41913: Bayes, AI and Deep Learning

Nick Polson

Fall, 2020

October 1, 2020
Getting Started

▶ Syllabus
http://faculty.chicagobooth.edu/nicholas.polson/teaching/41913/

▶ General Expectations
  1. Read the notes/Practice
  2. Be on schedule
  3. Polson and Scott: AIQ
  4. Polson and Sokolov: Deep Learning
Course Expectations

Homework: 20% Assignments. Handed in at Class
I encourage you to do assignments in groups.
Otherwise it’s no fun!
Grading is ✓−, ✓, ✓+

Final: 80% Week 11

Grading: PhD course
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Slide</th>
</tr>
</thead>
<tbody>
<tr>
<td>Section 1: Bayesian Machine Learning</td>
<td>Slide 7</td>
</tr>
<tr>
<td>Section 2: Artificial Intelligence</td>
<td>Slide 156</td>
</tr>
<tr>
<td>Section 3: Deep Learning</td>
<td>Slide 245</td>
</tr>
</tbody>
</table>
AIQ: People and Machines
Smarter Together

- Florence Nightingale
- Sir Isaac Newton
- Grace Hopper
Outline

1. Bayes
   (Hierarchical Models, Shrinkage, Modern Regression, Asset Allocation)

2. AI
   Predictive Analytics

3. Deep Learning
   (Kolmogorov-Arnold, NNs, SGD, Dropout, Applications)
Bayes, AI and Deep Learning: 41913

Week 1: Bayesian Machine Learning
Conditional Probability

Nick Polson

http://faculty.chicagobooth.edu/nicholas.polson/teaching/41913/
Probability

Subjective Probability (de Finetti, Ramsey, Savage, von Neumann, ...)

Principle of Coherence:

*A set of subjective probability beliefs must avoid sure loss*

1. If an event $A$ is certain to occur, it has probability 1
2. Either an event $A$ occurs or it does not.
   \[ P(A) = 1 - P(\text{not } A) \]

3. If two events are mutually exclusive (both cannot occur simultaneously) then
   \[ P(A \text{ or } B) = P(A) + P(B) \]

4. Joint probability, when events are independent
   \[ P(A \text{ and } B) = P(A)P(B) \]
Conditional, Joint and Marginal Distributions

Use probability to describe outcomes involving more than one variable at a time. Need to be able to measure what we think will happen to one variable relative to another

In general the notation is ...

- \( P(X = x, Y = y) \) is the joint probability that \( X = x \) and \( Y = y \)
- \( P(X = x \mid Y = y) \) is the conditional probability that \( X \) equals \( x \) given \( Y = y \)
- \( P(X = x) \) is the marginal probability of \( X = x \)
Conditional, Joint and Marginal Distributions

Relationship between the joint and conditional ...

\[ P(x, y) = P(x)P(y \mid x) \]
\[ = P(y)P(x \mid y) \]

Relationship between the joint and marginal ...

\[ P(x) = \sum_y P(x, y) \]
\[ P(y) = \sum_x P(x, y) \]
Bayes Rule

The computation of $P(x \mid y)$ from $P(x)$ and $P(y \mid x)$ is called Bayes theorem ...

$$P(x \mid y) = \frac{P(y, x)}{P(y)} = \frac{P(y, x)}{\sum_x P(y, x)} = \frac{P(y \mid x)P(x)}{\sum_x P(y \mid x)P(x)}$$

This shows now the conditional distribution is related to the joint and marginal distributions.
You’ll be given all the quantities on the r.h.s.
Bayes Rule

Key fact: $P(x \mid y)$ is generally different from $P(y \mid x)$!

Example: Most people would agree

$$Pr(\text{Practice hard} \mid \text{Play in NBA}) \approx 1$$
$$Pr(\text{Play in NBA} \mid \text{Practice hard}) \approx 0$$

The main reason for the difference is that $P(\text{Play in NBA}) \approx 0$. 
Independence

Two random variable $X$ and $Y$ are independent if

$$P(Y = y \mid X = x) = P(Y = y)$$

for all possible $x$ and $y$ values. Knowing $X = x$ tells you nothing about $Y$!

**Example:** Tossing a coin twice. What’s the probability of getting $H$ in the second toss given we saw a $T$ in the first one?
Bookies vs Betters: The Battle of Probabilistic Models

Source: The Secret Betting Strategy That Beats Online Bookmakers
Bookies vs Betters: The Battle of Probabilistic Models

- Bookies set odds that reflect their best guess on probabilities of a win, draw, or loss. Plus their own margin
- Bookies have risk aversion bias. When many people bet for an underdog (more popular team)
- Bookies hedge their bets by offering more favorable odds to the opposed team
- Simple algorithm: calculate average odds across many bookies and find outliers with large deviation from the mean
We can express probabilities in terms of Odds via

\[ O(A) = \frac{1 - P(A)}{P(A)} \quad \text{or} \quad P(A) = \frac{1}{1 + O(A)} \]

- For example if \( O(A) = 1 \) then for ever $1 bet you will payout $1. An event with probability \( \frac{1}{2} \).
- If \( O(A) = 2 \) or 2 : 1, then for a $1 bet you’ll payback $3. In terms of probability \( P = \frac{1}{3} \).
The following problem is known as the “exchange paradox”.

- A swami puts $m$ dollars in one envelope and $2m$ in another. He hands on envelope to you and one to your opponent.

The amounts are placed randomly and so there is a probability of $\frac{1}{2}$ that you get either envelope.

You open your envelope and find $x$ dollars. Let $y$ be the amount in your opponent’s envelope.
You know that $y = \frac{1}{2}x$ or $y = 2x$. You are thinking about whether you should switch your opened envelope for the unopened envelope of your friend. It is tempting to do an expected value calculation as follows

$$E(y) = \frac{1}{2} \cdot \frac{1}{2}x + \frac{1}{2} \cdot 2x = \frac{5}{4}x > x$$

Therefore, it looks as if you should switch no matter what value of $x$ you see. A consequence of this, following the logic of backwards induction, that even if you didn’t open your envelope that you would want to switch!
Bayes Rule

- Where’s the flaw in this argument? Use Bayes rule to update the probabilities of which envelope your opponent has! Assume $p(m)$ of dollars to be placed in the envelope by the swami.
- Such an assumption then allows us to calculate an odds ratio

$$\frac{p \left( y = \frac{1}{2}x \mid x \right)}{p \left( y = 2x \mid x \right)}$$

concerning the likelihood of which envelope your opponent has.
- Then, the expected value is given by

$$E(y) = p \left( y = \frac{1}{2}x \mid x \right) \cdot \frac{1}{2}x + p \left( y = 2x \mid x \right) \cdot 2x$$

and the condition $E(y) > x$ becomes a decision rule.
Three prisoners $A, B, C$.

Each believe are equally likely to be set free.

Prisoner $A$ goes to the warden $W$ and asks if s/he is getting axed.

▶ The Warden can’t tell $A$ anything about him.
▶ He provides the new information: $WB = "B \text{ is to be executed}"

Prisoner’s Dilemma
Prisoner’s Dilemma

Uniform Prior Probabilities:

<table>
<thead>
<tr>
<th>Prior</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P(\text{Pardon}) )</td>
<td>0.33</td>
<td>0.33</td>
<td>0.33</td>
</tr>
</tbody>
</table>

Posterior: Compute \( P(\mathcal{A}|\mathcal{WB}) \)?

What happens if \( \mathcal{C} \) overhears the conversation?

Compute \( P(\mathcal{C}|\mathcal{WB}) \)?
Game Show Problem

Named after the host of the long-running TV show, *Let’s make a Deal*.

- A contestant is given the choice of 3 doors. There is a *prize (a car, say)* behind one of the doors and something worthless behind the other two doors: two goats.
- The optimal strategy is *counter-intuitive*
The game is as follows:

- You pick a door.
- Monty then opens one of the other two doors, revealing a goat.
- You have the choice of switching doors.

Is it advantageous to switch?

Assume you pick door $A$ at random. Then $P(A) = (1/3)$.

You need to figure out $P(A|MB)$ after Monte reveals $B$ is a goat.
Probability and Psychology

How do people form probabilities or expectations in reality? Psychologists have categorized many different biases that people have in their beliefs or judgments.

**Loss Aversion**  The most important finding of Kahneman and Tversky is that people are loss averse. Utilities are defined over gains and losses rather than over final (or terminal) wealth, an idea first proposed by Markowitz. This is a violation of the EU postulates. Let \((x, y)\) denote a bet with gain \(x\) with probability \(y\). To illustrate this subjects were asked

*In addition to whatever you own, you have been given $1000, now choose between the gambles \(A = (1000, 0.5)\) and \(B = (500, 1)\).*

\(B\) was the more popular choice.
Example

The same subjects were then asked: *In addition to whatever you own, you have been given $2000, now choose between the gambles $C = (-1000, 0.5)$ and $D = (-500, 1)$.*

- This time $C$ was more popular.
- The key here is that their final wealth positions are identical yet people chose differently. The subjects are apparently focusing only on gains and losses. When they are not given any information about prior winnings, they choose $B$ over $A$ and $C$ over $D$. Clearly for a risk averse people this is the rational choice.
- This effect is known as loss aversion.
Representativeness

Representativeness When people try to determine the probability that evidence $A$ was generated by model $B$, they often use the representative heuristic. This means that they evaluate the probability by the degree to which $A$ reflects the essential characteristics of $B$.

A common bias is *base rate neglect* or ignoring prior evidence. For example, in tossing a fair coin the sequence *HHTHTHHTHH* with seven heads is likely to appear and yet people draw conclusions from too few data points and think 7 heads is representative of the true process and conclude $p = 0.7$. 
Expected Utility (EU) Theory

Normative

Let $P, Q$ be two probability distributions or risky gambles/lotteries. $pP + (1 - p)Q$ is the compound or mixture lottery.

The rational agent (You) will have preferences between gambles.

▶ We write $P \succeq Q$ if and only if You strictly prefer $P$ to $Q$. If two lotteries are indifferent we write $P \sim Q$.

▶ EU – a number of plausible axioms – completeness, transitivity, continuity and independence – then preferences are an expectation of a utility function.

▶ The theory is a normative one and not necessarily descriptive. It suggests how a rational agent should formulate beliefs and preferences and not how they actually behave.

▶ Expected utility $U(P)$ of a risky gamble is then

$$ P \succeq Q \iff U(P) \geq U(Q) $$
Attitudes to Risk

The solution depends on your risk preferences:

- **Risk neutral**: a risk neutral person is indifferent about fair bets. Linear Utility

- **Risk averse**: a risk averse person prefers certainty over fair bets.

\[ \mathbb{E}(U(X)) < U(\mathbb{E}(X)) \]

Concave utility

- **Risk loving**: a risk loving person prefer fair bets over certainty.

Depends on your preferences.
Ellsberg Paradox

*Probability is counter-intuitive!!!*

- Two urns
  1. 100 balls with 50 red and 50 blue.
  2. A mix of red and blue but you don’t know the proportion.
- Which urn would you like to bet on?
- People don’t like the “uncertainty” about the distribution of red/blue balls in the second urn.
Likelihood of Death

120 Stanford graduates:

<table>
<thead>
<tr>
<th>Cause</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Heart disease</td>
<td>34</td>
</tr>
<tr>
<td>cancer</td>
<td>23</td>
</tr>
<tr>
<td>Other natural causes</td>
<td>35</td>
</tr>
<tr>
<td>Total</td>
<td>58</td>
</tr>
<tr>
<td>actual</td>
<td>92</td>
</tr>
<tr>
<td>accident</td>
<td>5</td>
</tr>
<tr>
<td>homicide</td>
<td>1</td>
</tr>
<tr>
<td>other unnatural causes</td>
<td>2</td>
</tr>
<tr>
<td>Total</td>
<td>32</td>
</tr>
<tr>
<td>actual</td>
<td>8</td>
</tr>
</tbody>
</table>

The P’s don’t even sum up to one!
People vastly overestimate probability of violent death
St. Petersburg Paradox

What are you willing to pay to enter the following game?

- I toss a fair game and when the first head appears, on the $T$th toss, I pay you $2^T$ dollars.
- First, probability of first head on $T$th toss is $2^{-T}$

$$E(X) = \sum_{T=1}^{\infty} 2^T 2^{-T}$$

$$= 2(1/2) + 4(1/4) + 8(1/8) + \ldots$$

$$= 1 + 1 + 1 + \ldots \to \infty$$

- Bernoulli (1754) constructed utility to value bets with $E(u(X))$. 
Allais Paradox

You have to make a choice between the following gambles

- First compare the “Gambles”
  \[ G_1: \$5 \text{ million with certainty} \]
  \[ G_2: \$25 \text{ million } p = 0.10 \]
  \[ \$5 \text{ million } p = 0.89 \]
  \[ \$0 \text{ million } p = 0.01 \]

- Now choose between the Gambles
  \[ G_3: \$5 \text{ million } p = 0.11 \]
  \[ \$0 \text{ million } p = 0.89 \]
  \[ G_4: \$25 \text{ million } p = 0.10 \]
  \[ \$0 \text{ million } p = 0.90 \]

Fact: If \( G_1 \geq G_2 \) then \( G_3 \geq G_4 \) and vice-versa.
Solution: Expected Utility

Given (subjective) probabilities \( P = (p_1, p_2, p_3) \). Write \( E(U|P) \) for expected utility. W.l.o.g. set \( u(0) = 0 \) and for the high prize set \( u($25 million) = 1 \). Which leaves one free parameter \( u = u($5 million) \).

- Hence to compare gambles with probabilities \( P \) and \( Q \) we look at the difference

\[
E(u|P) - E(u|Q) = (p_2 - q_2)u + (p_3 - q_3)
\]

- For comparing \( G_1 \) and \( G_2 \) we get

\[
E(u|G_1) - E(u|G_2) = 0.11u - 0.1
\]
\[
E(u|G_3) - E(u|G_4) = 0.11u - 0.1
\]

The order is the same, given your \( u \).

- If your utility satisfies \( u < 0.1/0.11 = 0.909 \) you take the “riskier” gamble.
Power and log-utilities

- Constant relative risk aversion (CRRA).
- Advantage that the optimal rule is unaffected by wealth effects. The CRRA utility of wealth takes the form

\[ U_\gamma(W) = \frac{W^{1-\gamma} - 1}{1 - \gamma} \]

- The special case \( U(W) = \log(W) \) for \( \gamma = 1 \). This leads to a myopic Kelly criterion rule.
Kelly Criterion

Kelly Criterion corresponds to betting under binary uncertainty.

- Consider a sequence of i.i.d. bets where
  
  \[ p(X_t = 1) = p \text{ and } p(X_t = -1) = q = 1 - p \]

  The optimal allocation is \( \omega^* = p - q = 2p - 1 \).

- Maximising the expected long-run growth rate leads to the solution

  \[
  \max_{\omega} \mathbb{E} (\ln(1 + \omega W_T)) = p \ln(1 + \omega) + (1 - p) \ln(1 - \omega) \\
  \leq p \ln p + q \ln q + \ln 2 \text{ and } \omega^* = p - q
  \]
Let $p$ denote the probability of a gain and $O = (1 - p)/p$ the odds. We can generalize the rule to the case of asymmetric payouts $(a, b)$ where

$$p(X_t = 1) = p \quad \text{and} \quad p(X_t = -1) = q = 1 - p$$

Then the expected utility function is

$$p \ln(1 + b\omega) + (1 - p) \ln(1 - a\omega)$$

The optimal solution is

$$\omega^* = \frac{bp - aq}{ab} = \frac{p - q}{\sigma}$$
Kelly Criterion

- If $a = b = 1$ this reduces to the pure Kelly criterion.
- A common case occurs when $a = 1$ and market odds $b = O$. The rule becomes

$$\omega^* = \frac{p \cdot O - q}{O}$$

Example

- Two possible market opportunities: one where it offers you 4/1 when you have personal odds of 3/1 and a second one when it offers you 12/1 while you think the odds are 9/1. In expected return these two scenarios are identical both offering a 33% gain. In terms of maximizing long-run growth, however, they are not identical.
Example

▶ Table 1 shows the Kelly criteria advises an allocation that is twice as much capital to the lower odds proposition: 1/16 weight versus 1/40.

<table>
<thead>
<tr>
<th>Market</th>
<th>You</th>
<th>p</th>
<th>(\omega^*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4/1</td>
<td>3/1</td>
<td>1/4</td>
<td>1/16</td>
</tr>
<tr>
<td>12/1</td>
<td>9/1</td>
<td>1/10</td>
<td>1/40</td>
</tr>
</tbody>
</table>

**Table: Kelly rule**

▶ The optimal allocation \(\omega^* = \frac{pO - q}{O}\) is

\[
\frac{\left(\frac{1}{4}\right) \times 4 - \left(\frac{3}{4}\right)}{4} = \frac{1}{16} \quad \text{and} \quad \frac{\left(\frac{1}{10}\right) \times 12 - \left(\frac{9}{10}\right)}{12} = \frac{1}{40}
\]
SuperBowl XLVII: Ravens vs 49ers

Figure: SuperBowl XLVII
Super Bowl XLVII: Ravens vs 49ers

Super Bowl XLVII was held at the Superdome in New Orleans on February 3, 2013.

We will track $X(t)$ which corresponds to the Raven’s lead over the 49ers at each point in time. Table 3 provides the score at the end of each quarter.

<table>
<thead>
<tr>
<th>$t$</th>
<th>0</th>
<th>$\frac{1}{4}$</th>
<th>$\frac{1}{2}$</th>
<th>$\frac{3}{4}$</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ravens</td>
<td>0</td>
<td>7</td>
<td>21</td>
<td>28</td>
<td>34</td>
</tr>
<tr>
<td>49ers</td>
<td>0</td>
<td>3</td>
<td>6</td>
<td>23</td>
<td>31</td>
</tr>
<tr>
<td>$X(t)$</td>
<td>0</td>
<td>4</td>
<td>15</td>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>

Super Bowl XLVII by Quarter
Initial Market

Initial point spread Ravens being a four point underdog, \( \mu = -4 \).

\[
\mu = \mathbb{E}(X(1)) = -4.
\]

The Ravens upset the 49ers by \(34 - 31\) and \(X(1) = 34 - 31 = 3\) with the point spread being beaten by 7 points.

- To determine the markets' assessment of the probability that the Ravens would win at the beginning of the game we use the money-line odds.

San Francisco \(-175\) and Baltimore Ravens \(+155\). This implies that a bettor would have to place $175 to win $100 on the 49ers and a bet of $100 on the Ravens would lead to a win of $155.

Convert both of these money-lines to implied probabilities of the each team winning

\[
p_{SF} = \frac{175}{100 + 175} = 0.686 \quad \text{and} \quad p_{Bal} = \frac{100}{100 + 155} = 0.392
\]
Probabilities of Winning

The probabilities do not sum to one. This "excess" probability is the "market vig", also known as the bookmaker's edge.

\[ p_{SF} + p_{Bal} = 0.686 + 0.392 = 1.078 \]

providing a 7.8% edge for the bookmakers.

- Put differently, if bettors place money proportionally across both teams then the bookies will make 7.8% of the total staked.

We use the mid-point of the spread to determine \( p \) implying that

\[ p = \frac{1}{2} p_{Bal} + \frac{1}{2} (1 - p_{SF}) = 0.353 \]

From the Ravens perspective, we have \( p = \Pr(X(1) > 0) = 0.353 \). Baltimore’s win probability started trading at \( p_{0}^{mkt} = 0.38 \).
The Ravens took a commanding 21 – 6 lead at half time. Market was trading at $p_{mkt}^{1/2} = 0.90$.

- During the 34 minute blackout 42760 contracts changed hands with Baltimore’s win probability ticking down from 95 to 94.
- The win probability peak of 95% occurred after a third-quarter kickoff return for a touchdown.
- At the end of the four quarter, however, when the 49ers nearly went into the lead with a touchdown, Baltimore’s win probability had dropped to 30%.
Implied Volatility

To calculate the implied volatility of the Superbowl we substitute the pair \((\mu, p) = (-4, .353)\) into our definition and solve for \(\sigma_{IV}\).

\[
\sigma = \frac{\mu}{\Phi^{-1}(p)},
\]

- We obtain

\[
\sigma_{IV} = \frac{\mu}{\Phi^{-1}(p)} = \frac{-4}{-0.377} = 10.60
\]

where \(\Phi^{-1}(p) = qnorm(0.353) = -0.377\). The 4 point advantage assessed for the 49ers is under a \(\frac{1}{2}\sigma\) favorite.

- The outcome \(X(1) = 3\) was within one standard deviation of the pregame model which had an expectation of \(\mu = -4\) and volatility of \(\sigma = 10.6\).
Half Time Probabilities

What’s the probability of the Ravens winning given their lead at half time?
At half time Baltimore led by 15 points, 21 to 6.

The conditional mean for the final outcome is $15 + 0.5 \times (-4) = 13$
and the conditional volatility is $10.6 \sqrt{1-t}$.
These imply a probability of .96 for Baltimore to win the game.

A second estimate of the probability of winning given the half time lead can be obtained directly from the betting market.
From the online betting market we also have traded contracts on TradeSports.com that yield a half time probability of $p_{\frac{1}{2}} = 0.90$. 
What’s the implied volatility for the second half?

$p_{t}^{mkt}$ reflects all available information

- For example, at half-time $t = \frac{1}{2}$ we would update

$$\sigma_{IV,t=\frac{1}{2}} = \frac{l + \mu(1-t)}{\Phi^{-1}(p_t)\sqrt{1-t}} = \frac{15 - 2}{\Phi^{-1}(0.9)/\sqrt{2}} = 14$$

where $qnorm(0.9) = 1.28$.

- As $14 > 10.6$, the market was expecting a more volatile second half—possibly anticipating a comeback from the 49ers.
How can we form a valid betting strategy?

Given the initial implied volatility $\sigma = 10.6$.

At half time with the Ravens having a $l + \mu(1 - t) = 13$ points edge

▶ We would assess with $\sigma = 10.6$

$$p_{1/2} = \Phi \left( \frac{13}{(10.6/\sqrt{2})} \right) = 0.96$$

probability of winning versus the $p_{1/2}^{\text{mkt}} = 0.90$ rate.

▶ To determine our optimal bet size, $\omega_{\text{bet}}$, on the Ravens we might appeal to the Kelly criterion (Kelly, 1956) which yields

$$\omega_{\text{bet}} = p_{1/2} - \frac{q_{1/2}}{O_{\text{mkt}}} = 0.96 - \frac{0.1}{1/9} = 0.60$$
Bayes, AI and Deep Learning: 41913

Week 2: Bayesian Updating

Nick Polson

http://faculty.chicagobooth.edu/nicholas.polson/teaching/41913/
Bayes Rule

In its simplest form.

▶ Two events $A$ and $B$. Bayes rule

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B|A)P(A)}{P(B)}$$

▶ Law of Total Probability

$$P(B) = P(B|A)P(A) + P(B|\bar{A})P(\bar{A})$$

Hence we can calculate the denominator of Bayes rule.
Bayes Theorem

Many problems in decision making can be solved using Bayes rule.

- AI: Rule-based decision making.
- It’s counterintuitive! But gives the “right” answer.

Bayes Rule:

\[
P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B|A)P(A)}{P(B)}
\]

Law of Total Probability:

\[
P(B) = P(B|A)P(A) + P(B|\bar{A})P(\bar{A})
\]
Apple Watch

The Apple Watch Series 4 can perform a single-lead ECG and detect atrial fibrillation. The software can correctly identify 98% of cases of atrial fibrillation (true positives) and 99% of cases of non-atrial fibrillation (true negatives).

However, what is the probability of a person having atrial fibrillation when atrial fibrillation is identified by the Apple Watch Series 4?

Bayes’ Theorem:

\[ P(A|B) = \frac{P(B|A)P(A)}{P(B)} \]
Apple Watch

<table>
<thead>
<tr>
<th>Predicted</th>
<th>atrial fibrillation</th>
<th>no atrial fibrillation</th>
</tr>
</thead>
<tbody>
<tr>
<td>atrial fibrillation</td>
<td>1960</td>
<td>980</td>
</tr>
<tr>
<td>no atrial fibrillation</td>
<td>40</td>
<td>97020</td>
</tr>
</tbody>
</table>

\[ 0.6667 = \frac{0.98 \cdot 0.02}{0.0294} \]

The conditional probability of having atrial fibrillation when the Apple Watch Series 4 detects atrial fibrillation is about 67%.
Abraham Wald

How Abraham Wald improved aircraft survivability. Raw Reports from the Field

<table>
<thead>
<tr>
<th>Type of damage suffered</th>
<th>Returned (316 total)</th>
<th>Shot down (60 total)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engine</td>
<td>29</td>
<td>?</td>
</tr>
<tr>
<td>Cockpit</td>
<td>36</td>
<td>?</td>
</tr>
<tr>
<td>Fuselage</td>
<td>105</td>
<td>?</td>
</tr>
<tr>
<td>None</td>
<td>146</td>
<td>0</td>
</tr>
</tbody>
</table>

This fact would allow Wald to estimate:

\[ P(\text{damage on fuselage} \mid \text{returns safely}) = \frac{105}{316} \approx 32\% \]

You need the inverse probability:

\[ P(\text{returns safely} \mid \text{damage on fuselage}) \]

Completely different!
Abraham Wald

Wald invented a method to implement the missing data, which is called by data scientist as “imputation”. Wald Invented A Method for Reconstructing the Full Table

<table>
<thead>
<tr>
<th>Type of damage suffered</th>
<th>Returned (316 total)</th>
<th>Shot down (60 total)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engine</td>
<td>29</td>
<td>31</td>
</tr>
<tr>
<td>Cockpit</td>
<td>36</td>
<td>21</td>
</tr>
<tr>
<td>Fuselage</td>
<td>105</td>
<td>8</td>
</tr>
<tr>
<td>None</td>
<td>146</td>
<td>0</td>
</tr>
</tbody>
</table>

Then Wald got:

\[
P(\text{returns safely} \mid \text{damage on fuselage}) = \frac{105}{105 + 8} \approx 93% \]

\[
P(\text{returns safely} \mid \text{damage on engine}) = \frac{29}{29 + 31} \approx 48% \]
“Personalization” = “Conditional Probability”

Conditional probability is how AI systems express judgments in a way that reflects their partial knowledge.

