

GA²M and EBM

Generalized Additive Models
with Interactions

Explainable Boosting Machines

BLUF: They work.

Dataset/AUROC	Domain	Logistic Regression	Random Forest	XGBoost	Explainable Boosting Machine
Adult Income	Finance	.907±.003	.903±.002	.927±.001	.928±.002
Heart Disease	Medical	.895±.030	.890±.008	.851±.018	.898±.013
Breast Cancer	Medical	.995±.005	.992±.009	.992±.010	.995±.006
Telecom Churn	Business	.849±.005	.824±.004	.828±.010	.852±.006
Credit Fraud	Security	.979±.002	.950±.007	.981±.003	.981±.003

GAM

What's a GAM? A linear model, where each term is allowed to be a nonlinear function.

$$g(E[y]) = \sum f_i(x_i)$$

Link function.

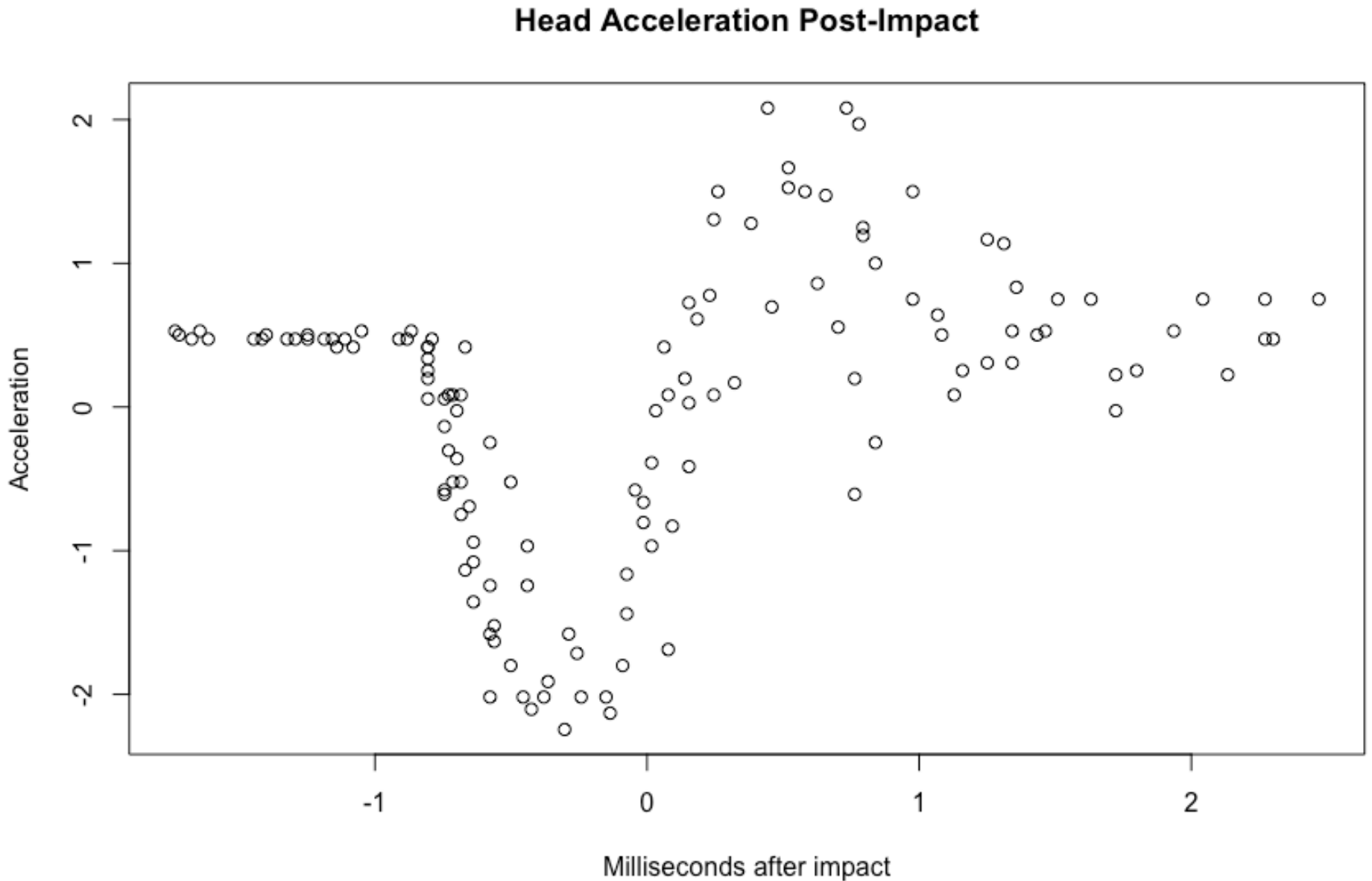
(logit, probit, any
GLM thing)

Functions of single input predictors

Any function here, no matter how complicated, still provides interpretability.

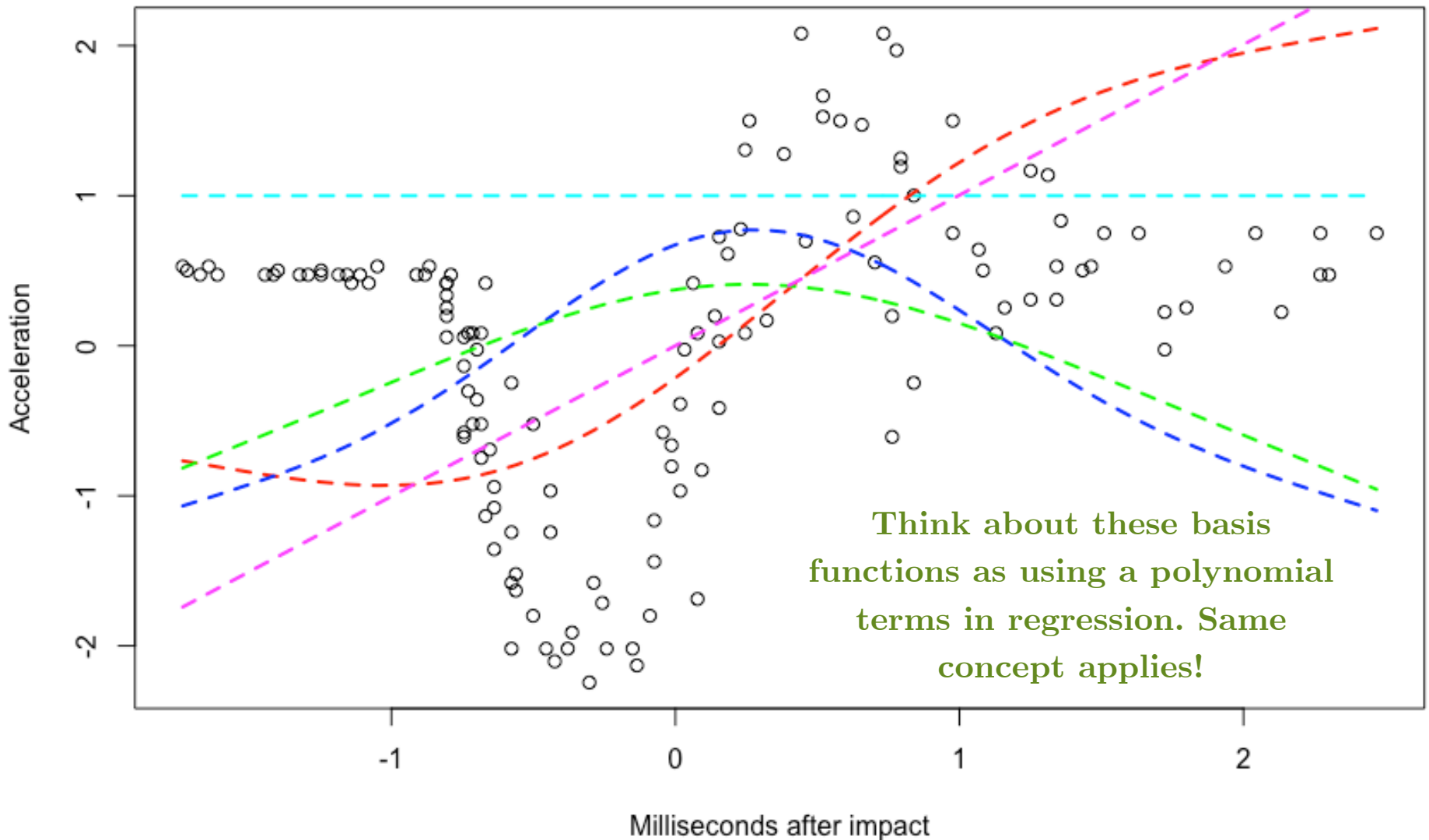
And we don't need to choose! In practice, just use splines to create piecewise smoothing functions.

