Statistical Learning with Big Data

Trevor Hastie
Department of Statistics
Department of Biomedical Data Science
Stanford University

Thanks to Rob Tibshirani for some slides
Some Take Home Messages

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

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

**Machine Learning** constructs algorithms that can learn from data.

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

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

All of these are very similar — with different emphases.
Some Definitions

**Machine Learning** constructs algorithms that can learn from data.

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

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

All of these are very similar — with different emphases.

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

Sexiest man alive?
Sexiest man alive?
Sexiest man alive?
Sexiest man alive?
Sexiest man alive?
Sexiest man alive?
Sexiest man alive?
Sexiest man alive?
Sexiest man alive?
Sexiest man alive?
Sexiest man alive?
The Supervising Learning Paradigm

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

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

- **IMPORTANT!** Don’t trust me or anyone who says they have a wonderful machine learning algorithm, unless you see the results of a careful internal validation.

- Eg: divide data into two parts $A$ and $B$. Run algorithm on part $A$ and then test it on part $B$. Algorithm must not have seen any of the data in part $B$.

- If it works in part $B$, you have (some) confidence in it.
Internal Model Validation

- **IMPORTANT!** Don’t trust me or anyone who says they have a wonderful machine learning algorithm, unless you see the results of a careful internal validation.

- Eg: divide data into two parts A and B. Run algorithm on part A and then test it on part B. Algorithm must not have seen any of the data in part B.

- If it works in part B, you have (some) confidence in it

  Simple? Yes
Internal Model Validation

• IMPORTANT! Don’t trust me or anyone who says they have a wonderful machine learning algorithm, unless you see the results of a careful internal validation.

• Eg: divide data into two parts $A$ and $B$. Run algorithm on part $A$ and then test it on part $B$. Algorithm must not have seen any of the data in part $B$.

• If it works in part B, you have (some) confidence in it

  Simple? Yes
  Done properly in practice? Rarely
Internal Model Validation

• **IMPORTANT!** Don’t trust me or anyone who says they have a wonderful machine learning algorithm, unless you see the results of a careful internal validation.

• Eg: divide data into two parts $A$ and $B$. Run algorithm on part $A$ and then test it on part $B$. Algorithm must not have seen any of the data in part $B$.

• If it works in part $B$, you have (some) confidence in it

Simple? Yes
Done properly in practice? Rarely

*In God we trust. All others bring data.*
Big data vary in *shape*. These call for different approaches.

**Wide Data**

Thousands / Millions of Variables  
Hundreds of Samples  
Screening and fdr, Lasso, SVM, Stepwise

**Tall Data**

Tens / Hundreds of Variables  
Thousands / Millions of Samples  
GLM, Random Forests, Boosting, Deep Learning

Sometimes simple models (linear) don’t suffice.  
We have enough samples to fit nonlinear models with many interactions, and not too many variables.  
Good automatic methods for doing this.
Big data vary in *shape*. These call for different approaches.

**Tall and Wide Data**

- Thousands / Millions of Variables
- Millions to Billions of Samples

**Tricks of the Trade**

- Exploit sparsity
- Random projections / hashing
- Variable screening
- Subsample rows
- Divide and recombine
- Case/ control sampling
- MapReduce
- ADMM (divide and conquer)
Big data vary in *shape*. These call for different approaches.

**Tricks of the Trade**

- Exploit sparsity
- Random projections / hashing
- Variable screening
- Subsample rows
- Divide and recombine
- Case/ control sampling
- MapReduce
- ADMM (divide and conquer)
- join Google
Examples of Big Data Learning Problems

Click-through rate. Based on the search term, knowledge of this user (IPAddress), and the Webpage about to be served, what is the probability that each of the 30 candidate ads in an ad campaign would be clicked if placed in the right-hand panel. Logistic regression with billions of training observations. Each ad exchange does this, then bids on their top candidates, and if they win, serve the ad — all within 10ms!
Examples of Big Data Learning Problems