Personalization runs on conditional probabilities, all of which must be estimated from massive data sets in which you are the conditioning event.

Many Business Applications!! Suggestions vs Search, ....
Probability as Evidence

evidence: known facts about criminal (e.g. blood type, DNA, ...)

suspect: matches a trait with evidence at scene of crime

Let $G$ denote the event that the suspect is the criminal.

- Bayes computes the conditional probability of guilt

$$P(G | \text{evidence})$$

Evidence $E$: suspect and criminal possess a common trait
Probability as Evidence

Bayes Theorem yields

$$P(G|\text{evidence}) = \frac{P(\text{evidence}|G)P(G)}{P(\text{evidence})}$$

In terms of relative odds

$$\frac{P(I|\text{evidence})}{P(G|\text{evidence})} = \frac{P(\text{evidence}|I)P(I)}{P(\text{evidence}|G)P(G)}$$
Bayes Factors

There are two terms:

1. Prior Odds of Guilt $O(\mathcal{G}) = P(\mathcal{I}) / P(\mathcal{G})$?

   How many people on the island?
   Sensitivity “what if” analysis?

2. The Bayes factor

   $$\frac{P(\text{evidence}|\mathcal{I})}{P(\text{evidence}|\mathcal{G})}$$

   is common to all observers and updates everyone’s initials odds
Prosecutor’s Fallacy

The most common fallacy is confusing

\[ P(\text{evidence}|\mathcal{G}) \] with \[ P(\mathcal{G}|\text{evidence}) \]

- Bayes rule yields

\[ P(\mathcal{G}|\text{evidence}) = \frac{P(\text{evidence}|\mathcal{G})p(\mathcal{G})}{P(\text{evidence})} \]

- Your assessment of \[ P(\mathcal{G}) \] will matter.
Island Problem

Suppose there’s a criminal on a island of $N + 1$ people.

- Let $I$ denote innocence and $G$ guilt.
- Evidence $E$: the suspect matches a trait with the criminal.
- The probabilities are

$$p(E|I) = p \quad \text{and} \quad p(E|G) = 1$$
Bayes factors are likelihood ratios

- The Bayes factor is given by

\[
\frac{p(E|I)}{p(E|G)} = p
\]

- If we start with a uniform prior distribution we have

\[
p(I) = \frac{1}{N + 1} \quad \text{and} \quad \text{odds}(I) = N
\]

- Priors will matter!
Island Problem

Posterior Probability related to Odds

\[
p(I|y) = \frac{1}{1 + \text{odds}(I|y)}
\]

-Prosecutors’ fallacy

The posterior probability \( p(I|y) \neq p(y|I) = p \).

- Suppose that \( N = 10^3 \) and \( p = 10^{-3} \). Then

\[
p(I|y) = \frac{1}{1 + 10^3 \cdot 10^{-3}} = \frac{1}{2}
\]

The odds on innocence are \( \text{odds}(I|y) = 1 \).

There’s a 50/50 chance that the criminal has been found.
Sally Clark was accused and convicted of killing her two children. They could have both died of SIDS.

- The chance of a family which are non-smokers and over 25 having a SIDS death is around 1 in 8,500.
- The chance of a family which has already had a SIDS death having a second is around 1 in 100.
- The chance of a mother killing her two children is around 1 in 1,000,000.
Bayes or Independence

1. Under Bayes

\[
P(\text{both SIDS}) = P(\text{first SIDS}) \cdot P(\text{Second SIDS} | \text{first SIDS})
\]

\[
= \frac{1}{8500} \cdot \frac{1}{100} = \frac{1}{850,000}
\]

The \( \frac{1}{100} \) comes from taking into account genetics.

2. Independence, as the court did, gets you

\[
P(\text{both SIDS}) = \left(\frac{1}{8500}\right) \left(\frac{1}{8500}\right) = \left(\frac{1}{73,000,000}\right)
\]

3. By Bayes rule

\[
\frac{p(I|E)}{p(G|E)} = \frac{P(E \cap I)}{P(E \cap G)}
\]

\[
P(E \cap I) = P(E|I)P(I) \text{ needs discussion of } p(I).
\]
Comparison

- Hence putting these two together gives the odds of guilt as

\[
\frac{p(I|E)}{p(G|E)} = \frac{1/850,000}{1/1,000,000} = 1.15
\]

In terms of posterior probabilities

\[
p(G|E) = \frac{1}{1 + O(G|E)} = 0.465
\]

- If you use independence

\[
\frac{p(I|E)}{p(G|E)} = \frac{1}{73} \text{ and } p(G|E) \approx 0.99
\]

The suspect looks guilty.
OJ Simpson

The O.J. Simpson trial was possibly the trail of the century. The murder of his wife Nicole Brown Simpson, and a friend, Ron Goldman, in June 1994 and the trial dominated the TV networks.

- DNA evidence and probability: $p(E|I)$
- Bayes Theorem: $p(G|E)$
- Prosecutor’s Fallacy: $p(G|E) \neq p(E|G)$

Odds ratio with $I = \bar{G}$ gives

$$\frac{p(I|E)}{p(G|E)} = \frac{p(E|I)}{p(E|G)} \frac{p(I)}{p(G)}$$

Prior odds conditioned on background information.
Suppose that you are a juror in a murder case of a husband who is accused of killing his wife.

The husband is known is have battered her in the past.

Consider the three events:

1. $G$ “husband murders wife in a given year”
2. $M$ “wife is murdered in a given year”
3. $B$ “husband is known to batter his wife”
OJ Simpson: Bayes Theorem

- Only 1/10th of one percent of husbands who batter their wife actually murder them. Conditional on eventually murdering their wife, there a one in ten chance it happens in a given year.

In a given year, there are about 5 murders per 100,000 of population in the United States.

In 1994, 5000 women were murdered, 1500 by their husband. Given a population of 100 million women at the time

\[ p(M|I) = \frac{3500}{1 \times 10^8} \approx \frac{1}{30,000}. \]

We’ll also need \( p(M|I, B) \)
OJ Simpson: Prosecutor’s Fallacy

Let $G = $ Guilt and $E = $ Evidence

*Prosecutor’s Fallacy:* $P(G|E) \neq P(E|G)$.

DNA evidence gives $P(E|I)$ – the $p$-value.

What’s the “match probability” for a rare event

Bayes theorem in Odds

$$\frac{p(G|M,B)}{p(I|M,B)} = \frac{p(M|G,B) \cdot p(G|B)}{p(M|I,B) \cdot p(I|B)}$$
By assumption,

- \( p(M|G,B) = 1 \)
- \( p(M|I,B) = \frac{1}{20,000} \)
- \( p(G|B) = \frac{1}{1000} \) and so

\[
\frac{p(G|B)}{p(I|B)} = \frac{1}{10,000}
\]

Therefore,

\[
\frac{p(G|M,B)}{p(I|M,B)} = 2 \quad \text{and} \quad p(G|M,B) = \frac{2}{3}
\]

More than a 50/50 chance that your spouse murdered you!
Fallacy $p(G|B) \neq p(G|B,M)$

The defense stated to the press: in any given year

"Fewer than 1 in 2000 of batterers go on to murder their wives".

▶ Now estimate $p(M|\bar{G},B) = p(M|\bar{G}) = \frac{1}{20,000}$.

▶ The Bayes factor is then

$$\frac{p(G|M,B)}{p(\bar{G}|M,B)} = \frac{1/2000}{1/20,000} = 10$$

which implies posterior probabilities

$$p(\bar{G}|M,B) = \frac{1}{1+10} \text{ and } p(G|M,B) = \frac{10}{11}$$

Hence its over 90% chance that O.J. is guilty based on this information!

Defense intended this information to exonerate O.J.
More than One Criminal

Let $E$ two criminals and suspect matches $C$.

- Rare blood type ($p = 0.1$) and Common blood type $C$ ($p = 0.6$)
- Evidence for the prosecution?
- Bayes: $P(C|I) = 0.6$ and $P(C|G) = 0.5$

$$BF = \frac{P(C|G)}{P(C|I)} = \frac{0.5}{0.6} = 0.833 < 1!$$

Evidence for the Defense. Posterior Odds on Guilt are decreased.
"Witness" 80% certain saw a "checker" C taxi in the accident.

- What's your P(C|E) ?
- Need P(C). Say P(C) = 0.2 and P(E) = 0.8.
- Then your posterior is

\[
P(C|E) = \frac{0.8 \cdot 0.2}{0.8 \cdot 0.2 + 0.2 \cdot 0.8} = 0.5
\]

Therefore O(C) = 1 a 50/50 bet.
Updating Fallacies

Most people don’t update quickly enough in light of new data.
Wards Edwards 1960s

When you have a small sample size, Bayes rule still updates probabilities:

- Two players: either 70 % A or 30 % A
- Observe A beats B 3 times out of 4.
- What’s $P(A = 70\% \text{ player})$?
Suppose that $X \sim F_X(x)$ and let $Y = g(X)$.
How do we find $F_Y(y)$ and $f_Y(y)$?

- **von Neumann**
  Given a uniform $U$, how do we find $X = g(U)$?

- **In the bivariate case** $(X, Y) \rightarrow (U, V)$.
  We need to find $f_{(U,V)}(u,v)$ from $f_{X,Y}(x,y)$

- **Applications:** Simulation, MCMC and PF.
Transformations

The cdf identity gives

\[ F_Y(y) = \mathbb{P}(Y \leq y) = \mathbb{P}(g(X) \leq y) \]

▫ Hence if the function \( g(\cdot) \) is monotone we can invert to get

\[ F_Y(y) = \int_{g(x) \leq y} f_X(x) \, dx \]

▫ If \( g \) is increasing \( F_Y(y) = P(X \leq g^{-1}(y)) = F_X(g^{-1}(y)) \)
  If \( g \) is decreasing \( F_Y(y) = P(X \geq g^{-1}(y)) = 1 - F_X(g^{-1}(y)) \)
1. Theorem 1: Let $X$ have pdf $f_X(x)$ and let $Y = g(X)$. Then if $g$ is a monotone function we have

$$f_Y(y) = f_X(g^{-1}(y)) \left| \frac{d}{dy}g^{-1}(y) \right|$$

There’s also a multivariate version of this that we’ll see later.

- Suppose $X$ is a continuous rv, what’s the pdf for $Y = X^2$?
- Let $X \sim N(0, 1)$, what’s the pdf for $Y = X^2$?
Theorem

Suppose that $U \sim U[0, 1]$, then for any continuous distribution function $F$, the random variable $X = F^{-1}(U)$ has distribution function $F$.

- Remember that for $u \in [0, 1]$, $P(U \leq u) = u$, so we have

$$P(X \leq x) = P\left(F^{-1}(U) \leq x\right) = P(U \leq F(x)) = F(x)$$

Hence, $X = F_X^{-1}(U)$. 

Probability Integral Transform
Simulation and Transformations

An important application is how to transform multiple random variables?

- Suppose that we have random variables:

\[(X, Y) \sim f_{X,Y}(x, y)\]

A transformation of interest given by:

\[U = g(X, Y) \quad \text{and} \quad V = h(X, Y)\]

- The problem is how to compute \(f_{U,V}(u, v)\)? Jacobian

\[J = \frac{\partial (x, y)}{\partial (u, v)} = \begin{vmatrix} \frac{\partial x}{\partial u} & \frac{\partial x}{\partial v} \\ \frac{\partial y}{\partial u} & \frac{\partial y}{\partial v} \end{vmatrix}\]
Bivariate Change of Variable

▶ *Theorem:* (change of variable)

\[ f_{U,V}(u, v) = f_{X,Y}(h_1(u, v), h_2(u, v)) \left| \frac{\partial (x, y)}{\partial (u, v)} \right| \]

The last term is the Jacobian.
This can be calculated in two ways.

\[ \left| \frac{\partial (x, y)}{\partial (u, v)} \right| = 1/ \left| \frac{\partial (u, v)}{\partial (x, y)} \right| \]

▶ So we don’t always need the inverse transformation
\[(x, y) = (g^{-1}(u, v), h^{-1}(u, v))\]
Inequalities and Identities

1. Markov

\[ \mathbb{P} (g(X) \geq c) \leq \frac{\mathbb{E}(g(X))}{c} \quad \text{where} \quad g(X) \geq 0 \]

2. Chebyshev

\[ \mathbb{P} (|X - \mu| \geq c) \leq \frac{\text{Var}(X)}{c^2} \]

3. Jensen

\[ \mathbb{E} (\phi(X)) \leq \phi (\mathbb{E}(X)) \]

4. Cauchy-Schwarz

\[ \text{corr}(X, Y) \leq 1 \]

Chebyshev follows from Markov. Mike Steele and Cauchy-Schwarz.
Special Distributions

See *Common Distributions*

1. Bernoulli and Binomial
2. Hypergeometric
3. Poisson
4. Negative Binomial
5. Normal Distribution
6. Gamma Distribution
7. Beta Distribution
8. Multinomial Distribution
9. Bivariate Normal Distribution
10. Wishart Distribution

...
Example: Markov Dependence

- We can always factor a joint distribution as

\[ p(X_n, X_{n-1}, \ldots, X_1) = p(X_n | X_{n-1}, \ldots, X_1) \cdots p(X_2 | X_1) p(X_1) \]

Example

- A process has the *Markov Property* if

\[ p(X_n | X_{n-1}, \ldots, X_1) = p(X_n | X_{n-1}) \]

- Only the current history matter when determining the probabilities.
SP500 daily ups and downs

- Daily return data from 1948 to 2007 for the SP500 index of stocks
- Can we calculate the probability of ups and downs?
A real world probability model

Hidden Markov Models

Are stock returns a random walk?

Hidden Markov Models (Baum-Welch, Viterbi)

- Daily returns on the SP500 stock market index.
  Build a hidden Markov model to predict the ups and downs.
- Suppose that stock market returns on the next four days are $X_1, \ldots, X_4$.
- Let’s empirical determine conditionals and marginals
Marginal and Bivariate Distributions


\[
\begin{array}{c|cc}
 x & \text{Down} & \text{Up} \\
 P(X_i) = x & 0.474 & 0.526 \\
\end{array}
\]

Finding \( p(X_2|X_1) \) is twice as much computational effort: counting \(UU, UD, DU, DD\) transitions.

\[
\begin{array}{c|cc}
 X_i & \text{Down} & \text{Up} \\
 X_{i-1} = \text{Down} & 0.519 & 0.481 \\
 X_{i-1} = \text{Up} & 0.433 & 0.567 \\
\end{array}
\]
Conditioned on two days

Let's do $p(X_3|X_2, X_1)$

<table>
<thead>
<tr>
<th>$X_{i-2}$</th>
<th>$X_{i-1}$</th>
<th>Down</th>
<th>Up</th>
</tr>
</thead>
<tbody>
<tr>
<td>Down</td>
<td>Down</td>
<td>0.501</td>
<td>0.499</td>
</tr>
<tr>
<td>Down</td>
<td>Up</td>
<td>0.412</td>
<td>0.588</td>
</tr>
<tr>
<td>Up</td>
<td>Down</td>
<td>0.539</td>
<td>0.461</td>
</tr>
<tr>
<td>Up</td>
<td>Up</td>
<td>0.449</td>
<td>0.551</td>
</tr>
</tbody>
</table>

We could do the distribution $p(X_2, X_3|X_1)$. This is a joint, marginal and conditional distribution all at the same time. 

*Joint* because more than one variable $(X_2, X_3)$, *marginal* because it ignores $X_4$ and *conditional* because its given $X_1$. 
Joint Probabilities

- Under Markov dependence

\[
P(UUD) = p(X_1 = U)p(X_2 = U|X_1 = U)p(X_3|X_2 = U, X_1 = U)
\]
\[
= (0.526)(0.567)(0.433)
\]

- Under independence we would have got

\[
P(UUD) = P(X_1 = U)p(X_2 = U)p(X_3 = D)
\]
\[
= (.526)(.526)(.474)
\]
\[
= 0.131
\]
| Team        | Best Odds | Sign Up Offers | £20 | £20 | £30 | £100 | £40 | £25 | £40 | £20 | £10 | £50 | £10 | £50 | £30 | £20 | £50 | £20 | £10 | £20 | £20 | £10 | £10 |
|-------------|-----------|----------------|-----|-----|-----|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| Man City    | 2/5       | 4/9            | 1/2 |     |     |      |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| Liverpool   | 3         | 11/4           | 11/4|     |     | 11/1| 11/1|     |     |     |     |     |     |     |     |     |     |     |     |     |     |     |
| Tottenham   | 20        | 18             | 16  |     |     | 20   | 16  | 20 | 20 | 16 |     |     |     |     |     |     |     |     |     |     |     |     |
| Chelsea     | 40        | 50             | 33  |     |     | 50   | 33  | 50 | 50 | 33 |     |     |     |     |     |     |     |     |     |     |     |     |
| Man Utd     | 40        | 50             | 33  |     |     | 50   | 28  | 40 | 40 | 33 |     |     |     |     |     |     |     |     |     |     |     |     |
| Arsenal     | 50        | 50             | 40  |     |     | 45   | 40  | 45 | 40 | 33 |     |     |     |     |     |     |     |     |     |     |     |     |
| Everton     | 150       | 250            | 200 |     |     | 250  | 200 | 250| 250| 200|     |     |     |     |     |     |     |     |     |     |     |     |
| Wolves      | 200       | 200            | 200 |     |     | 250  | 150 | 250| 150| 150|     |     |     |     |     |     |     |     |     |     |     |     |
| Leicester   | 300       | 250            | 200 |     |     | 250  | 200 | 250| 200| 250|     |     |     |     |     |     |     |     |     |     |     |     |
| West Ham    | 500       | 750            | 500 |     |     | 500  | 500 | 500| 500| 500|     |     |     |     |     |     |     |     |     |     |     |     |
| Newcastle   | 1000      | 1500           | 750 |     |     | 1500 | 1000| 1500|1000| 750|     |     |     |     |     |     |     |     |     |     |     |     |
| Aston Villa | 1000      | 1500           | 750 |     |     | 1500 | 500 | 1500|500 | 1000|     |     |     |     |     |     |     |     |     |     |     |     |
Calculate **Odds for the possible scores in a match?**

\[ 0 - 0, 1 - 0, 0 - 1, 1 - 1, 2 - 0, \ldots \]

Let \( X = \) Goals scored by Arsenal

\( Y = \) Goals scored by Liverpool

What’s the odds of a **team winning?** \( P (X > Y) \)  Odds of a **draw?** \( P (X = Y) \)

\[
\begin{align*}
z1 & = \text{rpois}(100,0.6) \\
z2 & = \text{rpois}(100,1.4) \\
\text{sum}(z1<z2)/100 & \quad \# \text{ Team 2 wins} \\
\text{sum}(z1=z2)/100 & \quad \# \text{ Draw}
\end{align*}
\]
Chelsea EPL 2017

Let’s take a historical set of data on scores. Then estimate $\lambda$ with the sample mean of the home and away scores.

<table>
<thead>
<tr>
<th>home team</th>
<th>results</th>
<th>visit team</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chelsea</td>
<td>2 1</td>
<td>West Ham</td>
</tr>
<tr>
<td>Chelsea</td>
<td>5 1</td>
<td>Sunderland</td>
</tr>
<tr>
<td>Watford</td>
<td>1 2</td>
<td>Chelsea</td>
</tr>
<tr>
<td>Chelsea</td>
<td>3 0</td>
<td>Burnley</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
EPL Chelsea

(a) Chelsea against

(b) Chelsea for

Our Poisson model fits the empirical data!!
### EPL: Attack and Defence Strength

Each team gets an “attack” strength and “defence” weakness rating. Adjust home and away average goal estimates.

<table>
<thead>
<tr>
<th>Team</th>
<th>Points</th>
<th>Goals for</th>
<th>’Attack strength’</th>
<th>Goals against</th>
<th>’Defence weakness’</th>
</tr>
</thead>
<tbody>
<tr>
<td>Man United</td>
<td>87</td>
<td>67</td>
<td>1.46</td>
<td>24</td>
<td>0.52</td>
</tr>
<tr>
<td>Liverpool</td>
<td>83</td>
<td>74</td>
<td>1.61</td>
<td>26</td>
<td>0.57</td>
</tr>
<tr>
<td>Chelsea</td>
<td>80</td>
<td>65</td>
<td>1.41</td>
<td>22</td>
<td>0.48</td>
</tr>
<tr>
<td>Arsenal</td>
<td>69</td>
<td>64</td>
<td>1.39</td>
<td>36</td>
<td>0.78</td>
</tr>
<tr>
<td>Everton</td>
<td>60</td>
<td>53</td>
<td>1.15</td>
<td>37</td>
<td>0.80</td>
</tr>
<tr>
<td>Aston Villa</td>
<td>59</td>
<td>53</td>
<td>1.15</td>
<td>48</td>
<td>1.04</td>
</tr>
<tr>
<td>Fulham</td>
<td>53</td>
<td>39</td>
<td>0.85</td>
<td>32</td>
<td>0.70</td>
</tr>
<tr>
<td>Tottenham</td>
<td>51</td>
<td>44</td>
<td>0.96</td>
<td>42</td>
<td>0.91</td>
</tr>
<tr>
<td>West Ham</td>
<td>48</td>
<td>40</td>
<td>0.87</td>
<td>44</td>
<td>0.96</td>
</tr>
<tr>
<td>Man City</td>
<td>47</td>
<td>57</td>
<td>1.24</td>
<td>50</td>
<td>1.09</td>
</tr>
<tr>
<td>Stoke</td>
<td>45</td>
<td>37</td>
<td>0.80</td>
<td>51</td>
<td>1.11</td>
</tr>
<tr>
<td>Wigan</td>
<td>42</td>
<td>33</td>
<td>0.72</td>
<td>45</td>
<td>0.98</td>
</tr>
<tr>
<td>Bolton</td>
<td>41</td>
<td>41</td>
<td>0.89</td>
<td>52</td>
<td>1.13</td>
</tr>
<tr>
<td>Portsmouth</td>
<td>41</td>
<td>38</td>
<td>0.83</td>
<td>56</td>
<td>1.22</td>
</tr>
<tr>
<td>Blackburn</td>
<td>40</td>
<td>40</td>
<td>0.87</td>
<td>60</td>
<td>1.30</td>
</tr>
<tr>
<td>Sunderland</td>
<td>36</td>
<td>32</td>
<td>0.70</td>
<td>51</td>
<td>1.11</td>
</tr>
<tr>
<td>Hull</td>
<td>35</td>
<td>39</td>
<td>0.85</td>
<td>63</td>
<td>1.37</td>
</tr>
<tr>
<td>Newcastle</td>
<td>34</td>
<td>40</td>
<td>0.87</td>
<td>58</td>
<td>1.26</td>
</tr>
<tr>
<td>Middlesbrough</td>
<td>32</td>
<td>27</td>
<td>0.59</td>
<td>55</td>
<td>1.20</td>
</tr>
<tr>
<td>West Brom</td>
<td>31</td>
<td>36</td>
<td>0.78</td>
<td>67</td>
<td>1.46</td>
</tr>
</tbody>
</table>
EPL: Hull vs ManU

Poisson Distribution

ManU Average away goals = 1.47. Prediction:
$1.47 \times 1.46 \times 1.37 = 2.95$
Attack strength times Hull’s defense weakness times average

Hull Average home goals = 1.47. Prediction:
$1.47 \times 0.85 \times 0.52 = 0.65$. Simulation

<table>
<thead>
<tr>
<th>Team</th>
<th>Expected Goals</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Man U</td>
<td>2.95</td>
<td>7</td>
<td>22</td>
<td>26</td>
<td>12</td>
<td>11</td>
<td>13</td>
</tr>
<tr>
<td>Hull City</td>
<td>0.65</td>
<td>49</td>
<td>41</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
EPL Predictions

A model is only as good as its predictions

▶ In our simulation Man U wins 88 games out of 100, we should bet when odds ratio is below 88 to 100.
▶ Most likely outcome is 0-3 (12 games out of 100)
▶ The actual outcome was 0-1 (they played on August 27, 2016)
▶ In out simulation 0-1 was the fourth most probable outcome (9 games out of 100)
Bayes, AI and Deep Learning: 41913

Week 3: Bayes
Hierarchical Distributions

Nick Polson

http://faculty.chicagobooth.edu/nicholas.polson/teaching/41913/
Bayesian Methods

Modern Statistical/Machine Learning

▶ Bayes Rule and Probabilistic Learning
▶ Computationally challenging: MCMC and Particle Filtering
▶ Many applications in Finance:
    Asset pricing and corporate finance problems.