(Smooth) GAM - Single $f_i(x_i)$ Example



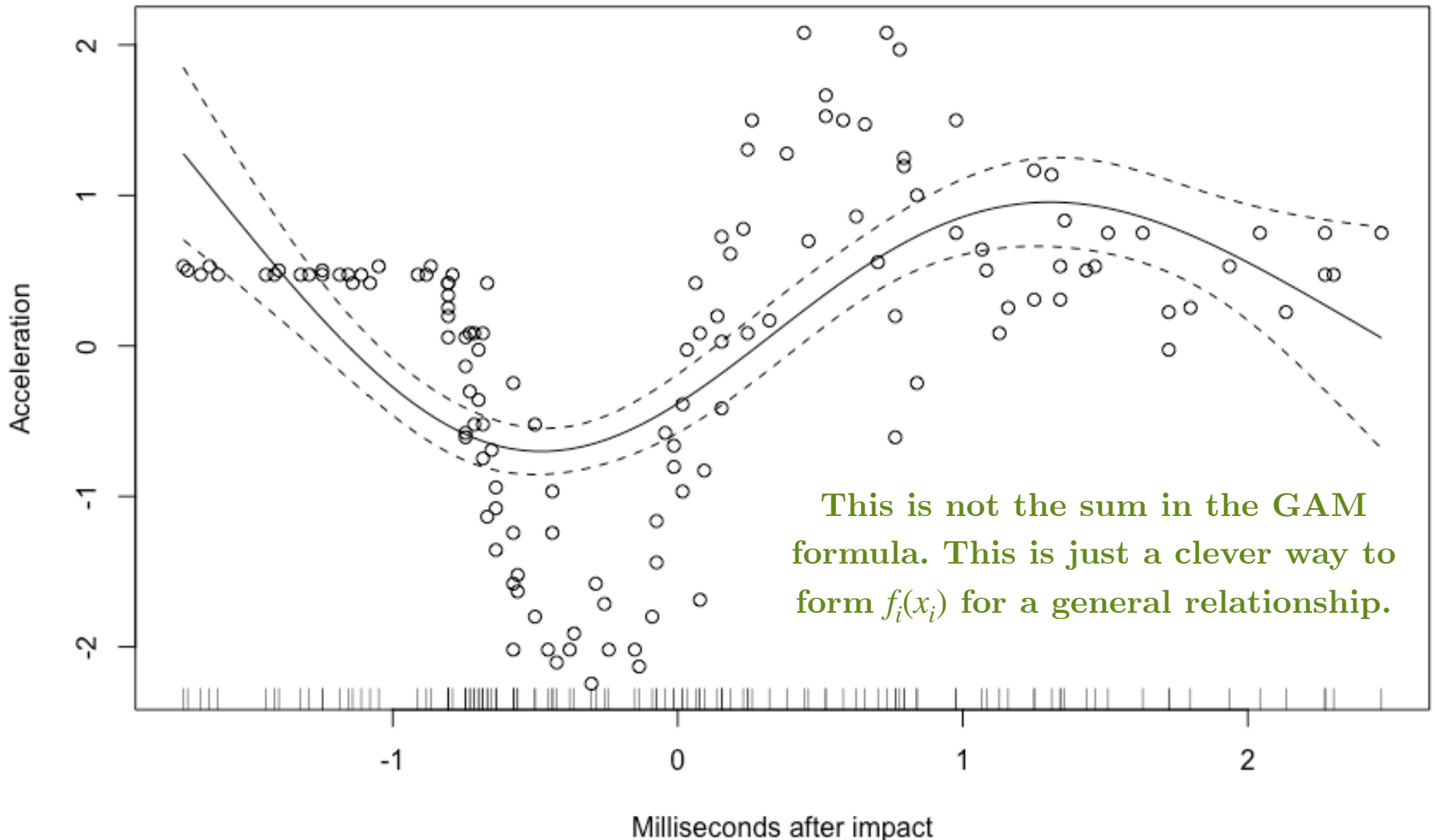
(Smooth) GAM - Single $f_i(x_i)$ Example

5 Basis Functions for Spline



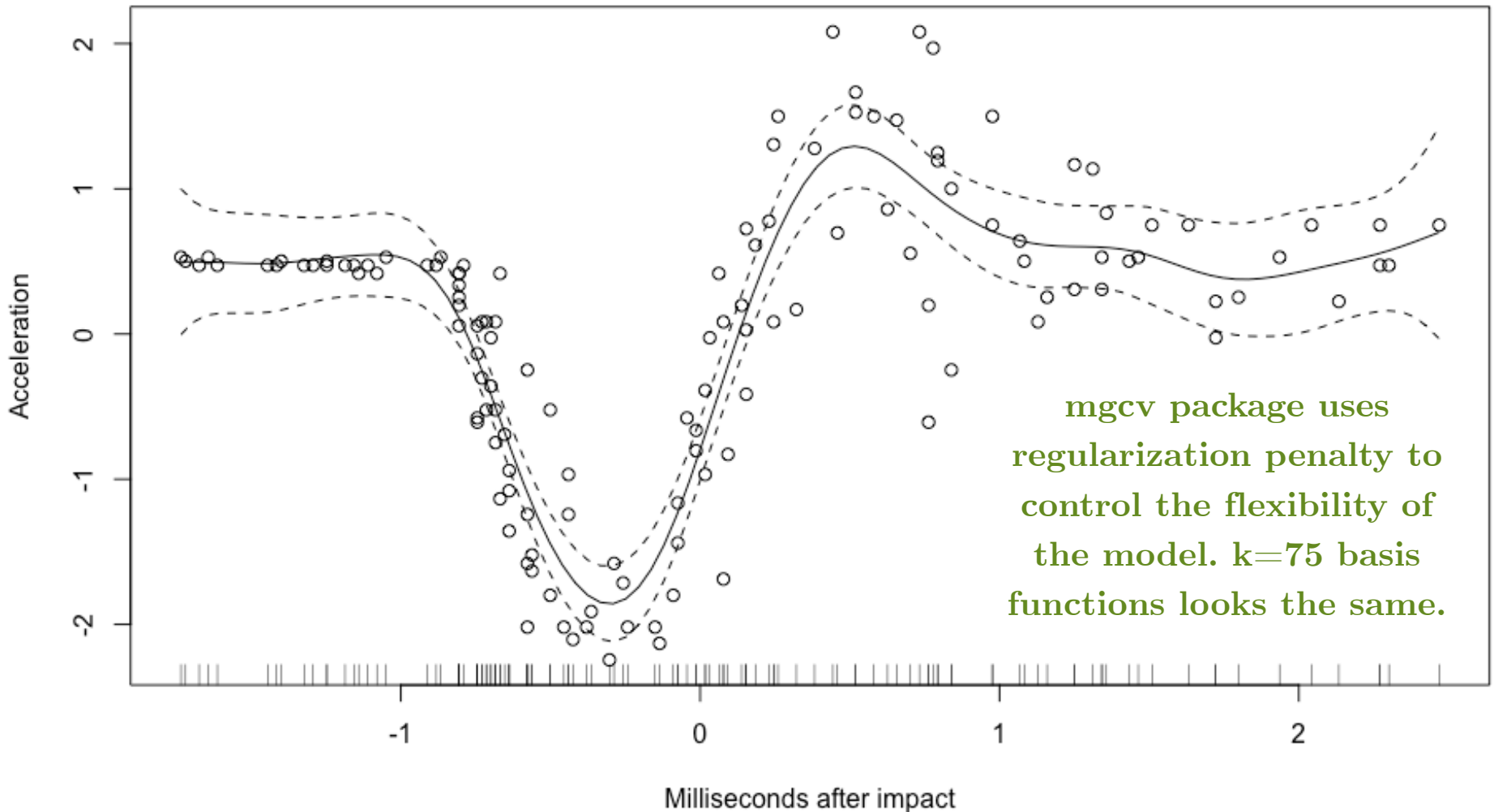
(Smooth) GAM - Single $f_i(x_i)$ Example

