

# **Econ.AI** | The role for Deep Neural Networks in Economics

Matt Taddy – UChicago and Microsoft

# What is AI?

Domain Structure + Data Generation + General Purpose ML

Econ Theory / Biz Frame  
Structural Econometrics  
Relaxations and Heuristics

Reinforcement Learning  
Sensor Networks, IOT  
Simulation/GANs






Deep Neural Nets  
SGD + OOS + GPUs  
Video/Audio/Text

*Self-training structures of ML predictors that automate and accelerate human tasks*

# Hybrid Reward Architecture



Level: 201

-	30425 × 10 =	304250
■	801 × 50 =	40050
	17 × 200 =	3400
	6 × 400 =	2400
	3 × 800 =	2400
	1 × 1600 =	1600
	42 × 100 =	4200
	40 × 200 =	8000
	33 × 500 =	16500
	43 × 700 =	30100
	48 × 1000 =	48000
	47 × 2000 =	94000
	89 × 5000 =	445000

999900



Tell me about your problem and I'll help you find the solution you need.

how do I upgrade windows>

Here's what I think you are asking about: *How to install or upgrade to Windows 10*. Is that correct?

Yes

Okay, you're looking for some info on Windows 10. What would you like to do?

Upgrading to Windows 10

#### How to get Windows 10

1. Windows 10 is available by buying a [new device](#) or a [full version of the software](#).
2. The Windows 10 free upgrade through the Get Windows 10 (GWX) app ended on July 29, 2016. [Click here to find more information about upgrading to Windows 10](#)

Select an option or enter your response here



# THE WALL STREET JOURNAL

TECH

## Microsoft Aims to Make Business AI Cheaper, Faster, Simpler

New line of software will take on IBM, others, in growing artificial-intelligence market

*By Ted Greenwald*

Sept. 25, 2017 9:00 a.m. ET



Microsoft Corp. plans Monday to unveil its first product in a new line of software aimed at taking on International Business Machines Corp. and others in the growing market to apply artificial intelligence to everyday business needs.

The new product, a customer-service virtual assistant, is designed to let people

# The Economics of AI

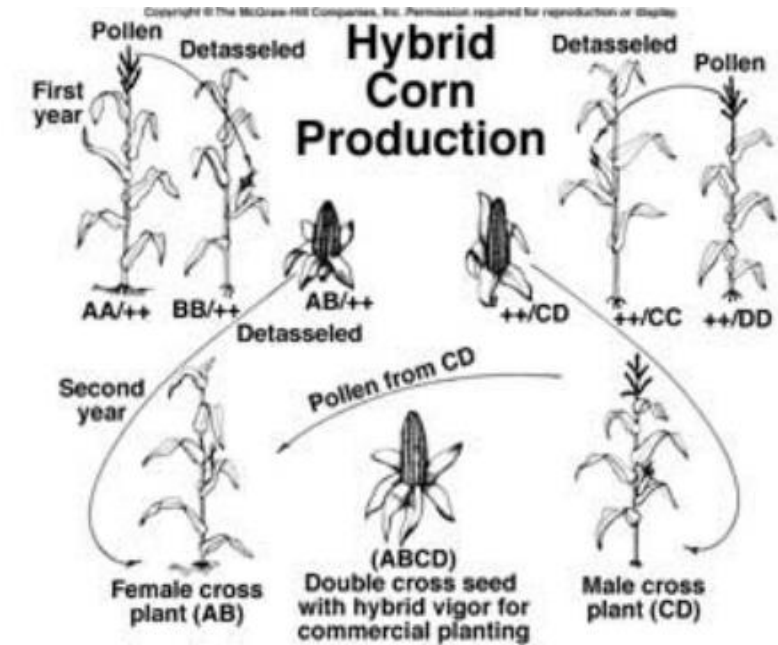
DNNs are GPT and 'method for invention'

- Broad impact, up and down the value chain
- Gets better, faster, and cheaper in time
- Can suffer from underinvestment
- Productivity gains lag invention

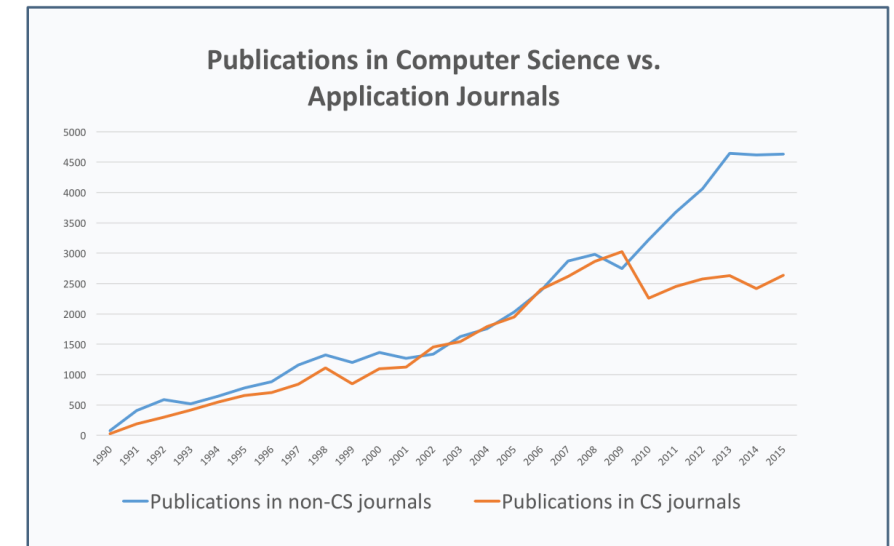
Automation, inequality, skill acquisition

Data ownership, markets, and privacy

High-info contracts and outcome pricing



AI research in computer science journals vs. other application sectors.



Graph from Cockburn/Henderson/Stern

# What about the impact of AI on the practice of Econom[etr]ics?

## Predictions for Economics

*Susan Athey:*

- ▶ Adoption of off-the-shelf ML methods for their intended tasks (prediction, classification, and clustering, e.g. for textual analysis)
  - ▶ Extensions and modifications of prediction methods to account for considerations such as fairness, manipulability, and interpretability
  - ▶ Development of new econometric methods based on machine learning designed to solve traditional social science estimation tasks, e.g. causal inference
  - ▶ Increased emphasis on model robustness and other supplementary analysis to assess credibility of studies
  - ▶ Adoption of new methods by empiricists at large scale
  - ▶ Revival and new lines of research in
- ▶ New methods for the design and analysis of large administrative data, including merging these sources
  - ▶ Increase in interdisciplinary research
  - ▶ Changes in organization, dissemination, and funding of economic research
  - ▶ “Economist as engineer” engages with firms, government to design and implement policies in digital environment
  - ▶ Design and implementation of digital experimentation, both one-time and as an ongoing process, in collaboration with firms and government
  - ▶ Increased use of data analysis in all levels of economics teaching; increase in interdisciplinary data science programs
  - ▶ Research on the impact of AI and ML on economy

Alt Text: A close up of a newspaper

Econometrics breaks systemic questions into sets of prediction tasks

- Prediction after controlling for confounders
- Two-stage prediction with IVs
- Heterogeneous treatment effect prediction
- Structural equation systems

Machine Learning can automate and accelerate tasks in applied econometric workflows

# Example: short-term price elasticity

If I drop price 1%, by what % will quantity sold increase?