Lindley, D.V. *Making Decisions*

Bernardo, J. and A.F.M. Smith *Bayesian Theory*
Bayesian Books

- Hierarchical Models and MCMC
- Bayesian Nonparametrics Machine Learning
- Dynamic State Space Models ...
Popular Books

McGrayne (2012): The Theory that would not Die

- History of Bayes-Laplace
- Code breaking
- Bayes search: Air France
Nate Silver: 538 and NYT

Silver (2012): The Signal and The Noise

- Presidential Elections
- Bayes dominant methodology
- Predicting College Basketball/Oscars . . .
Things to Know

Explosion of Models and Algorithms starting in 1950s
- Bayesian Regularisation and Sparsity
- Hierarchical Models and Shrinkage
- Hidden Markov Models
- Nonlinear Non-Gaussian State Space Models

Algorithms
- Monte Carlo Method (von Neumann and Ulam, 1940s)
- Metropolis-Hastings (Metropolis, 1950s)
- Gibbs Sampling (Geman and Geman, Gelfand and Smith, 1980s)
- Sequential Particle Filtering
Probabilistic Reasoning

Bayesians only make Probability statements

- Bayesian Probability (Ramsey, 1926, de Finetti, 1931)
  1. Beta-Binomial Learning: Black Swans
  2. Elections: Nate Silver
  3. Baseball: Kenny Lofton and Derek Jeter

- Monte Carlo (von Neumann and Ulam, Metropolis, 1940s)
  Shrinkage Estimation (Lindley and Smith, Efron and Morris, 1970s)
Bayesian Inference

**Key Idea:** Explicit use of probability for summarizing uncertainty.

1. A **probability distribution** for data given parameters

   \[ f(y|\theta) \text{ Likelihood} \]

2. A **probability distribution** for unknown parameters

   \[ p(\theta) \text{ Prior} \]

3. Inference for unknowns conditional on observed data
   - Inverse probability (Bayes Theorem);
   - Formal decision making (Loss, Utility)
Posterior Inference

**Bayes theorem** to derive posterior distributions

\[
p(\theta | y) = \frac{p(y | \theta)p(\theta)}{p(y)}
\]

\[
p(y) = \int p(y | \theta)p(\theta)d\theta
\]

Allows you to make probability statements

- They can be very different from p-values!
- Hypothesis testing and Sequential problems
- Markov chain Monte Carlo (MCMC) and Filtering (PF)
Conjugate Priors

Example

Definition: Let \( \mathcal{F} \) denote the class of distributions \( f(y|\theta) \).
A class \( \Pi \) of prior distributions is conjugate for \( \mathcal{F} \) if the posterior
distribution is in the class \( \Pi \) for all \( f \in \mathcal{F}, \pi \in \Pi, y \in \mathcal{Y} \).

Example: Binomial/Beta:
Suppose that \( Y_1, \ldots, Y_n \sim Ber(p) \).
Let \( p \sim Beta(\alpha, \beta) \) where \( (\alpha, \beta) \) are known hyper-parameters.
The beta-family is very flexible
Prior mean \( E(p) = \frac{\alpha}{\alpha + \beta} \).
Bayes Learning: Beta-Binomial

How do I update my beliefs about a coin toss?

Likelihood for Bernoulli

\[
p (y|\theta) = \prod_{t=1}^{T} p (y_t|\theta) = \theta^{\sum_{t=1}^{T} y_t} (1 - \theta)^{T - \sum_{t=1}^{T} y_t}.
\]

Initial prior distribution \( \theta \sim \mathcal{B}(a, A) \) given by

\[
p (\theta|a, A) = \frac{\theta^{a-1} (1 - \theta)^{A-1}}{B(a, A)}
\]
Bayes Learning: Beta-Binomial

Updated posterior distribution is also Beta

\[ p(\theta|y) \sim \mathcal{B}(a_T, A_T) \quad \text{and} \quad a_T = a + \sum_{t=1}^{T} y_t, A_T = A + T - \sum_{t=1}^{T} y_t \]

The posterior mean and variance are

\[ E[\theta|y] = \frac{a_T}{a_T + A_T} \quad \text{and} \quad \text{var}[\theta|y] = \frac{a_T A_T}{(a_T + A_T)^2 (a_T + A_T + 1)} \]
Suppose you’re only see a sequence of White Swans, having never seen a Black Swan.

What’s the Probability of Black Swan event *sometime* in the future?

Suppose that after $T$ trials you have only seen successes $(y_1, \ldots, y_T) = (1, \ldots, 1)$. The next trial being a success has

$$p(y_{T+1} = 1|y_1, \ldots, y_T) = \frac{T + 1}{T + 2}$$

For large $T$ is almost certain. Here $a = A = 1$. 
Black Swans

Principle of Induction (Hume)

The probability of never seeing a Black Swan is given by

\[ p(y_{T+1} = 1, \ldots, y_{T+n} = 1 | y_1, \ldots, y_T) = \frac{T + 1}{T + n + 1} \to 0 \]

Black Swan will eventually happen – don’t be surprised when it actually happens.
Lindley’s Paradox

Often evidence which, for a Bayesian statistician, strikingly supports the null leads to rejection by standard classical procedures.

- Do Bayes and Classical always agree?
  - Bayes computes the probability of the null being true given the data \( p(H_0|D) \). That’s not the p-value. Why?

- Surely they agree asymptotically?

- How do we model the prior and compute likelihood ratios \( L(H_0|D) \) in the Bayesian framework?
Bayes $t$-ratio

Edwards, Lindman and Savage (1963)

Simple approximation for the likelihood ratio.

$$L(p_0) \approx \sqrt{2\pi \sqrt{n}} \exp \left( -\frac{1}{2} t^2 \right)$$

- **Key**: Bayes test will have the factor $\sqrt{n}$
  This will asymptotically favour the null.
- There is only a big problem when $2 < t < 4$ – **but** this is typically the most interesting case!
**Coin Tossing**

**Intuition:** Imagine a coin tossing experiment and you want to determine whether the coin is “fair” $H_0 : p = \frac{1}{2}$.

There are four experiments.

<table>
<thead>
<tr>
<th>Expt</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>50</td>
<td>100</td>
<td>400</td>
<td>10,000</td>
</tr>
<tr>
<td>r</td>
<td>32</td>
<td>60</td>
<td>220</td>
<td>5098</td>
</tr>
<tr>
<td>$L(p_0)$</td>
<td>0.81</td>
<td>1.09</td>
<td>2.17</td>
<td>11.68</td>
</tr>
</tbody>
</table>
Implications:

- Classical: In each case the $t$-ratio is approx 2. They we just $H_0$ (a fair coin) at the 5% level in each experiment.

- Bayes: $L(p_0)$ grows to infinity and so they is overwhelming evidence for $H_0$. Connelly shows that the Monday effect disappears when you compute the Bayes version.
Hierarchical Models

Let’s do some cool applications ...

- Bayes MoneyBall
  Batter-pitcher match-up:
  Kenny Lofton and Derek Jeter
- Bayes Elections
- SAT scores
Example: Baseball

Batter-pitcher match-up?

Prior information on overall ability of a player. Small sample size, pitcher variation.

Let $p_i$ denote Jeter’s ability. Observed number of hits $y_i$

$$ (y_i|p_i) \sim Bin(T_i, p_i) \text{ with } p_i \sim Be(\alpha, \beta) $$

where $T_i$ is the number of at-bats against pitcher $i$. A priori

$$ E(p_i) = \frac{\alpha}{(\alpha + \beta)} = \bar{p}_i. $$

The extra heterogeneity leads to a prior variance

$$ Var(p_i) = \bar{p}_i(1 - \bar{p}_i)\phi \text{ where } \phi = (\alpha + \beta + 1)^{-1}. $$
### Kenny Lofton versus individual pitchers.

<table>
<thead>
<tr>
<th>Pitcher</th>
<th>At-bats</th>
<th>Hits</th>
<th>ObsAvg</th>
</tr>
</thead>
<tbody>
<tr>
<td>J.C. Romero</td>
<td>9</td>
<td>6</td>
<td>.667</td>
</tr>
<tr>
<td>S. Lewis</td>
<td>5</td>
<td>3</td>
<td>.600</td>
</tr>
<tr>
<td>B. Tomko</td>
<td>20</td>
<td>11</td>
<td>.550</td>
</tr>
<tr>
<td>T. Hoffman</td>
<td>6</td>
<td>3</td>
<td>.500</td>
</tr>
<tr>
<td>K. Tapani</td>
<td>45</td>
<td>22</td>
<td>.489</td>
</tr>
<tr>
<td>A. Cook</td>
<td>9</td>
<td>4</td>
<td>.444</td>
</tr>
<tr>
<td>J. Abbott</td>
<td>34</td>
<td>14</td>
<td>.412</td>
</tr>
<tr>
<td>A.J. Burnett</td>
<td>15</td>
<td>6</td>
<td>.400</td>
</tr>
<tr>
<td>K. Rogers</td>
<td>43</td>
<td>17</td>
<td>.395</td>
</tr>
<tr>
<td>A. Harang</td>
<td>6</td>
<td>2</td>
<td>.333</td>
</tr>
<tr>
<td>K. Appier</td>
<td>49</td>
<td>15</td>
<td>.306</td>
</tr>
<tr>
<td>R. Clemens</td>
<td>62</td>
<td>14</td>
<td>.226</td>
</tr>
<tr>
<td>C. Zambrano</td>
<td>9</td>
<td>2</td>
<td>.222</td>
</tr>
<tr>
<td>N. Ryan</td>
<td>10</td>
<td>2</td>
<td>.200</td>
</tr>
<tr>
<td>E. Hanson</td>
<td>41</td>
<td>7</td>
<td>.171</td>
</tr>
<tr>
<td>E. Milton</td>
<td>19</td>
<td>1</td>
<td>.056</td>
</tr>
<tr>
<td>M. Prior</td>
<td>7</td>
<td>0</td>
<td>.000</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>7630</td>
<td>2283</td>
<td>.299</td>
</tr>
</tbody>
</table>
Kenny Lofton (career .299 average, and current .308 average for 2006 season) was facing the pitcher Milton (current record 1 for 19)

He was *rested* and replaced by a .275 hitter.

▶ Is putting in a weaker player really a better bet?

▶ *Over-reaction to bad luck?*

\[ P(\leq 1 \text{ hit in 19 attempts}|p = 0.3) = 0.01 \]

An unlikely 1-in-100 event.
Bayes solution: shrinkage. Borrow strength across pitchers

Bayes estimate: use the posterior mean

Lofton’s batting estimates that vary from .265 to .340. The lowest being against Milton.

.265 < .275

**Conclusion:** resting Lofton against Milton was justified!!
Bayes Batter-pitcher match-up

Here’s our model again ... 

- Small sample sizes and pitcher variation.
- Let $p_i$ denote Lofton’s ability. Observed number of hits $y_i$

\[
(y_i | p_i) \sim Bin(T_i, p_i) \quad \text{with} \quad p_i \sim Be(\alpha, \beta)
\]

where $T_i$ is the number of at-bats against pitcher $i$. 

Estimate $(\alpha, \beta)$
Example: Derek Jeter

## Table: Derek Jeter hierarchical model estimates

<table>
<thead>
<tr>
<th>Pitcher</th>
<th>At-bats</th>
<th>Hits</th>
<th>ObsAvg</th>
<th>EstAvg</th>
<th>95% Int</th>
</tr>
</thead>
<tbody>
<tr>
<td>R. Mendoza</td>
<td>6</td>
<td>5</td>
<td>.833</td>
<td>.322</td>
<td>(.282,.394)</td>
</tr>
<tr>
<td>H. Nomo</td>
<td>20</td>
<td>12</td>
<td>.600</td>
<td>.326</td>
<td>(.289,.407)</td>
</tr>
<tr>
<td>A.J. Burnett</td>
<td>5</td>
<td>3</td>
<td>.600</td>
<td>.320</td>
<td>(.275,.381)</td>
</tr>
<tr>
<td>E. Milton</td>
<td>28</td>
<td>14</td>
<td>.500</td>
<td>.324</td>
<td>(.291,.397)</td>
</tr>
<tr>
<td>D. Cone</td>
<td>8</td>
<td>4</td>
<td>.500</td>
<td>.320</td>
<td>(.218,.381)</td>
</tr>
<tr>
<td>R. Lopez</td>
<td>45</td>
<td>21</td>
<td>.467</td>
<td>.326</td>
<td>(.291,.401)</td>
</tr>
<tr>
<td>K. Escobar</td>
<td>39</td>
<td>16</td>
<td>.410</td>
<td>.322</td>
<td>(.281,.386)</td>
</tr>
<tr>
<td>J. Wettland</td>
<td>5</td>
<td>2</td>
<td>.400</td>
<td>.318</td>
<td>(.275,.375)</td>
</tr>
<tr>
<td>T. Wakefield</td>
<td>81</td>
<td>26</td>
<td>.321</td>
<td>.318</td>
<td>(.279,.364)</td>
</tr>
<tr>
<td>P. Martinez</td>
<td>83</td>
<td>21</td>
<td>.253</td>
<td>.312</td>
<td>(.254,.347)</td>
</tr>
<tr>
<td>K. Benson</td>
<td>8</td>
<td>2</td>
<td>.250</td>
<td>.317</td>
<td>(.264,.368)</td>
</tr>
<tr>
<td>T. Hudson</td>
<td>24</td>
<td>6</td>
<td>.250</td>
<td>.315</td>
<td>(.260,.362)</td>
</tr>
<tr>
<td>J. Smoltz</td>
<td>5</td>
<td>1</td>
<td>.200</td>
<td>.314</td>
<td>(.253,.355)</td>
</tr>
<tr>
<td>F. Garcia</td>
<td>25</td>
<td>5</td>
<td>.200</td>
<td>.314</td>
<td>(.253,.355)</td>
</tr>
<tr>
<td>B. Radke</td>
<td>41</td>
<td>8</td>
<td>.195</td>
<td>.311</td>
<td>(.247,.347)</td>
</tr>
<tr>
<td>D. Kolb</td>
<td>5</td>
<td>0</td>
<td>.000</td>
<td>.316</td>
<td>(.258,.363)</td>
</tr>
<tr>
<td>J. Julio</td>
<td>13</td>
<td>0</td>
<td>.000</td>
<td>.312</td>
<td>(.243,.350)</td>
</tr>
<tr>
<td>Total</td>
<td>6530</td>
<td>2061</td>
<td>.316</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Derek Jeter 2006 season versus individual pitchers.
Bayes Estimates

Stern stimates $\hat{\phi} = (\alpha + \beta + 1)^{-1} = 0.002$ for Jeter
 Doesn’t vary much across the population of pitchers.

The extremes are shrunk the most also matchups with the smallest sample sizes.

Jeter had a season .308 average.
Bayes estimates vary from .311 to .327–he’s very consistent.

If all players had a similar record then a constant batting average would make sense.
Predicting the Electoral Vote (EV)

- Multinomial-Dirichlet: $\hat{p}|p) \sim Multi(p), (p|\alpha) \sim Dir(\alpha)$

\[ p_{Obama} = (p_1, \ldots, p_{51}|\hat{p}) \sim Dir(\alpha + \hat{p}) \]

- Flat uninformative prior $\alpha \equiv 1$.

http://www.electoral-vote.com/evp2012/Pres/prespolls.csv
Bayes Elections: Nate Silver

Simulation

Calculate probabilities via simulation: \texttt{rdirichlet}

\[ p(p_j, O|\text{data}) \quad \text{and} \quad p(EV > 270|\text{data}) \]

The election vote prediction is given by the sum

\[ EV = \sum_{j=1}^{51} EV(j) \mathbb{E}(p_j|\text{data}) \]

where \( EV(j) \) are for individual states
Polling Data: electoral-vote.com

<table>
<thead>
<tr>
<th>State</th>
<th>Mitt.pct</th>
<th>Obama.pct</th>
<th>EV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alabama</td>
<td>58</td>
<td>36</td>
<td>9</td>
</tr>
<tr>
<td>Alaska</td>
<td>55</td>
<td>37</td>
<td>3</td>
</tr>
<tr>
<td>Arizona</td>
<td>50</td>
<td>46</td>
<td>10</td>
</tr>
<tr>
<td>Arkansas</td>
<td>51</td>
<td>44</td>
<td>6</td>
</tr>
<tr>
<td>California</td>
<td>33</td>
<td>55</td>
<td></td>
</tr>
<tr>
<td>Colorado</td>
<td>45</td>
<td>52</td>
<td>9</td>
</tr>
<tr>
<td>Connecticut</td>
<td>31</td>
<td>56</td>
<td>7</td>
</tr>
<tr>
<td>Delaware</td>
<td>38</td>
<td>56</td>
<td>3</td>
</tr>
<tr>
<td>D.C.</td>
<td>13</td>
<td>82</td>
<td>3</td>
</tr>
<tr>
<td>Florida</td>
<td>46</td>
<td>50</td>
<td>27</td>
</tr>
<tr>
<td>Georgia</td>
<td>52</td>
<td>47</td>
<td>15</td>
</tr>
<tr>
<td>Hawaii</td>
<td>32</td>
<td>63</td>
<td>4</td>
</tr>
<tr>
<td>Idaho</td>
<td>68</td>
<td>26</td>
<td>4</td>
</tr>
<tr>
<td>Illinois</td>
<td>35</td>
<td>59</td>
<td>21</td>
</tr>
<tr>
<td>Indiana</td>
<td>48</td>
<td>48</td>
<td>11</td>
</tr>
<tr>
<td>Iowa</td>
<td>37</td>
<td>54</td>
<td>7</td>
</tr>
<tr>
<td>Kansas</td>
<td>63</td>
<td>31</td>
<td>6</td>
</tr>
<tr>
<td>Kentucky</td>
<td>51</td>
<td>42</td>
<td>8</td>
</tr>
<tr>
<td>Louisiana</td>
<td>50</td>
<td>43</td>
<td>9</td>
</tr>
<tr>
<td>Maine</td>
<td>35</td>
<td>56</td>
<td>4</td>
</tr>
<tr>
<td>Maryland</td>
<td>39</td>
<td>54</td>
<td>10</td>
</tr>
<tr>
<td>Massachusetts</td>
<td>34</td>
<td>53</td>
<td>12</td>
</tr>
<tr>
<td>Michigan</td>
<td>37</td>
<td>53</td>
<td>17</td>
</tr>
<tr>
<td>Minnesota</td>
<td>42</td>
<td>53</td>
<td>10</td>
</tr>
<tr>
<td>Mississippi</td>
<td>46</td>
<td>33</td>
<td>6</td>
</tr>
<tr>
<td>Missouri</td>
<td>48</td>
<td>48</td>
<td>11</td>
</tr>
<tr>
<td>Montana</td>
<td>49</td>
<td>46</td>
<td>3</td>
</tr>
<tr>
<td>Nebraska</td>
<td>60</td>
<td>34</td>
<td>5</td>
</tr>
<tr>
<td>Nevada</td>
<td>43</td>
<td>47</td>
<td>5</td>
</tr>
<tr>
<td>New.Hampshire</td>
<td>42</td>
<td>53</td>
<td>4</td>
</tr>
<tr>
<td>New.Jersey</td>
<td>34</td>
<td>55</td>
<td>15</td>
</tr>
<tr>
<td>New.Mexico</td>
<td>43</td>
<td>51</td>
<td>5</td>
</tr>
<tr>
<td>New.York</td>
<td>31</td>
<td>62</td>
<td>31</td>
</tr>
<tr>
<td>North.Carolina</td>
<td>49</td>
<td>46</td>
<td>15</td>
</tr>
<tr>
<td>North.Dakota</td>
<td>43</td>
<td>45</td>
<td>3</td>
</tr>
<tr>
<td>Ohio</td>
<td>47</td>
<td>45</td>
<td>20</td>
</tr>
<tr>
<td>Oklahoma</td>
<td>61</td>
<td>34</td>
<td>7</td>
</tr>
<tr>
<td>Oregon</td>
<td>34</td>
<td>48</td>
<td>7</td>
</tr>
<tr>
<td>Pennsylvania</td>
<td>46</td>
<td>52</td>
<td>21</td>
</tr>
<tr>
<td>Rhode.Island</td>
<td>31</td>
<td>45</td>
<td>4</td>
</tr>
<tr>
<td>South.Carolina</td>
<td>59</td>
<td>39</td>
<td>8</td>
</tr>
<tr>
<td>South.Dakota</td>
<td>48</td>
<td>41</td>
<td>3</td>
</tr>
<tr>
<td>Tennessee</td>
<td>55</td>
<td>39</td>
<td>11</td>
</tr>
<tr>
<td>Texas</td>
<td>57</td>
<td>38</td>
<td>34</td>
</tr>
<tr>
<td>Utah</td>
<td>55</td>
<td>32</td>
<td>5</td>
</tr>
<tr>
<td>Vermont</td>
<td>36</td>
<td>57</td>
<td>3</td>
</tr>
<tr>
<td>Virginia</td>
<td>44</td>
<td>47</td>
<td>13</td>
</tr>
<tr>
<td>Washington</td>
<td>39</td>
<td>51</td>
<td>11</td>
</tr>
<tr>
<td>West.Virginia</td>
<td>53</td>
<td>44</td>
<td>5</td>
</tr>
<tr>
<td>Wisconsin</td>
<td>42</td>
<td>53</td>
<td>10</td>
</tr>
<tr>
<td>Wyoming</td>
<td>58</td>
<td>32</td>
<td>3</td>
</tr>
</tbody>
</table>
Figure: Election 2008 Prediction. Obama 370
**Figure:** Election 2012 Prediction. Obama 332.
**SAT Scores**

SAT (200 – 800): 8 high schools and estimate effects.

<table>
<thead>
<tr>
<th>School</th>
<th>Estimated $y_j$</th>
<th>St. Error $\sigma_j$</th>
<th>Average Treatment $\theta_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>28</td>
<td>15</td>
<td>?</td>
</tr>
<tr>
<td>B</td>
<td>8</td>
<td>10</td>
<td>?</td>
</tr>
<tr>
<td>C</td>
<td>-3</td>
<td>16</td>
<td>?</td>
</tr>
<tr>
<td>D</td>
<td>7</td>
<td>11</td>
<td>?</td>
</tr>
<tr>
<td>E</td>
<td>-1</td>
<td>9</td>
<td>?</td>
</tr>
<tr>
<td>F</td>
<td>1</td>
<td>11</td>
<td>?</td>
</tr>
<tr>
<td>G</td>
<td>18</td>
<td>10</td>
<td>?</td>
</tr>
<tr>
<td>H</td>
<td>12</td>
<td>18</td>
<td>?</td>
</tr>
</tbody>
</table>

$\theta_j$ average effects of coaching programs

$y_j$ estimated treatment effects, for school $j$, standard error $\sigma_j$. 
Estimates

Two programs appear to work (improvements of 18 and 28)

- Large standard errors. Overlapping Confidence Intervals?
- Classical hypothesis test fails to reject the hypothesis that the $\theta_j$'s are equal.
- Pooled estimate has standard error of 4.2 with

$$
\hat{\theta} = \frac{\sum_j (y_j / \sigma_j^2)}{\sum_j (1 / \sigma_j^2)} = 7.9
$$

- Neither separate or pooled seems sensible.

Bayesian shrinkage!
Bayesian Model

Hierarchical Model ($\sigma_j^2$ known) is given by

$$\bar{y}_j | \theta_j \sim N(\theta_j, \sigma_j^2)$$

Unequal variances–differential shrinkage.

- Prior Distribution: $\theta_j \sim N(\mu, \tau^2)$ for $1 \leq j \leq 8$. Traditional random effects model.
  Exchangeable prior for the treatment effects.
  As $\tau \to 0$ (complete pooling) and as $\tau \to \infty$ (separate estimates).

- Hyper-prior Distribution: $p(\mu, \tau^2) \propto 1/\tau$. The posterior $p(\mu, \tau^2|y)$ can be used to “estimate” $(\mu, \tau^2)$. 

Joint Posterior Distribution $y = (y_1, \ldots, y_J)$

$$p(\theta, \mu, \tau | y) \propto p(y | \theta)p(\theta | \mu, \tau)p(\mu, \tau)$$

$$\propto p(\mu, \tau^2) \prod_{i=1}^{8} N(\theta_i | \mu, \tau^2) \prod_{j=1}^{8} N(y_j | \theta_j)$$

$$\propto \tau^{-9} \exp \left( -\frac{1}{2} \sum_j \frac{1}{\tau^2} (\theta_j - \mu)^2 - \frac{1}{2} \sum_j \frac{1}{\sigma_j^2} (y_j - \theta_j)^2 \right)$$

MCMC!
### Posterior Inference

Report posterior quantiles

<table>
<thead>
<tr>
<th>School</th>
<th>2.5%</th>
<th>25%</th>
<th>50%</th>
<th>75%</th>
<th>97.5%</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>-2</td>
<td>6</td>
<td>10</td>
<td>16</td>
<td>32</td>
</tr>
<tr>
<td>B</td>
<td>-5</td>
<td>4</td>
<td>8</td>
<td>12</td>
<td>20</td>
</tr>
<tr>
<td>C</td>
<td>-12</td>
<td>3</td>
<td>7</td>
<td>11</td>
<td>22</td>
</tr>
<tr>
<td>D</td>
<td>-6</td>
<td>4</td>
<td>8</td>
<td>12</td>
<td>21</td>
</tr>
<tr>
<td>E</td>
<td>-10</td>
<td>2</td>
<td>6</td>
<td>10</td>
<td>19</td>
</tr>
<tr>
<td>F</td>
<td>-9</td>
<td>2</td>
<td>6</td>
<td>10</td>
<td>19</td>
</tr>
<tr>
<td>G</td>
<td>-1</td>
<td>6</td>
<td>10</td>
<td>15</td>
<td>27</td>
</tr>
<tr>
<td>H</td>
<td>-7</td>
<td>4</td>
<td>8</td>
<td>13</td>
<td>23</td>
</tr>
<tr>
<td>µ</td>
<td>-2</td>
<td>5</td>
<td>8</td>
<td>11</td>
<td>18</td>
</tr>
<tr>
<td>τ</td>
<td>0.3</td>
<td>2.3</td>
<td>5.1</td>
<td>8.8</td>
<td>21</td>
</tr>
</tbody>
</table>

Schools A and G are similar!
Bayesian Shrinkage

Bayesian shrinkage provides a way of modeling complex datasets.