Click-through rate. Based on the search term, knowledge of this user (IP Address), and the Webpage about to be served, what is the probability that each of the 30 candidate ads in an ad campaign would be clicked if placed in the right-hand panel.
Examples of Big Data Learning Problems

Click-through rate. Based on the search term, knowledge of this user (IPAddress), and the Webpage about to be served, what is the probability that each of the 30 candidate ads in an ad campaign would be clicked if placed in the right-hand panel. Logistic regression with billions of training observations. Each ad exchange does this, then bids on their top candidates, and if they win, serve the ad — all within 10ms!
Examples of Big Data Learning Problems

Recommender systems. Amazon online store, online DVD rentals, Kindle books, ...
Based on my past experiences, and those of others like me, what else would I choose?
Recommender systems. Amazon online store, online DVD rentals, Kindle books, ...
Examples of Big Data Learning Problems

Recommender systems. Amazon online store, online DVD rentals, Kindle books, ...
Based on my past experiences, and those of others like me, what else would I chose?
Examples of Big Data Learning Problems

- **Adverse drug interactions.** US FDA (Food and Drug Administration) requires physicians to send in adverse drug reports, along with other patient information, including disease status and outcomes. Massive and messy data.

- **Social networks.** Based on who my friends are on Facebook or LinkedIn, make recommendations for who else I should invite. Predict which ads to show me. There are more than a billion Facebook members, and two orders of magnitude more connections. Knowledge about friends informs our knowledge about you. Graph modeling is a hot area of research. (e.g. Leskovec lab, Stanford CS.)
Examples of Big Data Learning Problems

- **Adverse drug interactions.** US FDA (Food and Drug Administration) requires physicians to send in adverse drug reports, along with other patient information, including disease status and outcomes. Massive and messy data. Using natural language processing, Stanford BMI researchers (Altman lab) found drug interactions associated with good and bad outcomes.
Examples of Big Data Learning Problems

- **Adverse drug interactions.** US FDA (Food and Drug Administration) requires physicians to send in adverse drug reports, along with other patient information, including disease status and outcomes. Massive and messy data. Using natural language processing, Stanford BMI researchers (Altman lab) found drug interactions associated with good and bad outcomes.

- **Social networks.** Based on who my friends are on Facebook or LinkedIn, make recommendations for who else I should invite. Predict which ads to show me.
Examples of Big Data Learning Problems

- **Adverse drug interactions.** US FDA (Food and Drug Administration) requires physicians to send in adverse drug reports, along with other patient information, including disease status and outcomes. Massive and messy data. Using natural language processing, Stanford BMI researchers (Altman lab) found drug interactions associated with good and bad outcomes.

- **Social networks.** Based on who my friends are on Facebook or LinkedIn, make recommendations for who else I should invite. Predict which ads to show me. There are more than a billion Facebook members, and two orders of magnitude more connections. Knowledge about friends informs our knowledge about you. Graph modeling is a hot area of research. (e.g. Leskovec lab, Stanford CS.)
The Netflix Recommender

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

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

Never   Sometimes   Often
41K teams participated! Competition ran for nearly 3 years. Winner “BellKor’s Pragmatic Chaos”, essentially tied with “The Ensemble”.

41K teams participated! Competition ran for nearly 3 years. Winner “BellKor’s Pragmatic Chaos”, essentially tied with “The Ensemble”. ➤ our Lester Mackey ➤
The Netflix Data Set

<table>
<thead>
<tr>
<th></th>
<th>movie I</th>
<th>movie II</th>
<th>movie III</th>
<th>movie IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>User A</td>
<td>1</td>
<td>?</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>User B</td>
<td>?</td>
<td>2</td>
<td>3</td>
<td>?</td>
</tr>
<tr>
<td>User C</td>
<td>4</td>
<td>1</td>
<td>2</td>
<td>?</td>
</tr>
<tr>
<td>User D</td>
<td>?</td>
<td>5</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>User E</td>
<td>1</td>
<td>2</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

- **Training Data:**
  - 480K users, 18K movies,
  - 100M ratings (1–5)
  - (99% ratings missing)

- **Goal:**
  - $1M prize for 10% reduction
  - in RMSE over Cinematch

- **BellKor’s Pragmatic Chaos**
  - declared winners on
  - 9/21/2009

Used ensemble of models, an important ingredient being
low-rank factorization (SVD)
Strategies for modeling big data

Once the data have been cleaned and organized, we are often left with a massive matrix of observations.