Problem: both prices and sales respond to underlying demand

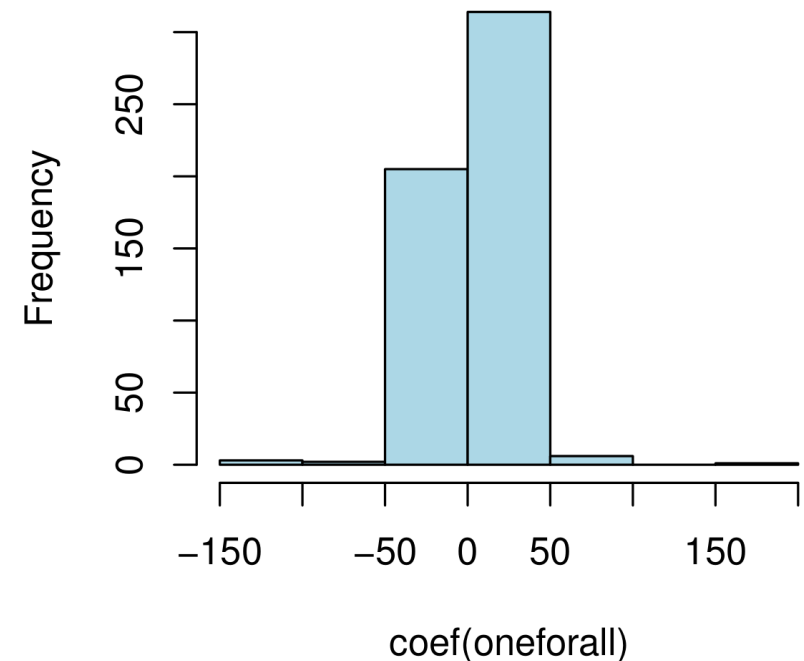
Need a causal effect of price on sales, not their co-movement

## Beer Data

A single shared elasticity gives tiny -0.23

Separate elasticity for each: noisy zeros

We need to group the products together using brand, pack, etc.





# Beer Elasticity

Say  $w_{bk} = 1$  if word  $k$  is in description for beer  $b$

@transaction  $t$ :  $y_{tb} = \gamma_b p_{tb} + f_t(\mathbf{w}_b) + \varepsilon_{tb}$ ,

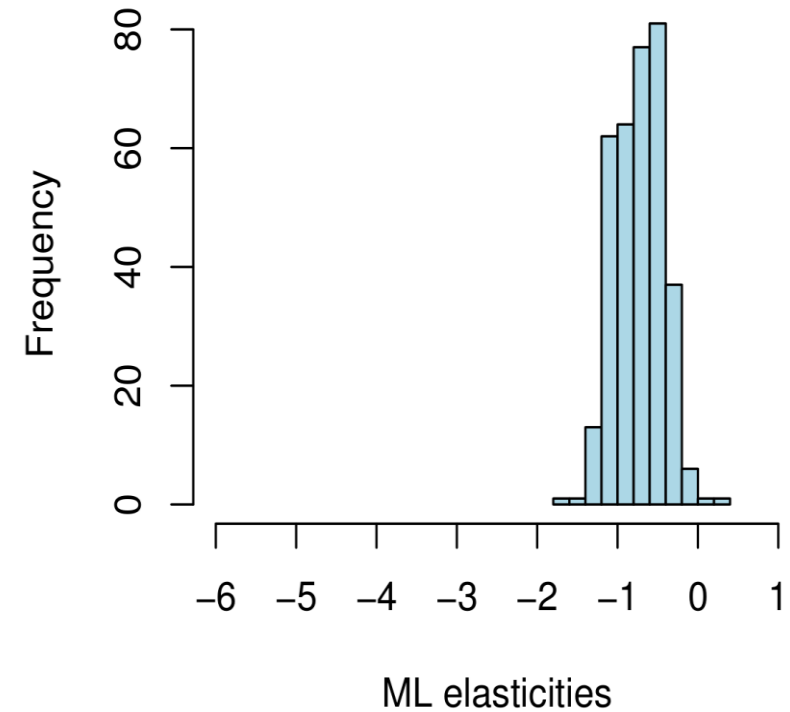
$$\gamma_b = \mathbf{w}_b' \boldsymbol{\beta}$$

$$p_{tb} = h_t(\mathbf{w}_b) + v_{tb}$$

Creates a large number of parameters

Just throw it all in a lasso?

Yields unbelievably small elasticities



The naïve ML conflates two problems:

selecting controls and predicting response after controlling for confounders.

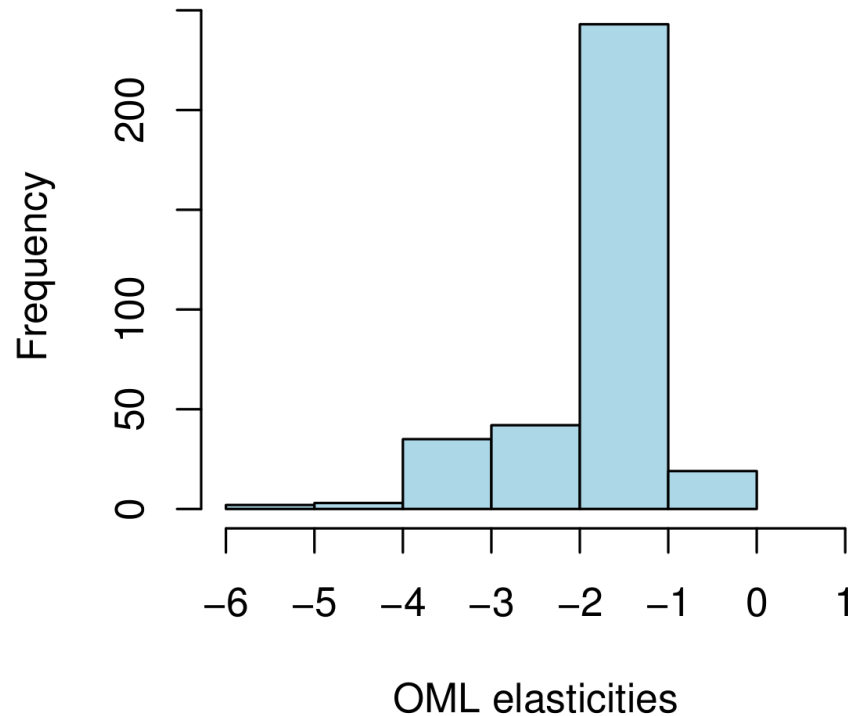
Instead, use Orthogonal ML (*Chernozhukov et al, 2016 and earlier*)

- Estimate **nuisance** functions  $\mathbb{E}[\mathbf{y}_{tb}|\mathbf{w}_b]$  and  $\mathbb{E}[\mathbf{p}_{tb}|\mathbf{w}_b]$
- Orthogonalize the score against these nuisance functions (**data split**)
- Then estimation for  $\boldsymbol{\gamma}$  is robust to slow-learned nuisances

Estimation breaks into a series of ML tasks:

1. Predict sales from the demand variables:  $\mathbf{y}_{tb} \approx g(t, \mathbf{w}_b)$
2. Predict prices from the demand variables:  $\mathbf{p}_{tb} \approx h(t, \mathbf{w}_b)$
3. Get OOS residuals:  $\tilde{\mathbf{y}}_t = \mathbf{y}_t - \hat{g}_{\bar{t}}(t, \mathbf{w}_b), \quad \tilde{\mathbf{p}}_t = \mathbf{p}_t - \hat{h}_{\bar{t}}(t, \mathbf{w}_b)$
4. And fit the final regression:  $\mathbb{E}[\tilde{\mathbf{y}}_t] = \boldsymbol{\Gamma} \tilde{\mathbf{p}}_t = \text{diag}(\boldsymbol{\gamma}) \tilde{\mathbf{p}}_t$

# Orthogonal ML for Beer



There's no ground truth,  
but these are economically realistic

The text encodes a natural hierarchy

Many beers are *IPA* or *Cider*

But individual brands also load

Most Price Sensitive

```
> names(sort(e1)[1:5])  
[1] "GUINNESS DRAUGHT 6PK BTL  
[2] "GUINNESS DRAUGHT 4PK CAN  
[3] "PYRAMID OUTBURST IMP IPA 6PK  
[4] "ELYSIAN IMPORTAL IPA 6PK  
[5] "PYRAMID OUTBURST IMP IPA 12PK
```

Least Price Sensitive

```
> names(sort(-e1)[1:5])  
[1] "2 TOWNS CRISP APPLE CIDER  
[2] "2 TOWNS BAD APPLE CIDER  
[3] "ATLAS BLKBRY APPLE CIDER  
[4] "D'S WICKED BAKED APPLE CIDER  
[5] "D'S WICKED GREEN APPLE CIDER
```

# Econ + ML

This is what econometricians do: break systems into measurable pieces

Another common example: Instrumental Variables

Endogenous errors:

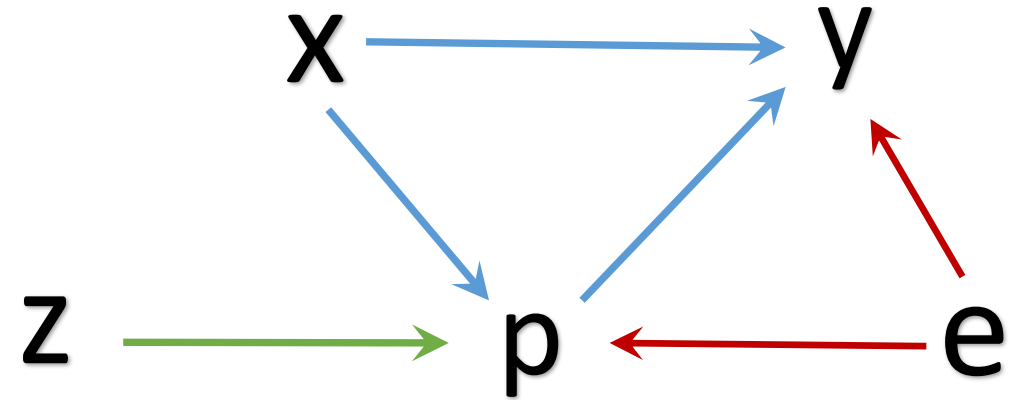
$$y = g(p, \mathbf{x}) + e \text{ and } \mathbb{E}[p e] \neq 0$$

If you estimate this using naïve ML, you'll get

$$E[y|p, \mathbf{x}] = E_{e|p}[g(p, \mathbf{x}) + e] = g(p, \mathbf{x}) + E[e|p, \mathbf{x}]$$

But, with instruments...

# Instrumental Variables



The *exclusion structure* implies

$$\mathbb{E}[y|x, z] = \int g(p, x) dF(p|x, z)$$

You can observe and estimate  $\hat{\mathbb{E}}[y|x, z]$  and  $\hat{F}(p|x, z)$

$\Rightarrow$  to solve for *structural*  $g(p, x)$  we have an inverse problem.

cf Newey+Powell 2003

$$\min_{g \in G} \sum \left( y_i - \int g(p, x_i) dF(p|x_i, z_i) \right)^2$$

**2SLS:**  $p = \beta z + v$  and  $g(p) = \tau p$  so that  $\int g(p) dF(p|z) = \tau \mathbb{E}[p|z]$

So you first regress  $p$  on  $z$  then regress  $y$  on  $\hat{p}$  to recover  $\hat{\tau}$ .

**Sieve:**  $g(p, x_i) \approx \sum_k \gamma_k \varphi_k(p, x_i)$ ,  $\mathbb{E}_F[\varphi_k(p, x_i)] \approx \sum_j \alpha_{kj} \beta_j(x_i, z_i)$

*Also Blundell, Chen, Kristensen, , Chen + Pouzo, Darolles et al, Hall+Horowitz*

$$\min_{g \in G} \sum \left( y_i - \int g(p, x_i) dF(p|x_i, z_i) \right)^2$$

Deep IV uses DNNs to target the integral loss function directly

- First, fit  $\hat{F}$  using a network with multinomial response
- Second (preferably on another sample) fit  $\hat{g}$  following

$$\nabla \hat{L}(x_i, y_i, z_i, \theta) = -2(y_i - g_\theta(\dot{p}, x_i)) g'_\theta(\ddot{p}, x_i), \quad \dot{p}, \ddot{p} \sim \hat{F}(p|x_i, z_i)$$

# Stochastic Gradient Descent

You have loss  $L(\mathbf{D}, \theta)$  where  $\mathbf{D} = [\mathbf{d}_1 \dots \mathbf{d}_N]$

In the usual GD, you iteratively descend

$$\theta_t = \theta_{t-1} - \mathbf{C}_t \nabla L(\mathbf{D}, \theta_{t-1})$$

In SGD, you instead follow *noisy* but *unbiased* sample gradients

$$\theta_t = \theta_{t-1} - \mathbf{C}_t \nabla L(\{\mathbf{d}_{t_b}\}_{b=1}^B, \theta_{t-1})$$



