ML for Demand Systems

Greg Lewis (MSR + NBER) Matt Taddy (MSR + Chicago) Kui Tang (Columbia)
Jason Hartford (UBC) Richard Li (UW) Mengting Wan (UCSD) James Zou (Stanford)
Matt Goldman (MSFT) Justin Rao (MSR) Di Wang (MSR)

What is a demand system?

Suppose that you have transactions 't' on products 'j'.

Write the quantity bought 'q' as

$$q_{tj} = \alpha_j(\boldsymbol{d}_t) + \gamma_j \log p_{tj} + \varepsilon_{tj}$$

a function of utility we can ($\alpha_j(\boldsymbol{d}_t)$) and can't ($\varepsilon_{tj}$) see, plus price p_{tj}.

You need to have a model like this to target customers or set prices.

But it's a system!

For example: There many different products

Demand for j depends on substitutes and complements

Or: where does price come from?

$$\log p_{tj} = \varphi_t(\boldsymbol{c}_j) + \psi_j q^{\star}_{tj} + \nu_{tj}$$

and the *demand system* is in equilibrium when $q^{\star}_{tj} = q_{tj}$

This equilibrium introduces `price endogeneity': $\mathbb{E}[p_{tj}\varepsilon_{tj}] \neq 0$

Economic models

Demand models let us estimate pieces of simplified systems

- Discrete choice of BLP and McFadden for one-hot selections
- Hedonics of Rosen and Bajari+Benkard for characteristic demand
- Almost Ideal for spending with big sets of complements and substitutes
- IV strategies for focused parameter inference

What can stat/machine/deep learning offer?

Sometimes it's just regression

If we treat ε_{tj} as independent this is a simple prediction problem
e.g., model store transactions with covariates $\boldsymbol{x}_{tj}$ as

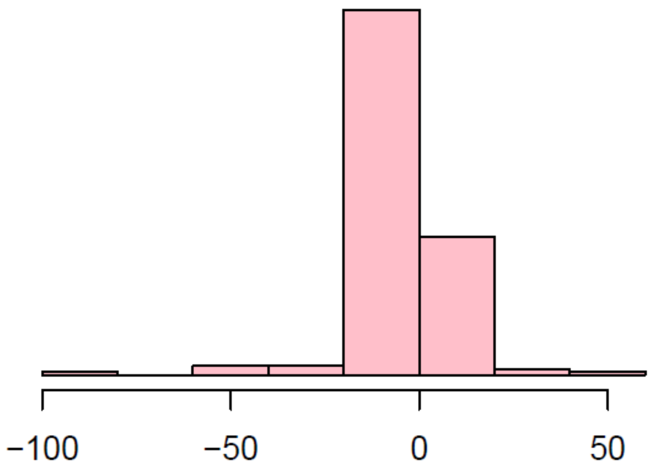
$$\mathbb{E}\log q_{tj} = \boldsymbol{x}'_{tj}\boldsymbol{\beta} + \log p_{tj}\, \boldsymbol{x}'_{tj}\boldsymbol{\gamma}$$

Elasticities:

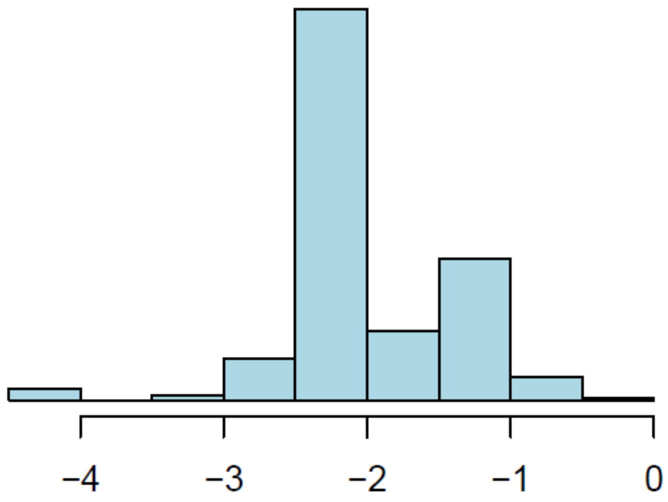
one shared: $x_{tj} = 1$

$$\frac{dq}{dp}\frac{p}{q} = -0.23$$

brand-specific: $x_{tjk} = \mathbb{1}_{[k=j]}$



$\boldsymbol{x}_{tj}$= *featurized description*

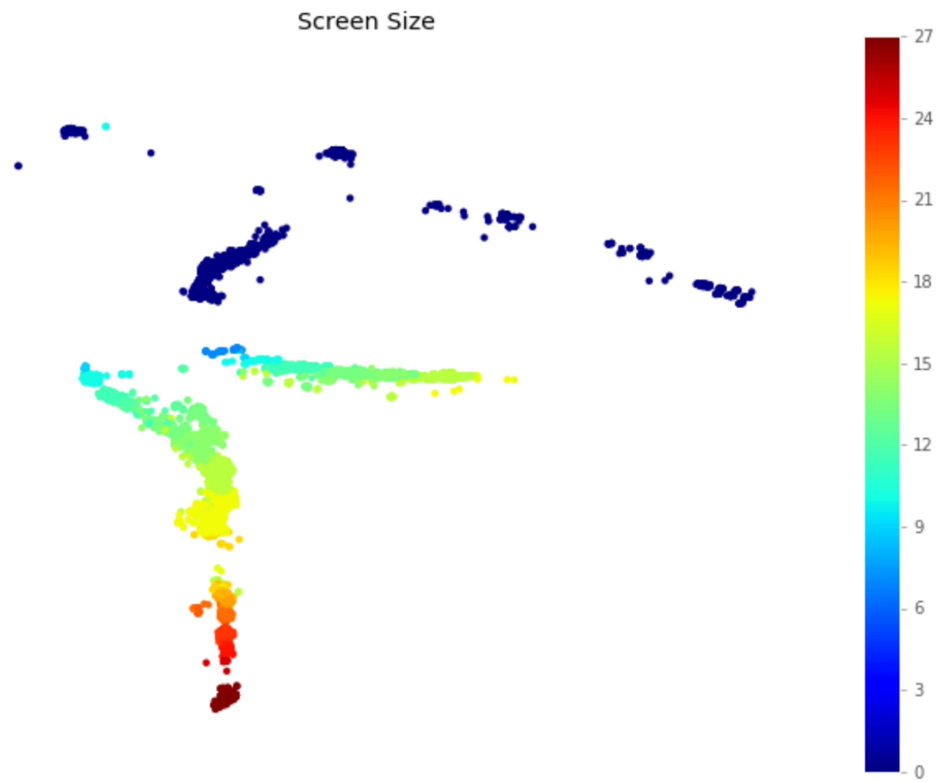


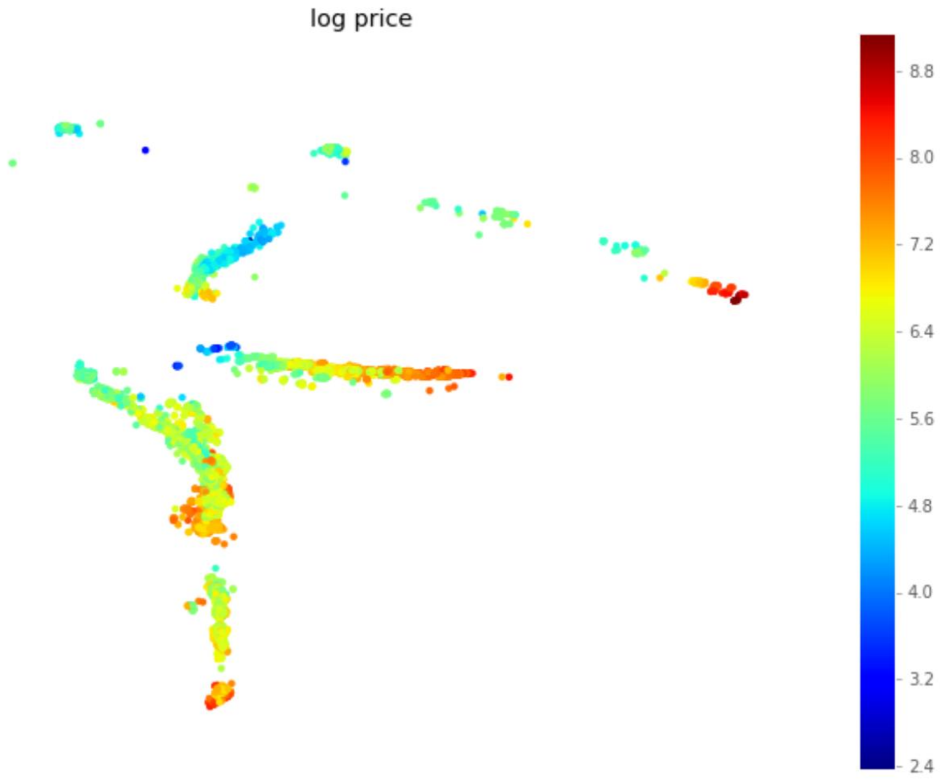
Deep Hedonics

Hedonic regression models *prices* from product *characteristics*

If the characteristic space is continuous and low-D
then you can invert this to get demand for characteristics