Weighted Sum of Basis Functions



(Smooth) GAM - Single $f_i(x_i)$ Example

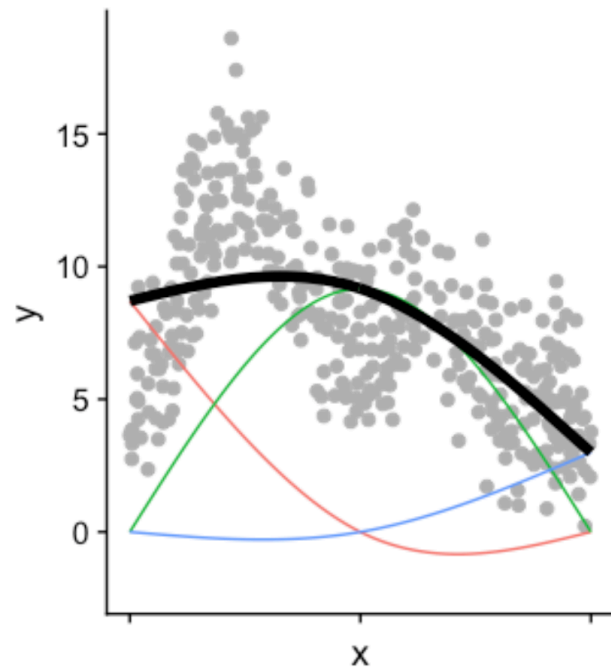
Weighted Sum of 22 Basis Functions



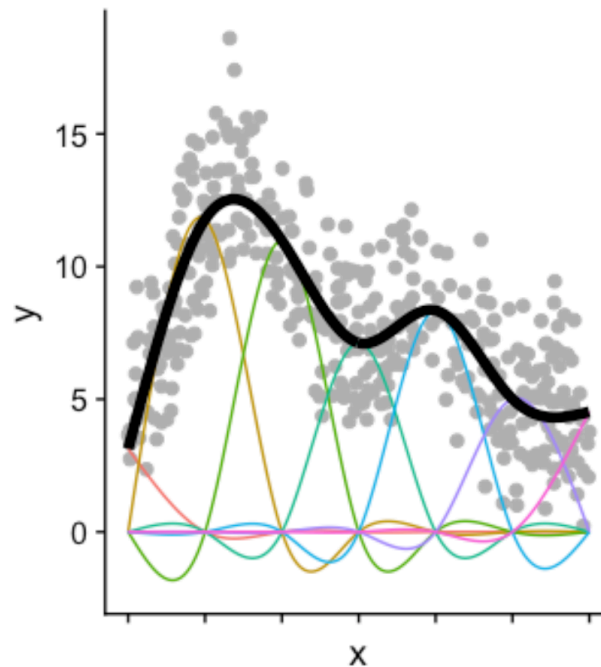
Regularization

- Number of basis functions for the spline affects the fit.
- Not that important since **overfitting is regularized out**.
Need sufficient #, just avoid too many for compute time.