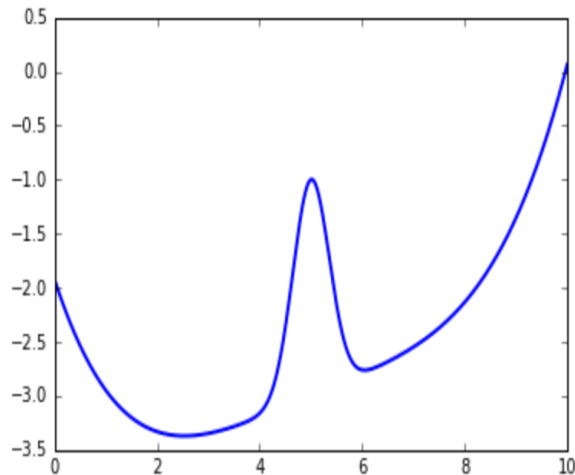
# A pricing simulation

$$y = 100 + s\psi_t + (\psi_t - 2)p + e,$$

$$p = 25 + (z + 3)\psi_t + v$$

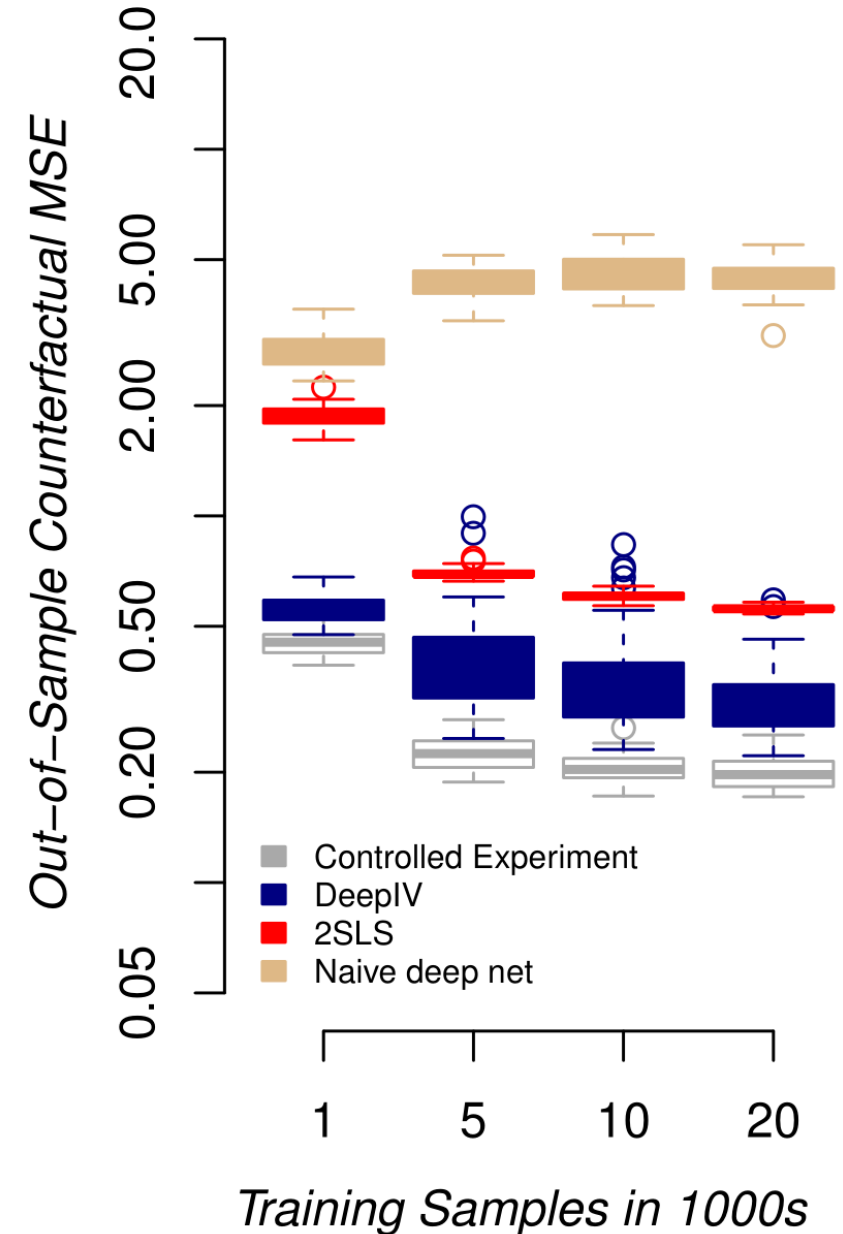
$$z, v \sim N(0, 1) \text{ and } e \sim N(\rho v, 1 - \rho^2),$$

Time-dependent  $\psi_t$



Customer type 's'

1 1 5 4 3  
7 5 3 5 3  
5 5 9 0 6  
3 5 2 0 0



# Biased?

$\nabla \hat{L}(x_i, y_i, z_i, \theta) = -2(y_i - g_\theta(\dot{p}, x_i)) g'_\theta(\dot{p}, x_i)$  is *biased* for our loss  
but unbiased for an upper bound on that loss (via Jensen's)

It works pretty well on OOS Loss:

Loss Function	# Samples	Mean	Stdev
Upper bound	1	<b>0.32</b>	0.085
Unbiased	2	0.48	0.107
Unbiased	4	0.50	0.158
Unbiased	8	0.44	0.100
Unbiased	16	<b>0.39</b>	0.098