We use auto-encoders to learn low-D latent characteristics



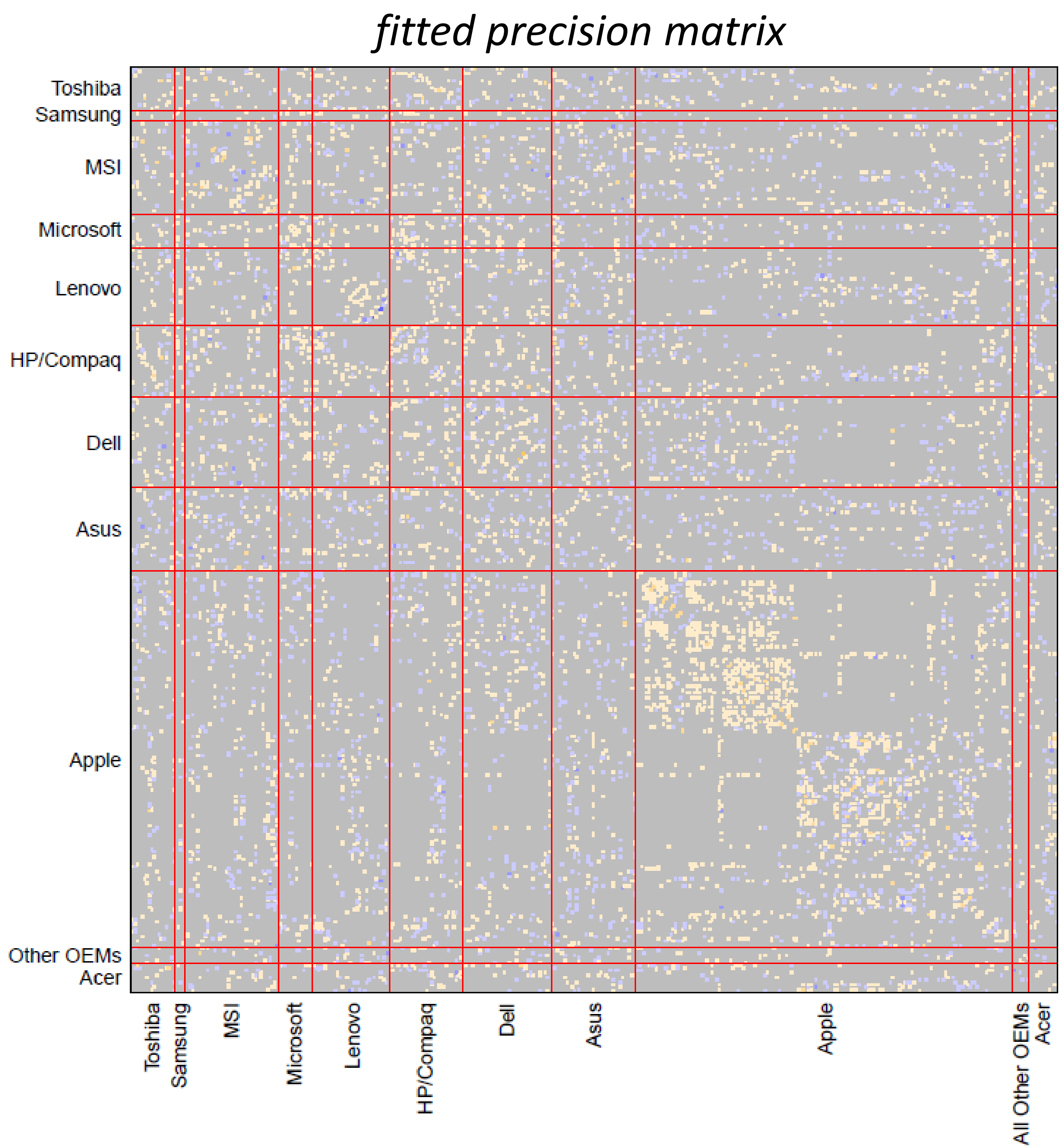


Cross-Product Sales Effects

Products move together

We can fit a graph of conditional dependencies (i.e., the precision) as a function of distance in both observables and latent space.

(using roll-outs for identification)



Product Co-occurrence

$$q_{tj} = \alpha_j(\boldsymbol{d}_t) + \gamma_j \log p_{tj} + \varepsilon_{tj}$$

Ignoring price (and demand shifters), a market with many products yields many big J-vectors of correlated quantities $\boldsymbol{q}_t = [q_{t1} \cdots q_{tJ}]'$.

Classic "data mining" seeks *association rules* in *market baskets*:

Find $j \neq k$ pairs so that $\mathbb{E}[q_{tj}q_{tk}] \gg \mathbb{E}q_{tj}\mathbb{E}q_{tk}$

Contemporary ML (e.g., from NLP) has much better tools for this.

Embedding

Tools like word2vec and glove map from the J-dimensional discrete vocabulary word-occurrence space to [two] vector *embeddings* in $\mathbb{R}^S$, and $S \ll J$

Say $q_{tj} = \mathbb{1}[\text{word } j \text{ occurs in sentence } t]$

$$\max\left\{ \sum_t \sum_{j,k} q_{tj} q_{tk} \boldsymbol{u}_j' \boldsymbol{v}_k \; - \; A(\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{C}) \right\}$$

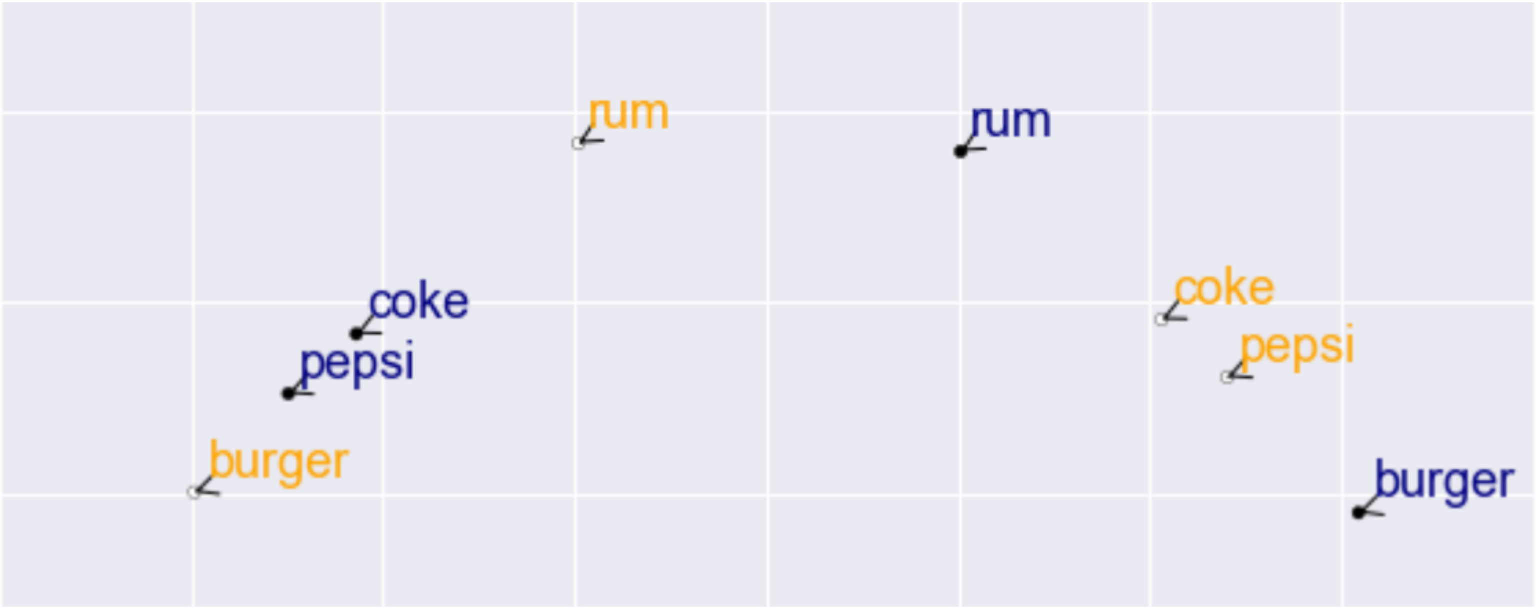
where $c_{jk} = \sum_t q_{tj} q_{tk}$ and $A(\cdot)$ is a normalizing constant

