TabPFN: SOTA Tabular AutoML in 1 Second?

Frank Hutter

University of Freiburg & Bosch Center for AI
fh@cs.uni-freiburg.de

Based on joint work with Samuel Müller, Noah Hollmann & Katharina Eggensperger
ICLR 2023 & best paper at the NeurIPS 2022 Workshop on Tabular Data

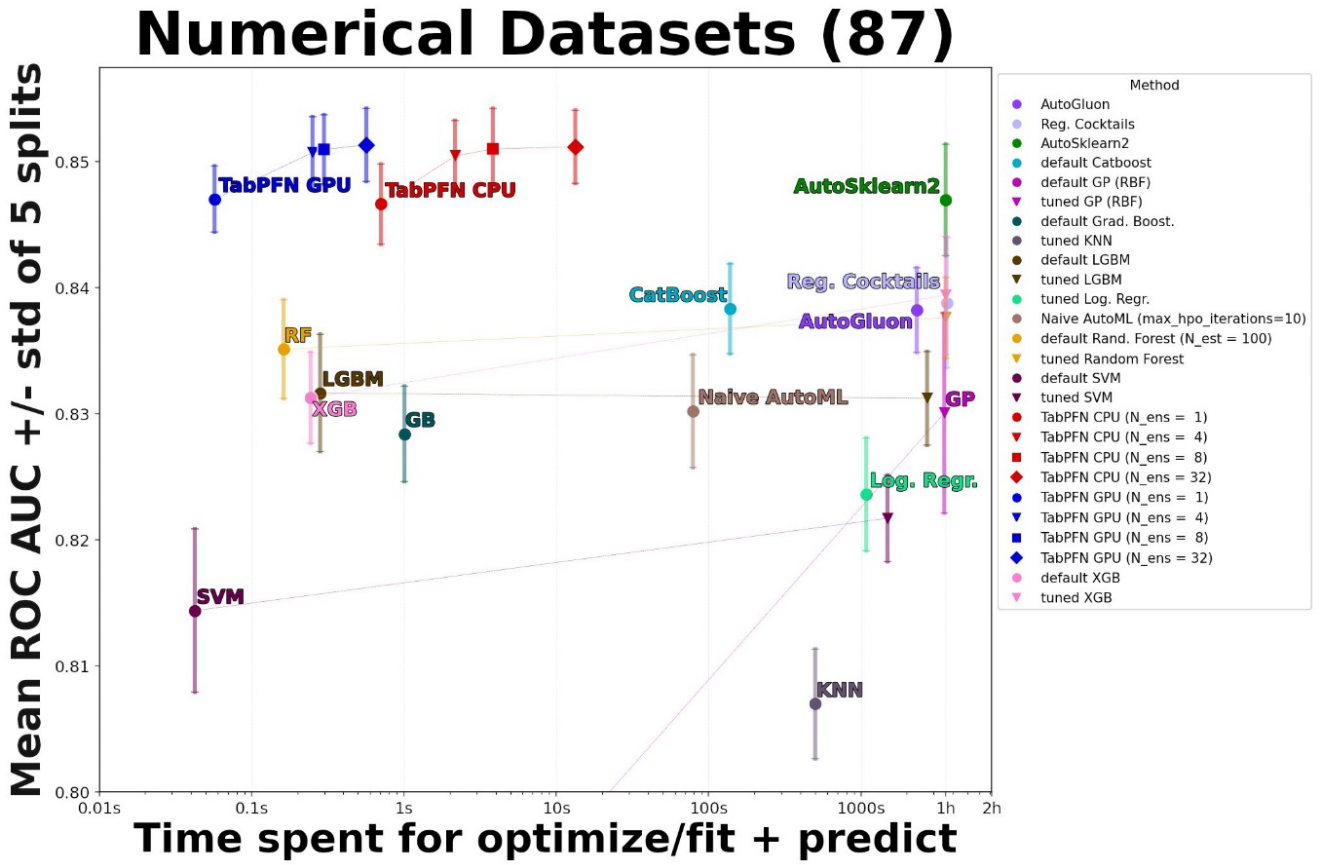
@FrankRHutter
@AutoML_org

These slides are available at www.automl.org/talks

Preview

- A radically new (GPT-3 like) approach for tabular classification
- **Better performance in 1s than than any other ML / AutoML method in 1h**

- **Current limitations**
  - Size: up to 1000 data points, 100 features, 10 classes
  - Not (yet) designed for: categorical features, missing values, uninformative features
  - High inference time