- If data are sparse (lots of zeros or NAs), store using sparse-matrix methods.

Quantcast example next: fit a sequence of logistic regression models using glmnet in R with 54M rows and 7M predictors. Extremely sparse X matrix, stored in memory (256G) — took 2 hours to fit 100 models of increasing complexity.

- If not sparse, use distributed, compressed databases. Many groups are developing fast algorithms and interfaces to these databases. For example H2O [CRAN] by H2O interfaces from R to highly compressed versions of data, using Java-based implementations of many of the important modeling tools.
Strategies for modeling big data

Once the data have been cleaned and organized, we are often left with a massive matrix of observations.

- If data are sparse (lots of zeros or NAs), store using sparse-matrix methods. Quantcast example next: fit a sequence of logistic regression models using \texttt{glmnet} in R with 54M rows and 7M predictors. Extremely sparse X matrix, stored in memory (256G) — took 2 hours to fit 100 models of increasing complexity.
Strategies for modeling big data

Once the data have been cleaned and organized, we are often left with a massive matrix of observations.

- If data are sparse (lots of zeros or NAs), store using sparse-matrix methods. Quantcast example next: fit a sequence of logistic regression models using glmnet in R with 54M rows and 7M predictors. Extremely sparse X matrix, stored in memory (256G) — took 2 hours to fit 100 models of increasing complexity.

- If not sparse, use distributed, compressed databases. Many groups are developing fast algorithms and interfaces to these databases.
Strategies for modeling big data

Once the data have been cleaned and organized, we are often left with a massive matrix of observations.

- If data are sparse (lots of zeros or NAs), store using sparse-matrix methods. Quantcast example next: fit a sequence of logistic regression models using `glmnet` in R with 54M rows and 7M predictors. Extremely sparse X matrix, stored in memory (256G) — took 2 hours to fit 100 models of increasing complexity.

- If not sparse, use distributed, compressed databases. Many groups are developing fast algorithms and interfaces to these databases. For example `H2O` [CRAN] by H2O interfaces from R to highly compressed versions of data, using Java-based implementations of many of the important modeling tools.
glmnet

Fit regularization paths for a variety of GLMs with lasso and elastic net penalties; e.g. logistic regression

\[
\begin{align*}
\log \frac{Pr(Y = 1 \mid X = x)}{Pr(Y = 0 \mid X = x)} &= \beta_0 + \sum_{j=1}^{p} x_j \beta_j \\
\end{align*}
\]

• Lasso penalty [Tibshirani, 1996] induces sparsity in coefficients: \( \sum_{j=1}^{p} |\beta_j| \leq s \). It shrinks them toward zero, and sets many to zero.

• Fit efficiently using coordinate descent. Handles sparse \( X \) naturally, and exploits sparsity of solutions, warms starts, variable screening, and includes methods for model selection using cross-validation.

glmnet team: TH, Jerome Friedman, Rob Tibshirani, Noah Simon, Junyang Qian.
Example: Large Sparse Logistic Regression

Quantcast is a digital marketing company.* Data are five-minute internet sessions. Binary target is type of family ($\leq 2$ adults vs adults plus children). 7 million features of session info (web page indicators and descriptors). Divided into training set (54M), validation (5M) and test (5M).

- All but 1.1M features could be screened because $\leq 3$ nonzero values.
- Fit 100 models in 2 hours in R using glmnet.
- Richest model had 42K nonzero coefficients, and explained 10% deviance (like R-squared).