# Validation and model tuning

We can do OOS *causal validation*

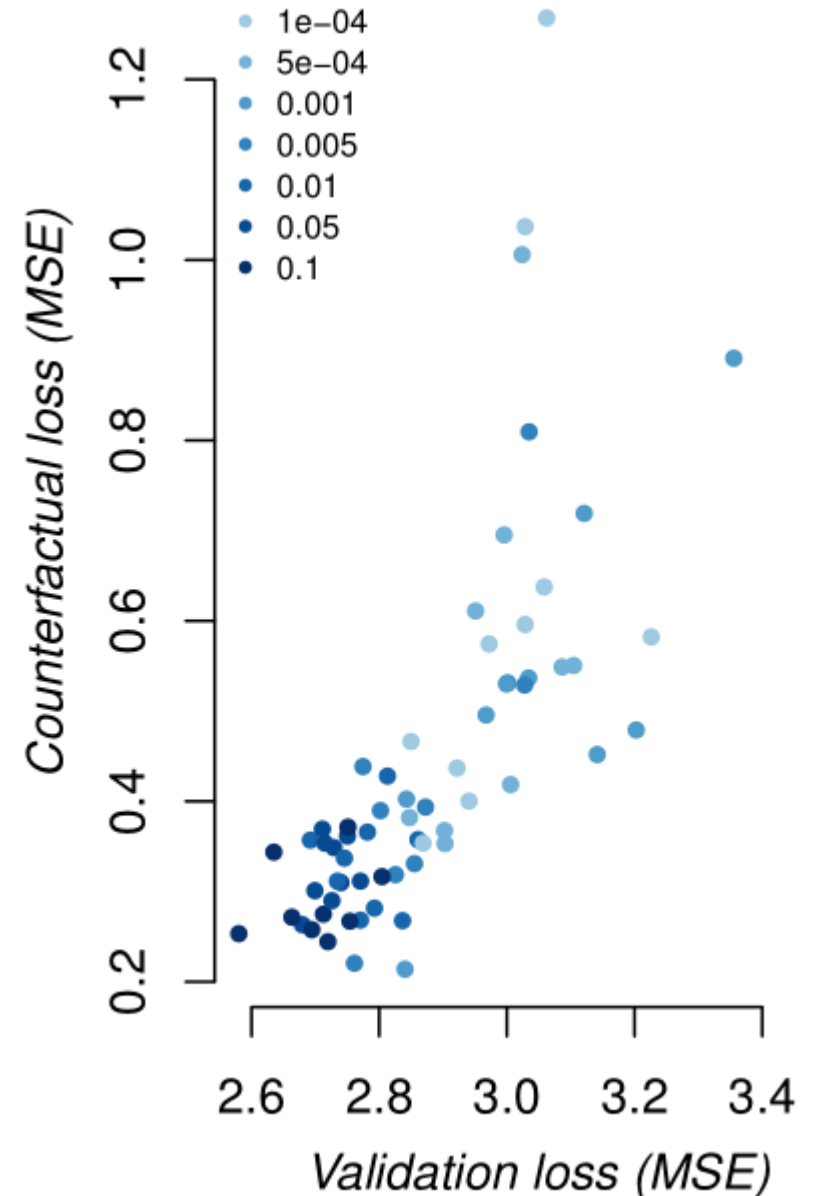
Leave-out deviance on first stage

$$\sum_{i \in LO} -\log \hat{f}(p|x_i, z_i)$$

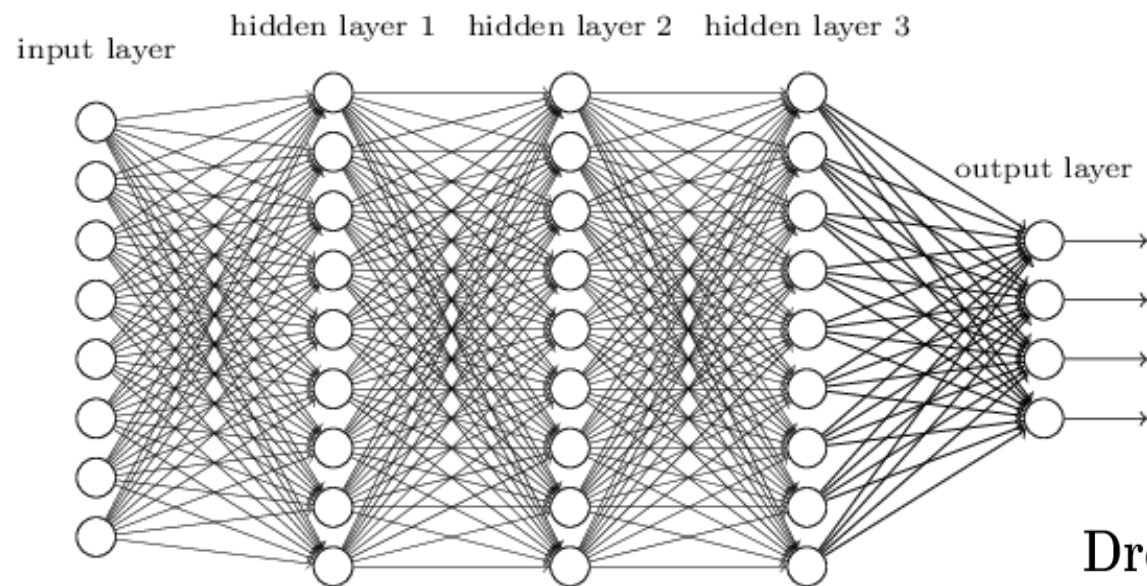
Leave-out loss on second

$$\sum_{i \in LO} (y_i - \int g_{\theta}(p, x_i) d\hat{F}(p|x_i, z_i))^2$$

You want to minimize both of these (in order).



# DEEP NEURAL NETWORKS



Train faster, generalize better:  
Stability of stochastic gradient descent

## Adaptive Subgradient Methods for Online Learning and Stochastic Optimization\*

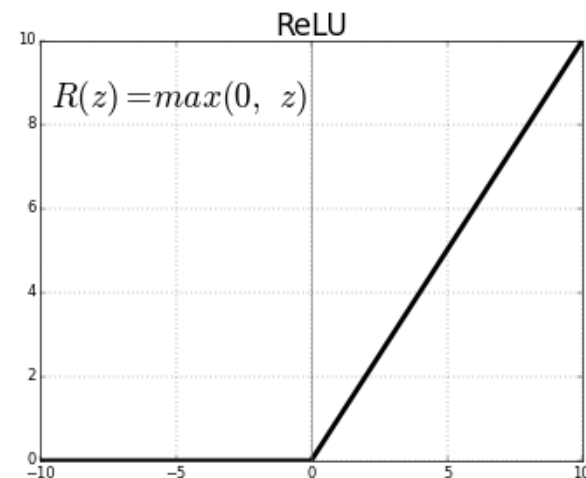
Dropout: A Simple Way to Prevent Neural Networks from  
Overfitting

## The Microsoft Cognitive Toolkit

A free, easy-to-use, open-source, commercial-grade toolkit that  
trains deep learning algorithms to learn like the human brain.

GET STARTED >

$$\mathcal{L} + \beta \left[ \frac{1}{2} \|w\|_2^2 \right]$$

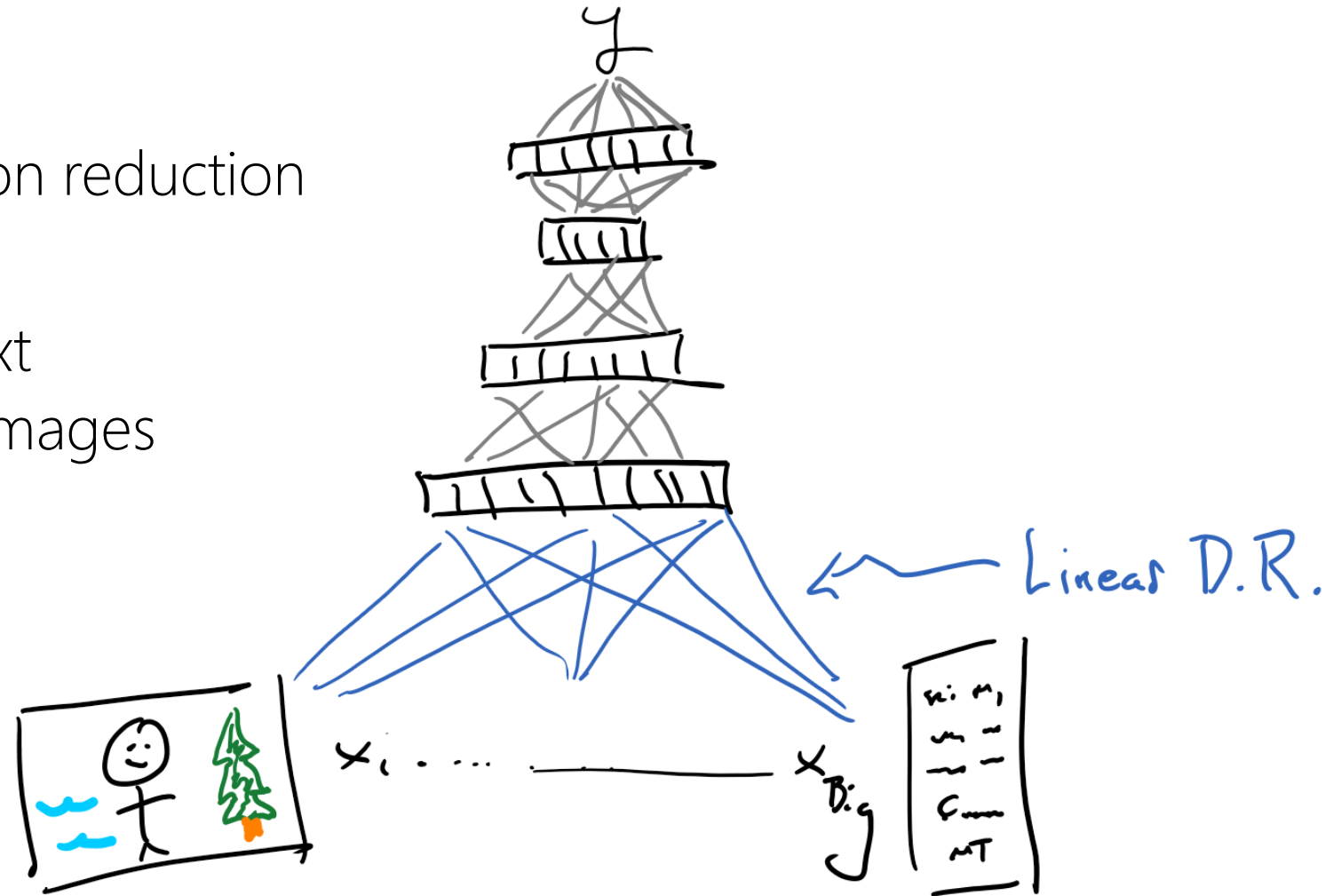


# Deep nets are **not** nonparametric sieves

The 1<sup>st</sup> layer is a big dimension reduction

For example,

- word embedding for text
- matrix convolution for images



# Why? Heterogeneity!

Example: ads application from Goldman and Rao (2014)