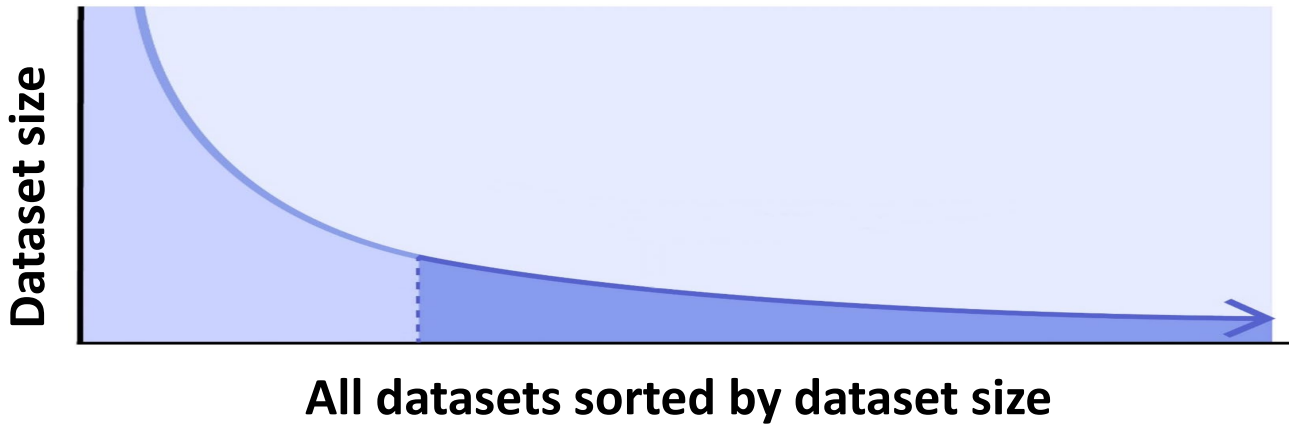
Premises

- **Tabular data** is the most common type of data
  - Yet, deep learning did not traditionally excel on it

company	division	sector	tryint
00nil_Combined_Company	00nil_Combined_Division	00nil_Combined_Sector	14625
apple	00nil_Combined_Division	00nil_Combined_Sector	10125
apple	hardware	00nil_Combined_Sector	4500
apple	hardware	business	1350
apple	hardware	consumer	3150
apple	software	00nil_Combined_Sector	5625
apple	software	business	4950
apple	software	consumer	675
microsoft	00nil_Combined_Division	00nil_Combined_Sector	4500
microsoft	hardware	00nil_Combined_Sector	1890
microsoft	hardware	business	855
microsoft	hardware	consumer	1035
microsoft	software	00nil_Combined_Sector	2610
microsoft	software	business	1215
microsoft	software	consumer	1395

- **Neural networks excel for large amounts of data**
  - But they are **slow** to train
  - But they are **prone to overfitting on small datasets**

- We care about the **long tail of small datasets**
  - Biological data
  - Medical data
  - Climate data
  - …



TabPFN is Similar to Language Models Like GPT-3

- TabPFN is a **transformer pretrained to do tabular classification**
- Framed as next-word prediction: $x_1, y_1, \ldots, x_n, y_n, x_{n+1}, ?$