Word2vec uses A from binomial logit models (Poisson model works nicely also)

Glove uses A for the WLS obj. $\sum_{j,k} w_{jk} \left(c_{jk} - \boldsymbol{u}_j' \boldsymbol{v}_k\right)^2$ with $w_{jk} = \mathbb{1}[c_{jk} > 0]$

Product Embeddings

words are like products and sentences are like baskets



substitutes (synonyms) are close in the same vector space
complements (topical words) are close across vector spaces

Supermarkets products are organized into narrowly defined modules

If we believe that modules contain mostly substitutes, then same-space embedding neighbors should often be in the same module

1-Nearest Neighbor Module Prediction

Recall WLS objective

$$\sum_{j,k} w_{jk} \left(c_{jk} - \boldsymbol{u}_j'\boldsymbol{v}_k\right)^2$$

Moving inside a demand system (AIDS)

It's *almost* ideal:

$$\boldsymbol{s}_t \;=\; \boldsymbol{\alpha} \;+\; \boldsymbol{\Gamma}\log(\boldsymbol{p}_t) \;+\; \boldsymbol{\beta}\log\frac{e_t}{\phi_t} \;+\; \boldsymbol{\varepsilon}_t$$

s_{tj} is the budget share for product j in basket t and e_t is the budget

($e_t = \sum_j \$_{tj}$ and $s_{tj} = \$_{tj}/e_t$)

ϕ_t is the translog price index $\sum_j \log p_{tj} \left[\alpha_j + \sum_k \gamma^{\star}_{jk} \log p_{tk}\right]$

(which we will replace with a plug-in for estimation)

This is meaningful after aggregation, and we can actually estimate it

Factorizing Γ

The price terms are key to finding complements and substitutes

$$\mathbb{E}s_{tj} = \alpha_j + \sum_k \gamma_{jk} \log p_{tj} + \beta_j \frac{e_t}{\phi_t}$$

$\boldsymbol{\Gamma}$ is $J \times J$, so we need to reduce dimension if J is going to go big