74 mil click-rates over 4 hour increments for 10k search terms

Treatment: **ad position 1-2**

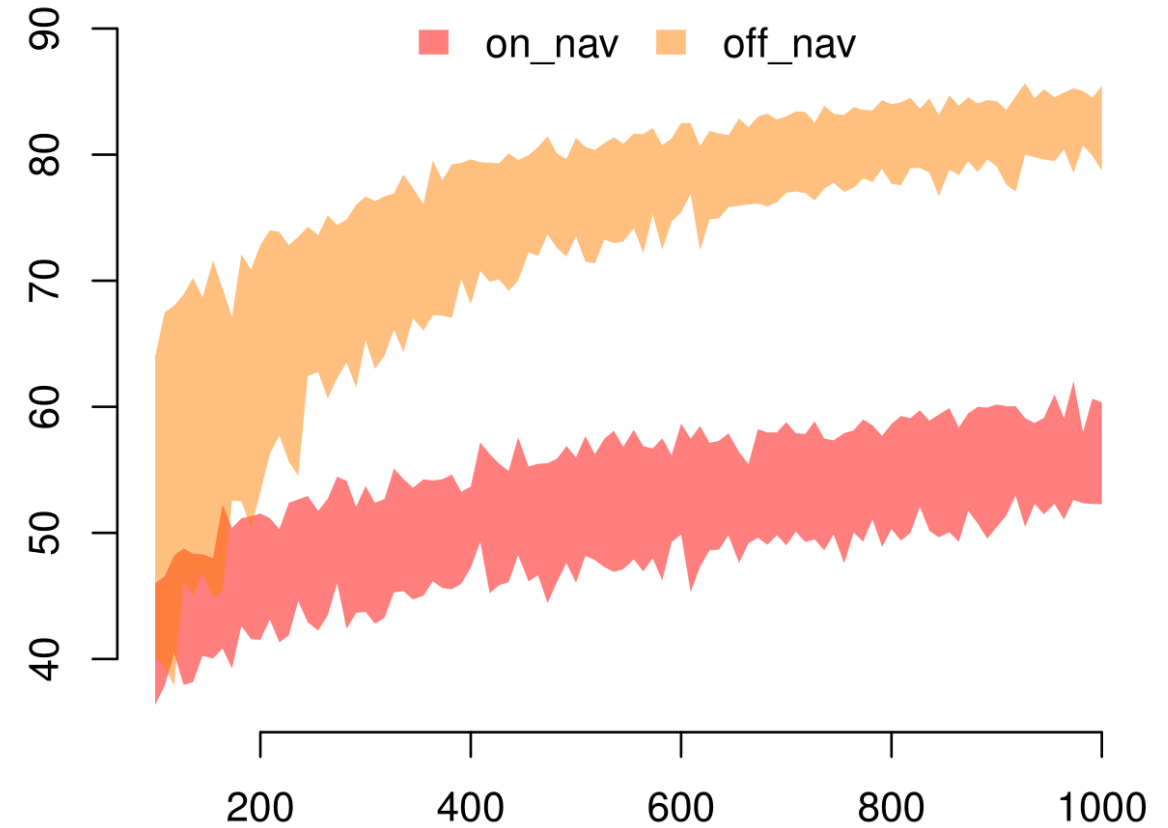
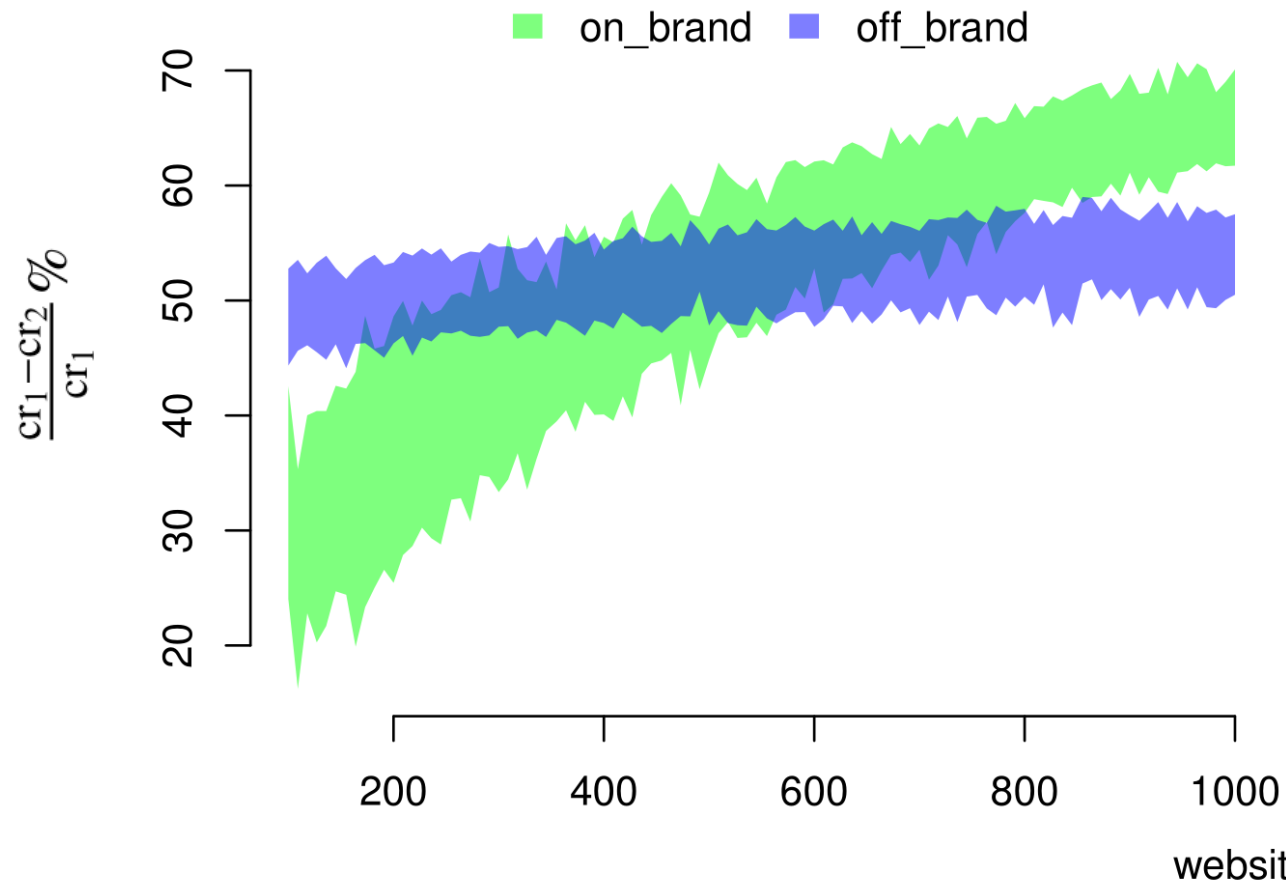
Instrument: **background AB testing (bench of ~ 100 tests)**