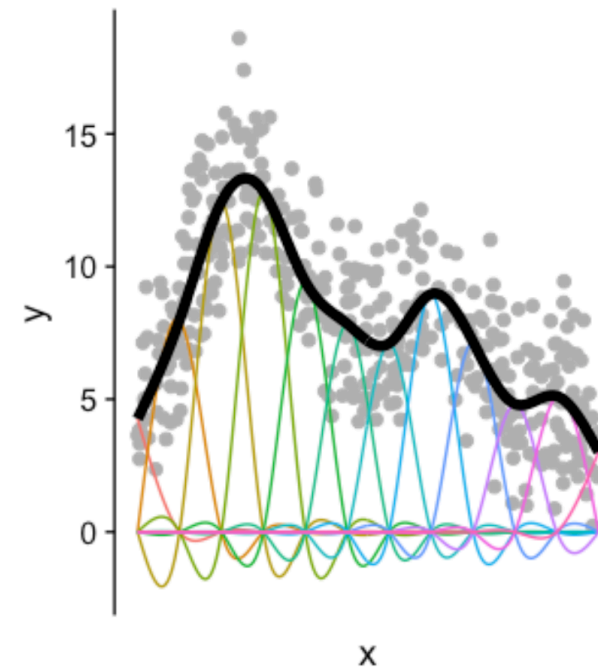
3 Basis Functions



7 Basis Functions

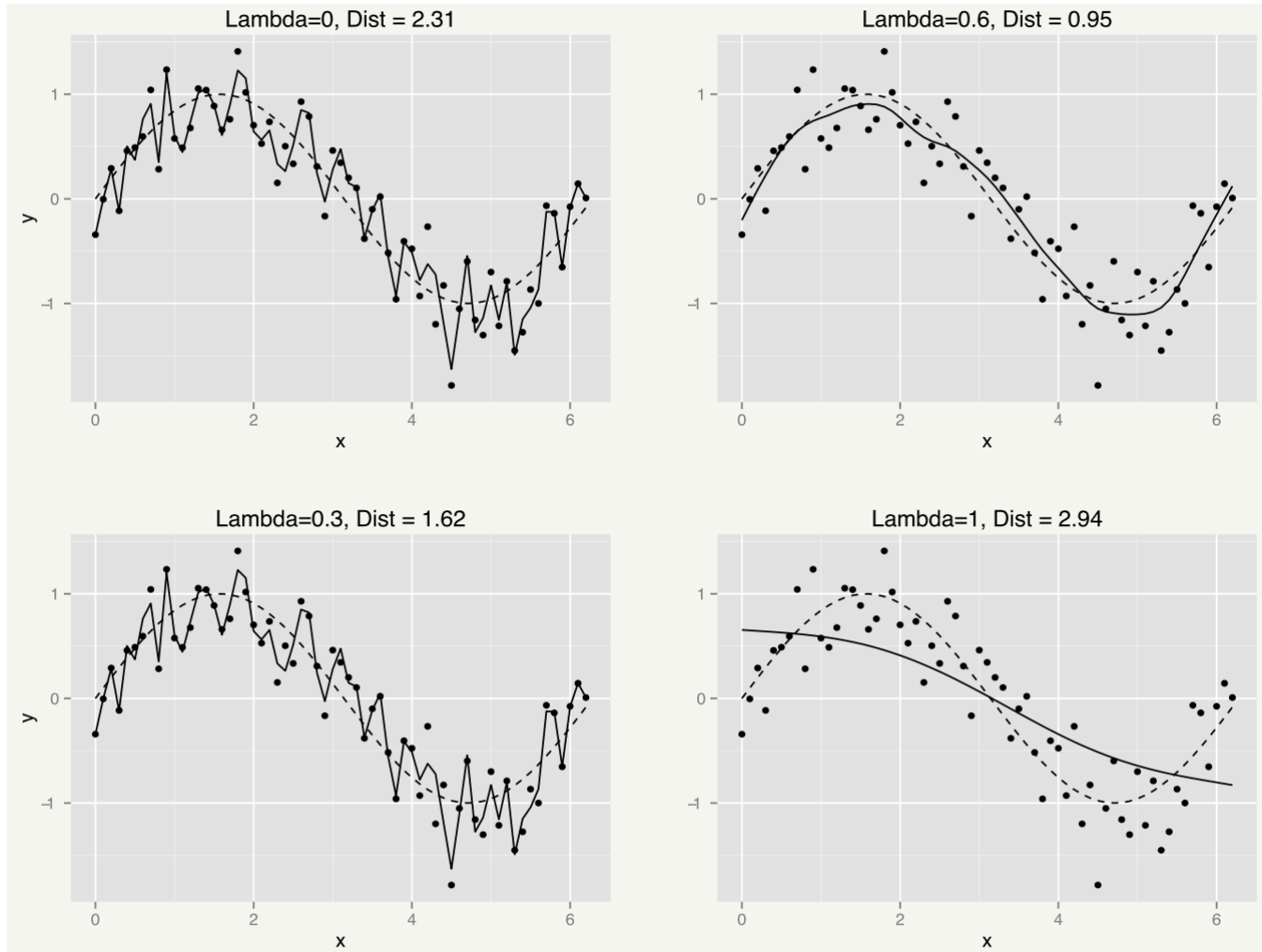


12 Basis Functions



Regularization

λ (regularization parameter) is optimized for you in mgcv package.



GAMs are Explainable

But not in the *global* sense of linear model. Each component is nonlinear. This means the variables do not have a per-unit-increase type of effect!

```
library(gamair)
library(mgcv)
data("mpg", package="gamair")
```

```
model <- gam(city.mpg ~ s(weight) + s(length) + s(price),
              data = mpg, method = "REML")
```

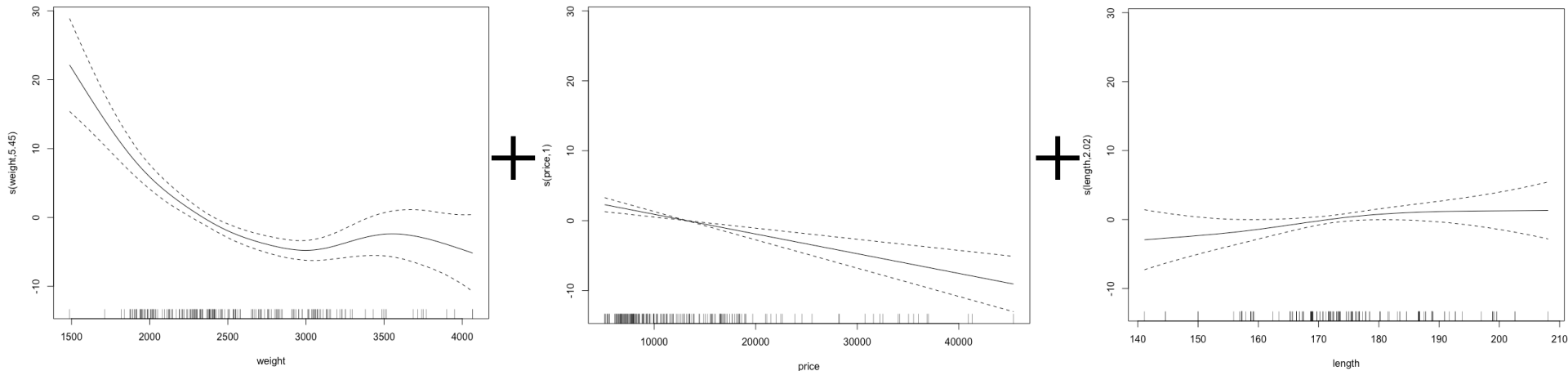


Spline smoothing functions

GAMs are Explainable

```
# Plot the model
```

```
plot(model, all.terms = TRUE, pages = 1)
```



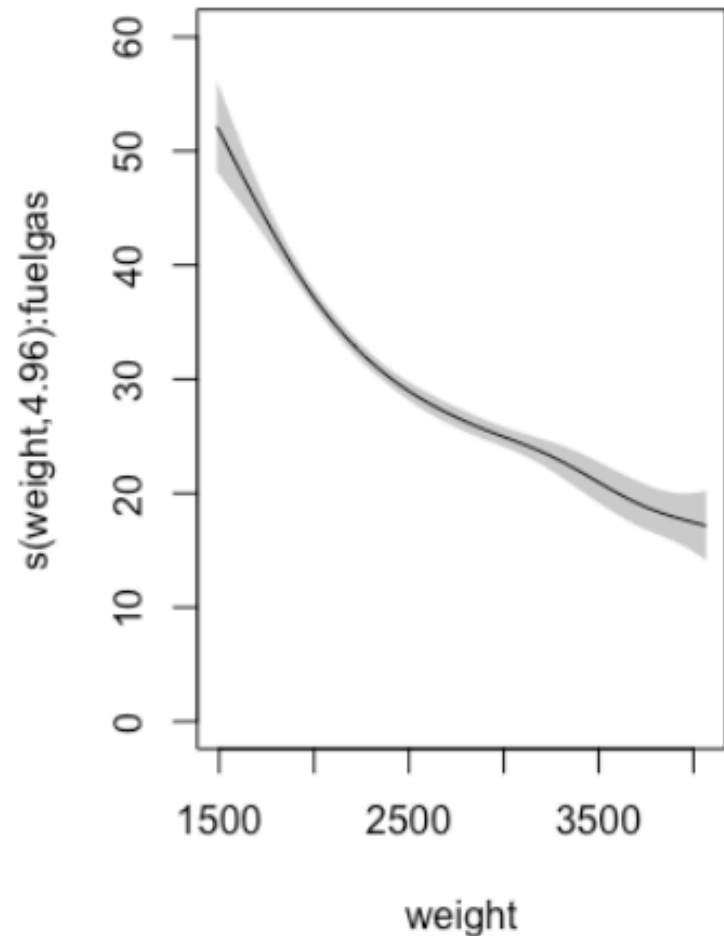
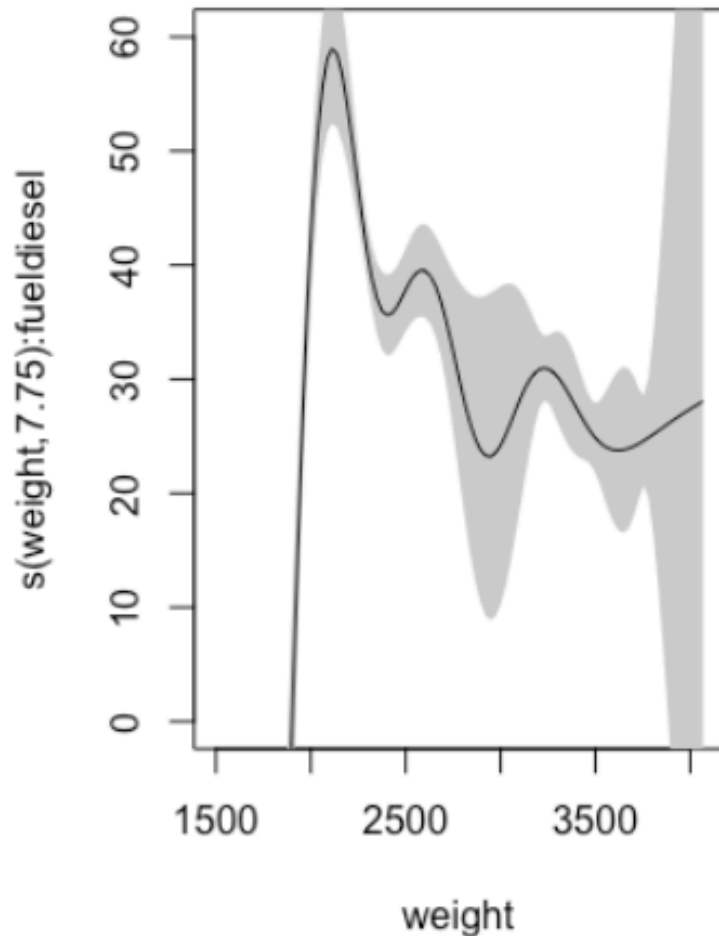
Each variable's contribution to the prediction is perfectly transparent