* TH on SAB
54M train, 5M val, 5M test

% Deviance Explained on Training Data

Misclassification Error

Validation
Test
Train

54M train, 5M val, 5M test
H2O Billion Row Machine Learning Benchmark
GLM Logistic Regression

Hadoop/Mahout

H2O 16 EC2 nodes
34.9 sec, 3 iterations
numerical and categorical

H2O 16 EC2 nodes
16.5 sec, 2 iterations
numerical

H2O 48 EC2 nodes
14.2 sec, 3 iterations
numerical and categorical

H2O 48 EC2 nodes
5.6 sec, 2 iterations
numerical

Compute Hardware: AWS EC2 c3.2xlarge - 8 cores and 15 GB per node, 1 GbE interconnect
Airline Dataset 1987-2013, 42 GB CSV, 1 billion rows, 12 input columns, 1 outcome column
9 numerical features, 3 categorical features with cardinalities 30, 376 and 380

* TH on SAB
Strategies for modeling big data

- Online (stochastic) learning algorithms are popular — need not keep data in memory.

When modeling click-through rate, there is typically 1 positive example per 10,000 negatives. You do not need all the negatives, because beyond some point the variance comes from the paucity of positives. 1 in 15 is sufficient.


Think out of the box! How much accuracy do you need? Timeliness can play a role, as well as the ability to explore different approaches. Explorations can be done on subsets of the data.
Strategies for modeling big data

- Online (stochastic) learning algorithms are popular — need not keep data in memory.
- Subsample if possible!

When modeling click-through rate, there is typically 1 positive example per 10,000 negatives. You do not need all the negatives, because beyond some point the variance comes from the paucity of positives. 1 in 15 is sufficient.


Think out of the box! How much accuracy do you need? Timeliness can play a role, as well as the ability to explore different approaches. Explorations can be done on subsets of the data.
Strategies for modeling big data

- Online (stochastic) learning algorithms are popular — need not keep data in memory.

- Subsample if possible! When modeling click-through rate, there is typically 1 positive example per 10,000 negatives. You do not need all the negatives, because beyond some point the variance comes from the paucity of positives. 1 in 15 is sufficient.

Strategies for modeling big data

- Online (stochastic) learning algorithms are popular — need not keep data in memory.

- Subsample if possible! When modeling click-through rate, there is typically 1 positive example per 10,000 negatives. You do not need all the negatives, because beyond some point the variance comes from the paucity of positives. 1 in 15 is sufficient.


- Think out of the box!
Strategies for modeling big data

- Online (stochastic) learning algorithms are popular — need not keep data in memory.
- Subsample if possible! When modeling click-through rate, there is typically 1 positive example per 10,000 negatives. You do not need all the negatives, because beyond some point the variance comes from the paucity of positives. 1 in 15 is sufficient.


- Think out of the box! How much accuracy do you need? Timeliness can play a role, as well as the ability to explore different approaches. Explorations can be done on subsets of the data.
Lasso regression path: 70 mins.
Split data into 25 parts, distribute, and average: 30 secs.
In addition, free prediction standard errors and CV error.
Predicting the Pathogenicity of Missense Variants

Goal: prioritize list of candidate genes for prostate cancer

Joint work with Epidemiology colleagues Weiva Sieh, Joe Rothstein, Nilah Monnier Ioannidis, and Alice Whittemore
Approach

- A number of existing scores for disease status do not always agree (e.g. SIFT, Polyphen).
- Idea is to use a Random Forest algorithm to integrate these scores into a single consensus score for predicting disease.
- We will use existing functional prediction scores, conservation scores, etc as features — 12 features in all.
- Data acquired through SwissVar. 52K variants classified as
  - disease — 21K variants
  - neutral — 31K variants
Correlation of Features
Trees use the features to create subgroups in the data to refine the estimate of disease.
Trees use the features to create subgroups in the data to refine the estimate of disease. Shallow trees are too coarse/inaccurate.
Random Forests

Leo Breiman (1928–2005)

• Deep trees (fine subgroups) are more accurate, but very noisy.
• Idea: fit many (1000s) different and very-deep trees, and average their predictions to reduce the noise.
• How to get different trees?
  - Grow trees to bootstrap subsampled versions of the data.
  - Randomly ignore variables as candidates for splits.