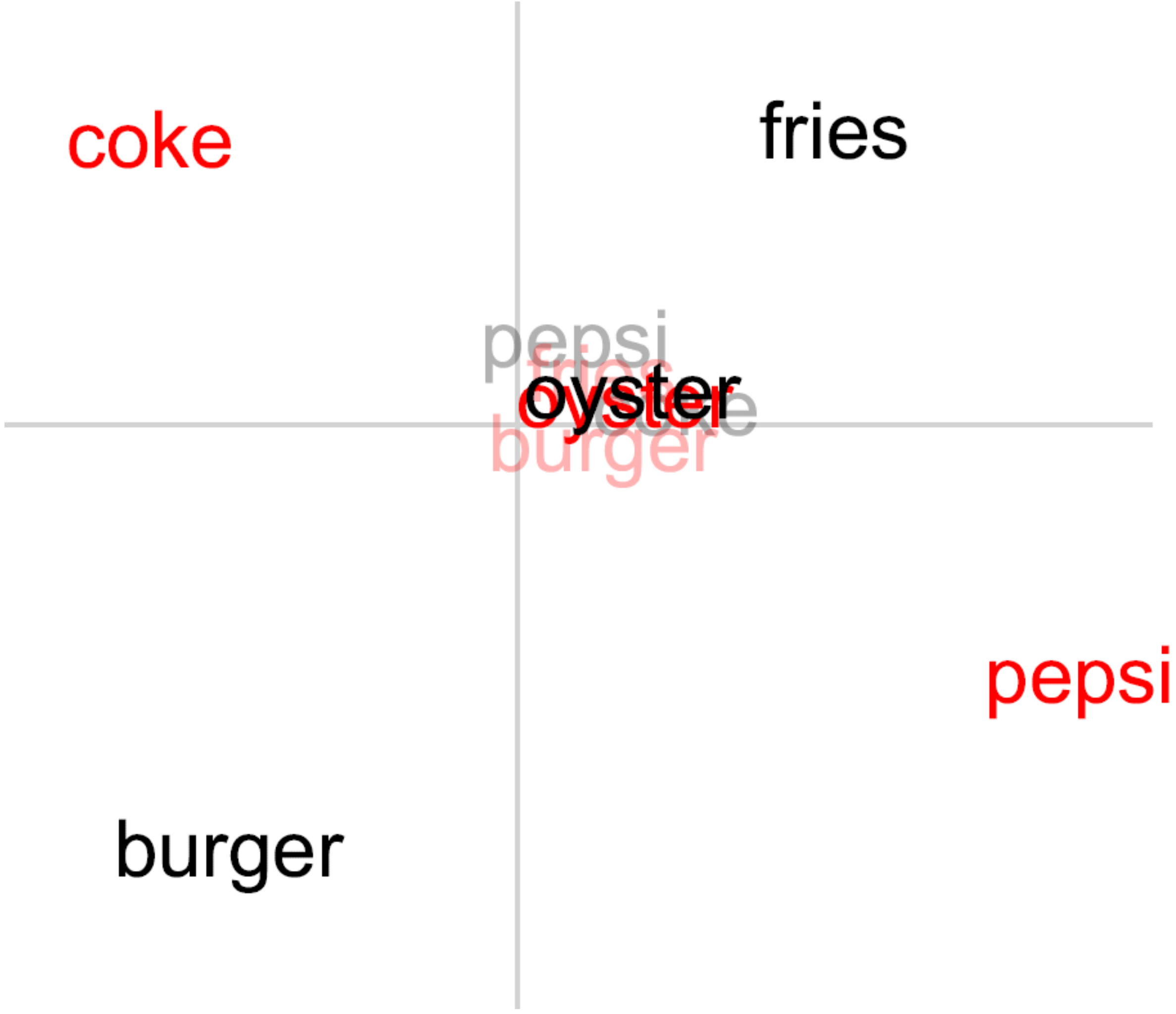
One option: square matrix factorizations from word/prod embedding

Rewrite $\boldsymbol{\Gamma} = \boldsymbol{UV}' + \boldsymbol{D}$ where $\boldsymbol{u}_j, \boldsymbol{v}_j$ are S-vectors and D is J-diagonal

Embedding lunch on the cape

Γ	coke	pepsi	burger	fries	oyster
coke	-0.25	0.25	0.000	0.000	0.0
pepsi	0.25	-0.25	0.000	0.000	0.0
burger	0.00	0.00	0.125	-0.125	0.0
fries	0.00	0.00	-0.125	0.125	0.0
oyster	0.00	0.00	0.000	0.000	0.0

	coke	pepsi	burger	fries	oyster
alpha	0.15	0.15	0.50	0.1	0.10
beta	0.02	0.02	-0.15	0.1	0.01
mean price	0.50	0.50	3.00	1.0	0.50