GAMs with interactions

- Generally speaking, GAMs have always been able to handle interactions. This is not new.
- We can create component functions of the form $f_{ij}(x_i, x_j)$ that add a 2-dimensional component to our model.
- Still interpretable - can see exactly the relationship of x_i on the target, even though it may depend on x_j .

GAMs with interactions

Still interpretable - can see exactly the relationship of x_i on the target, even though it may depend on x_j .



GAMs with interactions

But how to find interactions in a fast, efficient way?



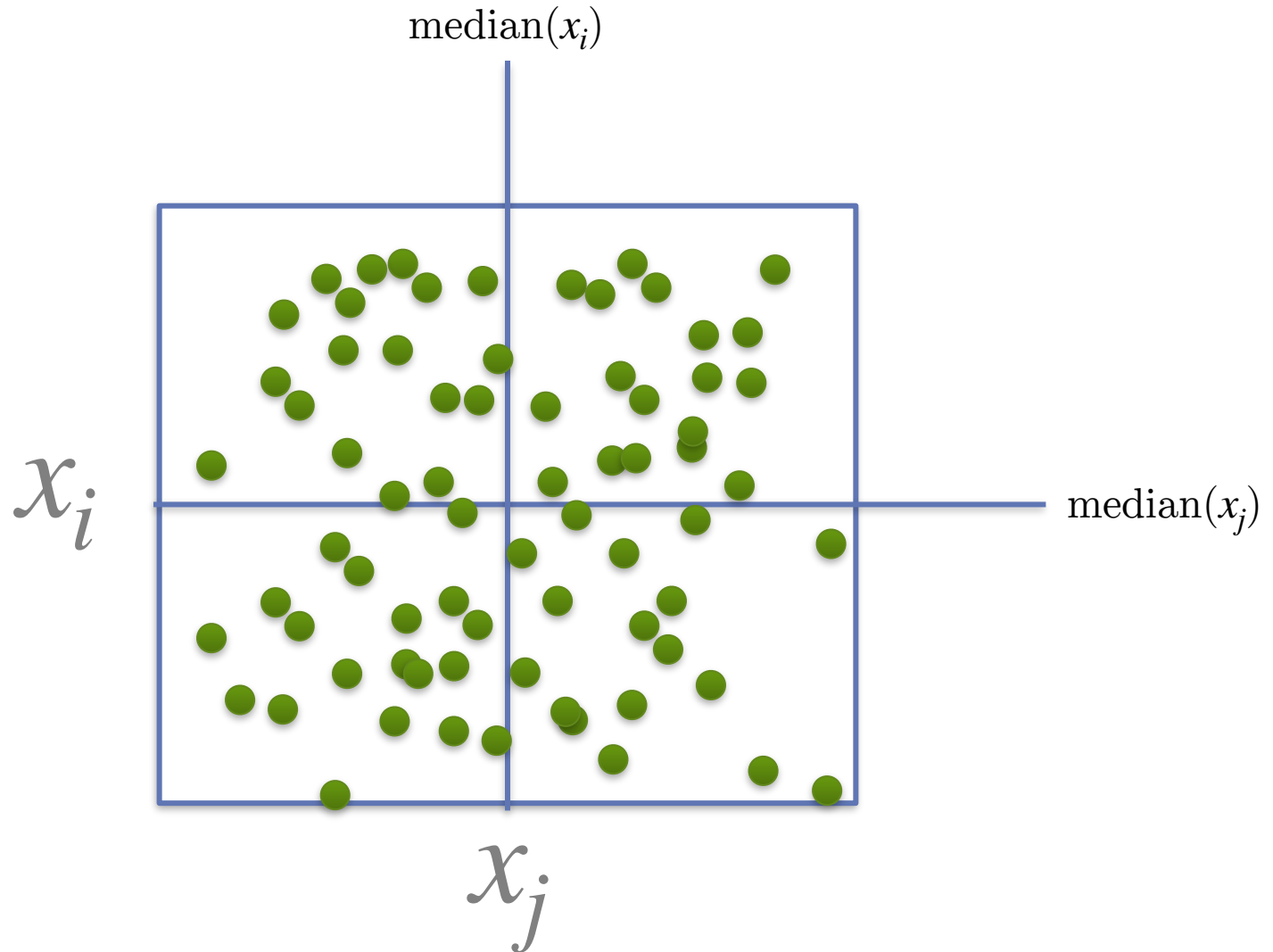
GUIDE Algorithm

...

Generalized, Unbiased, Interaction Detection and Estimation.