Covariates: **advertiser id and ad properties, search text, time period**

The reduction in clicks due to a drop in position is search and ad dependent

## Example product (click rate response)

- Treatment effect is small for on-navigation queries ('searches' for microsoft.com)
- Effect rises for less popular websites (brands)
- Off brand effect is flat with website popularity

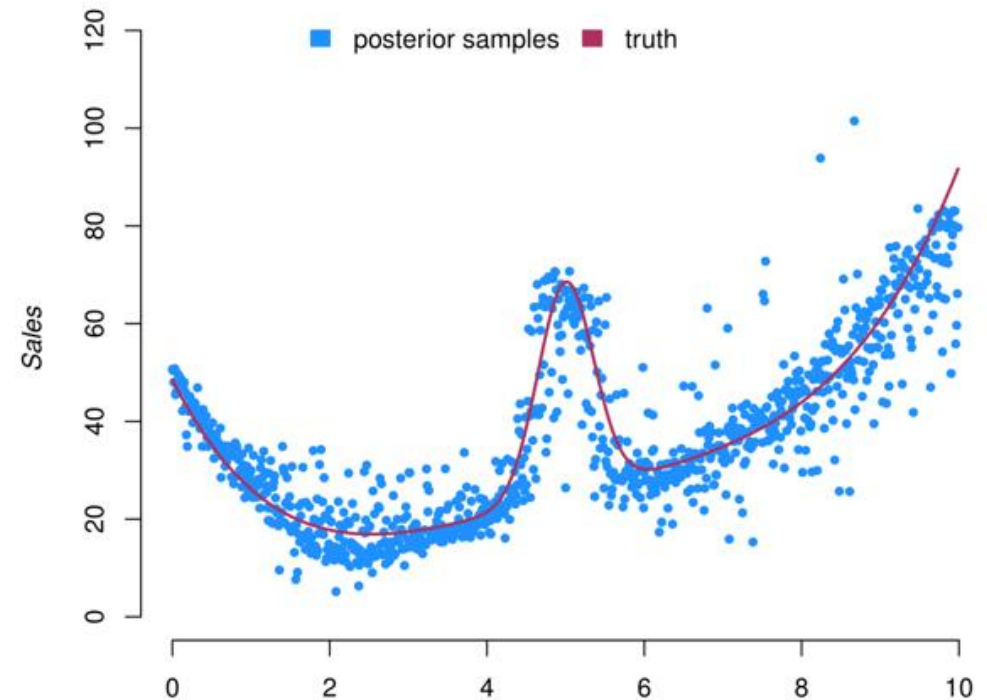


# Inference? Good question

Data Splitting

Variational Dropout

Quantile Regression



*with Jason Hartford UBC*



# Data Split

- Fit DNNs that map from inputs to output layer  $\psi_k(x)$ ,  $k = 1 \dots K$
- Use **out-of-sample**  $x_i$  to obtain 'features'  $\psi_{ik} = \psi_k(x_i)$
- Possibly do PCA on  $\boldsymbol{\eta}_i$  to get a nonsingular design
- Fit OLS  $y_i \approx \boldsymbol{\psi}_i' \boldsymbol{\beta}$  to get  $\hat{\boldsymbol{\beta}}$  with variance

$$\text{var}(\hat{\boldsymbol{\beta}}) = (\boldsymbol{\Psi}'\boldsymbol{\Psi})^{-1} \boldsymbol{\Psi}' \text{diag}(\mathbf{y} - \boldsymbol{\Psi}\hat{\boldsymbol{\beta}}) \boldsymbol{\Psi} (\boldsymbol{\Psi}'\boldsymbol{\Psi})^{-1}$$

This can be used to get  $\text{var}(\mathbb{E}[y | x])$

# Variational Bayes and Dropout

- VB fits  $q$  to minimize  $\mathbb{E}_q[\log q(W) - \log p(\mathbf{D}|W) - \log p(W)]$
- We train with **dropout SGD**:

At each update of weights  $\omega$ , use gradients for  $w = \xi\omega$ ,  $\xi \sim \text{Bern}(c)$

- This corresponds to VB under

$$q(W) = \prod_l \prod_k c \mathbb{1}_{[W_k = \Omega_k]} + (1 - c) \mathbb{1}_{[W_k = 0]}$$

This can be used to get  $\text{var}(\mathbb{E}[y | x])$

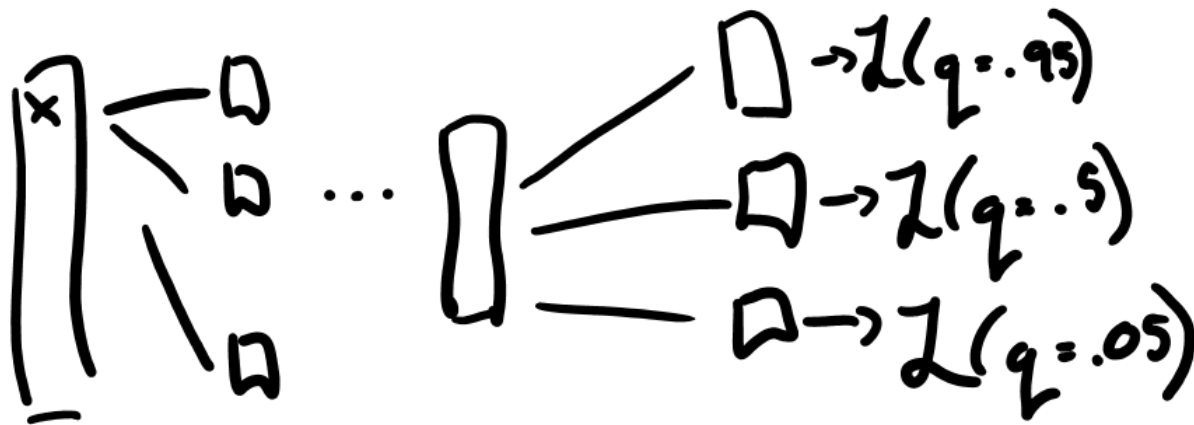
# Quantile Regression

Instead of targeting MSE or logit loss, minimize quantile loss

$$L_q = (y - \eta_q(x)) (q - 1_{y < \eta_q(x)})$$

Where  $q$  is your desired probability and  $\eta_q(x)$  is the quantile function

Better yet, architect a net to fit multiple quantiles at once...