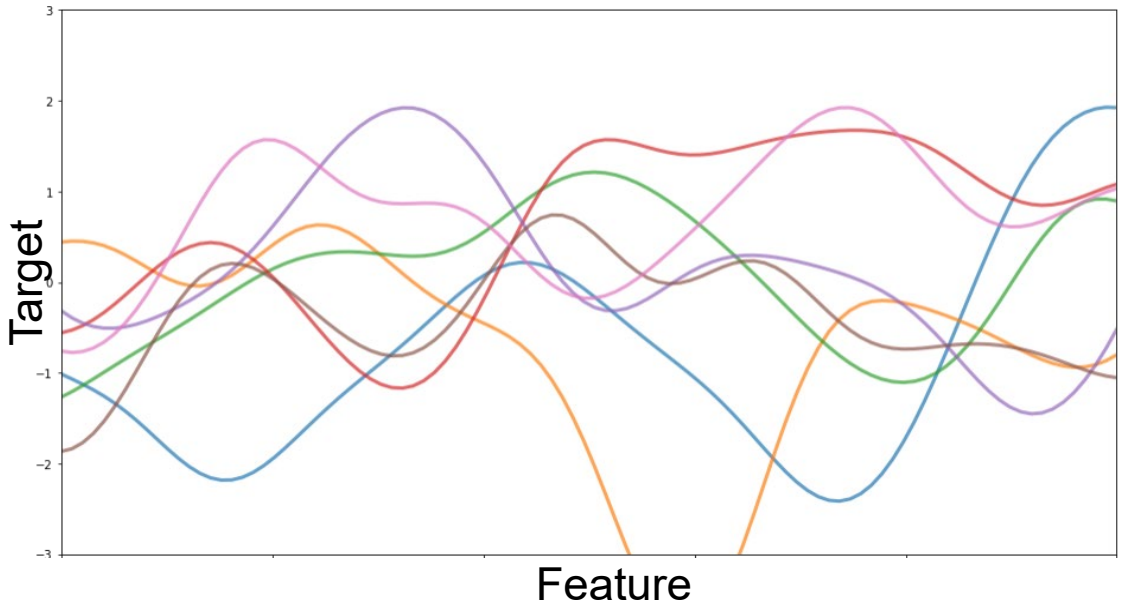
- To be more precise:

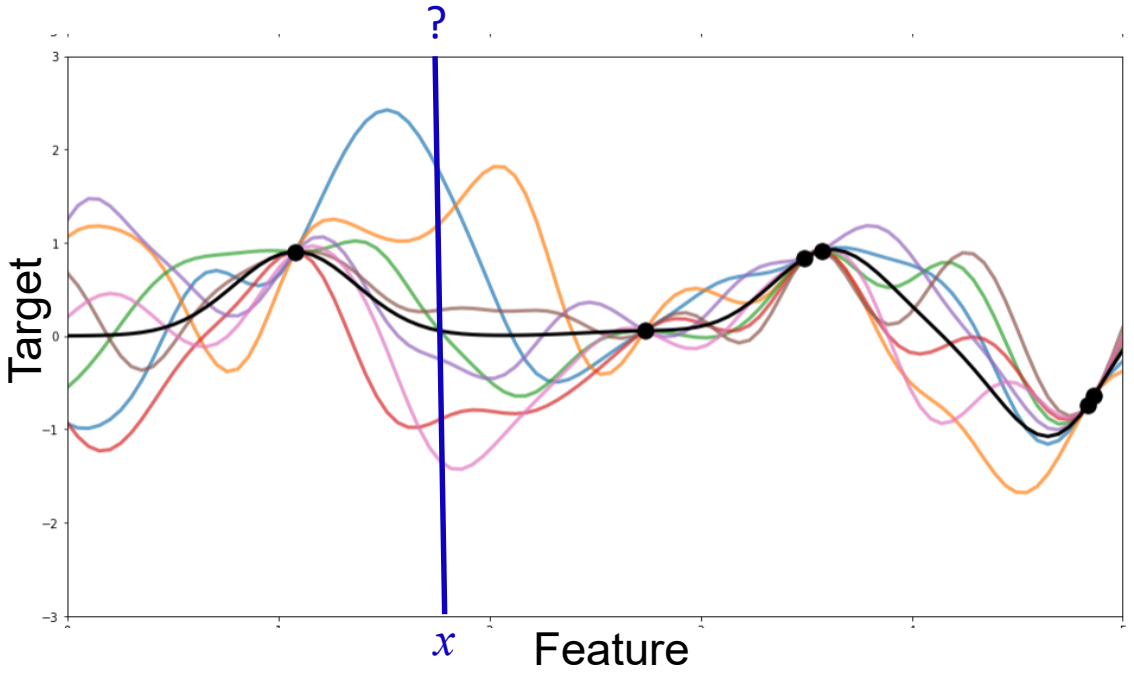
$$\{(x_1, y_1), \ldots, (x_n, y_n)\}, x_{n+1} \longrightarrow \boxed{\text{TabPFN}} \longrightarrow \hat{y}_{n+1}$$

- To be even more precise:

$$\{(x_1, y_1), \ldots, (x_n, y_n)\}, x_{n+1} \longrightarrow \boxed{\text{TabPFN}} \longrightarrow p(y_{n+1} \mid x_{n+1}, \{(x_1, y_1), \ldots, (x_n, y_n)\})$$