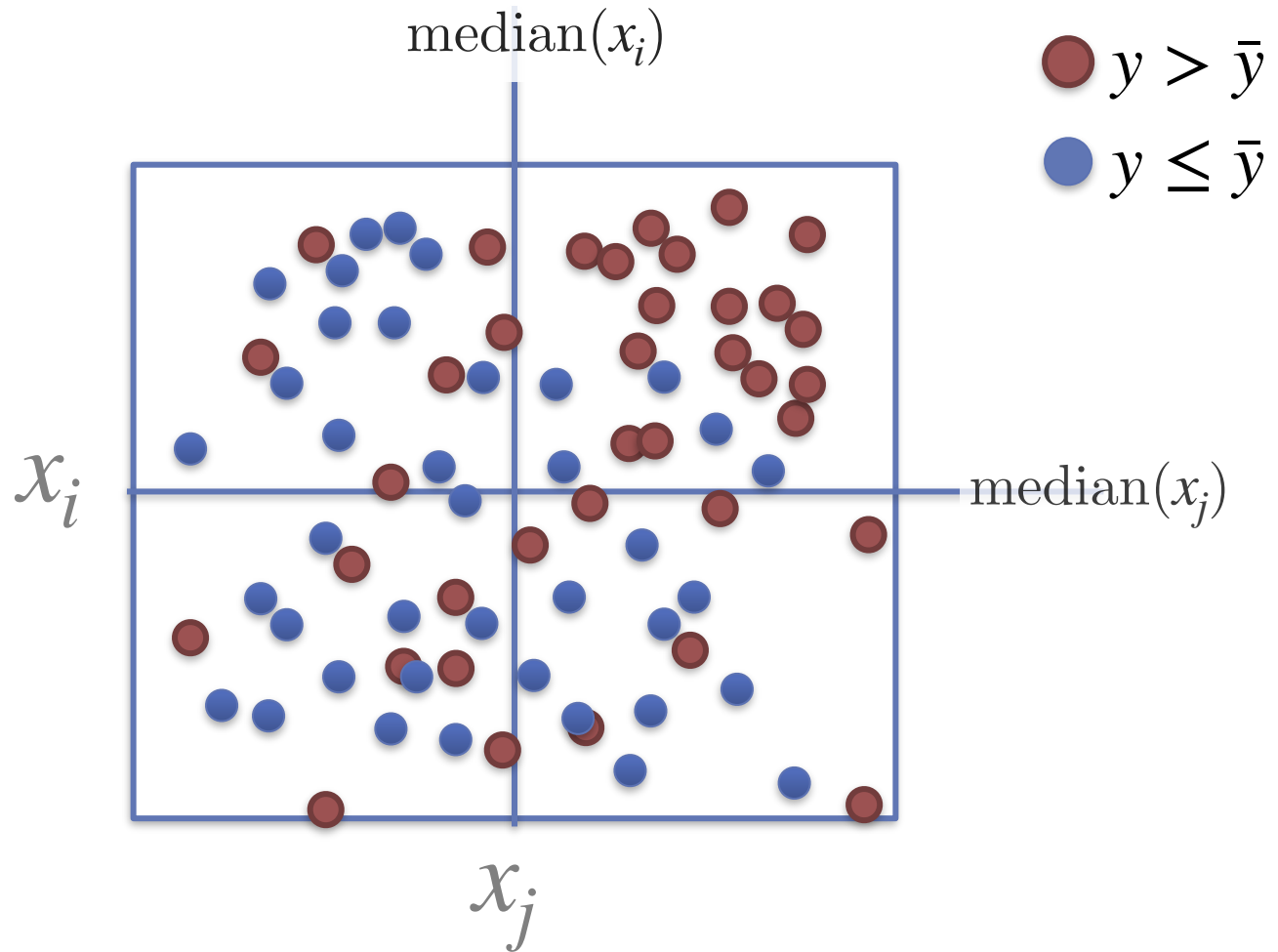
GUIDE Algorithm

(Continuous Attributes)

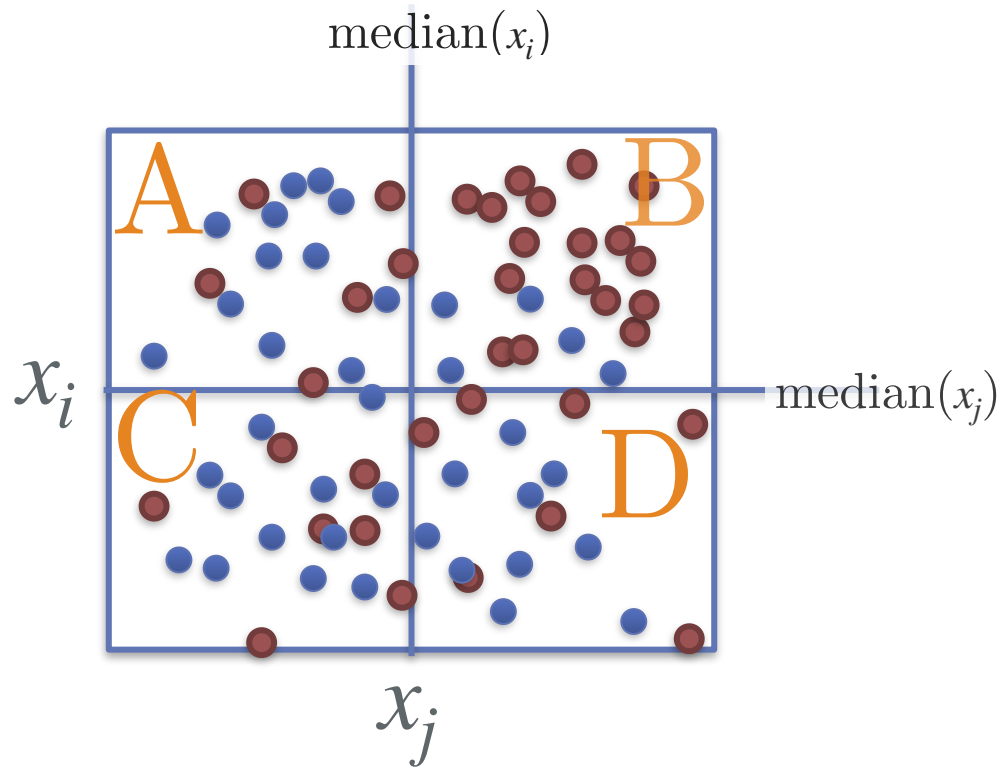


GUIDE Algorithm



(Continuous Attributes)



GUIDE Algorithm: Continuous Attributes



Chi-square test on this table comparing proportion of positive residuals from a constant model in each quadrant of space.

	A	B	C	D
 $y > \bar{y}$	6	17	7	7
 $y \leq \bar{y}$	12	5	12	10

GUIDE Algorithm: Categorical Attributes

$x_i = \text{Project Ongoing}$

Yes

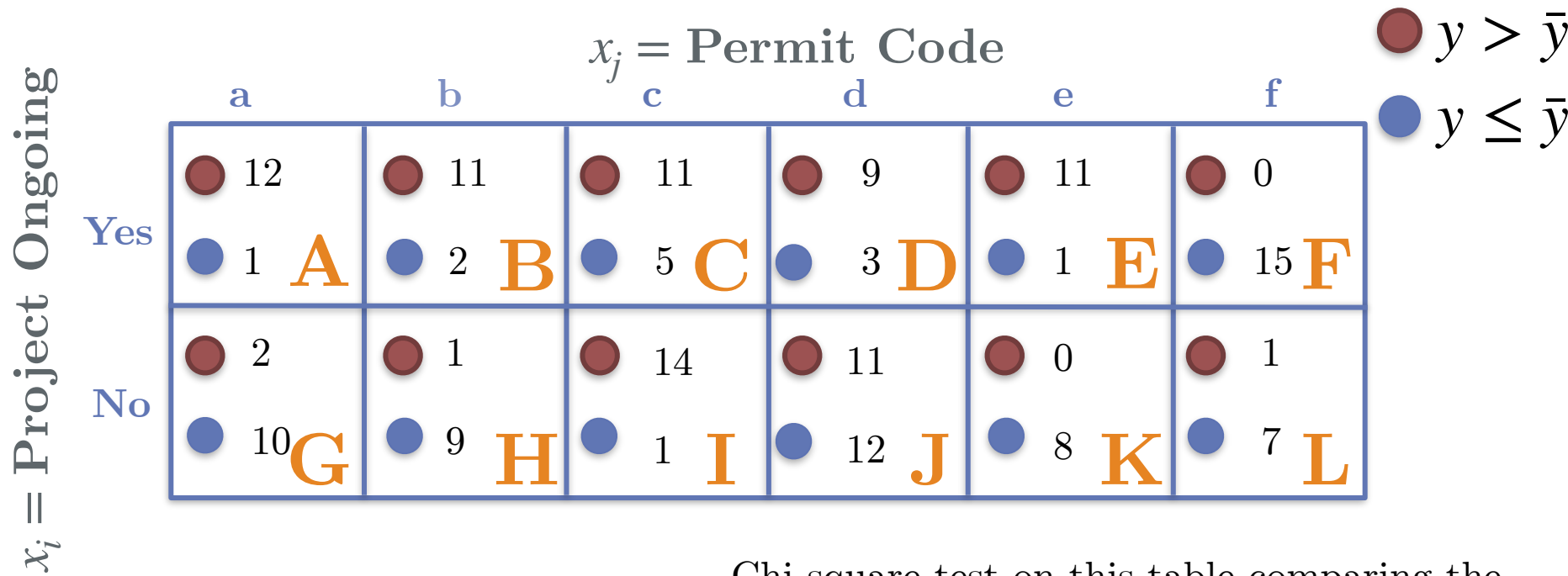
No

$x_j = \text{Permit Code}$					
a	b	c	d	e	f
<div><div></div>12</div> <div><div></div>1</div> <div>A</div>	<div><div></div>11</div> <div><div></div>2</div> <div>B</div>	<div><div></div>11</div> <div><div></div>5</div> <div>C</div>	<div><div></div>9</div> <div><div></div>3</div> <div>D</div>	<div><div></div>11</div> <div><div></div>1</div> <div>E</div>	<div><div></div>0</div> <div><div></div>15</div> <div>F</div>
<div><div></div>2</div> <div><div></div>10</div> <div>G</div>	<div><div></div>1</div> <div><div></div>9</div> <div>H</div>	<div><div></div>14</div> <div><div></div>1</div> <div>I</div>	<div><div></div>11</div> <div><div></div>12</div> <div>J</div>	<div><div></div>0</div> <div><div></div>8</div> <div>K</div>	<div><div></div>1</div> <div><div></div>7</div> <div>L</div>

$y > \bar{y}$

$y \leq \bar{y}$

GUIDE Algorithm: Categorical Attributes











Chi-square test on this table comparing the proportion of positive residuals from a constant model in each combination of levels.