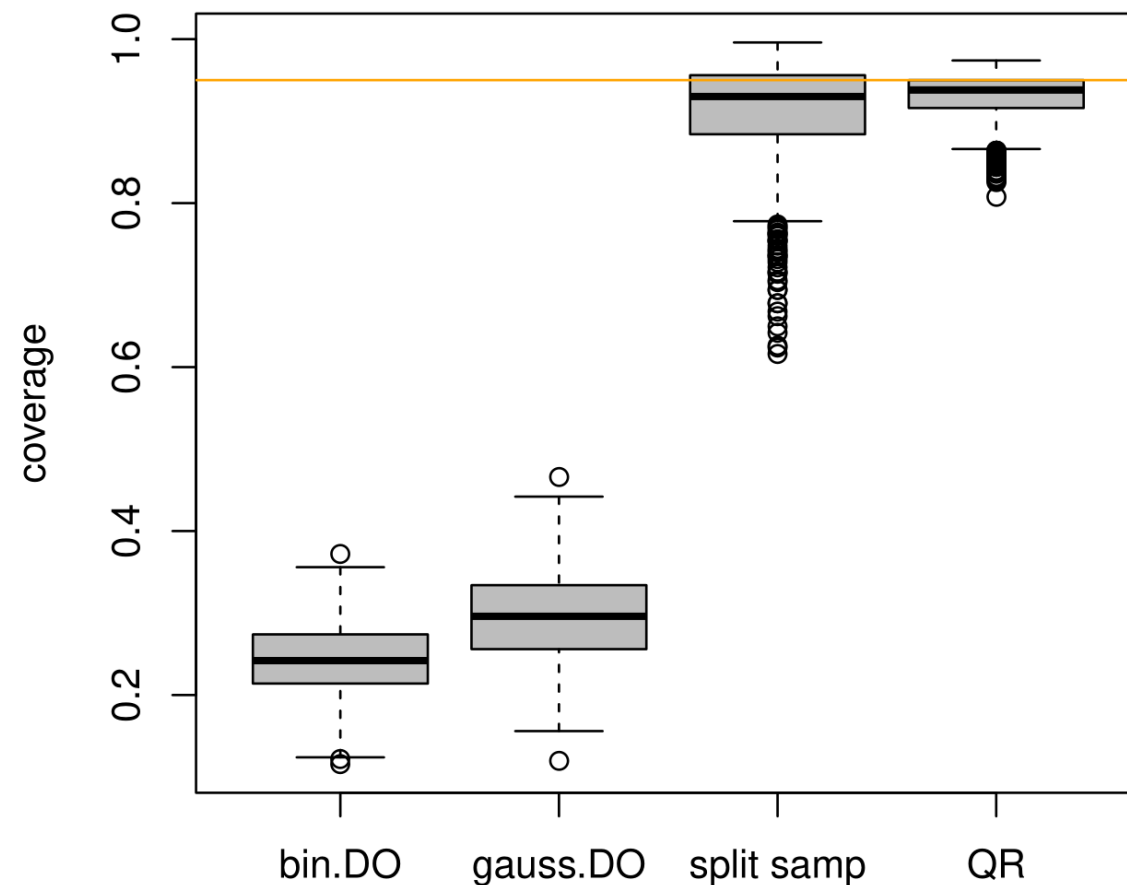
This can be used  
to get prediction  
intervals for  $y | x$



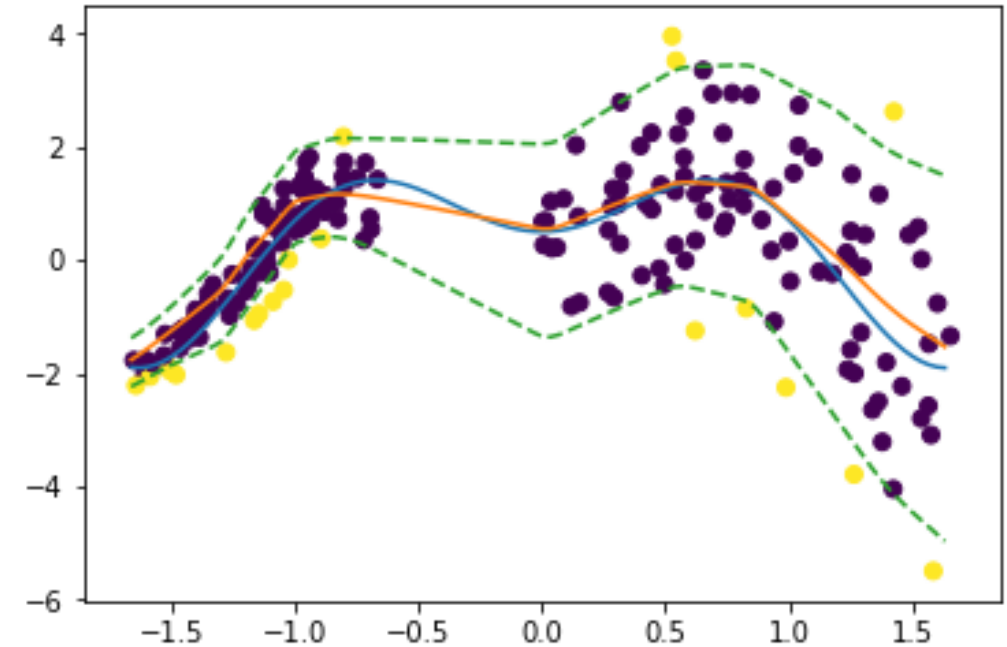
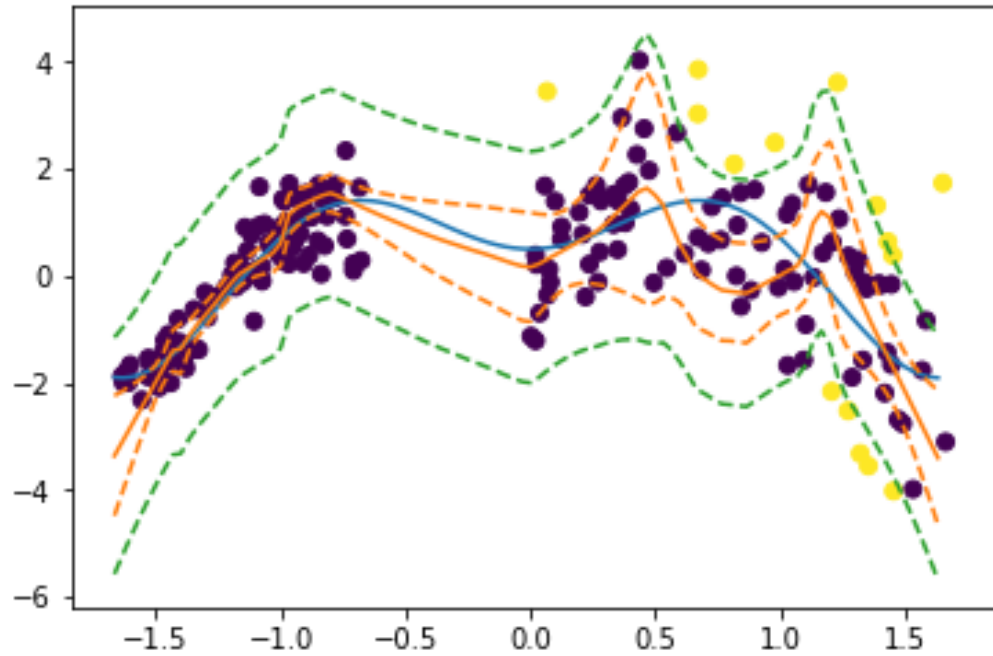
## Million Song Dataset

- A dataset of a million songs
- Inputs are timbre features
- Output is the release year
- Test and train are split to have no overlap on artists

*PI coverage around random songs*



If you want Prediction Intervals, you should use quantile regression



For Confidence Intervals, sample splitting can't be beat

# Economic AI

The ML doesn't create new economic insights or replace economists  
It **automates and accelerates** subjective labor-intensive measurement

- Instruments are everywhere inside firms
- With reinforcement learning there will be even more
- Reduced forms are low fruit; structural econometrics is next
  - Need to link long term rewards to short term signals

# Business AI

Deep learning revolution: good **low-dev-cost** off-the-shelf ML

As the tools become plug-n-play, teams get interdisciplinary

**The next big gains in AI are coming from domain context**

- Use domain structure to break questions into ML problems
- Don't re-learn things you already know with baby AI