TabPFN approximates Bayesian predictions





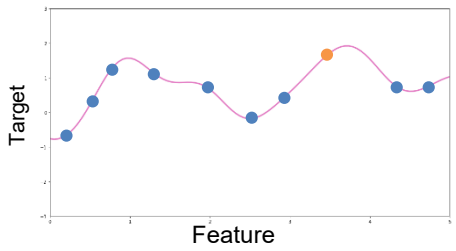
Prior over functions
parameterized by latents t

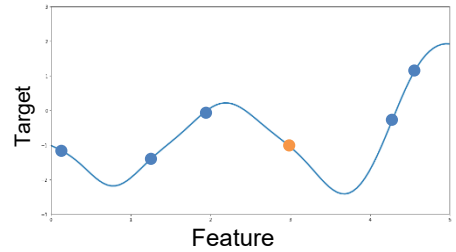
Posterior $p(t|D) = \frac{p(D|t)p(t)}{\int p(D|t)dt}$

Intractable to compute exactly!

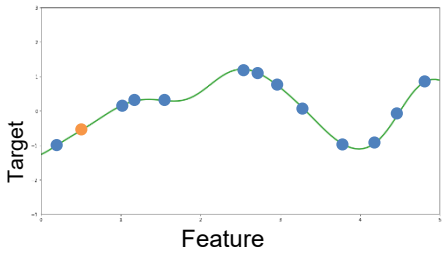
Posterior predictive distribution $p(y|x, D) = \int p(y|x, t)p(t|D)dt$

Illustration of Prior-Fitted Networks (PFNs) [ICLR 2022]





- (500M times)



train
test

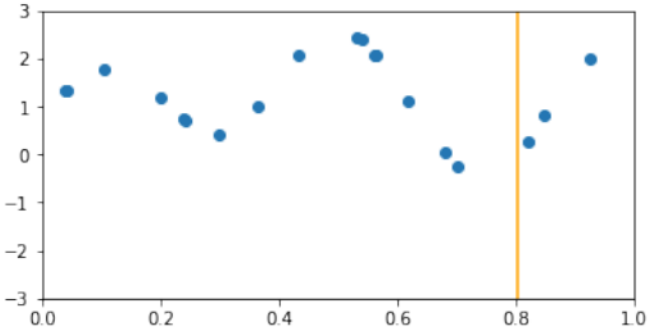
Samples from the prior:
a data-generating mechanism

Learn a model to predict test from train

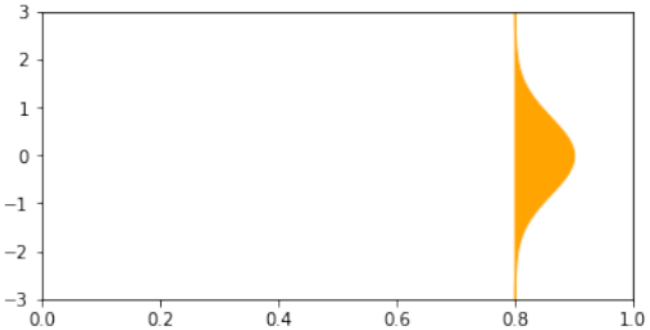
We call this model a
prior-fitted network (PFN)

We have thus meta-learned to approximate Bayesian inference, purely by supervised learning of data generated with the prior

Actual Data ?