1. Coin Tossing: Lindley’s Paradox
2. Baseball batting averages: Stein’s Paradox
3. Toxoplasmosis
4. SAT scores
5. Clinical Trials
Binomial-Beta

\[ p(p|\bar{y}) \] is the **posterior distribution** for \( p \)

\( \bar{y} \) is a sufficient statistic.

▶ **Bayes theorem gives**

\[
p(p|y) \propto f(y|p)p(p|\alpha, \beta)
\]

\[
\propto p^{\sum y_i}(1-p)^{n-\sum y_i} \cdot p^{\alpha-1}(1-p)^{\beta-1}
\]

\[
\propto p^{\alpha+\sum y_i-1}(1-p)^{n-\sum y_i+\beta-1}
\]

\[
\sim Beta(\alpha + \sum y_i, \beta + n - \sum y_i)
\]

▶ The posterior mean is a shrinkage estimator

Combination of sample mean \( \bar{y} \) and prior mean \( E(p) \)

\[
E(p|y) = \frac{\alpha + \sum_{i=1}^{n} y_i}{\alpha + \beta + n} = \frac{n}{n + \alpha + \beta} \bar{y} + \frac{\alpha + \beta}{\alpha + \beta + n} \frac{\alpha}{\alpha + \beta}
\]
Example

Poisson/Gamma: Suppose that $Y_1, \ldots, Y_n \sim Poi(\lambda)$.

Let $\lambda \sim Gamma(\alpha, \beta)$

$(\alpha, \beta)$ are known hyper-parameters.

- The posterior distribution is

$$p(\lambda|y) \propto \exp(-n\lambda)\lambda^{\sum y_i} \lambda^{\alpha-1} \exp(-\beta\lambda)$$

$$\sim Gamma(\alpha + \sum y_i, n + \beta)$$
Example: Clinical Trials

Novick and Grizzle: Bayesian Analysis of Clinical Trials

Four treatments for duodenal ulcers.
Doctors assess the state of the patient.

Sequential data
($\alpha$-spending function, can only look at prespecified times).

<table>
<thead>
<tr>
<th>Treat</th>
<th>Excellent</th>
<th>Fair</th>
<th>Death</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>76</td>
<td>17</td>
<td>7</td>
</tr>
<tr>
<td>B</td>
<td>89</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>86</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>88</td>
<td>9</td>
<td>3</td>
</tr>
</tbody>
</table>

Conclusion: Cannot reject at the 5% level

Conjugate binomial/beta model+sensitivity analysis.
Binomial-Beta

Let $p_i$ be the death rate proportion under treatment $i$.

- To compare treatment $A$ to $B$ directly compute $P(p_1 > p_2 | D)$.
- Prior $beta(\alpha, \beta)$ with prior mean $E(p) = \frac{\alpha}{\alpha + \beta}$.
  Posterior $beta(\alpha + \sum x_i, \beta + n - \sum x_i)$

- For $A$, $beta(1,1) \rightarrow beta(8,94)$
  For $B$, $beta(1,1) \rightarrow beta(2,100)$
- Inference: $P(p_1 > p_2 | D) \approx 0.98$
Sensitivity Analysis

Important to do a sensitivity analysis.

<table>
<thead>
<tr>
<th>Treat</th>
<th>Excellent</th>
<th>Fair</th>
<th>Death</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>76</td>
<td>17</td>
<td>7</td>
</tr>
<tr>
<td>B</td>
<td>89</td>
<td>10</td>
<td>1</td>
</tr>
<tr>
<td>C</td>
<td>86</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>88</td>
<td>9</td>
<td>3</td>
</tr>
</tbody>
</table>

Poisson-Gamma, prior $\Gamma(m, z)$ and $\lambda_i$ be the expected death rate.

Compute $P\left(\frac{\lambda_1}{\lambda_2} > c|D\right)$

<table>
<thead>
<tr>
<th>Prob</th>
<th>$(0, 0)$</th>
<th>$(100, 2)$</th>
<th>$(200, 5)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P\left(\frac{\lambda_1}{\lambda_2} &gt; 1.3</td>
<td>D\right)$</td>
<td>0.95</td>
<td>0.88</td>
</tr>
<tr>
<td>$P\left(\frac{\lambda_1}{\lambda_2} &gt; 1.6</td>
<td>D\right)$</td>
<td>0.91</td>
<td>0.80</td>
</tr>
</tbody>
</table>
Normal-Normal Model

Using Bayes rule we get

\[ p(\mu|y) \propto p(y|\mu)p(\mu) \]

Posterior is given by

\[ p(\mu|y) \propto \exp \left( -\frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \mu)^2 - \frac{1}{2\tau^2} (\mu - \mu_0)^2 \right) \]

Hence \( \mu|y \sim N(\hat{\mu}_B, V_\mu) \) where

\[ \hat{\mu}_B = \frac{n/\sigma^2}{n/\sigma^2 + 1/\tau^2} \bar{y} + \frac{1/\tau^2}{n/\sigma^2 + 1/\tau^2} \mu_0 \quad \text{and} \quad V_\mu^{-1} = \frac{n}{\sigma^2} + \frac{1}{\tau^2} \]

A shrinkage estimator.
Bayes Learning: Update our beliefs in light of new information

- In the 2014-2015 season.
  The Bears suffered back-to-back 50-points defeats.
  Partiots-Bears 51 – 23
  Packers-Bears 55 – 14

- Their next game was at home against the Minnesota Vikings.
  Current line against the Vikings was −3.5 points.
  Slightly over a field goal

What’s the Bayes approach to learning the line?
Hierarchical Model

Hierarchical model for the current average win/lose this year

\[
\bar{y} | \theta \sim N \left( \theta, \frac{\sigma^2}{n} \right) \sim N \left( \theta, \frac{18.34^2}{9} \right)
\]

\[
\theta \sim N(0, \tau^2)
\]

Here \( n = 9 \) games so far. With \( s = 18.34 \) points

Pre-season prior mean \( \mu_0 = 0 \), standard deviation \( \tau = 4 \).

Record so-far. Data \( \bar{y} = -9.22 \).
Chicago Bears

Bayes Shrinkage estimator

\[ \mathbb{E} (\theta | \bar{y}, \tau) = \frac{\tau^2}{\tau^2 + \frac{\sigma^2}{n}} \bar{y} \]

The Shrinkage factor is 0.3!!

That’s quite a bit of shrinkage. Why?

- Our updated estimator is

\[ \mathbb{E} (\theta | \bar{y}, \tau) = -2.75 > -0.35 \]

where current line is \(-3.5\).

- Based on our hierarchical model this is an over-reaction. One point change on the line is about 3% on a probability scale. Alternatively, calculate a market-based \(\tau\) given line \(= -3.5\).
Chicago Bears

Last two defeats were 50 points scored by opponent (2014-15)

```r
bears<-c(-3,8,8,-21,-7,14,-13,-28,-41)
> mean(bears)
[1] -9.222222
> sd(bears)
[1] 18.34242
> tau=4

> sig2=sd(bears)*sd(bears)/9
> tau^2/(sig2+tau^2)
[1] 0.2997225
> 0.29997*-9.22
[1] -2.765723
> pnorm(-2.76/18)
[1] 0.4390677
```

Home advantage is worth 3 points. Vikings an average record.

Result: Bears 21, Vikings 13
Shrinkage Estimation

**Stein paradox:** possible to make a uniform improvement on the MLE in terms of MSE.

- Mistrust of the statistical interpretation of Stein’s result.
  In particular, the loss function.
- Difficulties in adapting the procedure to special cases
- Long familiarity with good properties for the MLE

Any gains from a “complicated” procedure could not be worth the extra trouble (Tukey, savings not more than 10% in practice)
# Baseball Batting Averages

Data: 18 major-league players after 45 at bats (1970 season)

| Player        | $\bar{y}_i$ | $E(p_j | D)$ | average season |
|---------------|-------------|-------------|----------------|
| Clemente      | 0.400       | 0.290       | 0.346          |
| Robinson      | 0.378       | 0.286       | 0.298          |
| Howard        | 0.356       | 0.281       | 0.276          |
| Johnstone     | 0.333       | 0.277       | 0.222          |
| Berry         | 0.311       | 0.273       | 0.273          |
| Spencer       | 0.311       | 0.273       | 0.270          |
| Kessinger     | 0.311       | 0.268       | 0.263          |
| Alvarado      | 0.267       | 0.264       | 0.210          |
| Santo         | 0.244       | 0.259       | 0.269          |
| Swoboda       | 0.244       | 0.259       | 0.230          |
| Unser         | 0.222       | 0.254       | 0.264          |
| Williams      | 0.222       | 0.254       | 0.256          |
| Scott         | 0.222       | 0.254       | 0.303          |
| Petrocelli    | 0.222       | 0.254       | 0.264          |
| Rodriguez     | 0.222       | 0.254       | 0.226          |
| Campanens     | 0.200       | 0.259       | 0.285          |
| Munson        | 0.178       | 0.244       | 0.316          |
| Alvis         | 0.156       | 0.239       | 0.200          |
Baseball Data

First Shrinkage Estimator: Efron and Morris

Figure: Baseball Shrinkage
Shrinkage

Let $\theta_i$ denote the end of season average

- Lindley: shrink to the overall grand mean

$$c = 1 - \frac{(k - 3)\sigma^2}{\sum(\bar{y}_i - \bar{y})^2}$$

where $\bar{y}$ is the overall grand mean and

$$\hat{\theta} = c\bar{y}_i + (1 - c)\bar{y}$$

- Baseball data: $c = 0.212$ and $\bar{y} = 0.265$. Compute $\sum(\hat{\theta}_i - \bar{y}_i^{obs})^2$ and see which is lower:

$$MLE = 0.077 \quad STEIN = 0.022$$

That’s a factor of 3.5 times better!
'Clemente' batting averages over 1970 season:
.400 after 45 at bats; .346 for remainder; .352 overall
Baseball Paradoxes

**Shrinkage on Clemente** too severe:

\[ z_{CL} = 0.265 + 0.212(0.400 - 0.265) = 0.294. \]

The 0.212 seems a little severe

- Limited translation rules, maximum shrinkage eg. 80%
- Not enough shrinkage eg O’Connor (\( y = 1, n = 2 \)).
  \[ z_{OC} = 0.265 + 0.212(0.5 - 0.265) = 0.421. \]
  Still better than Ted Williams 0.406 in 1941.

- Foreign car sales (\( k = 19 \)) will further improve MSE performance! It will change the shrinkage factors.

- Clearly an improvement over the Stein estimator is

\[ \hat{\theta}_{S+} = \max \left( \left( 1 - \frac{k - 2}{\sum \bar{Y}_i^2} \right), 0 \right) \bar{Y}_i \]
Baseball Prior

Include extra prior knowledge

Empirical distribution of all major league players

\[ \theta_i \sim N(0.270, 0.015) \]

The 0.270 provides another origin to shrink to and the prior variance 0.015 would give a different shrinkage factor.

To fully understand maybe we should build a probabilistic model and see what the posterior mean is as our estimator for the unknown parameters.
Model $Y_i|\theta_i \sim N(\theta_i, D_i)$ where $\theta_i \sim N(\theta_0, A) \sim N(0.270, 0.015)$.

- The $D_i$ can be different – unequal variances
- Bayes posterior means are given by

$$E(\theta_i|Y) = (1 - B_i)Y_i$$

where $B_i = \frac{D_i}{D_i + \hat{A}}$

where $\hat{A}$ is estimated from the data, see Efron and Morris (1975).

- Different shrinkage factors as different variances $D_i$.
  $D_i \propto n_i^{-1}$ and so smaller sample sizes are shrunk more.
  Makes sense.
Example: Toxoplasmosis Data

Disease of Blood that is endemic in tropical regions.

Data: 5000 people in El Salvador (varying sample sizes) from 36 cities.

- Estimate “true” prevalences $\theta_i$ for $1 \leq i \leq 36$
- Allocation of Resources: should we spend funds on the city with the highest observed occurrence of the disease? Same shrinkage factors?
- Shrinkage Procedure (Efron and Morris, p315)

$$z_i = c_i y_i$$

where $y_i$ are the observed relative rates (normalized so $\bar{y} = 0$). The smaller sample sizes will get shrunk more. The most gentle are in the range $0.6 \rightarrow 0.9$ but some are $0.1 \rightarrow 0.3$. 
de Finetti and Markowitz: Mean-variance portfolio shrinkage: $\frac{1}{\gamma} \Sigma^{-1} \mu$

Different shrinkage factors for different history lengths.

Portfolio Allocation in the SP500 index

Entry/exit; splits; spin-offs etc. For example, 73 replacements to the SP500 index in period 1/1/94 to 12/31/96.

**Advantage:** $E(\alpha|D_t) = 0.39$, that is 39 bps per month which on an annual basis is $\alpha = 468$ bps.

The posterior mean for $\beta$ is $p(\beta|D_t) = 0.745$

$\bar{x}_M = 12.25\%$ and $\bar{x}_{PT} = 14.05\%$. 
## SP Composition

<table>
<thead>
<tr>
<th>Date</th>
<th>Symbol</th>
<th>6/96</th>
<th>12/89</th>
<th>12/79</th>
<th>12/69</th>
</tr>
</thead>
<tbody>
<tr>
<td>General Electric</td>
<td>GE</td>
<td>2.800</td>
<td>2.485</td>
<td>1.640</td>
<td>1.569</td>
</tr>
<tr>
<td>Coca Cola</td>
<td>KO</td>
<td>2.342</td>
<td>1.126</td>
<td>0.606</td>
<td>1.051</td>
</tr>
<tr>
<td>Exxon</td>
<td>XON</td>
<td>2.142</td>
<td>2.672</td>
<td>3.439</td>
<td>2.957</td>
</tr>
<tr>
<td>ATT</td>
<td>T</td>
<td>2.030</td>
<td>2.090</td>
<td>5.197</td>
<td>5.948</td>
</tr>
<tr>
<td>Philip Morris</td>
<td>MO</td>
<td>1.678</td>
<td>1.649</td>
<td>0.637</td>
<td>*****</td>
</tr>
<tr>
<td>Royal Dutch</td>
<td>RD</td>
<td>1.636</td>
<td>1.774</td>
<td>1.191</td>
<td>*****</td>
</tr>
<tr>
<td>Merck</td>
<td>MRK</td>
<td>1.615</td>
<td>1.308</td>
<td>0.773</td>
<td>0.906</td>
</tr>
<tr>
<td>Microsoft</td>
<td>MSFT</td>
<td>1.436</td>
<td></td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Johnson/Johnson</td>
<td>JNJ</td>
<td>1.320</td>
<td>0.845</td>
<td>0.689</td>
<td>*****</td>
</tr>
<tr>
<td>Intel</td>
<td>INTC</td>
<td>1.262</td>
<td></td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Procter and Gamble</td>
<td>PG</td>
<td>1.228</td>
<td>1.040</td>
<td>0.871</td>
<td>0.993</td>
</tr>
<tr>
<td>Walmart</td>
<td>WMT</td>
<td>1.208</td>
<td>1.084</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>IBM</td>
<td>IBM</td>
<td>1.181</td>
<td>2.327</td>
<td>5.341</td>
<td>9.231</td>
</tr>
<tr>
<td>Hewlett Packard</td>
<td>HWP</td>
<td>1.105</td>
<td>0.477</td>
<td>0.497</td>
<td>*****</td>
</tr>
<tr>
<td>Pepsi</td>
<td>PEP</td>
<td>1.061</td>
<td>0.719</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Pfizer</td>
<td>PFE</td>
<td>0.918</td>
<td>0.491</td>
<td>0.408</td>
<td>0.486</td>
</tr>
<tr>
<td>Dupont</td>
<td>DD</td>
<td>0.910</td>
<td>1.229</td>
<td>0.837</td>
<td>1.101</td>
</tr>
<tr>
<td>AIG</td>
<td>AIG</td>
<td>0.910</td>
<td>0.723</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Mobil</td>
<td>MOB</td>
<td>0.906</td>
<td>1.093</td>
<td>1.659</td>
<td>1.040</td>
</tr>
<tr>
<td>Bristol Myers Squibb</td>
<td>BMY</td>
<td>0.878</td>
<td>1.247</td>
<td>*****</td>
<td>0.484</td>
</tr>
<tr>
<td>GTE</td>
<td>GTE</td>
<td>0.849</td>
<td>0.975</td>
<td>0.593</td>
<td>0.705</td>
</tr>
<tr>
<td>General Motors</td>
<td>GM</td>
<td>0.848</td>
<td>1.086</td>
<td>2.079</td>
<td>4.399</td>
</tr>
<tr>
<td>Disney</td>
<td>DIS</td>
<td>0.839</td>
<td>0.644</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Citicorp</td>
<td>CCI</td>
<td>0.831</td>
<td>0.400</td>
<td>0.418</td>
<td>*****</td>
</tr>
<tr>
<td>BellSouth</td>
<td>BLS</td>
<td>0.822</td>
<td>1.190</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Motorola</td>
<td>MOT</td>
<td>0.804</td>
<td>*****</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Company</td>
<td>Ticker</td>
<td>Year 1</td>
<td>Year 2</td>
<td>Year 3</td>
<td>Year 4</td>
</tr>
<tr>
<td>-----------------------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
<td>--------</td>
</tr>
<tr>
<td>Ford</td>
<td>F</td>
<td>0.798</td>
<td>0.883</td>
<td>0.485</td>
<td>0.640</td>
</tr>
<tr>
<td>Chervon</td>
<td>CHV</td>
<td>0.794</td>
<td>0.990</td>
<td>1.370</td>
<td>0.966</td>
</tr>
<tr>
<td>Amoco</td>
<td>AN</td>
<td>0.733</td>
<td>1.198</td>
<td>1.673</td>
<td>0.758</td>
</tr>
<tr>
<td>Eli Lilly</td>
<td>LLY</td>
<td>0.720</td>
<td>0.814</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Abbott Labs</td>
<td>ABT</td>
<td>0.690</td>
<td>0.654</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>AmerHome Products</td>
<td>AHP</td>
<td>0.686</td>
<td>0.716</td>
<td>0.606</td>
<td>0.793</td>
</tr>
<tr>
<td>FedNatlMortgage</td>
<td>FNM</td>
<td>0.686</td>
<td>*****</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>McDonald’s</td>
<td>MCD</td>
<td>0.686</td>
<td>0.545</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Ameritech</td>
<td>AIT</td>
<td>0.639</td>
<td>0.782</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Cisco Systems</td>
<td>CSCO</td>
<td>0.633</td>
<td>*****</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>CMB</td>
<td>CMB</td>
<td>0.621</td>
<td>*****</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>SBC</td>
<td>SBC</td>
<td>0.612</td>
<td>0.819</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Boeing</td>
<td>BA</td>
<td>0.598</td>
<td>0.584</td>
<td>0.462</td>
<td>*****</td>
</tr>
<tr>
<td>MMM</td>
<td>MMM</td>
<td>0.581</td>
<td>0.762</td>
<td>0.838</td>
<td>1.331</td>
</tr>
<tr>
<td>BankAmerica</td>
<td>BAC</td>
<td>0.560</td>
<td>*****</td>
<td>0.577</td>
<td>*****</td>
</tr>
<tr>
<td>Bell Atlantic</td>
<td>BEL</td>
<td>0.556</td>
<td>0.946</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Gillette</td>
<td>G</td>
<td>0.535</td>
<td>*****</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Kodak</td>
<td>EK</td>
<td>0.524</td>
<td>0.570</td>
<td>1.106</td>
<td>*****</td>
</tr>
<tr>
<td>Chrysler</td>
<td>C</td>
<td>0.507</td>
<td>*****</td>
<td>*****</td>
<td>0.367</td>
</tr>
<tr>
<td>Home Depot</td>
<td>HD</td>
<td>0.497</td>
<td>*****</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Colgate</td>
<td>COL</td>
<td>0.489</td>
<td>0.499</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Wells Fargo</td>
<td>WFC</td>
<td>0.478</td>
<td>*****</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Nations Bank</td>
<td>NB</td>
<td>0.453</td>
<td>*****</td>
<td>*****</td>
<td>*****</td>
</tr>
<tr>
<td>Amer Express</td>
<td>AXP</td>
<td>0.450</td>
<td>0.621</td>
<td>*****</td>
<td>*****</td>
</tr>
</tbody>
</table>
Keynes versus Buffett

CAPM

keynes = 15.08 + 1.83 market
buffett = 18.06 + 0.486 market

<table>
<thead>
<tr>
<th>Year</th>
<th>Keynes</th>
<th>Market</th>
</tr>
</thead>
<tbody>
<tr>
<td>1928</td>
<td>-3.4</td>
<td>7.9</td>
</tr>
<tr>
<td>1929</td>
<td>0.8</td>
<td>6.6</td>
</tr>
<tr>
<td>1930</td>
<td>-32.4</td>
<td>-20.3</td>
</tr>
<tr>
<td>1931</td>
<td>-24.6</td>
<td>-25.0</td>
</tr>
<tr>
<td>1932</td>
<td>44.8</td>
<td>-5.8</td>
</tr>
<tr>
<td>1933</td>
<td>35.1</td>
<td>21.5</td>
</tr>
<tr>
<td>1934</td>
<td>33.1</td>
<td>-0.7</td>
</tr>
<tr>
<td>1935</td>
<td>44.3</td>
<td>5.3</td>
</tr>
<tr>
<td>1936</td>
<td>56.0</td>
<td>10.2</td>
</tr>
<tr>
<td>1937</td>
<td>8.5</td>
<td>-0.5</td>
</tr>
<tr>
<td>1938</td>
<td>-40.1</td>
<td>-16.1</td>
</tr>
<tr>
<td>1939</td>
<td>12.9</td>
<td>-7.2</td>
</tr>
<tr>
<td>1940</td>
<td>-15.6</td>
<td>-12.9</td>
</tr>
<tr>
<td>1941</td>
<td>33.5</td>
<td>12.5</td>
</tr>
<tr>
<td>1942</td>
<td>-0.9</td>
<td>0.8</td>
</tr>
<tr>
<td>1943</td>
<td>53.9</td>
<td>15.6</td>
</tr>
<tr>
<td>1944</td>
<td>14.5</td>
<td>5.4</td>
</tr>
<tr>
<td>1945</td>
<td>14.6</td>
<td>0.8</td>
</tr>
</tbody>
</table>

Keynes vs Cash

King’s College Cambridge
Bayes, AI and Deep Learning: 41913

Section 2: AI
Week 4: Reinforcement and Q-Learning

Nick Polson

http://faculty.chicagobooth.edu/nicholas.polson/teaching/41913/
Q-Learning and Deal-No Deal

Rule of Thumb: Continue as long as there are two large prizes left.

- Continuation value is large.
  For example, with three prizes and two large ones. Risk averse people will naively choose deal, when if they incorporated the continuation value they would choose no deal.

- Suzanne and Frank’s choices on Dutch version of the show

See Korsos and Polson (2014)
## Susanne’s Choices

<table>
<thead>
<tr>
<th>Prize</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>€0.01</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€0.20</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€0.50</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€20</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€50</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€100</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€200</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€300</td>
<td></td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€400</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€500</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€1,000</td>
<td></td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€2,500</td>
<td></td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€5,000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€7,500</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€10,000</td>
<td></td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€12,500</td>
<td></td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€15,000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€20,000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€25,000</td>
<td></td>
<td></td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€50,000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€100,000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€150,000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€250,000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Average €**

|       | 32,094 | 21,431 | 26,491 | 34,825 | 46,417 | 50,700 | 62,750 | 83,667 | 125,000 |

**Offer €**

|       | 3,400  | 4,350  | 10,000 | 15,600 | 25,000 | 31,400 | 46,000 | 75,300 | 125,000 |

**Offer %**

|       | 11%    | 20%    | 38%    | 45%    | 54%    | 62%    | 73%    | 90%    | 100%    |

**Decision**

|       | No Deal | No Deal | No Deal | No Deal | No Deal | No Deal | No Deal | No Deal | No Deal |

<p>| | | | | | | | | | |
|       |        |        |        |        |        |        |        |        |        |</p>
<table>
<thead>
<tr>
<th>Prize</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>€0.01</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€0.20</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€0.50</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>€1</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€10</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>€20</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>€100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€500</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€1,000</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€2,500</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€5,000</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€7,500</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€10,000</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>€25,000</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€50,000</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€75,000</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€100,000</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€200,000</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€300,000</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€400,000</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€500,000</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€1,000,000</td>
<td>×</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€2,500,000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>€5,000,000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Average €</td>
<td>383,427</td>
<td>64,502</td>
<td>85,230</td>
<td>95,004</td>
<td>85,005</td>
<td>102,006</td>
<td>2,508</td>
<td>3,343</td>
<td>5,005</td>
</tr>
<tr>
<td>Offer €</td>
<td>17,000</td>
<td>8,000</td>
<td>23,000</td>
<td>44,000</td>
<td>52,000</td>
<td>75,000</td>
<td>2,400</td>
<td>3,500</td>
<td>6,000</td>
</tr>
<tr>
<td>Offer %</td>
<td>4%</td>
<td>12%</td>
<td>27%</td>
<td>46%</td>
<td>61%</td>
<td>74%</td>
<td>96%</td>
<td>105%</td>
<td>120%</td>
</tr>
<tr>
<td>Decision</td>
<td>No Deal</td>
<td>No Deal</td>
<td>No Deal</td>
<td>No Deal</td>
<td>No Deal</td>
<td>No Deal</td>
<td>No Deal</td>
<td>No Deal</td>
<td>No Deal</td>
</tr>
</tbody>
</table>
There’s a matrix of $Q$-values that solves the problem.