Random Forests are very effective and give accurate predictions. They are automatic, and give good CV estimates of prediction error (for free!). R package RandomForest.
Results for Random Forests

Performance evaluated using OOB (out-of-bag) predictions for:
- All disease vs neutral variants (AUC 0.984)
- Cancer vs neutral variants (AUC 0.935)
Two New Methods

**Glinternet**

With past PhD student Michael Lim (JCGS 2014).
Main effect + two-factor interaction models selected using the group lasso.

**Gamsel**

With past Ph.D student Alexandra Chouldechova, using overlap group lasso.
Automatic, sticky selection between zero, linear or nonlinear terms in GAMs:

$$\eta(x) = \sum_{j=1}^{p} f_j(x_j)$$
Example: GWAS with $p = 27K$ Snps , each a 3-level factor, and a binary response, $N = 3500$.

- Let $X_j$ be $N \times 3$ indicator matrix for each Snp, and $X_{j:k} = X_j \star X_k$ be the $N \times 9$ interaction matrix.

- We fit model

$$
\log \frac{\Pr(Y = 1 | X)}{\Pr(Y = 0 | X)} = \alpha + \sum_{j=1}^{p} X_j \beta_j + \sum_{j<k} X_{j:k} \theta_{j:k}
$$

- note: $X_{j:k}$ encodes main effects and interactions.

- Maximize group-lasso penalized likelihood:

$$
\ell(y, p) - \lambda \left[ \sum_{j=1}^{p} \| \beta_j \|_2 + \sum_{j<k} \| \theta_{j:k} \|_2 \right]
$$

- Solutions map to traditional hierarchical main-effects/interactions model (with effects summing to zero).
Strong rules for feature filtering essential here — parallel and distributed computing useful too. GWAS search space of 729M interactions!

Formulated for all types of interactions, not just categorical variables.

**Glinternet** very fast — two-orders of magnitude faster than competition, with similar performance.
Example: Mining Electronic Health Records for Synergistic Drug Combinations

Using Oncoshare database (EHR from Stanford Hospital and Palo Alto Medical Foundation) looked for synergistic effects between 296 drugs in treatment of 9,945 breast cancer patients.

Used **GLINTERNET** to discover three potential synergies. Joint work with Yen Low, Michael Lim, TH, Nigam Shah and others.
Variables (nodes) are connected by edges if they synergistically interact with each other such that the pair is more strongly associated with lower mortality (blue edges) or higher mortality (red edges):

- Demographic variable
- Tumor variable
- Comorbidity variable
- Treatment variable
- Drug variable
- Drug class variable
\[
\frac{1}{2} \left\| y - \sum_{j=1}^{p} \alpha_j x_j - \sum_{j=1}^{p} U_j \beta_j \right\|^2 + \lambda \sum_{j=1}^{p} \left\{ (1 - \gamma) |\alpha_j| + \gamma \|\beta_j\|_{D_j^*} \right\} \\
+ \frac{1}{2} \sum_{j=1}^{p} \psi_j \|\beta_j\|^2_{D_j}
\]