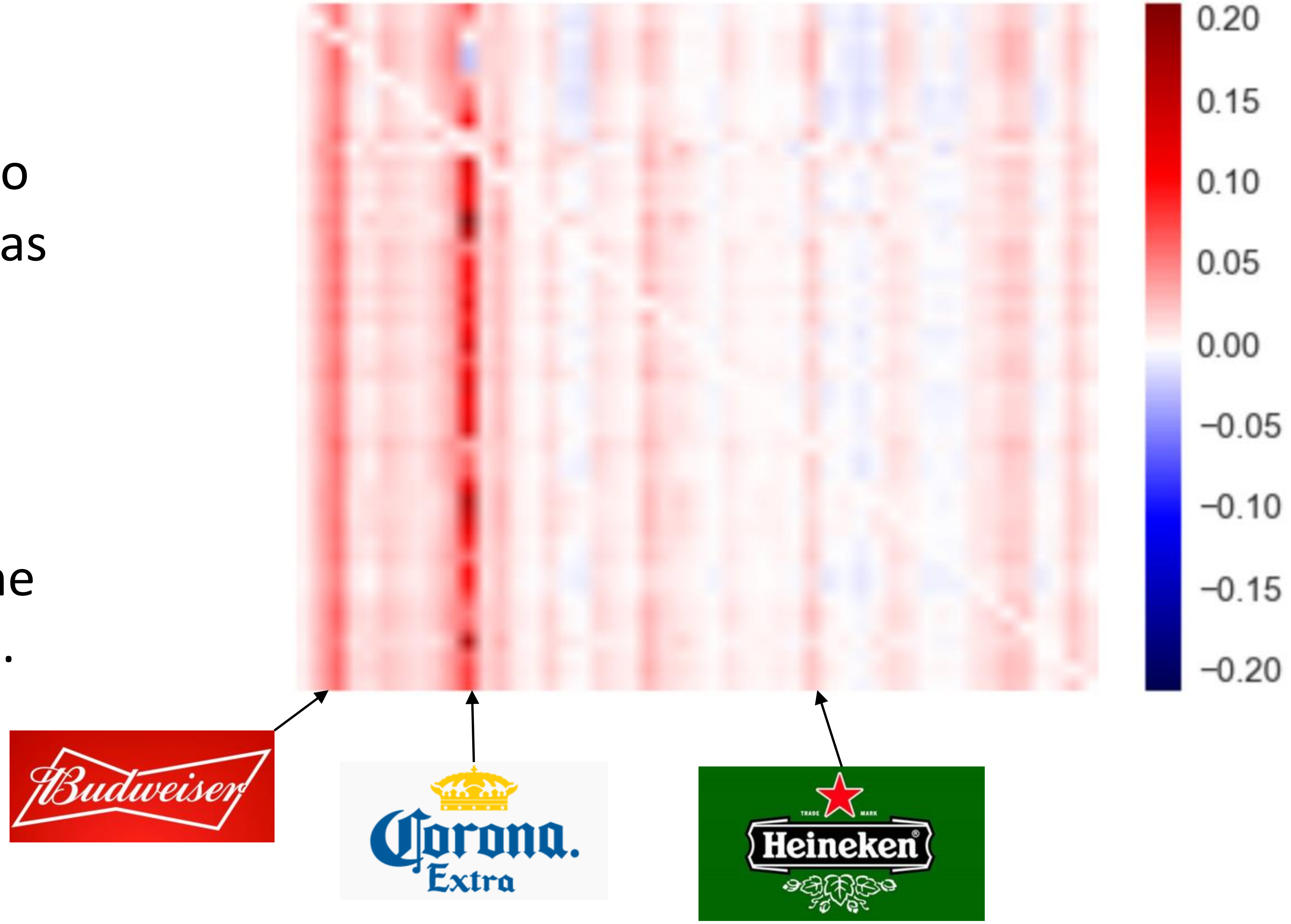
Back to beer

Translate the γ_{jk} values into [compensated] elasticities as

$$\frac{\gamma_{jk}}{\overline{s_j}} - \overline{s_k} - \mathbb{1}_{[k=j]}$$

and you can also get income effects, uncomp. elast., etc.

Elasticity matrix (omitting diagonal)



Just a start

We're doing all this to scale up to Really Big J Products, keeping things fast enough to have parameters change for subgroups of transactions.

AIDS implies restrictions: $\gamma_{jk} = \gamma_{kj}, \quad \sum_j \gamma_{jk} = \sum_j \gamma_{kj} = \sum_j \beta_j = 0$

We're going to use these (esp. symmetric matrix factorization).

The big agenda: bring ML into models that Econ says are measurable