- Let $s$ denote the current state of the system and $a$ an action.
- The $Q$-value, $Q_t(s, a)$, is the value of using action $a$ today and then proceeding optimally in the future. We use $a = 1$ to mean no deal and $a = 0$ means deal.
- The Bellman equation for $Q$-values becomes

$$Q_t(s, a) = u(s, a) + \sum_{s^*} P(s^*|s, a) \max_a Q_{t+1}(s^*, a)$$
Value Function

The value function and optimal action are given by

\[ V(s) = \max_a Q(s,a) \quad \text{and} \quad a^* = \arg\max Q(s,a) \]

▶ Transition Matrix.
Consider the problem where you have three prizes left. Now \( s \) is the current state of three prizes.

\[ s^* = \{ \text{all sets of two prizes} \} \quad \text{and} \quad P(s^*|s,a = 1) = \frac{1}{3} \]

where the transition matrix is uniform to the next state.

▶ There’s no continuation for \( P(s^*|s,a = 0) \).
Imitation Learning

Machines can learn from billions of cases ....

- Learn from the best. Take all the moves of a chess grand master: 
  \( \{s_i, a_i\}_{i=1}^{N} \) state-action pairs
- Learn conditional distribution over actions \( \pi_\theta(a_t | s_t) \)
- Use deep neural network

---

Source: Mariusz 2016
Reward Function

- What if some observed actions are bad, e.g., in a self-play setting?
- How do we distinguish a good action from a bad action?
- Introduce the reward function $r(s, a)$
Q-learning

There’s a matrix of Q-values that solves the problem.

- Let $s$ denote the current state of the system and $a$ an action.
- The Q-value, $Q_t(s,a)$, is the value of using action $a$ today and then proceeding optimally in the future. We use $a = 1$ to mean no deal and $a = 0$ means deal.
- The Bellman equation for Q-values becomes

$$Q_t(s,a) = u(s,a) + \sum_{s^*} P(s^*|s,a) \max_a Q_{t+1}(s^*,a)$$

where $P$ denotes the transition matrix of states.

The value function and optimal action are given by

$$V(s) = \max_a Q(s,a) \text{ and } a^* = \text{argmax}_a Q(s,a)$$
Google DeepMind’ playing Atari

Google DeepMind

- Google DeepMind created an AI using deep reinforcement learning that plays Atari games and improves itself to a superhuman level.
- Capable of playing many Atari games and uses a combination of deep artificial neural networks and reinforcement learning.
- This was the beginning for Google DeepMind.

OpenAI and Dota2 is the current state-of-the-art
Policy Gradient

- Specify parametric $\pi$ and $r$ (e.g. deep learning)
- Generate state-action samples $s^i_t, a^i_t$ and associated reward $r^i_t$, $i = 1, \ldots, N$

$$E_\theta \left[ \sum_t r_t \right] \approx \frac{1}{N} \sum_{i=1}^N \sum_t r^i_t$$

- Run a step of batch SGD to update $\theta$: policy update via backpropagation, after seeing the reward
- Only works for deterministic dynamics!
- Naive algorithm
Reinforcement Learning

- Generate their training data via optimally designed experiments
  Maximize a reward to find an optimal policy \( d(x_t \mid \theta) \)

- Action function predicts response of the system to event \( t \) with characteristics \( x_t \). e.g. \( t \) - customer arriving on the website and \( d \) is the list of ads to be shown Customer generates response \( y_t \), e.g. click/no click

- Calculate the reward \( r(d(x_t \mid \theta), y_t) \)
  Sequentially learn parameters \( \theta, \ldots, \theta, \ldots \)

Goal: find an optimal configuration \( \theta^* \) that minimizes the objective

\[
\sum_{t=1}^{T} \left[ r(d(x_t \mid \theta), y_t) - r(d(x_t \mid \theta), y_t) \right]
\]
Example: Reinforcement Learning

Here’s the basic step-up for Google Ad placements

- $t$ - user lending on your web page
- $r(d(x_t \mid \theta), y_t)$ pick one of the $J$ possible ads
- Reward $y_t = 1$ if the user clicks on the ad and $y_t = 0$ otherwise
- Parameters are $\theta_j = p(y_t = 1 \mid d_t = j)$
Multi-Armed Bandit: Thompson Sampling

- \( r(d(x_t \mid \theta), y_t) \) pick one of the \( J \) possible ads
- Reward \( y_t = 1 \) if the user clicks on the ad and \( y_t = 0 \) otherwise
- Parameters are \( \theta_j = p(y_t = 1 \mid d_t = j) \)

\[
p(d_{t+1} = j) = p \left( \theta_j = \max_k \theta_k \mid y_t \right)
\]

Replace it with a Monte-Carlo simulation draw

\[
\theta_{t+1} \sim p(\theta \mid y^t)
\]

then the best decision is

\[
d_{t+1} = \arg \max_j \theta_{t+1,j}
\]
Beta-Bernoulli

Traditional approach is

\[ p(w \mid \alpha, \beta) = \prod_{j=1}^{J} \text{Beta}(d_j \mid \alpha, \beta) \]

Count success / failures after \( t \) steps

\[ n_{j,0} = \sum_{i=1}^{t} \mathbb{I}(y_i = 0, d_i = j) \]

\[ n_{j,1} = \sum_{i=1}^{t} \mathbb{I}(y_i = 1, d_i = j) \]

The posterior

\[ p(w \mid y^t) = \prod_{j=1}^{J} \text{Beta}(d_j \mid \alpha + n_{j,1}, \beta + n_{j,0}) \]

Key technology behind Google Ad placement
Pattern Matching: Chess

Shannon number $10^{152}$

- Hand-coded Rules
- Chess.com
  Dataset on all games. Openings, Middle game, End game
- 23 million human games. Logistic regression, predict next move of Grandmaster given board position. 57% accuracy
- Deep and Reinforcement Learning!
  Maximize probability of winning. Build Value and Policy functions.
- alphaGoZero. Play itself billions of times!!
  No need for humans . . . Spatial not lines of play
AlphaGo and AlphaGo Zero

Hand-crafted heuristic rules with deep learners

▸ Maximize probability of winning (Value function)
▸ Use SGD to update network weights based on self-play samples
▸ 4 hours to train grand-master level algorithm with no human inputs
▸ Same idea can be applied to many other settings: replace models of the world with neural nets
▸ Humans do the same. Tennis players do not use Newton’s laws to predict trajectory of a ball

AlphaGo Movie Trailer
Value Function

\[ v_\theta(s) \]

\( s \)

\( \theta \)
Policy Function

Move probabilities

Position

$p_{\sigma}(a|s)$
Full Tree
Monte-Carlo rollouts
Reducing depth with value network

- Value function approximates probability of winning.
- Pick the path with highest approximated chance to win the game
- No need to explore the tree till the end
Reducing breadth with policy network

- Policy function gives a histogram over possible moves
- Pick a few with highest probabilities
- No need to explore low probability moves, reduce breadth of the search
Summary: \( \alpha \text{Go} \)

**alphaGo Movie**

Supervised and Reinforcement Learning, Value Function and Tree Search

**Convenient**
- Fully observed
- Discrete action space
- Perfect simulator
- Relatively short game
- Trial-and error experience
- Large human datasets

**Inconvenient**
- Actions executed awkwardly
- Incomplete information
- Imperfect simulator
- Longer tasks, hard to assess value
- Hard to practice millions of times
- Small human data sources

AlphaGoZero’s Strategies
You Need Both Exploration and Exploitation to Achieve the Grand Master Level

Source: AlphaStar: Grandmaster level in StarCraft II using multi-agent reinforcement learning by DeepMind

Nature Video
Bayes, AI and Deep Learning: 41913

Week 5: AI
Modern Regression Methods

Nick Polson

http://faculty.chicagobooth.edu/nicholas.polson/teaching/41913/
The goal is **model selection**

- Why not include all the variables?
  Big models tend to over-fit and find specific features.
- Need to trade-off fit for making good predictions.
  Friedman: A good model is one that predicts!
MSE: Out-of-Sample Prediction

A very popular statistical criterion is **mean squared error**

MSE is defined by

\[
MSE = \sum (Y - \hat{Y})^2
\]

- You make a prediction \( \hat{Y} \) about the variable \( Y \).
- After the outcome \( Y \), you calculate \( (Y - \hat{Y})^2 \).
- In data mining, it is popular to use a holdout sample and after you’ve built your statistical model to test it out-of-sample in terms of its mean squared error performance.
Cross-Validation

Cross-Validation: Fit the model on training data.

Use model to predict $\hat{Y}$-values for the holdout sample.

Calculate predicted MSE $\frac{1}{N} \sum_{j=1}^{N} (Y_j - \hat{Y}_j)^2$.

Smallest MSE wins.
Ridge Regression

Ridge Regression is a modification of the least squares criteria that minimizes (as a function of β’s):

\[
\sum_{i=1}^{n} \left( Y_i - \beta_0 - \sum_{j=1}^{p} \beta_j X_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2
\]

for some value of \( \lambda > 0 \)

- The “blue” part of the equation is the traditional objective function of LS
- The “red” part is the shrinkage penalty, ie, something that makes costly to have big values for β
Ridge Regression

\[ \sum_{i=1}^{n} \left( Y_i - \beta_0 - \sum_{j=1}^{p} \beta_j X_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \]

- if \( \lambda = 0 \) we are back to least squares
- when \( \lambda \to \infty \), it is "too expensive" to allow for any \( \beta \) to be different than 0...
- So, for different values of \( \lambda \) we get a different solution to the problem
Ridge regression explores the bias-variance trade-off! The larger the $\lambda$ the more bias (towards zero) is being introduced in the solution, ie, the less flexible the model becomes... at the same time, the solution has less variance

- As always, the trick to find the “right” value of $\lambda$ that makes the model not too simple but not too complex!
- Whenever possible, we will choose $\lambda$ by comparing the out-of-sample performance (usually via cross-validation)
Ridge Regression

- Ridge is computationally very attractive as the “computing cost” is almost the same of least squares (contrast that with subset selection!)
- It’s a good practice to always center and scale the X’s before running ridge
The LASSO is a shrinkage method that performs automatic selection. Lasso provides solutions that are \textit{sparse}, some $\beta$’s exactly equal to 0! This facilitates interpretation of the results...
The LASSO solves the following problem:

$$\arg \min_{\beta} \left\{ \sum_{i=1}^{n} \left( Y_i - \beta_0 - \sum_{j=1}^{p} \beta_j X_{ij} \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\}$$

- Once again, $\lambda$ controls how flexible the model gets to be
- Still a very efficient computational strategy
- Whenever possible, we will choose $\lambda$ by comparing the out-of-sample performance (usually via cross-validation).
Ridge vs. LASSO

Why does the LASSO outputs zeros?
Ridge vs. LASSO

Which one is better?

► LASSO will perform better than Ridge when a relative small number of predictors have a strong effect in Y while Ridge will do better when Y is a function of many of the X’s and the coefficients are of moderate size

► LASSO can be easier to interpret (the zeros help!)

► But, if prediction is what we care about the only way to decide which method is better is comparing their out-of-sample performance
Choosing $\lambda$

The idea is to solve the ridge or LASSO objective function over a grid of possible values for $\lambda$...
Gauss invented the concept of least squares

The $L^2$-regression objective function

$$\arg \min_\beta \|y - X\beta\|^2$$

Here parameter vector is $\beta = (\beta_1, \ldots, \beta_p)$.

This has solution given by

$$\hat{\beta} = (X^\top X)^{-1} X^\top y$$

This can be numerically unstable when $X^\top X$ is ill-conditioned.

Happens when $p$ is large.
Ridge Regression $L^2 + L^2$-norm

Ridge Regression has the solution

$$\hat{\beta}_{\text{ridge}} = (X^\top X + \lambda I)^{-1} X^\top y$$

You can plot the coefficients over a regularisation path of $\lambda$’s.

This can also be interpreted as a Bayesian hierarchical model with a normal likelihood and prior
Optimization: LASSO $L^1$-norm

Least Absolute Shrinkage and Selection Operator (LASSO)

The solution to the lasso objective function

$$\arg\min_\beta \left\{ \frac{1}{2} (y - \beta)^2 + \lambda |\beta| \right\}$$

is the soft-thresholding operator defined by

$$\hat{\beta} = \text{soft}(y; \lambda) = (y - \lambda \text{sgn}(y))_+$$

Here $\text{sgn}$ is the sign function and $(x)_+ = \max(x, 0)$.

Define a slack variable $z = |\beta|$ and solve the joint constrained optimisation problem which is differentiable
Linear Regression

Model \( y_i = x_i^\top \beta + \epsilon_i \) where \( \beta = (\beta_1, \ldots, \beta_p) \) for \( 1 \leq i \leq n \).
Equivalently: \( y = X\beta + \epsilon \) and \( \min_\beta ||y - X\beta||^2 \).

▶ Predictive Ability

The MLE (maximum likelihood) or OLS (Ordinary Least Squares) estimator is designed to have zero bias.
That means it can suffer with high variance!
There’s a variance/bias tradeoff that we have to trade-off.

Main Advantage: Interpretability
Example

Simulated Data: \( n = 50, p = 30 \) and \( \sigma^2 = 1 \).

True model: linear with 10 large coefficients between 0.5 and 1.

- Linear Regression
  Bias squared = 0.006 and variance = 0.627.
  Prediction error = 1 + 0.006 + 0.627 = 1.633

- We’ll do better by shrinking the coefficients to reduce the variance

- How big a gain will we get with Ridge/Bayes Regression?
Example: True Coefficients

Figure: Shrinkage will Help
Example: Prediction error

Ridge Regression At best: Bias squared = 0.077 and variance = 0.402. Prediction error = 1 + 0.077 + 0.403 = 1.48
Bias-Variance Tradeoff

Simulated data: \( n = 50, p = 30 \).

Figure: Ridge
Prostate Data

Cross-Validation: CV and Regularization Path

Figure: Lasso
What about Ridge?

Figure: Ridge Coefficient Estimates
Prostate Data

CV Regularisation Path

Hastie, Friedman and Tibshriani (2013). *Elements of Statistical Learning*

Figure: Lasso
What about Ridge?

![Figure: Ridge: CV regularisation](image)

**Figure:** Ridge: CV regularisation
Bayes, AI and Deep Learning: 41913

Week 6: AI
Predictive Analytics

Nick Polson

http://faculty.chicagobooth.edu/nicholas.polson/teaching/41913/
What Does “AI” Really Mean?
Think of an algorithm.
Two distinguishing features of AI algorithms:

1. Algorithms typically deal with probabilities rather than certainties.
2. There’s the question of how these algorithms “know” what instructions to follow.
Three AI enablers

1. Moore’s law: speed of computers (Intel, Nvidia)
2. New Moore’s law: Explosive growth in the amount of data, as all of humanity’s information is digitized
3. Cloud-computing (Nvidia, Google, AWS, Facebook, Azure)

Together with
1. Massive Data: Fitting models to describe complicated patterns without over-fitting requires millions or billions of data points.
2. Trial and Error: A Billion Times per sec better than 10,000 hours.
3. Pattern Recognition: Deep Learning
AI: Pattern Recognition

Predictive Analytics uses Pattern Recognition
Good prediction rule to map input to output
Two key ideas

1. In AI, a “pattern” is a prediction rule mapping an input to expected output.
2. “Learning a pattern” means fitting a good prediction rule to a data set.

In AI, prediction rules are often referred to as “models”. The process of using data to find a good prediction rule is called “training the model”.

Ethics of Automating your job
Predictive Analytics

General Introduction

Predictive Analytics is the most widely used tool for high dimensional input-output analysis

\[ Y = F(X) \text{ where } X = (X_1, \ldots, X_p) \]

- Consumer Demand (Amazon, Airbnb, ...)
- Maps (Bing, Uber)
- Pricing
- Healthcare

The applications are endless ....
## Kaggle: Predictive Culture

<table>
<thead>
<tr>
<th>Competition</th>
<th>Title</th>
<th>Description</th>
<th>Award</th>
<th>Teams</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Passenger Screening Algorithm Challenge</td>
<td>Improve the accuracy of the Department of Homeland Security’s threat recognition algorithms</td>
<td>$1,500,000</td>
<td>518 teams</td>
</tr>
<tr>
<td></td>
<td>Zillow Prize: Zillow’s Home Value Prediction (Zestimate)</td>
<td>Can you improve the algorithm that changed the world of real estate?</td>
<td>$1,200,000</td>
<td>3,775 teams</td>
</tr>
<tr>
<td></td>
<td>Data Science Bowl 2017</td>
<td>Can you improve lung cancer detection?</td>
<td>$1,000,000</td>
<td>1,972 teams</td>
</tr>
<tr>
<td></td>
<td>Heritage Health Prize</td>
<td>Identify patients who will be admitted to a hospital within the next year using historical claims data.</td>
<td>$500,000</td>
<td>1,351 teams</td>
</tr>
</tbody>
</table>
AI at Google

Growing Use of Deep Learning at Google

Across many products/areas:
- Android
- Apps
- GMail
- Image Understanding
- Maps
- NLP
- Photos
- Speech
- Translation
- many research uses...
- YouTube
- ... many others ...

# of directories containing model description files

Unique project directories

Time

AI at Google

Google is AI first, placing bets in all major areas
Major investments in non-core ventures and initiatives

Healthcare
- DeepMind Health
- Calico
- verily
- editas medicine
- one medical group
- XandMe
- PiERian
- FLATIRON

Finance & Insurance
- Tala
- LendUp
- CircleUp

Cyber security
- IONIC
- ANOMALI
- Recorded Future

Transport & Logistics
- WAYMO
- Sidewalk Labs
- urban engines
- UBER
- Airware
- SKYcatch
- Saviok

IOT & Industry 4.0, Robotics
- nest
- Google Home

AI Enablers
- Mighty Ai
- ClearStory
- COHESITY
- MindMeld
- Research at Google
- DeepMind
- api.ai
- DataFox
- clarifai
- MAPD
- PREMISE

Credit: https://blog.aimultiple.com/ai-is-already-at-the-heart-of-google/
Target and other retailers use predictive analytics to study consumer purchasing behaviour to see what type of coupons or promotions you might like.

Here’s a famous story about a father and his daughter. Target predicted that his daughter was pregnant from her purchasing behaviour long before they were buying diapers.

Here’s the original link...

Target and Pregnancy
Walmart began using predictive analytics in 2004. Mining trillions of bytes’ worth of sales data from recent hurricanes determine what customers most want to purchase leading up to a storm.

Strawberry Pop-Tarts are one of the most purchased food items, especially after storms, as they require no heating and can be eaten at any meal.

Walmart and Hurricanes
Airbnb New User Bookings Prediction Competition

New users on Airbnb can book a place to stay in 34,000+ cities across 190+ countries.

Accurately predict where a new user will book their first travel experience.

Airbnb can then personalized content, decrease the average time to first booking, and better forecast demand.

12 classes—major destinations, and a did not book category.
List of users, demographics, web session records, and content data

Winner has the best out-of-sample prediction!!
Predicting Consumer Demand

Customers were less likely to return merchandise if it arrived within two days.

If an item takes longer to arrive, it gives customers more time to spot the product in a shop for less money and buy it, forcing Otto to forgo the sale and eat the shipping costs.

While customers are less likely to return merchandise that arrives quickly, also they prefer to receive everything at once.
Sorted daters into seven clusters, like "Diverse" and "Mindful," each with distinct characteristics.

Wired article

NOVA Video
Germany’s Otto

Otto sells other brands, does not stock those goods itself, hard to avoid one of the two evils: shipping delays until all the orders are ready for fulfilment, or lots of boxes arriving at different times.

▶ Analyze around 3 billion past transactions and 200 variables—past sales, searches on Otto’s site and weather information. They predict what customers will buy a week before they order. This system has proved so reliable, predicting with 90% accuracy what will be sold within 30 days, that Otto allows it automatically to purchase around 200,000 items a month from third-party brands with no human intervention.

Economist

Germany’s Otto
Stitch Fix CEO says AI Incredibly Valuable

Stitch Fix asks customers for insights and feedback alongside their size and color preference for items, even the ones customers didn’t like or buy, in exchange for a clear value proposition.

The breadth and depth of their data are valuable.

Their model relies on a combination of data science – machine learning, AI and natural language processing – and human stylists; on top of complex customer profiles built by data, stylists can layer the nuances of buying and wearing clothes.
Bayes predicts where you’re going to be dropped off.

Uber constructs prior probabilities for riders, Uber cars, and popular places.

Combine to construct a joint probability table

Then calculate the posterior probability of destination for each person and pool travellers together
Logistic Regression: Classification

When the $Y$ we are trying to predict is *categorical* (or *qualitative*) we say that we have a *classification* problem.

For a numeric (or *quantitative*) $Y$ we predict its value

For a binary output we predict the probability its going to happen

$$p(Y = 1 \mid X = x)$$

where $X$ is our usual list of predictors, $X_1, \ldots, X_p$
Logistic Regression

Suppose that we have a binary response, $Y$ taking the value 0 or 1

- Win or lose
- Sick or healthy
- Buy or not buy
- Pay or default

The goal is to predict the **probability that $Y$ equals 1**

You can then do **classification** and categorize a new data-point
Example: Default Data

Here’s a typical problem

Assessing credit risk and default data ...

- $Y$: whether or not a customer defaults on their credit card (No or Yes)
- $X$: The average balance that customer has remaining on their credit card after making their monthly payment.

... plus as many other features you think might predict $Y$ ...
Logistic Regression

$Y$ is an indicator: $Y = 0$ or $1$.

$X$ is our usual set of predictors/covariates

We need to model the probability that $Y = 1$ as

$$p(Y = 1 \mid X_1, \ldots, X_p) = f (\beta_1 X_1 + \ldots + \beta_p X_p)$$

where $f$ is increasing and $0 < f(X) < 1$ The logit-transform is given by $f(x) = e^x / (1 + e^x)$
Logistic Regression

The logistic regression model is linear in log-odds

\[
\log \left( \frac{p(Y = 1 \mid X)}{1 - p(Y = 1 \mid X)} \right) = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p
\]

These model are easy to fit in R:

\[
glm(Y \sim X_1 + X_2, family = \text{binomial})
\]

- “g” is for generalized; binomial indicates \(Y = 0\) or \(1\)
- “glm” has a bunch of other options.
Example: NBA point spread

Does the Vegas point spread predict whether the favorite wins or not?

Turquoise = Favorites does win, Purple = Favorite does not win
**R: Logistic Regression**

**In R:** the output gives us ...

```r
nbareg = glm(favwin~spread-1, family=binomial)
summary(nbareg)
```

Call:
```
glm(formula = favwin ~ spread - 1, family = binomial)
```

Coefficients:
```
                         Estimate  Std. Error   z value  P(> |z| )
spread                0.156000   0.013771  11.33333   <2e-16 ***
```

# prediction
```
newweek=c(8,4)
```

The $\beta$ measures how our log-odds change! $\beta = 0.156$
NBA Point Spread Prediction

“Plug-in” the values for the new game into our logistic regression

\[
P_{\text{favwin} | \text{spread}} = \frac{e^{\beta x}}{1 + e^{\beta x}}
\]

Check that when \( \beta = 0 \) we have \( p = \frac{1}{2} \).

- Given our new values \( \text{spread} = 8 \) or \( \text{spread} = 4 \),
  The win probabilities are 77% and 65%, respectively.

  0.7769474, 0.6511238

Clearly, the bigger spread means a higher chance of winning.
Bill Benter did the impossible: He wrote an algorithm that couldn’t lose at the track, close to a billion dollars later.

Benter’s model required his undivided attention. It monitored only about 20 inputs—just a fraction of the infinite factors that influence a horse’s performance, from wind speed to what it ate for breakfast, and the Jockey Club’s publicly available betting odds
Horse race prediction

We use the run.csv data from Kaggle. We want to use individual variables to predict the chance of winning of a horse.

For the simplicity of computation, we only consider horses with id \( \leq 500 \), and train the model with \( \ell_1 \)-regularized logistic regression.

And we include lengths\_behind, horse\_age, horse\_country, horse\_type, horse\_rating, horse\_gear, declared\_weight, actual\_weight, draw, win\_odds, place\_odds as predicting variables in our model.
Horse Race Predictions

Since most of the variables, such as country, gear, type, are categorical, after spanning them into binary indicators, we have more than 800 columns in the design matrix.

We try two logistic regression models. The first one includes win_odds given by the gambling company. The second one does not include the win_odds and we use win_odds to test the power of our model. We tune both models with a 10-fold cross-validation to find the best penalty parameter $\lambda$. 


Predict with win_odds

In this model, we fit the logistic regression with full dataset. The best $\lambda$ we find is $5.699782 e - 06$.

#variables w.r.t. $\lambda$

Coefficient of top 20 variables
Predict with win_odds

In this model, we randomly partition the dataset into training(70%) and testing(30%) parts. We fit the logistic regression with training dataset. The best $\lambda$ we find is $4.792637e - 06$.