	A	B	C	D	E	F	G	H	I	J	K	L
● $y > \bar{y}$	12	11	11	9	11	0	2	1	14	11	0	1
● $y \leq \bar{y}$	1	2	5	3	1	15	10	9	1	12	8	7

GUIDE Algorithm: Mixed Attributes



$x_j = \text{Project Ongoing}$

$x_i = \text{Market Price}$

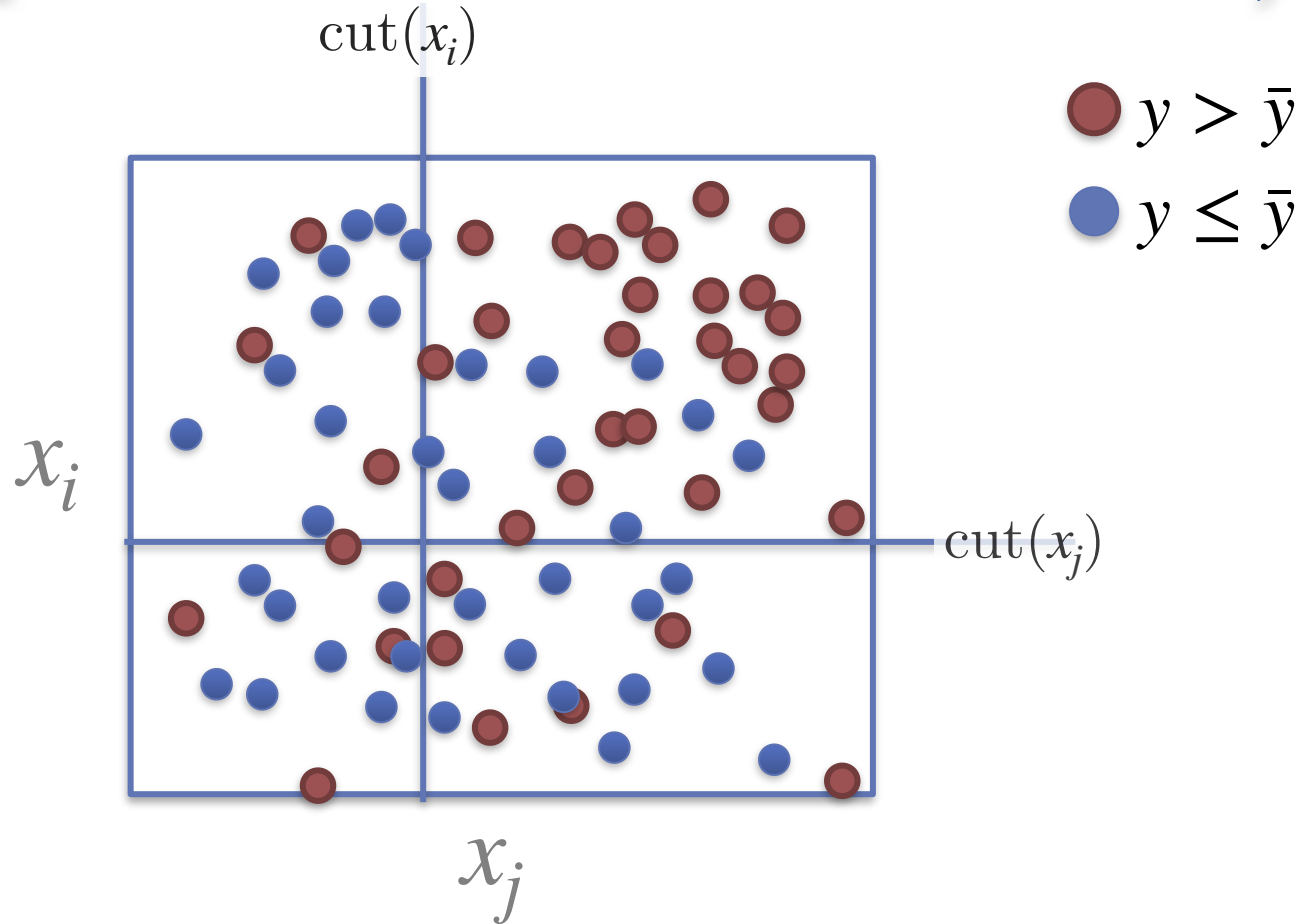
		Yes	No	
$x_i > \text{med}(x_i)$	 62	 20		
	 22 A	 30 B		
$x_i \leq \text{med}(x_i)$	 25	 40		
	 35 C	 9 D		

 $y > \bar{y}$
 $y \leq \bar{y}$

Chi-square test on this table comparing the proportion of positive residuals from a constant model in each combination of levels.

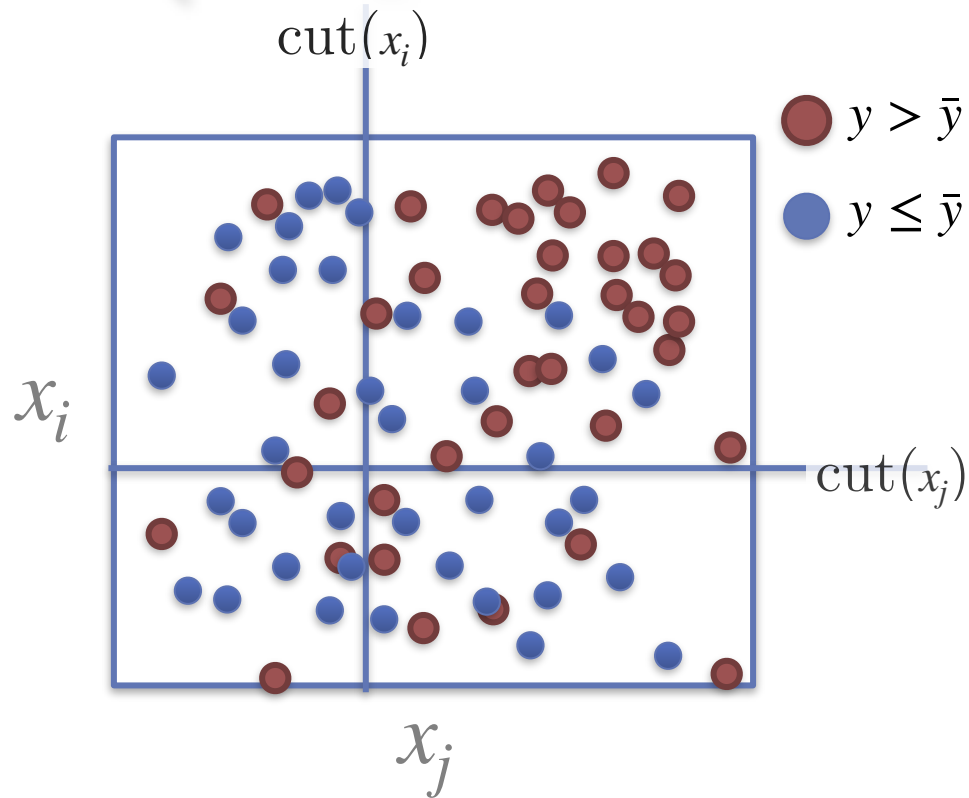
	A	B	C	D
 $y > \bar{y}$	62	20	25	40
 $y \leq \bar{y}$	22	30	35	9

FAST Algorithm (Improvement in GA²M)



Rather than use the median, compute the ideal splits: $\text{cut}(x_i)$ and $\text{cut}(x_j)$
The strength of interaction is measured by loss function on this simple tree model.