- \( U_j = [x_j \ p_1(x_j) \cdots p_k(x_j)] \) where the \( p_i \) are orthogonal Demmler-Reinsch spline basis functions of increasing degree.
- \( D_j = \text{diag}(d_{j0}, d_{j1}, \ldots, d_{jk}) \) diagonal penalty matrix with \( 0 = d_{j0} < d_{j1} \leq d_{j2} \leq \cdots \leq d_{jk} \), and \( D_j^* = D_j \) but with \( d_{j0} = d_{j1} \).
Step = 1  \lambda = 125.43
Step = 2  lambda = 114.18
Step = 3   lambda = 103.94
Step = 4   lambda = 94.61
Step = 5    lambda = 86.13
Step = 6   \( \lambda = 78.4 \)
Step = 7  lambda = 71.37
Step = 8  lambda = 64.97
Step = 9    lambda = 59.14
Step = 10  lambda = 53.83
Step = 11   lambda = 49.01
Step = 12   lambda = 44.61
Step = 13  lambda = 40.61
Step = 14  lambda = 36.97
Step = 16   lambda = 30.63
Step = 17   lambda = 27.88
Step = 18   lambda = 25.38
Step = 19  \lambda = 23.11
Step = 20   lambda = 21.03
Step = 21  \lambda = 19.15
Step= 22   lambda = 17.43
Step = 23   lambda = 15.87
Step = 24  lambda = 14.44
Step = 25 \quad \lambda = 13.15
Step = 26   lambda = 11.97
Step = 27  lambda = 10.89
Step = 28  lambda = 9.92
Step = 29   lambda = 9.03
Step = 30   lambda = 8.22
Step = 31  lambda = 7.48
Step = 32   lambda = 6.81
Step = 33  lambda = 6.2
Step = 34   lambda = 5.64
Step = 35  lambda = 5.14
Step = 36  \lambda = 4.68
Step = 37   lambda = 4.26
Step = 38   lambda = 3.87
Step = 39  lambda = 3.53
Step = 40  \lambda = 3.21
Step = 41   lambda = 2.92
Step = 42    lambda = 2.66

v1
f(v1)

v2
f(v2)

v3
f(v3)

v4
f(v4)

v5
f(v5)

v6
f(v6)

v7
f(v7)

v8
f(v8)

v9
f(v9)

v10
f(v10)

v11
f(v11)

v12
f(v12)
Step = 43  \( \lambda = 2.42 \)
Step = 44   lambda = 2.2
Step = 45  lambda = 2.01
Step = 46   lambda = 1.83
Step = 47   lambda = 1.66
Step = 48    lambda = 1.51
Step = 49   lambda = 1.38

\[ f(v_1) \]

\[ f(v_2) \]

\[ f(v_3) \]

\[ f(v_4) \]

\[ f(v_5) \]

\[ f(v_6) \]

\[ f(v_7) \]

\[ f(v_8) \]

\[ f(v_9) \]

\[ f(v_{10}) \]

\[ f(v_{11}) \]

\[ f(v_{12}) \]
Step = 50  lambda = 1.25

1. $f(v_1)$
2. $f(v_4)$
3. $f(v_7)$
4. $f(v_{10})$
5. $f(v_2)$
6. $f(v_5)$
7. $f(v_8)$
8. $f(v_{11})$
9. $f(v_3)$
10. $f(v_6)$
11. $f(v_9)$
12. $f(v_{12})$
useR! 2016

All the tools I described are implemented in R, which is wonderful free software that gets increasingly more powerful as it interfaces with other systems. R can be found on CRAN: http://cran.us.r-project.org

27–30 June 2016, R user conference at Stanford!
useR! 2016

All the tools I described are implemented in R, which is wonderful free software that gets increasingly more powerful as it interfaces with other systems. R can be found on CRAN: http://cran.us.r-project.org

27–30 June 2016, R user conference at Stanford!

⋯ and now for some cheap marketing ⋯
useR! 2016

All the tools I described are implemented in R, which is wonderful free software that gets increasingly more powerful as it interfaces with other systems. R can be found on CRAN: http://cran.us.r-project.org

27–30 June 2016, R user conference at Stanford!

⋯ and now for some cheap marketing ⋯
useR! 2016

All the tools I described are implemented in R, which is wonderful free software that gets increasingly more powerful as it interfaces with other systems. R can be found on CRAN: http://cran.us.r-project.org

27–30 June 2016, R user conference at Stanford!

⋯ and now for some cheap marketing …

Thank you!