#variables w.r.t. $\lambda$  

Coefficient of top 20 variables

The out-of-sample mean squared error for win_odds is 0.0668.
LinkedIn Study: How to Become an Executive

Analyze the career paths of about 459,000 LinkedIn members who worked at a Top 10 consultancy between 1990 and 2010 and became a VP, CXO, or partner at a company with at least 200 employees. About 64,000 members reached this milestone. \( \hat{p} = 0.1394 \).

- Look at their profiles – educational background, gender, work experience, and career transitions.
- Build a model to predict the probability of becoming an executive.

Conditional on making it into the database ....
Logistic Regression

Logistic regression with 8 key features (a.k.a. covariates):

$$\log \left( \frac{p}{1-p} \right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_8 X_8$$

- $p$: Probability of “success” – reach VP/CXO/Partner at a company with at least 200 employees.
- $X_i (i = 1, 2, \ldots, 8)$: Features to predict the “success” probability.
Features

Location Features: **X1 Metro region**: whether a member has worked in one of the top 10 largest cities in the U.S. or globally.

Personal Features: **X2 Gender**: Inferred from member names: ‘male’, or ‘female’.

Education Features: **X3 Graduate education type**: whether a member has an MBA from a top U.S. program / a non-top program / a top non-U.S. program / another advanced degree.

**X4 Undergraduate education type**: whether a member has attended a school from the U.S. News national university rankings / a top 10 liberal arts college /a top 10 non-U.S. school.
Features

Work Experience:

X5 Company count: # different companies in which a member has worked.

X6 Function count: # different job functions in which a member has worked.

X7 Industry sector count: # different industries in which a member has worked.

X8 Years of experience: # years of work experience, including years in consulting, for a member.
\( \hat{\beta}'s \) of Features

1. Location: \textit{Metro region:} 0.28
2. Personal: \textit{Gender(Male):} 0.31
3. Education: \textit{Graduate education type:} 1.16, \textit{Undergraduate education type:} 0.22
4. Work Experience: \textit{Company count:} 0.14, \textit{Function count:} 0.26, \textit{Industry sector count:} -0.22, \textit{Years of experience:} 0.09
Main Findings

1. Working across job functions, like marketing or finance, is good. Each additional job function provides a boost that, on average, is equal to three years of work experience. Switching industries has a slight negative impact. Learning curve? Lost network?

2. MBAs are worth the investment. But pedigree matters. *Top five program equivalent to 13 years of work experience!!!*

3. Location matters. NYC helps.
Examples

**Person A (p=6%)**: Male in Tulsa, Oklahoma, Undergraduate degree, 1 job function for 3 companies in 3 industries, 15-year experience.

**Person B (p=15%)**: Male in London, Undergraduate degree from top international school, Non-MBA Master, 2 different job functions for 2 companies in 2 industries, 15-year experience.

**Person C (p=63%)**: Female in New York City, Top undergraduate program, Top MBA program, 4 different job functions for 4 companies in 1 industry, 15-year experience.
Let’s re-design Person B!!

Person B (p=15%): Male in London, Undergraduate degree from top international school, Non-MBA Master, 2 different job functions for 2 companies in 2 industries, 15-year experience.

1. Work in one industry rather than two. Increase 3%
2. Undergrad from top 10 US program rather than top international school. 3%
3. Worked for 4 companies rather than 2. Another 4%
4. Move from London to NYC. 4%
5. Four job functions rather than two. 8%. A 1.5X effect.
6. Worked for 10 more years. 15%. A 2X effect.
Choices and Impact (Person B)
Bayes, AI and Deep Learning: 41913

Week 7 Deep Learning
Learning Algorithms and Theory

Nick Polson

http://faculty.chicagobooth.edu/nicholas.polson/teaching/41913/
Why do we care about DL?

Input space ($X$) includes numerical, text (word2vec), images, videos. Vectors, matrices and tensors, ...

- Google’s translation algorithm
  ~ 1-2 billion parameters
- Alexa’s speech recognition: 100 million parameters
  Networks will get larger and more efficient
- Google Waymo

Advances in computing speed (Nvidia) lets us train and implement Deep Learning in real-time. Google Waymo’s Lidar processes 6MB Data per second ...
Multi-Layer Deep Models

- NN models one layer!! Key is to use multi “deep” layers
- Learn weight and connections in hidden layers

Predicting House Prices ...

<table>
<thead>
<tr>
<th>Input(X)</th>
<th>Factor</th>
<th>Output(Y)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td></td>
<td></td>
</tr>
<tr>
<td># bedrooms</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zip(postal) Code</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wealth</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Family Size</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Commute (walkability)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>School Quality</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Price</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Deep Learning Predictors

Smart conditional averaging

The competitors: Trees and RF.

(a) Tree Kernel  (b) Random Forest Kernel

Few points will be neighbors in a high dimensional input space.
Kolmogorov-Arnold

There are no multivariate functions just superpositions of univariate ones

Let \( f_1, \ldots, f_L \) be given univariate activation functions. We set

\[
f^W,b_l = f_l \left( \sum_{j=1}^{N_l} W_{lj} X_j + b_l \right) = f_l(W_l X_l + b_l), \quad 1 \leq l \leq L,
\]

Our deep predictor has hidden units \( N_l \) and depth \( L \).

\[
\hat{Y}(X) = F(X) = \left( f^W_{1,b_1} \circ \cdots \circ f^W_{L,b_L} \right)(X)
\]

Put simply, we model a high dimensional mapping \( F \) via the superposition of univariate semi-affine functions.
Explicit Formulation

\[ Z^{(1)} = f^{(1)} \left( W^{(0)} X + b^{(0)} \right) \]
\[ Z^{(2)} = f^{(2)} \left( W^{(1)} Z^{(1)} + b^{(1)} \right) \]
\[ \vdots \]
\[ Z^{(L)} = f^{(L)} \left( W^{(L-1)} Z^{(L-1)} + b^{(L-1)} \right) \]
\[ \hat{Y}(X \mid W, b) = f^{L+1} \left( W^{(L)} Z^{(L)} + b^{(L)} \right). \]
The deep learning predictor uncovers non-linear features of the data through hierarchical feature interpretation.

The input of the first transformation is the \((L + 1)\)-th layer, and so on, with the output \(\hat{Y}\) as the first layer. The inner layers are called hidden layers.

Commonly used activation functions are

- Sigmoidal (e.g., \(1/(1 + \exp(-x))\), \(cosh(x)\), or \(tanh(x)\)),
- Heaviside gate functions (e.g., \(\mathbb{I}(x > 0)\)),
- or rectified linear units (ReLU) \(\max\{x, 0\}\).
Kolmogorov-Arnold Example

Interaction terms, $x_1x_2$ and $(x_1x_2)^2$, and max functions, $\max(x_1, x_2)$ can be expressed as nonlinear functions of semi-affine combinations. Specifically,

$$x_1x_2 = \frac{1}{4}(x_1 + x_2)^2 - \frac{1}{4}(x_1 - x_2)^2$$

$$\max(x_1, x_2) = \frac{1}{2}|x_1 + x_2| + \frac{1}{2}|x_1 - x_2|$$

$$(x_1x_2)^2 = \frac{1}{4}(x_1 + x_2)^4 + \frac{7}{4 \cdot 33}(x_1 - x_2)^4 - \frac{1}{2 \cdot 33}(x_1 + 2x_2)^4 - \frac{2^3}{33}(x_1 + \frac{1}{2}x_2)^4$$
Deep ReLU architectures can be viewed as Max-Sum networks via the following simple identity.

Define $x^+ = \max(x, 0)$. Let $f_x(b) = (x + b)^+$ where $b$ is an offset. Then $(x + y^+)^+ = \max(0, x, x + y)$.

Can show:

$$(f_{x_1} \circ \ldots \circ f_{x_k})(0) = (x_1 + (x_2 + \ldots + (x_{k-1} + x_k^+)^+)^+) = \max_{1 \leq j \leq k} (x_1 + \ldots + x_j)^+$$

A composition or convolution of max-layers is than a one layer max-sum network
Traditional Modeling Culture

Traditional stats models use one-layer transformations

- **SIR**
  \[ y = g(\beta_1^T x, \beta_2^T x, \ldots, \beta_k^T x, \epsilon), \]

- **GLM:**
  \[ y = g^{-1}(\beta^T x) \]

- **GAM**
  \[ y = \beta_0 + f_1(x_1) + \cdots + f_k(x_k) \]

- **PCR**
  \[ y = \beta^T (Wx), \ W \in \mathbb{R}^{k \times p}, \ k < p \]
Deep Architectures

MLP

auto-encoder

convolution

recurrent

Long / short term memory

http://www.asimovinstitute.org/neural-network-zoo/
What's wrong with Kernels?

2D image of 1000 uniform samples from a 50-dimensional ball $B_{50}$.

Marginal distribution shrinks as dimensionality of the space grows

(a) $p = 100$  
(b) $p = 200$  
(c) $p = 300$  
(d) $p = 400$
DL Solves the Problem

**Figure:** Hyperplanes defined by three neurons with ReLU activation functions.

\[
\hat{Y}(X) = \sum_{k \in K} w_k(X) \hat{Y}_k(X),
\]
Tree vs DL example

\[ Y = \text{softmax}(w^0 Z^2 + b^0) \]
\[ Z^2 = \tanh(w^2 Z^1 + b^2) \quad Z^1 = \tanh(w^1 X + b^1). \]

An advantage of deep architectures is that the number of hyper-planes grow exponentially with the number of layers.

Source: https://cs.stanford.edu/people/karpathy/convnetjs/
What is Wrong with DL

At the moment ....

- Point estimates
- No model selection mechanism
- No regularization mechanism
NN Terminology

- $x$: inputs
- $W_1, \ldots, W_L$ weights, $W_i \in \mathbb{R}^{n_{i-1} \times n_i}$
- $b_1, \ldots, b_L$ biases, $b_i \in \mathbb{R}$
- $\sigma_1, \ldots, \sigma_L$: activation functions, $\sigma : \mathbb{R} \to \mathbb{R}$
- $y$: output
- $n_1, \ldots, n_L, \sigma_1, \ldots, \sigma_L, L$: architecture
Supervised Learning

Given data $D = \{(x_i, y_i)\}, \ i = 1, \ldots, m$

- Find parameters of NN $F(x, \theta) \theta = (W_1, \ldots, W_L, b_1, \ldots, b_L)$ using the data
- Minimize the loss function

$$\min_\theta l(F(x, \theta), y)$$

- Regression loss

$$l(F(x, \theta), y) = (F(x, \theta) - y)^2$$

- Classification loss for class $k$

$$l(F(x, \theta), y = k) = -\log \frac{\exp(f_k(x, \theta))}{\sum_{i=1}^K \exp(f_i(x, \theta))}$$
Empirical Loss

We want to minimize the theoretical loss

$$\min_{\theta} E_D(l(F(x, \theta), y))$$

We approximate with the empirical loss

$$\min_{\theta} \frac{1}{m} \sum_{i=1}^{m} l(F(x_i, \theta), y_i)$$

Plus add regularization

$$\min_{\theta} \frac{1}{m} \sum_{i=1}^{m} l(F(x_i, \theta), y_i) + \phi(\theta)$$
Probabilistic View

\( Y \sim p ( Y \mid \hat{Y}(X \mid W, b)) \) and \( \hat{Y}(X \mid W, b) \) a predictor

Given parameters \((W, b)\), log-likelihood defines oss \( \mathcal{L} \) as

\[
\mathcal{L}(Y, \hat{Y}) = - \log p ( Y \mid \hat{Y}(X \mid W, b)) .
\]

Given a training sample \( D = \{(X_i, Y_i)\}_{i=1}^n \), the \( L_2 \)-norm,

\[
\mathcal{L}(Y_i, \hat{Y}(X_i)) = \frac{1}{n} \sum_{i=1}^{n} (Y_i - \hat{Y}(X_i))^2
\]

Traditional least squares or negative cross-entropy loss for classification

\[
\mathcal{L}(Y_i, \hat{Y}(X_i)) = - \sum_{i=1}^{n} Y_i \log \hat{Y}(X_i)
\]
Probabilistic View

Bias-variance trade-off controlled by prior

$$\mathcal{L}_\lambda(Y, \hat{Y}) = -\log p(Y | \hat{Y}(X | W, b)) - \log p(W, b | \lambda).$$

The regularization term is a negative log-prior distribution

$$-\log p(W, b | \lambda) = \lambda \phi(W, b),$$
$$p(W, b | \lambda) \propto \exp(-\lambda \phi(W, b)).$$

Deep predictors are regularized maximum a posteriori (MAP) estimators, where

$$-p(W, b|D) \propto -p(Y | \hat{Y}(X | W, b)) p(W, b | \lambda)$$
$$\propto \exp (- \log p(Y | \hat{Y}(X | W, b)) - \log p(W, b)).$$

Training requires the solution of a highly nonlinear optimization problem

$$\arg\max_{W, b} \log p(W, b | D).$$
Logistic Regression

We use a logistic function

\[
p(y = 1 \mid x) = h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}}
\]

\[
p(y = 0 \mid x) = 1 - p(y = 1 \mid x)
\]

The likelihood

\[
P(y \mid x; \theta) = h_\theta(x)^y (1 - h_\theta(x))^{(1-y)},
\]
Stochastic Gradient Descent (SGD)

Stochastic gradient descent (SGD) is a default gold standard for minimizing the function $f(W, b)$ (maximizing the likelihood) to find the deep learning weights and offsets. SGD simply minimizes the function by taking a negative step along an estimate $g^k$ of the gradient $\nabla f(W^k, b^k)$ at iteration $k$. Smooth convex optimization

$$\min_{x} f(x)$$

i.e., $f$ is convex and differentiable with $\text{dom}(f) = \mathbb{R}^n$. Denote the optimal criterion value by $f^* = \inf f(x)$, and a solution by $x^*$

**Gradient descent:** choose initial $x^{(0)} \in \mathbb{R}^n$, repeat:

$$x^{(k)} = x^{(k-1)} - t_k \nabla f(x^{(k-1)}), \ k = 1, 2, 3, \ldots$$

Stop at some point
\[ x^2 + y^2 \]
\sin\left(x^2 + y^2\right)
$x^2 + y^2$
\[ \sin\left(x^2 + y^2\right) \]
Example: Linear Regression for Big Data

\[
\text{minimize } \sum_{i=1}^{m} (a_i^T x - y_i)^2
\]

For small to medium scale problems, we can use our favorite linear algebra software to find pseudoinverse \( A^\dagger \). Depending on the solver, a QR or SVD decomposition will be used, cost is \( O(n^3) \)

What if \( n = 10^9 \)?

\[
\frac{\partial f}{\partial x_i} = 2a_i^T \sum_{i=1}^{m} (a_i^T x - y_i)
\]

\[
\nabla_x f(x) = \left( \frac{\partial f}{\partial x_1}, \ldots, \frac{\partial f}{\partial x_n} \right)
\]

Iterate until convergence

\[
x^+ = x - t\nabla_x f(x)
\]
Terminology

Step size $t_k$

- $t_k$ is called a step size (in optimization literature)
- $t_k$ is called learning rate (in ML literature)

In ML it is typical to use a constant learning rate $t_k = t = 0.1$. The gradient gets smaller as we approach optima, so GD still converges. There are better ways to choose $t_k$

In ML, GD method also called “batch” gradient descent to highlight the fact that the entire dataset is used at each iteration
Gradient descent interpretation

At each iteration, consider the expansion

\[ f(y) \approx f(x) + \nabla f(x)^T (y - x) + \frac{1}{2t} ||y - x||^2_2 \]

**Quadratic approximation**, replacing usual \( \nabla^2 f(x) \) by \( \frac{1}{t} I \)

\[
\begin{align*}
 f(x) + \nabla f(x)^T (y - x) & \quad \text{linear approximation to } f \\
 \frac{1}{2t} ||y - x||^2_2 & \quad \text{proximity term to } x, \text{ with weight } 1/(2t)
\end{align*}
\]

Choose next point \( y = x^+ \) to minimize quadratic approximation:

\[ x^+ = x - t \nabla f(x) \]
Blue point is x, red point is

\[ x^+ = \arg \min_y f(x) + \nabla f(x)^T (y - x) + \frac{1}{2t} \|y - x\|^2_2 \]
Fixed step size

Simply take $t_k = t$ for all $k = 1, 2, 3, \ldots$, can diverge if $t$ is too big. Consider $f(x) = (10x_1^2 + x_2^2)/2$, gradient descent after 8 steps:
Can be **slow** if $t$ is too small. Same example, gradient descent after 100 steps:
Same example, gradient descent after 40 appropriately sized steps:

Clearly there’s a tradeoff-convergence analysis later will give us a better idea
Backtracking line search

One way to adaptively choose the step size is to use backtracking line search:

- First fix parameters $0 < \beta < 1$ and $0 < \alpha \leq 1/2$
- At each iteration, start with $t = 1$, and while

$$f(x - t \nabla f(x)) > f(x) - \alpha t \|\nabla f(x)\|^2$$

shrink $t = \beta t$. Else perform gradient descent update

$$x^+ = x - t \nabla f(x)$$

Simple and tends to work well in practice (further simplification: just take $\alpha = 1/2$)
Backtracking interpretation

For us $\Delta x = -\nabla f(x)$
Backtracking picks up roughly the **right step size** (12 outer steps, 40 steps total):

\[ \alpha = \beta = 0.5 \]
Exact line Search

Could also choose step to do the best we can along direction of negative gradient, called exact line search:

\[ t = \arg \min_{s \geq 0} f(x - s \nabla f(x)) \]

Usually not possible to do this minimization exactly

Approximations to exact line search are often not much more efficient than backtracking, and it’s usually not worth it
quadratic problem in $\mathbb{R}^2$

$$f(x) = \frac{1}{2}(x_1^2 + \gamma x_2^2), \quad \gamma > 0$$

with exact line search, starting at $x^{(0)} = (\gamma, 1)$:

$$x_1^{(k)} = \gamma \left( \frac{\gamma - 1}{\gamma + 1} \right)^k, \quad x_2^{(k)} = \left( -\frac{\gamma - 1}{\gamma + 1} \right)^k$$

- very slow when $\gamma \gg 1$ or $\gamma \ll 1$
- example for $\gamma = 10$
Non-Quadratic example

\[ f(x) = e^{x_1 + 3x_2 - 0.1} + e^{x_1 - 3x_2 - 0.1} + e^{-x_1 - 0.1} \]
a problem in $\mathbb{R}^{100}$

$$f(x) = c^T x - \sum_{i=1}^{500} \log(b_i - a_i^T x)$$
Large Data Tricks: Preprocessing

- data scaling helps to improve the conditioning
- make data to be on the same scale: $\mu = 0$ and $\sigma = 1$, i.e. each $-1 \leq x_i \leq 1$
- when the input dimension varies by orders of magnitude, it is better to take the $\log(1 + x)$
- Doing so makes learning work much better

$$x_i^+ - x_i \propto \frac{\partial f}{\partial x_i}$$

If the average value of the $x$‘s is large (say, 100), then the updates will be very large and correlated, which makes learning bad and slow. Keeping things zero-mean and with small variance simply makes everything work much better.
Stochastic Gradient Descent

Consider sum of functions

$$\min_x \frac{1}{n} \sum_{i=1}^{n} f_i(x)$$

Gradient descent applied to this problem would repeat

$$x^{(k)} = x^{(k-1)} - t_k \cdot \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(x^{(k-1)}), \ k = 1, 2, 3, \ldots$$

In comparison, stochastic gradient descent (or incremental gradient descent) repeats

$$x^{(k)} = x^{(k-1)} - t_k \cdot \nabla f_{i_k}(x^{(k-1)}), \ k = 1, 2, 3, \ldots$$

where $i_k \in \{1, \ldots, n\}$ is some chosen index at iteration $k$
Notes:

- Typically we make a (uniform) random choice $i_k \in \{1, \ldots, n\}$
- Also common: mini-batch stochastic gradient descent, where we choose a random subset $I_k \subset \{1, \ldots, n\}$, of size $b \ll n$, and update according to

$$x^{(k)} = x^{(k-1)} - t_k \cdot \frac{1}{b} \sum_{i \in I_k} \nabla f_i(x^{(k-1)}), k = 1, 2, 3, \ldots$$

- In both cases, we are approximating the full gradient by a noisy estimate, and our noisy estimate is unbiased

$$\mathbb{E}[\nabla f_{i_k}(x)] = \nabla f(x)$$

$$\mathbb{E} \left[ \frac{1}{b} \sum_{i \in I_k} \nabla f_i(x) \right] = \nabla f(x)$$

The mini-batch reduces the variance by a factor $1/b$, but is also $b$ times more expensive!
Example with $n = 10,000, p = 20$, all methods employ fixed step sizes (diminishing step sizes give roughly similar results):
What’s happening? Iterations make better progress as mini-batch size $b$ gets bigger. But now let’s parametrize by flops:
Example: Online Learning

Say you are a retail company and you sell a product. Say you have a model $p(y|x, \theta)$, where $y \in \{\text{buy, not buy}\}$, $x$ is the characteristics of the buyer (age, gender, previous purchasing behavior...) and product characteristics, including price. $\theta$ is the model parameters (neural net, logistic regression,....)

Stochastic gradient descent is the mechanism to update $\theta$ every time user visits a product page.

You adjust price according to user’s characteristics!
Gradient descent
\[ x^+ = x - t \nabla f(x) \]

**MAP**
Machine 1: Use 1, ..., \( n/4 \) samples to calculate \( x^+_1 \)
Machine 2: Use \( n/4 + 1, ..., n/2 \) samples to calculate \( x^+_2 \)
Machine 3: Use \( n/2 + 1, ..., 3n/4 \) samples to calculate \( x^+_3 \)
Machine 4: Use \( 3n/4 + 1, ..., n \) samples to calculate \( x^+_4 \)

**REDUCE**
Average the results. For NN speedup is at least \#nodes / \( \log(\#\text{rows}) \) (http://arxiv.org/abs/1209.4129)
Can be used on multi-core machines
GD: Practicalities

Stopping rule: stop when $||\nabla f(x)||_2$ is small

- Recall $\nabla f(x^*) = 0$ at solution $x^*$
- If $f$ is strongly convex with parameter $m$, then

$$||\nabla f(x)||_2 \leq \sqrt{2m\epsilon} \Rightarrow f(x) - f^* \leq \epsilon$$

Pros and cons of gradient descent:

- Pro: simple idea, and each iteration is cheap
- Pro: very fast for well-conditioned, strongly convex problems
- Con: often slow, because interesting problems aren’t strongly convex or well-conditioned
- Con: can’t handle nondifferentiable functions
Automatic Differentiation (AD)

In general, there are three different ways to calculate those derivatives.

- numerical differentiation, when a gradient is approximated by a finite difference \( f'(x) = (f(x + h) - f(x))/h \)
- the numerical differentiation is not backward stable
- Symbolic differentiation uses a tree form representation of a function and applies chain rule
Similar to symbolic differentiations AD recursively applies the chain rule. It calculates the exact value of derivative and thus avoids the problem of numerical instability.

AD provides the value of derivative evaluated at a specific point rather than an analytical representation of the derivative.
Automatic Differentiation in Machine Learning: a Survey

\[ l_1 = x \]
\[ l_{n+1} = 4l_n(1 - l_n) \]
\[ f(x) = l_4 = 64x(1 - x)(1 - 2x)^2(1 - 8x + 8x^2)^2 \]

Manual Differentiation

\[ f'(x) = 128x(1 - x)(-8 + 16x)(1 - 2x)^2(1 - 8x + 8x^2)^2 + 64x(1 - x)(1 - 2x)^2(1 - 8x + 8x^2)^2 - 64x(1 - 2x)^2(1 - 8x + 8x^2)^2 - 256x(1 - x)(1 - 2x)(1 - 8x + 8x^2)^2 \]

Coding

\[ f(x): \]
\[
\begin{align*}
  v &= x \\
  \text{for } i = 1 \text{ to } 3 \\
  &\quad v = 4v*(1 - v) \\
  &\quad \text{return } v \\
\end{align*}
\]

or, in closed-form,

\[ f(x): \]
\[
\begin{align*}
  &\quad \text{return } 64x(1 - x)((1 - 2x)^2)(1 - 8x + 8x^2)^2 \\
\end{align*}
\]

Symbolic Differentiation of the Closed-form

\[ f'(x): \]
\[
\begin{align*}
  &\quad \text{return } 128x(1 - x)(-8 + 16x)(1 - 2x)^2(1 - 8x + 8x^2)^2 + 64x(1 - x)(1 - 2x)^2(1 - 8x + 8x^2)^2 - 64x(1 - 2x)^2(1 - 8x + 8x^2)^2 - 256x(1 - x)(1 - 2x)(1 - 8x + 8x^2)^2 \\
\end{align*}
\]

Automatic Differentiation

\[ f'(x): \]
\[
\begin{align*}
  (v, dv) &= (x, 1) \\
  \text{for } i = 1 \text{ to } 3 \\
  &\quad (v, dv) = (4v*(1 - v), 4* dv - 8*v*dv) \\
  &\quad \text{return } (v, dv) \\
\end{align*}
\]

Numerical Differentiation

\[ f'(x): \]
\[
\begin{align*}
  h &= 0.000001 \\
  &\quad \text{return } (f(x + h) - f(x)) / h \\
\end{align*}
\]

\[ f'(x_0) = f'(x_0) \]

Exact

\[ f'(x_0) \approx f'(x_0) \]

Approximate

Figure 2: The range of approaches for differentiating mathematical expressions and computer code, looking at the example of a truncated logistic map (upper left). Symbolic differentiation (center right) gives exact results but requires closed-form input and suffers from expression swell; numerical differentiation (lower right) has problems of accuracy due to round-off and truncation errors; automatic differentiation (lower left) is as accurate as symbolic differentiation with only a constant factor of overhead and support for control flow.
AD does not require analytical specification. AD can differentiate complex functions which involve IF statements and loops, and AD can be implemented using either forward or backward mode.

```python
def sigmoid(x,b,w):
    v1 = w*x;
    v2 = v1 + b
    v3 = 1/(1+exp(-v2))
```
Back Propagation

A specific type of automated differentiation algorithm applied. Take $c = a+b$; $d = b+1$; $e = c*d$

Source: https://colah.github.io/posts/2015-08-Backprop/
Forward Calculation

\[ a = 2 \]
\[ b = 1 \]
\[ c = a + b = 3 \]
\[ d = b + 1 = 2 \]
\[ e = c \times d = 6 \]
Differentiation via Backward Calculations (Backprop)

\[ \frac{\partial}{\partial a} (a + b) = \frac{\partial a}{\partial a} + \frac{\partial b}{\partial a} \quad \text{and} \quad \frac{\partial}{\partial u} uv = u \frac{\partial v}{\partial u} + v \frac{\partial u}{\partial u} \]
Backprop

# set some inputs
x = −2; y = 5; z = −4

# perform the forward pass
q = x + y # q becomes 3
f = q * z # f becomes −12

# perform the backward pass (backpropagation)
# in reverse order:
# first backprop through f = q * z
dfdz = q # df/dz = q, so gradient on z becomes 3
dfdq = z # df/dq = z, so gradient on q becomes −4
# now backprop through q = x + y
dfdx = 1.0 * dfdq # dq/dx = 1.
# And the multiplication here is the chain rule!
dfdy = 1.0 * dfdq # dq/dy = 1
That is, the (sub)gradient is 1 on the input that was larger and 0 on the other input. Intuitively, if the inputs are
whether they want their outputs to increase or decrease (and how strongly), so as to make the final output value
higher.