FAST Algorithm (Improvement in GA²M)



This can be done *super fast* with some clever book keeping.

We don't search every possible value for $\text{cut}(x_i)$, just d_i possible bins to split on.

This is done in $O(d_i, d_j)$

Rather than use the median, compute the ideal splits: $\text{cut}(x_i)$ and $\text{cut}(x_j)$
The strength of interaction is measured by loss function on this simple tree model.

GA²M quickly became
EBM

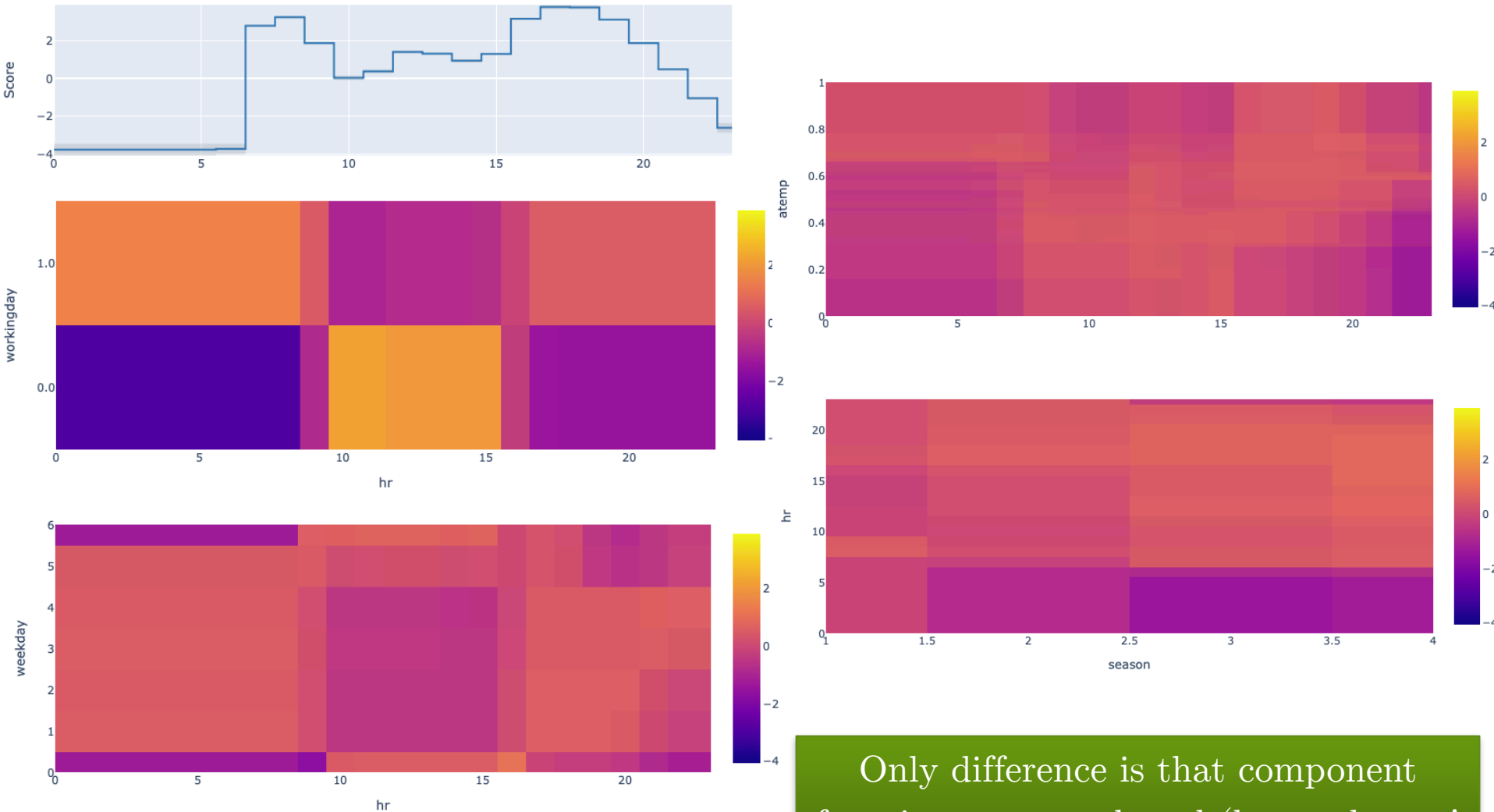
...

The Explainable Boosting Machine

Explainable Boosting Machines

- Microsoft's new Explainable Boosting Machines are GA²Ms that opt for gradient boosted trees over splines.
- First 1-dimensional components are learned (slowly, in round-robin format to avoid a result that depends on variable order).
- Then 2-dimensional components detected and trained on residual

EBMs are still fully transparent.



Only difference is that component functions are tree-based (boosted trees!)

Ideal Implementation

- Make your best GAM using component functions of 1 variable.
- Repeat until nothing is added to the model:
 1. Return residuals from that model and select the best interaction to predict the residual with FAST.
 2. Re-fit the model with your original components and the interaction term.

Alas, this is *expensive* for large datasets because of all that model fitting.

Faster Implementation

- Stage 1: Make your best GAM using component functions of 1 variable. Fix that part of the model, coefficients will not change in stage 2.
- Stage 2: Find best interaction to model residuals from stage 1 and create a shape function for that interaction.
 - Repeat until nothing is added to the model:
 1. Return residuals from previous model and select the best interaction to predict the residual with FAST.
 2. Re-fit the residual model with additional interaction term.

Here, we have interaction terms allowed to change with addition of new interaction terms, but not main effects.

Fastest Implementation

- Stage 1: Make your best GAM using component functions of 1 variable. Fix that part of the model, coefficients will not change in stage 2.
- Stage 2: Find k best interaction terms to model residuals from stage 1, ranked according to FAST chi-square tests. Fit residual model using all k interaction terms at once.

Here, we don't let the selection of interaction terms affect selection of other interaction terms.

Faster

vs.

Fastest

Suppose age*gender was most significant interaction, age*salary was second most.

After fitting age*gender into the model, you find age*salary is no longer most significant.

Well, that wouldn't happen because you'd take all top k interactions in isolation.

interpretML/interpret

EBM now available in Python and R

- `pip install interpret`
- Rapidly expanding library

Interpretability Technique	Type	Examples
Explainable Boosting	glassbox model	Notebooks
Decision Tree	glassbox model	Notebooks
Decision Rule List	glassbox model	Coming Soon
Linear/Logistic Regression	glassbox model	Notebooks
SHAP Kernel Explainer	blackbox explainer	Notebooks
SHAP Tree Explainer	blackbox explainer	Coming Soon
LIME	blackbox explainer	Notebooks
Morris Sensitivity Analysis	blackbox explainer	Notebooks
Partial Dependence	blackbox explainer	Notebooks

Resources

- [GAM course in R \(Free+Interactive\)](#)
 - By Noam Ross - also has videos/blogs/notebooks
- [Accurate Intelligible Models w/ Pairwise Interactions](#)
- Python package [interpret](#)
- R package [interpret](#) (minimal implementation)