Modularity: Sigmoid example

1. Let's start with a simple circuit diagram:

```
  +   * 
  |   |  
  |   |  
  v   v  
 x   y  
```

Inputs: x = -2, y = 5, z = -4

Gate outputs: q = 3, f = -12

2. Compute the output of the add gate:

```
q = x + y = -2 + 5 = 3
```

3. Compute the output of the multiply gate:

```
f = q * z = 3 * (-4) = -12
```

4. Compute the derivatives of both expressions separately, as seen in the previous section.

5. Notice that this has the desired effect: If we anthropomorphize the circuit as wanting to output a higher value
then the add gate's output would decrease, which in turn makes the multiply gate's output increase.

6. The rest of the circuit computed the final value, which is -12. During the backward pass in which the chain rule
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

7. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

8. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

9. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

10. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

11. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

12. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

13. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

14. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

15. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

16. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

17. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

18. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

19. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

20. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

21. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

22. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

23. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

24. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

25. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

26. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

27. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

28. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

29. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

30. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

31. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

32. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.

33. Notice that backpropagation is a beautifully local process. Every gate in a circuit diagram gets some inputs and
right away compute two things: 1. its output value and 2. the gradient of its output value on the final output of the entire circuit.

34. Chain rule (which can help with intuition), then we can think of the circuit as “wanting” the output of the add gate to be lower
that the gradient for its output was -4. If we anthropomorphize the circuit as wanting to output a higher value
is applied recursively backwards through the circuit, the add gate (which is an input to the multiply gate) learns
+1. The rest of the circuit computed the final value, which is -12.
Can you guess the function?

The point of this section is that the details of how the backpropagation is performed, and which parts of the created an intermediate variable.

Implementation protip: staged backpropagation

As we see, the gradient turns out to simplify and becomes surprisingly simple. For example, the sigmoid derivation (after a fun tricky part where we add and subtract a 1 in the numerator):

In the example above, we see a long chain of function applications that operates on the result of the dot product (image above), except this way it would be done with a single, simple and efficient expression (and with less

Where the functions (add, mul, max), there are four more:

# we're done! we have the gradients on the inputs to the circuit

# backward pass through the neuron (backpropagation)

# forward pass

The gates we introduced above are relatively arbitrary. Any kind of differentiable function can act as a gate, and we can group multiple gates into a single gate, or decompose a function into multiple gates whenever it is convenient. Let's look at another expression that illustrates this point:
Suppose that we have a shallow two layer neural net (a.k.a regression)

The loss function is simply $L^2$, namely

$$l(W_1, W_2) = (W_2 \sigma(W_1 x) - y)^2$$

Need to know how to calculate $\sigma'$ and $\frac{\partial a}{\partial W}$, $\frac{\partial a}{\partial x}$, where $a = Wx$
Backprop for Classification

The loss function is now given by

$$l(F(x, \theta), y = k) = -\log \frac{\exp(f_k(x, \theta))}{\sum_{i=1}^{K} \exp(f_i(x, \theta))}$$

Useful identities $g(z) = e^z / (1 + e^z) = 1 / (1 + e^{-z})$

$$g'(z) = g(z)(1 - g(z))$$

$$(\log g(z))' = 1 - g(z) = -g(-z)$$

Check your gradients using finite difference!
Logistic Likelihood

\[ L(\theta|x) = Pr(Y|X; \theta) = \prod_i Pr(y_i|x_i; \theta) = \prod_i h_\theta(x_i)^{y_i} (1 - h_\theta(x_i))^{(1-y_i)} \]

Taking logs gives cross-entropy loss

\[ J(\theta) = -\sum_i (y_i \log(h_\theta(x_i)) + (1 - y_i) \log(1 - h_\theta(x_i))) \]

\[ \frac{\partial J(\theta)}{\partial \theta_j} = \sum_i x_j^{(i)} (h_\theta(x^{(i)}) - y^{(i)}) \]

Written in its vector form, the entire gradient can be expressed as:

\[ \nabla_\theta J(\theta) = \sum_i x^{(i)} (h_\theta(x^{(i)}) - y^{(i)}) \]
Multinomial Logistic regression

\[ p(y = k \mid x) = h_{\theta k}(x) = \frac{e^{\theta_k^T x}}{\sum_j e^{-\theta_j^T x}} \]

So \( h \) is a vector now

\[
h_{\theta}(x) = \begin{bmatrix} P(y = 1 \mid x; \theta) \\ P(y = 2 \mid x; \theta) \\ \vdots \\ P(y = K \mid x; \theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{K} \exp(\theta_j^T x)} \begin{bmatrix} \exp(\theta_1^T x) \\ \exp(\theta_2^T) \\ \vdots \\ \exp(\theta_K^T) \end{bmatrix}
\]
Multinomial Logistic regression

Using one-hot encoding, i.e. \( y = 1 \) becomes \( T = (1, 0, \ldots, 0) \), we have the likelihood

\[
p(T|\theta) = \prod_{i=1}^{n} \prod_{k=1}^{K} p(y = k|\theta_i)^{t_{ik}} = \prod_{i=1}^{n} \prod_{k=1}^{K} y_{ik}^{t_{ik}}
\]

Where \( T \) is \( n \times K \) matrix of target variables with elements \( t_{ik} \). The negative log-likelihood is then

\[
- \log p(T|\theta) = - \sum_{i=1}^{n} \sum_{k=1}^{K} t_{ik} \log y_{ik}
\]
Activation Functions

ReLU networks have become popular

- Sigmoid or tanh: Gradient is zero at $|x| \geq 6$ and maximum at 0.25
- Every time you multiply gradients they decrease! Vanishing gradients
- ReLU: has dead nodes (zero activations). Sparsity.
Cross Validation in DL

We split our training data into complementary subset to then conduct analysis and validation on different sets.

▶ Aims to reduce over-fitting and increase out-of-sample performance.

▶ Provides a tool to decide what levels of regularization lead to good generalization (i.e., prediction), which is the classic variance-bias trade-off.

▶ A key advantage of cross validation (over say $t$-ratios and $p$-values) is that it also allows us to assess the size and depth of the hidden layers (a.k.a. model selection).
Finding Parameters: Very Doable

Well, generally, we are screwed

In practice, local minima from **Stochastic Gradient Descent (SGD)** is good enough
Back-Propagation

Stochastic gradient descent adapted to a deep learning setting.

Proximal Newton Algorithm: $\nabla L$ available for deep learners.

One caveat of back-propagation is the multi-modality of the system to be solved (and the resulting slow convergence properties).

Deep learning methods heavily rely on the availability of large computational power: Theano and TensorFlow.
Tensor Processing Unit

- **The problem**: Deep Learning is typically applied to large datasets.
  - A driverless car processes 6GB data per second.
- Applications need computational speed
- **The solution**: A specialized processor called Tensor Processing Unit (TPU, GPU, CPU)
  - Processing advances tied to TPU not CPU
  - Google TPU 2.0 and Nvidia Tesla V100
Google’s TPU AI Chip

Google data center runs on 30,000 TPUs. The chip has reduced the need for servers.
NVIDIA’s CUDA is Still Ahead
Bayes, AI and Deep Learning: 41913

Week 8: Deep Learning
Architectures

Nick Polson

http://faculty.chicagobooth.edu/nicholas.polson/teaching/41913/
Models are getting larger!

Figure 1: Top1 vs. network. Single-crop top-1 validation accuracies for top scoring single-model architectures. We introduce with this chart our choice of colour scheme, which will be used throughout this publication to distinguish effectively different architectures and their correspondent authors. Notice that networks of the same group share the same hue, for example ResNet are all variations of pink.

Figure 2: Top1 vs. operations, size / parameters. Top-1 one-crop accuracy versus amount of operations required for a single forward pass. The size of the blobs is proportional to the number of network parameters; a legend is reported in the bottom right corner, spanning from $5 \times 10^6$ to $155 \times 10^6$ params. Both these figures share the same y-axis, and the grey dots highlight the centre of the blobs.

Single run of VGG-16 (Simonyan & Zisserman, 2014) and GoogLeNet (Szegedy et al., 2014) are $870\%$ and $10.07\%$ respectively, revealing that VGG-16 performs better than GoogLeNet. When models are run with 10-crop sampling, then the errors become $9.33\%$ and $9.15\%$ respectively, and therefore VGG-16 will perform worse than GoogLeNet, using a single central-crop. For this reason, we decided to base our analysis on re-evaluations of top-1 accuracies for all networks with a single central-crop sampling technique (Zagoruyko, 2016).

For inference time and memory usage measurements we have used Torch7 (Collobert et al., 2011) with cuDNN-v5 (Chetlur et al., 2014) and CUDA-v8 back-end. All experiments were conducted on a JetPack-2.3 NVIDIA Jetson TX1 board (nVIDIA): an embedded visual computing system with a 64-bit ARM R_A57 CPU, a 1 T-Flop/s 256-core NVIDIA Maxwell GPU and 4 GB LPDDR4 of shared RAM. We use this resource-limited device to better underline the differences between network architecture, but similar results can be obtained on most recent GPUs, such as the NVIDIA K40 or Titan X, to name a few. Operation counts were obtained using an open-source tool that we developed (Paszke, 2016). For measuring the power consumption, a Keysight 1146B Hall effect current probe has been used with a Keysight MSO-X 2024A 200 MHz digital oscilloscope with a sampling period of $2s$ and $50 kSa/s$ sample rate. The system was powered by a Keysight E3645A GPIB controlled DC power supply.

RESULTS

In this section we report our results and comparisons. We analysed the following DDNs: AlexNet (Krizhevsky et al., 2012), batch normalised AlexNet (Zagoruyko, 2016), batch normalised Network In Network (NIN) (Lin et al., 2013), ENet (Paszke et al., 2016) for ImageNet (Culurciello, 2016), GoogLeNet (Szegedy et al., 2014), VGG-16 and -19 (Simonyan & Zisserman, 2014), ResNet-18, -34, -50, -101 and -152 (He et al., 2015), Inception-v3 (Szegedy et al., 2015) and Inception-v4 (Szegedy et al., 2016) since they obtained the highest performance, in these four years, on the ImageNet (Russakovsky et al., 2015) challenge.

In the original paper this network is called VGG-D, which is the best performing network. Here we prefer to highlight the number of layer utilised, so we will call it VGG-16 in this publication.

From a given image multiple patches are extracted: four corners plus central crop and their horizontal mirrored twins.

Accuracy and error rate always sum to 100, therefore in this paper they are used interchangeably.
Deep Architectures

MLP

auto-encoder

convolution

recurrent

Long / short term memory

http://www.asimovinstitute.org/neural-network-zoo/
Convolutional NN Analyze data with spatial structure (image - 2D/3D, voice - 1D, video - 3D)

Local model via convolutions
Filtering in Machine Vision

Code for filter banks:

www.robots.ox.ac.uk/~vgg/research/texclass/filters.html
Convolution of Two Signals

\[
\begin{bmatrix}
-1 & 0 & 1 \\
-1 & 0 & 1 \\
-1 & 0 & 1 \\
\end{bmatrix}
\]
CNN

Source: http://cs231n.github.io/convolutional-networks/
Source: http://cs231n.github.io/convolutional-networks/
Fully Connected Layer

Example: 200x200 image
40K hidden units

- Spatial correlation is local
- Waste of resources + we have not enough training samples anyway..

Ranzato @ Facebook
Locally Connected Layer

Example: 200x200 image
40K hidden units
Filter size: 10x10
4M parameters

Note: This parameterization is good when input image is registered (e.g., face recognition).

Ranzato @ Facebook
Locally Connected Layer

**STATIONARITY?** Statistics is similar at different locations

Example: 200x200 image
- 40K hidden units
- Filter size: 10x10
- 4M parameters

Ranzato @ Facebook
Convolutional Layer

Share the same parameters across different locations (assuming input is stationary):

Convolutions with learned kernels

Ranzato @ Facebook
CNN by Layer

Matthew D. Zeiler and Rob Fergus
Conv layer implementation

\[
\text{out}(N_i, C_{out_j}) = \text{bias}(C_{out_j}) + \sum_{k=0}^{C_{in}-1} \text{weight}(C_{out_j}, k) \ast \text{input}(N_i, k)
\]

- It is a linear operation and can be implemented using matrix-matrix multiplication
- Very efficient implementations for NVIDIA (cuDNN) and Intel (MKL DNN)
- Vinogradov method for Fourier transforms
Gradient of the conv layer

Just a matrix-matrix gradient, we know how to do that!

\[
y = \begin{pmatrix} k_1 & k_2 & 0 & k_3 & k_4 & 0 & 0 & 0 & 0 \\
0 & k_1 & k_2 & 0 & k_3 & k_4 & 0 & 0 & 0 \\
0 & 0 & 0 & k_1 & k_2 & 0 & k_3 & k_4 & 0 \\
0 & 0 & 0 & 0 & k_1 & k_2 & 0 & k_3 & k_4 \\
\end{pmatrix} x
\]

\[
\frac{dL}{dx} = \begin{pmatrix} k_1 & k_2 & 0 & k_3 & k_4 & 0 & 0 & 0 & 0 \\
0 & k_1 & k_2 & 0 & k_3 & k_4 & 0 & 0 & 0 \\
0 & 0 & 0 & k_1 & k_2 & 0 & k_3 & k_4 & 0 \\
0 & 0 & 0 & 0 & k_1 & k_2 & 0 & k_3 & k_4 \\
\end{pmatrix}^T \frac{dL}{dy}
\]
ImageNet

- 1.2m images of 1000 classes
  - Labeled with Amazon MTurk
ImageNet Challenge

Models are getting better
AlexNet

Krizhevsky 2012

- Max-pooling
- ReLU nonlinearities
- Fully-connected layers
- Large data set and many parameters (60M)
- Data augmentation (flips and random samples)
- Dropout
- GPUs
- 1 week of training on 2 GPUs
VGG 2014

- Cascade of 3x3 convolutions (smaller number of parameters)
- Most of the parameters are in dense layers
- Total 140m parameters vs 60m in AlexNet
- Most of the computations happen in conv layers
- Vanishing gradient problem (multi-stage training with different depths)
- 4 tatan black GPUs for 2-3 weeks
GoogleNet (2014)

- Max depth is 22 parametrized layers
- No fully connected layers
- 1/12 of the parameters compared to AlexNet

Source: https://leonardoaraujosantos.gitbooks.io/artificial-inteligence/content/googlenet.html
GoogleNet Inception

Nonlinear architecture, used inception module (9 of them)

1x1 conv layers reduce dimensionality

- Very deep, cannot be jointly learned
- Same block “copy-pasted” in multiple places
- Used BatchNorm and Residual blocks
ResNet

Skip connection (He et al., 2016)

\[ a = f(x) + x \]

so that

\[ \frac{\partial l}{\partial x} = f'(x) \frac{\partial l}{\partial y} + \frac{\partial l}{\partial a} \]
ResNet

GoogLeNet
Ultra deep! More than 100 layers

ResNet
## ResNet Table

### Layer-by-layer specs

<table>
<thead>
<tr>
<th>layer name</th>
<th>output size</th>
<th>18-layer</th>
<th>34-layer</th>
<th>50-layer</th>
<th>101-layer</th>
<th>152-layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>conv1</td>
<td>112×112</td>
<td></td>
<td></td>
<td></td>
<td>7×7, 64, stride 2</td>
<td></td>
</tr>
<tr>
<td>conv2.x</td>
<td>56×56</td>
<td>[3×3, 64] ×2</td>
<td>[3×3, 64] ×3</td>
<td>[1×1, 64]</td>
<td>[3×3, 64] ×3</td>
<td>[1×1, 64]</td>
</tr>
<tr>
<td>conv3.x</td>
<td>28×28</td>
<td>[3×3, 128] ×2</td>
<td>[3×3, 128] ×4</td>
<td>[1×1, 128]</td>
<td>[3×3, 128] ×4</td>
<td>[1×1, 128]</td>
</tr>
<tr>
<td>conv4.x</td>
<td>14×14</td>
<td>[3×3, 256] ×2</td>
<td>[3×3, 256] ×6</td>
<td>[1×1, 256]</td>
<td>[3×3, 256] ×6</td>
<td>[1×1, 256]</td>
</tr>
<tr>
<td>conv5.x</td>
<td>7×7</td>
<td>[3×3, 512] ×2</td>
<td>[3×3, 512] ×3</td>
<td>[1×1, 512]</td>
<td>[3×3, 512] ×3</td>
<td>[1×1, 512]</td>
</tr>
<tr>
<td>1×1</td>
<td></td>
<td></td>
<td></td>
<td>7×7, 64, stride 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>average pool, 1000-d fc, softmax</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FLOPs</td>
<td>1.8×10⁹</td>
<td>3.6×10⁹</td>
<td>3.8×10⁹</td>
<td>7.6×10⁹</td>
<td>11.3×10⁹</td>
<td></td>
</tr>
</tbody>
</table>


ResNet: Simply adding the layers is not good enough
In ResNet there is not fixed width

Different paths from $x$ to $y$ of different length

DenseNet (Huang 2016) is an extreme case when all of the layers are connected

Number of parameters $\propto L^2$

Still smaller number of parameters compared to ResNet, thin layers!
Classification problem using large sets of labeled images

- Many good architectures
- Image as input
- Label as output
- Use logistic loss (cross-entropy)
- Start with convolution layers then fully connected layers
- Can re-use existing architectures. Transfer learning
Object Detection

Goal: find object on the image, e.g. identify corners of a bounding box
Earlier Methods Based on R-CNN

Girshick et al., 2013

R-CNN: Regions with CNN features

1. Input image
2. Extract region proposals (~2k)
3. Compute CNN features
4. Classify regions

- Take multiple guesses on where object is and use CNN to verify each guess
- Output is class label and estimated position
- Problem is imbalanced positions of objects on the image
- Used special loss function called focal loss
Divide and conquer: divide calculations of convolutions among the guess
**Regional proposal network**

Ren et al., 2015

Use neural network to propose a hypotheses (guess)
Liu et al., 2016

- Work in real time or faster!
- From pixels to coordinates of bounding boxes.
- “Our system divides the input image into an SxS grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object.”
Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts $B$ bounding boxes, confidence for those boxes,
Segmentation

Label each pixel
Use fully connected CNN

Long et al., 2015
Problem: low resolution of the output
Use fully connected CNN

Use upconv, dialated conv,
Use detection for segmentation

Loose exact position because of maxpool

Use smooth pooling

Bilinear interpolation on of the pixels on the boundary
Mask R-CNN

Mask R-CNN

[He et al., 2017]
Image retrieval

- Goal is to find similar images
- Face detection
- One approach is to represent an input image using a small vector and then do nearest neighbor search using some distance metric, e.g., Euclidean
- There are fast approximate search algorithms
- Can use undertrained networks
Siamese networks

- We use the same network and then calculate distance between representations
- How to we train those models?

Image from [Simo-Serra et al., 2015]
Siamese nets

- $y = 1$ for same pair
- $y = 0$ for different pairs

Then use contrastive loss

$$l(x_1, x_2) = y\|x_1 - x_2\|^2 + (1 - y) \max \left(0, \|x_1 - x_2\|^2\right)$$

$m$ is the margin that controls the contrast. Need to normalize inputs $x$. 

Image from [Simo-Serra et al., 2015]
Siamese nets

How to define a loss function?

Triples loss
Simo-Serra et al., 2015; Gordo et al., 2016

Source:
Object Tracking in Video

Idea is to use one of the branches of the siamese net in a convolution
Bertinetto et al., 2016
Object Tracking in Video

Idea is to use one of the branches of the siamese net in a convolution

Bertinetto et al., 2016
Video classification

Identify type of action in a video

Swinging, Diving, Kicking, Lifting, Horse Riding, Running, Skateboard, High-bar, Golf, Walking
Recurrent Neural Network (RNNs)
RNN Architecture

Same block “copy-pasted”

Given predictors $x_t$, the observed data $y_t$ and a hidden state $h_t$, then

\[
\begin{align*}
    y_t &= \sigma(W_1 h_t + b_z) \\
    h_t &= \sigma(W_2 [x_t, h_{t-1}] + b_h).
\end{align*}
\]

Here $\sigma(x) = 1/(1 + e^{-x})$ is the sigmoid function applied component-wise and is used for calculating both the hidden vector $h_t$ and the output vector $y_t$. 
Depending on the task, a different input-output architectures to be used

Input, hidden state, output

Credit: Andrej Karpathy
Vanishing/Exploding Gradients

\[ h_t = \sigma(W h_{t-1} + b_h) \]

\[ \frac{\partial l(h_3)}{\partial h_0} = J_{h_0 h_0}^T W^T \sigma'(h_3) \frac{\partial}{\partial h_3} \]

\[ = J_{h_1 h_0}^T W^T \sigma'(h_2) W^T \sigma'(h_3) \frac{\partial}{\partial h_3} \]

\[ = W^T \sigma'(h_1) W^T \sigma'(h_2) W^T \sigma'(h_3) \frac{\partial}{\partial h_3} \]

What happens when \( ||W|| \gg 1 \) or \( ||W|| \ll 1 \)?

- Gradient clipping is one way to solve the problem
- LSTM or GRU is another and more often used in practice
LSTM

The hidden state will be generated via another hidden cell state $c_t$ that allows for long term dependencies to be “remembered”. Then, we generate

Output: $h_t = o_t \star \tanh(c_t)$

$$k_t = \tanh(W_c[h_{t-1}, x_t] + b_c)$$

$$c_t = f_t \star c_{t-1} + i_t \star k_t$$

State equations:

$$\begin{pmatrix} f_t \\ i_t \\ o_t \end{pmatrix} = \sigma(W[h_{t-1}, x_t] + b).$$

Where $\star$ denotes point-wise multiplication. Then, $f_t \star c_{t-1}$ introduces the long-range dependence. The states $(i_t, f_t, o_t)$ are input, forget, and output states.
Bayes, AI and Deep Learning: 41913

Week 9 and 10 Deep Learning Applications

Nick Polson

http://faculty.chicagobooth.edu/nicholas.polson/teaching/41913/
Pattern Matching: Mammograms

Pattern matching is a powerful empirical tool

- Original findings were done by looking at data!!
- Side-by-side images with and without breast cancer.
- To the human eye what’s different? White blobs.

Nowadays done by deep learning ...

NPR Article on Training A Computer To Read Mammograms As Well As A Doctor
Google’s AI: Music like Bach

Google has launched its first ever AI-powered Doodle. All you have to do is plunk down a few notes and the Doodle will add harmony to them in the signature style of Bach.

In this case, 306 chorale harmonizations composed by Bach were fed into a model. His chorales make for great training data because their structure is pretty consistent and concise – they all contain four voices, which take on a pleasing depth when layered on top of one another. After the model “learned” Bach’s style by picking out the patterns, the machine learning system was refitted to run within the confines of your humble web browser.
Google’s AI: Music like Bach

Try Yourself!

Harmonizing...

Machine learning models need time to learn how to find the right patterns. This machine trained for a few hours, despite the relatively small dataset of 306 Bach compositions.
Application: Training A New Rembrandt

Analyze all 346 of Rembrandt’s paintings

- Identify all geometric patterns used by Rembrandt.
- Reassemble into a fully formed face and bust

Nvidia Faces
Examples: Robotics

Two classic examples

- Rory and the Robot (Rory vs The Robot)
- Robot Back-flip (Be very afraid ... robots can now do backflips)

Key idea: train and fail many-many times until you “learn”

Deep Learning
Example: ChatBots and DeepFakes

AI can create speech and video content that humans cannot. You won’t believe what Obama says in this video!

You cannot distinguish human voice from the synthesized one: Google WaveNet

China’s AI news anchor

AI can talk to you and you won’t even know it: Silicon Valley: Gilfoyle Made A Bot (Season 6 Episode 1)
Which Person is Real?

Click on the person who is real.
Silicon Valley: Season 4 Episode 4: https://youtu.be/ACmydtFDTGs
David Bowie used machine learning (ML) to write his songs. Bowie comments that the questionable quality of the lyrics from his old band Tin Machine should be “blamed” on the computer.

Verbasizer automates a technique used by Bowie to write songs

Form random phrases based on shuffled word cut-outs from newspapers and other sources.
NLP: Watson and Jeopardy

Jeopardy

- IBM super computer Watson beat two former champions of TV game Jeopardy and took home one million dollars prize. Watson is a significant leap a machine’s ability to understand context in human language.

- IBM believes the technology behind Watson can be applied to a variety of fields, most notably medicine.
Image Processing: MNIST

Hand-written digits.
Multi-layer fully-connected neural network.
Convolution neural network
## Classification Dataset Results

### MNIST

**who is the best in MNIST?**

MNIST 50 results collected

Units: error %

Classify handwritten digits. Some additional results are available on the [original dataset page.](#)

<table>
<thead>
<tr>
<th>Result</th>
<th>Method</th>
<th>Venue</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.21%</td>
<td>Regularization of Neural Networks using DropConnect</td>
<td>ICML 2013</td>
<td></td>
</tr>
<tr>
<td>0.23%</td>
<td>Multi-column Deep Neural Networks for Image Classification</td>
<td>CVPR 2012</td>
<td></td>
</tr>
<tr>
<td>0.23%</td>
<td>APAC: Augmented PAttern Classification with Neural Networks</td>
<td>arXiv 2015</td>
<td></td>
</tr>
<tr>
<td>0.24%</td>
<td>Batch-normalized Maxout Network in Network</td>
<td>arXiv 2015</td>
<td>Details</td>
</tr>
<tr>
<td>0.29%</td>
<td>Generalizing Pooling Functions in Convolutional Neural Networks: Mixed, Gated, and Tree</td>
<td>AISTATS 2016</td>
<td></td>
</tr>
<tr>
<td>0.31%</td>
<td>Recurrent Convolutional Neural Network for Object Recognition</td>
<td>CVPR 2015</td>
<td></td>
</tr>
<tr>
<td>0.31%</td>
<td>On the Importance of Normalisation Layers in Deep Learning with Piecewise Linear Activation Units</td>
<td>arXiv 2015</td>
<td></td>
</tr>
</tbody>
</table>
DL Model

Dataset contains 60k training observations and 10k samples out-of-sample performance (validation). 2-hidden layers with ReLU activation function

\[
Y = \text{softmax}(W^3 \tilde{Y}^{(3)} + b^3)
\]

\[
\tilde{Y}^{(3)} = D^{(3)} \ast Z^2
\]

\[
D^{(3)} \sim \text{Ber}(0.5)
\]

\[
Z^2 = \max(W^2 \tilde{Y}^{(2)} + b^1, 0)
\]

\[
\tilde{Y}^{(2)} = D^{(2)} \ast Z^1
\]

\[
D^{(2)} \sim \text{Ber}(0.5)
\]

\[
Z^1 = \max(W^1 \tilde{Y}^{(1)} + b^1, 0)
\]

\[
\tilde{Y}^{(1)} = D^{(1)} \ast X, \quad X \in \mathbb{R}^{1024}
\]

\[
D^{(1)} \sim \text{Ber}(0.5)
\]

Cross-entropy–negative log-likelihood–as loss function to be optimized.
SGD and optimisation

Apply mini-batch SGD algorithm with Nesterov acceleration (momentum) with the following parameters

- initial learning rate = 0.01
- learning decayed each time validation performance stalls (divided by 2)
- momentum of 0.9
- batch size of 10
- L2 weight decay / Gaussian prior on all parameters = 1e-5
Examples: Image Classification

Pattern: relationship between the visual features and its class.

- Cucumber sorter of images to sort them into nine different classes.
- Toilet paper assignment at the Temple of Heaven in Beijing
- Identification of untagged friends on Facebook
- Detection collisions between subatomic particles at CERN

Ultimately, computers are agnostic about the type of input you give them, because to a computer, it’s all just numbers.
<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Waveform" /></td>
<td>Speech to text: “Chi-ca-go hot-dog.”</td>
</tr>
<tr>
<td>68F/20C, 70% humidity, mostly sunny</td>
<td>Numerical prediction: “Power consumption in London will be 25,500 megawatt-hours.”</td>
</tr>
</tbody>
</table>
Example: Google Search: BERT

Google is using deep learning to pre-process your search query. Bidirectional Encoder Representations from Transformers

Source: Understanding searches better than ever before
AI in China

Xinhua is a China’s state news agency uses virtual anchor

▶ Sogou, the anchor was designed to simulate human voice, facial expressions and gestures

▶ AI to measure level of focus and engagement among students
Shoppers lined up for identification checks outside the Kashgar Bazaar last fall. Members of the largely Muslim Uighur minority have been under Chinese surveillance and persecution for years. Credit. Paul Mozur

Source: “One Month, 500,000 Face Scans: How China Is Using A.I. to Profile a Minority” by NYT
SenseFace: AI for Profiling Uighurs in China

- Law enforcement in the central Chinese city of Sanmenxia screened whether residents were Uighurs 500,000 times in 1 month.
- Law enforcement from the central province of Shaanxi, requested a smart camera system last year that “should support facial recognition to identify Uighur/non-Uighur attributes”.

Facial Recognition is becoming a commodity. Can buy from IBM: Attribute detection with Body Camera Analytics.
Amazon Go Stores

- Fully automated. All of your actions are tracked using CNNs
- Items in your cart are charged to your credit card when you step out.

In Chicago since 2019

Amazon Go Video
Makoto Koike sorts cucumbers at his parents farm. 9 classes of cucumbers.
His mother spent up to eight hours per day at peak harvesting times.
Learning to sort cucumbers can take months.
Size, thickness, color, texture, scratches, crooked or have prickles.

- Data 7000 pictures of cucumbers sorted by his mother. What are the important “features”?
- Recognition accuracy exceeded 95% for test images, out-of-sample the accuracy drops to about 70%.

Farming and TensorFlow
Traditional hiring is biased.

▶ The deepest-rooted source of bias in AI is the human behavior it is simulating.

▶ Current hiring process leads to significant unconscious bias against women, minorities and older workers.

▶ Often only applicants to be considered are those coming from Ivy League campuses, passive candidates from competitors (HBR article)

Can AI eliminate unconscious human bias and accelerate the hiring process?
Smart cities. Copenhagen bicycles.

Copenhagen wants to cut bus travel times by 5 to 20 percent and cycling travel times by 10 percent. Reduce the number of times cyclists have to stop by 10 percent.

To better manage the traffic, there’s an AI that identifies cars/bicyclists and gives priority to cyclists at morning peak hours.

Installing 380 “intelligent traffic signals” that will spot, and prioritize, buses and bikes.
AI Diagnostics

Artificial intelligence-powered device that detects cancer cells

- Hundreds of times faster than previous methods.
- Can extract cancer cells from blood immediately after they are detected
- Larry Ellison claims it is the future (video)

Source: Deep learning enables scientists to identify cancer cells in blood in milliseconds
In effect, word2vec has learned to take the SAT Verbal test using only skills from the SAT Math test.

- Which hockey teams play in which cities: Canadiens - Montreal + Toronto. Word2vec’s answer: “Maple Leafs.”
Word vectors provide a clear-cut mathematical description of something that to any human listener seems simple: sometimes one word fits better, and sometimes the other.
Word2Vec

On the third day, he [rows / rose / telephoned... ] from the dead.

He planted 100 [ears / rows / rose... ] of corn.
Probability: 41913

Week 9 and 10: AI and Deep Learning

Nick Polson
The University of Chicago Booth School of Business
http://faculty.chicagobooth.edu/nicholas.polson/teaching/41913/
Deep Learning: Introduction

Deep Learning is the most widely used machine learning tool for high dimensional input-output problems

- Image Recognition
- Google Translate
- Driverless Cars

The applications are endless ....

Sir, you complain a machine can’t do something. If you can define the task, I can build a machine to do it

John von Neumann
Area of Applications

- Image recognition: Healthcare, MNIST
- Natural language processing: Watson
- Sports
- Business
- ...

Kolmogorov-Arnold

There are no multivariate functions just superpositions of univariate ones

Let $f_1, \ldots, f_L$ be given univariate activation functions. We set

$$f_l^{W, b} = f_l \left( \sum_{j=1}^{N_l} W_{lj} X_j + b_l \right) = f_l(W_l X_l + b_l), \quad 1 \leq l \leq L,$$

Deep predictor has hidden units $N_l$ and depth $L$.

$$\hat{Y}(X) = F(X) = \left( f_1^{W_1, b_1} \circ \ldots \circ f_L^{W_L, b_L} \right)(X)$$
Deep Architectures: TensorFlow

Many possible superpositions of univariate semi-affine functions

Auto-encoder    Recurrent    Long Short Term Memory

http://www.asimovinstitute.org/neural-network-zoo/
### Deep Learning Recovers Patterns

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pixels: <img src="image1" alt="Image" /></td>
<td>“lion”</td>
</tr>
<tr>
<td>Audio: <img src="image2" alt="Image" /></td>
<td>“see at tuhl res taur aun ts”</td>
</tr>
<tr>
<td><code>&lt;query, doc&gt;</code></td>
<td>P(click on doc)</td>
</tr>
<tr>
<td>“Hello, how are you?”</td>
<td>“Bonjour, comment allez-vous?”</td>
</tr>
<tr>
<td>Pixels: <img src="image3" alt="Image" /></td>
<td>“A close up of a small child holding a stuffed animal”</td>
</tr>
</tbody>
</table>

#### Matrix Algebra and Automatic Differentiation
TPU Advantages for real-time data analysis

- Matrix analysis programmed into chips
- XLA: Accelerated Linear Algebra
- AD: Automatic Differentiation

Keypoint detection ➔ Extract SIFT descriptors ➔ Classification

Prediction = Border Collie!
Image recognition has improved

Machines are becoming better than humans
Natural Language Revolution

How words become numbers:

1. numerical representation for words:

<table>
<thead>
<tr>
<th></th>
<th>Animal</th>
<th>Agreeable</th>
<th>Growls or grunts</th>
<th>Talks</th>
<th>Lives in London</th>
<th>Is a bear</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scrooge</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

2. word co-location statistics: “ketchup”, “fries”, “bun”

3. vector operation (word2vec): queen = king - man + woman

4. speech recognition: mathematical language provides crucial tie-breaking information for homophones
Natural Language Processing (NLP)

Use word2vec combined with classification model (inputs = word vectors) to solve different problems

► Sequence tagging: automated analysis of documents (extract item names, shipping address,... )
► Text classification: sentiment analysis (is this article negative about my product?)
► Seq2seq: machine translation, chat bots, summarization (law)

Example: conversational AI by NVIDIA
Example: Can a Machine Learn to Write for The New Yorker?
Train the model by asking it to fill-in the blanks

**Original Text**

Former President Barack Obama returned to his academic roots on Tuesday, visiting the University of Chicago campus where he spent 12 years, and the neighborhood where his life last held some semblance of normalcy.

**Masked Words**

Former ________ Barack Obama returned to his academic ________ on Tuesday, visiting the ____________ of Chicago campus where he spent 12 years, and the neighborhood where his ____ last held some semblance of normalcy.
Use pre-trained model for your task

▸ Train a model by asking it to fill-in the blanks on large corpuses of text (all digitized books, text from web pages, twitts and FB posts, Google queries,...)

▸ Fine tune this model on the individual task with small amounts of data

▸ No need to have large datasets!

▸ Some existing pre-trained models: ULMFiT, Transformer, Google’s BERT, Transformer-XL, OpenAI’s GPT-2, word2vec
The Robotics Revolution

▶ In the 1950s, the state of the art was Theseus, a life-size autonomous mouse built by Claude Shannon at Bell Labs, and powered by a bank of telephone relays. He would navigate by trial and error until he found the cheese.

▶ In the 1960s and ’70s, there was the Stanford Cart: a wagon-sized with four small bicycle wheels, an electric motor, and a single TV camera. The Cart could steer itself across a chair-filled room in five hours, without human intervention.

▶ Today? Self-driving cars are just the start.
The question is: Where am I?

In AI, this is called the SLAM problem, for “simultaneous localization and mapping.” The word “simultaneous” is key here.

Whether you’re a person or a robot, knowing where you are means doing two things at once:

1. constructing a mental map of an unknown environment
2. inferring your own unknown location within that environment

The solution to both problems involved Bayes’s rule.
Waymo: 9 billion hours of autonomous driving simulations

Alphabet-owned Waymo uses AI to run simulations in gaming-like environments. These simulations are being used to inform their autonomous vehicle efforts and the company has amassed an impressive 5 billion hours in edge case scenarios.

These edge cases are the really tricky 1% or 0.1% or even 0.001% of possible interactions that could occur when you vehicles moving through complex environments.

While Waymo have some actual cars, it’s clear their emphasis on safe simulations digitally, is the complete opposite to Tesla’s strategy.

The big difference is that Tesla is deploying Autopilot into real customer vehicles and taking the data from events where the humans have taken over, therefore identifying an edge case that wasn’t already accommodated in the system.
Which restaurants have the longest parking times and why?

How long does it take for the delivery-partner to walk to the restaurant?

Does a restaurant have a difficult pickup process for delivery-partners?

When should we dispatch the delivery-partner? Does the delivery-partner have to wait for the food? Is the food waiting for the delivery-partner?

Did the delivery-partner have trouble with a delivery?
Uber

Android activity recognition

Raw data collection on phone to processing it as part of a batch pipeline.
Machine learning (ML) aimed at improving the product experience for the delivery-partner and eater by accurately calculating delivery times
Sequence models are capable of finding the delivery-partner change-points amongst a sequence of state observations. These observations consist of activities or activities fused with other modalities, such as GPS and motion sensors.
The additional detail achieved in our Uber Eats Trip State Model

The ML model gives an in-depth look at how an Uber Eats trip proceeds, allows to control dispatch time
Google Energy: Data Center Cooling Costs Reduced by 40%

Monitoring real-time conditions and adjusting data center climate control based on past experience
Google Verily: Identifying Skin Cancer

- Dataset: 130,000 images of skin lesions/2,000 different diseases
- Test data: 370 high-quality, biopsy-confirmed images
- Better performance than 23 Stanford dermatologists
- 10,000 hours no match for deep learning and large datasets
Google’s AI: Heart disease from eye scan

Google’s Verily scans of the back of a patient’s eye able to accurately deduce individual’s age, blood pressure, and whether or not they smoke, etc.

Predict their risk of suffering a major cardiac event—such as a heart attack—with roughly the same accuracy as current leading methods.

Quicker to analyze a patient’s cardiovascular risk—doesn’t require a blood test. Training 300,000 patients. Eye scans plus general medical data. Deep learning to mine for patterns
Google’s Verily: Diabetic Retinopathy

Diabetic Retinopathy is the fastest growing cause of preventable blindness!

Two images of the fundus—interior rear of your eye.

The left is a regular image; the right shows how Google’s algorithm picks out blood vessels (in green) to predict blood pressure.

The DL algorithm outperforms ophthalmologists 82 vs 84

Source: https://blog.verily.com/2018/02/eyes-window-into-heart-health.html
Intelligent Scanning Using Deep Learning for MRI

The scan operator first acquires a set of low-resolution “localizer” images from which approximate location and orientation of desired landmarks can be identified. These anatomical references are then used to manually plan the exact locations, orientation, and required coverage for images that will be used for the high-resolution scans that are used for diagnosis.

To aid the scan operator we developed a deep-learning (DL) based framework for intelligent MRI slice placement (ISP) for several commonly used brain landmarks. TensorFlow library with the Keras interface is used to implement the DL based framework for ISP.

As compared to the classical approaches, a DL-based approach is less affected by factors that affect MRI image quality or appearance. And it can be easily extended across other anatomies.
Chester: Chest X-Ray Disease Prediction System
Google Verily: Gleason grading of prostate cancer

An Augmented Reality Microscope for Cancer Detection

Camera capture of current field of view

View seen by user

Output to accelerated compute unit

Augmented reality display

AI algorithm

Image Analysis: Gleason grading of prostate cancer

Deep learning outperforms pathologists

Enterprise AI

AI solutions slowly move from university/tech company labs to the rest of the world. Many solutions offered by tech companies to non-tech businesses. Some examples

- AI Data Hub by Pure Storage
- RAPIDS by NVIDIA. Walmart case study.
Human and Machines, Talking Together

A few likely trends stand out:

▶ Language models will become personalized.
   The machines around you will adapt to the way you speak, just as they adapt to your movie-watching preferences

▶ Good policy and thoughtful regulations will be hugely important.
   An algorithm that can write an episode of Friends seems cute, if a bit useless. That same algorithm will seem a lot more pernicious when someone can program it to flood the internet with fake news around election time.
As the number and kinds of available experiences on TripAdvisor grew rapidly in the last couple of years, a personalized website can increase user satisfaction significantly by providing travelers with an easy way to find experiences that are relevant to them.

► Approach

1. Training Data Collection
2. Entity Embeddings
3. RFY Model Architecture

► Result Analysis

1. Offline Evaluation
2. Online A/B test
Sports Analytics: Learning Player Trajectories: NHL, NBA and EPL

Characteristics of group dynamics from their trajectories alone.

![Trajectory Network](image)

player 1 is trying to pass the puck
player 2 is going to block player 1
Racehorse Big Data Unlocks the Formula for Human Superathletes

Jeff Seder had spent decades and millions of dollars collecting a huge database of physiological and biological data in an effort to discover which traits corresponded most closely with greatness. He pioneered portable ultrasound device that allowed him to examine horses on the inside and scanned tens of thousands of animals. It was only through this that he came to the conclusion that one of the most important data points in selecting a horse is the size of its heart – and American Pharoah had a huge one.

The wisdom of big data is increasingly being applied to human competition, although the field most certainly has room to grow. It will likely change not only how athletes perform, but how they look, as the tools and technologies of sports medicine become more and more sophisticated.
Homecourt Can Help You Improve Your Jump Shot

HomeCourt is a basketball training app that uses deep learning to record, track, and chart shots for basketball players in real time. Using NVIDIA Tesla P100 GPUs on the Google Cloud, with the cuDNN-accelerated TensorFlow and Keras deep learning framework, the team trained their neural network on hours of basketball footage they filmed in local Bay Area high schools.

Not only can you review their workout videos with instant stat analysis but also workouts from other players and engage, and interact with the broader basketball community through basketball on a device they have with them all the time.
No sensors.
HomeCourt is ready to use with iPhone for real-time shot tracking and analysis.

Interactive drills.
Your phone is transformed into your own virtual on-demand skills coach, with mobile AI-powered assessment tools.

Shot Science.
HomeCourt’s proprietary Shot Science technology provides meaningful insights for every shot you take.

Analyze every shot.
With data tracked from >10M shots, >9M dribbles from >150 countries.
Artificial Intelligence in Formula 1

Formula One

- Strategy teams at the race track and at the team’s HQ are constantly trying to predict the next best optimal move to improve their drivers’ positions.
- Teams are limited to 60 data scientists AI (a.k.a machine learning/deep learning) provides better predictions of when best to stop, when to change tyres, overtake, ...
- Best strategies can vary quickly from moment to moment.
Robot Shoot Hoops

This basketball-loving robot by Toyota played against pro ballers in a shooting competition. According to its creators, the bizarre-looking humanoid robot called ‘Cue’ learned how to shoot thanks to artificial intelligence.
Traffic Prediction

Google maps real-time travel predictions ...

- Path search algorithms to calculate fastest route
When Iowa’s snow piles up, TensorFlow can keep roads safe

To improve road safety and efficiency, the Iowa Department of Transportation has teamed up with researchers at Iowa State University to use machine learning, including our TensorFlow framework, to provide insights into traffic behavior. Iowa State’s technology helps analyze the visual data gathered from stationary cameras and cameras mounted on snow plows.

They also capture traffic information using radar detectors. Machine learning transforms that data into conclusions about road conditions, like identifying congestion and getting first responders to the scenes of accidents faster..

In California, snow may not be an issue, but traffic certainly is, and college students there used TensorFlow to identify pot holes and dangerous road cracks in Los Angeles.
Hans Moravec and Stanford Cart

- KL10 processor, at about 2.5 MIPS, Moravec was eventually able to use multi-ocular vision to navigate slowly around obstacles in a controlled environment. The cart moved in one meter spurts punctuated by ten to fifteen minute pauses for image processing and route planning. In 1979, the cart successfully crossed a chair-filled room without human intervention in about five hours.
Based on the new capabilities of Autopilot under version 9, the new computer vision neural net had to be significantly updated.

It can now track vehicles and other objects all around the car – meaning that it makes better use of the 8 cameras around the car and not just the front-facing ones.

Scaling computational power, training data, and industrial resources plays to Tesla’s strengths and involves less uncertainty than potentially more powerful but less mature techniques.
Automated Rotterdam Port

Rotterdam Port is one of the most automated ports and one of the largest ports in the world.

Automated container carriers are completely computer controlled, carrying containers to cranes. Meanwhile, the cranes are human controlled and move the containers to the ship.

With the fully automated cranes, the terminal can be run by a team of no more than 10 to 15 people on a day-to-day basis.
Automated Port: Port of Qingdao

Qingdao

- Port of Qingdao is the first automated container terminal in Asia. The terminal is called a “ghost port” since it is all controlled by AI and no workers found in sight.

- Through laser scanning and positioning, the program is able to locate the four corners of each container. It accurately grabs them and puts them onto the driverless trucks. And it is capable to work in complete darkness. The smart autopilot trucks, driven by electricity, have their routes and tasks under digital control. They even know when it’s time to go for a recharge.
Rio Tinto Mining Automation to Boost Efficiency

Rio Tinto

► 73 self-driving trucks that reportedly haul payloads at a cost 15 percent less than those operated by human drivers. In addition to the trucks, they also have robotic, rock-drilling rigs plugging away at the topography. In the near future, Rio Tinto is looking to upgrade the trains that haul the ore to port to not only drive themselves but also have the ability to load and unload automatically.

► 15% reduction in the cost of operating the automated trucks compared to those driven by humans, as hauling is among the largest costs to a mining operation.
Applying deep learning to Related Pins

One of the most popular ways people find ideas on Pinterest is through Related Pins, an item-to-item recommendations system that uses collaborative filtering.

Previously, candidates were generated using board co-occurrence, signals from all the boards a Pin is saved to. Now, for the first time, Pinterest is applying deep learning to make Related Pins even more relevant. Ultimately, they developed a scalable system that evolves with their product and people’s interests, the most relevant recommendations can surface through Related Pins.

Pin2Vec is built to embed all the Pins in a 128-dimension space. Pin tuples are used in supervised training to train the embedding matrix for each of the tens of millions of Pins of the vocabulary. TensorFlow is used as the trainer. At serving time, a set of nearest neighbors are found as Related Pins in the space for each of the Pins.
Recommendation Systems: YouTube

YouTube is experimenting with ways to make its algorithm even more addictive
Recommendation Systems: YouTube

“Recommendation algorithms are some of the most powerful machine-learning systems today because of their ability to shape the information we consume”

Goal is to maximize amount of time spent watching

Confirmation bias: can create an addictive experience that shuts out other views.
Recommendation Systems: YouTube Algorithm

1. Compiles a shortlist of several hundred videos by finding ones that match the topic and other features of the one you are watching

2. Ranks the list according to the user’s preferences, which it learns by feeding all your clicks, likes, and other interactions into a machine-learning algorithm

The effect is that over time, the system can push users further and further away from the videos they actually want to watch.

YouTube’s algorithms created an isolated far-right community, pushed users toward videos of children, and promoted misinformation (online extremism)
AI for Personalized Music

Inputs are auditory features extracted from a composition (dynamics, timbre, harmony, rhythm). Total of 74 inputs
Outputs are brain activity + subjective description of the listener. Used unpopular songs for training. To avoid confounding. Used equal number of “happy” and “sad songs”

Multimodal View into Music’s Effect on Human Neural, Physiological, and Emotional Experience
a) In this study population, pioglitazone does not appear to be significantly associated with an increased risk of bladder cancer in patients with type 2 diabetes.
b) The use of pioglitazone is associated with an increased risk of incident bladder cancer among people with type 2 diabetes.
Data Team vs Business Experts

At Netflix, Who Wins When It’s Hollywood vs. the Algorithm?

As the company plunges deeper into originals, its L.A. wing is doing the once-unthinkable: overriding the metrics

By Shalini Ramachandran and Joe Flint
Nov. 10, 2018 12:00 am ET

▶ The Netflix War: How to promote “Grace and Frankie”?  
▶ Product team only included Ms. Fonda’s co-star, Lily Tomlin.  
▶ Tests showed that more users clicked on the show when the photo didn’t include Ms. Fonda.  
▶ Could violate the contract!

Vulture’s Inside the Binge Factory
Cindy Holland and Ted Sarandos, who are in charge of content at Netflix: It’s 70 percent gut and 30 percent data
Apple reportedly buys music platform “able to find the next Justin Bieber”

[Photo: Daniel Cañibano/Unsplash]
In financial and policy applications interoperability is important. Cost of a mistake is high.

Explanable AI Article “A good deep learning approach could give us more comfort that we know what’s happening in the system than having 1000 of these human-created rules, created over decades.”
More Booth Stats!

Lots more to do:

41201: Big Data

41204: Machine Learning

HAVE FUN!!!