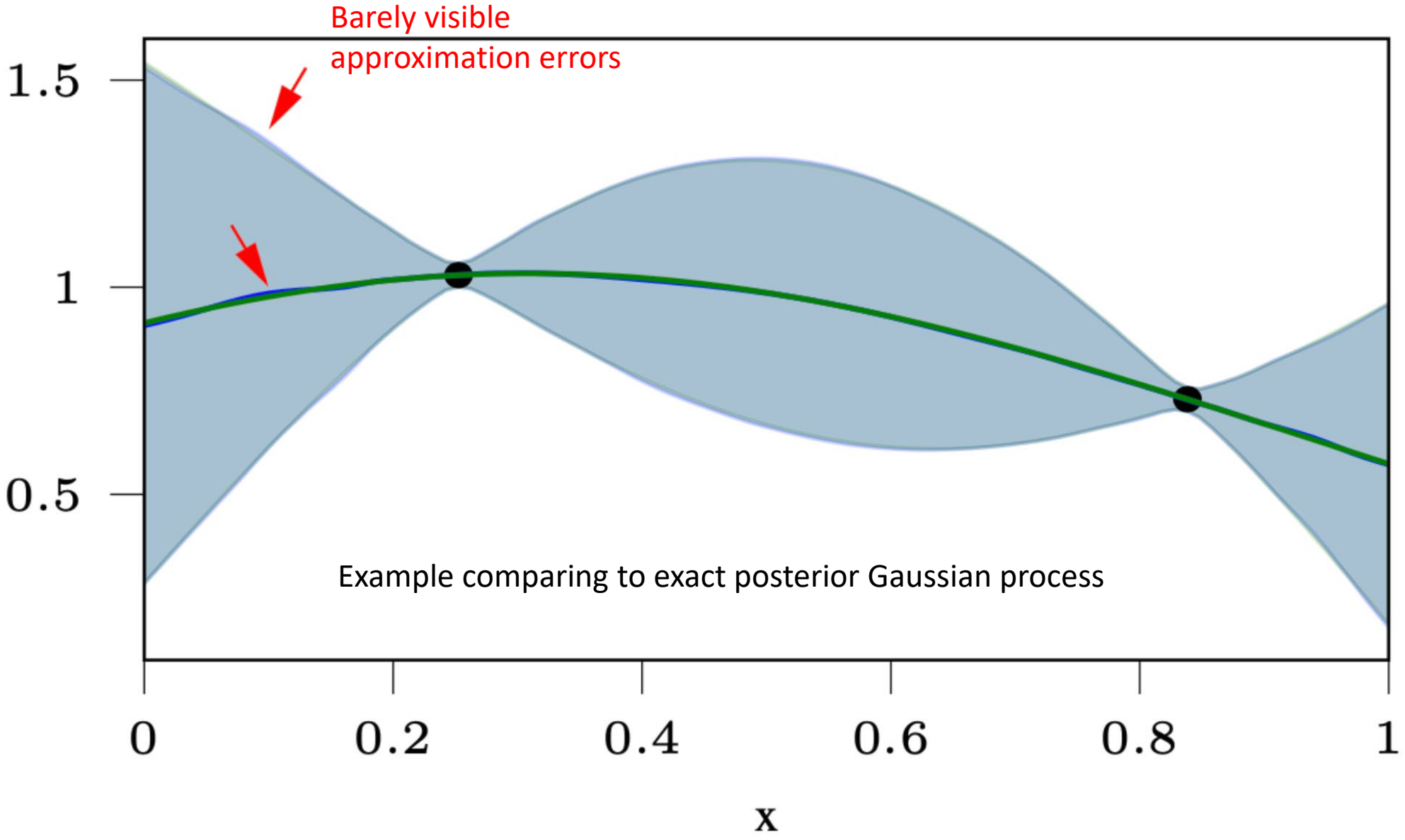


PFN forward pass



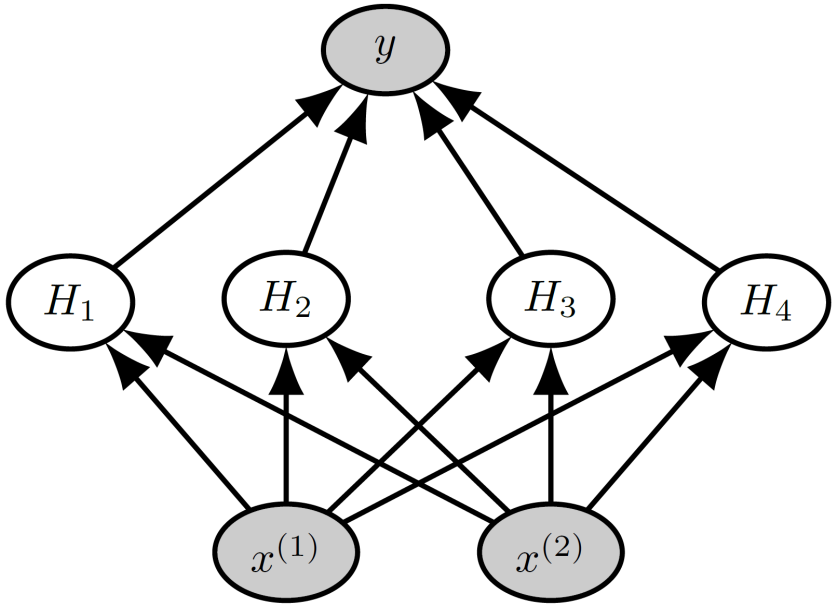
Posterior predictive distribution

PFNs can predict the true posterior arbitrarily closely [ICLR 2022]

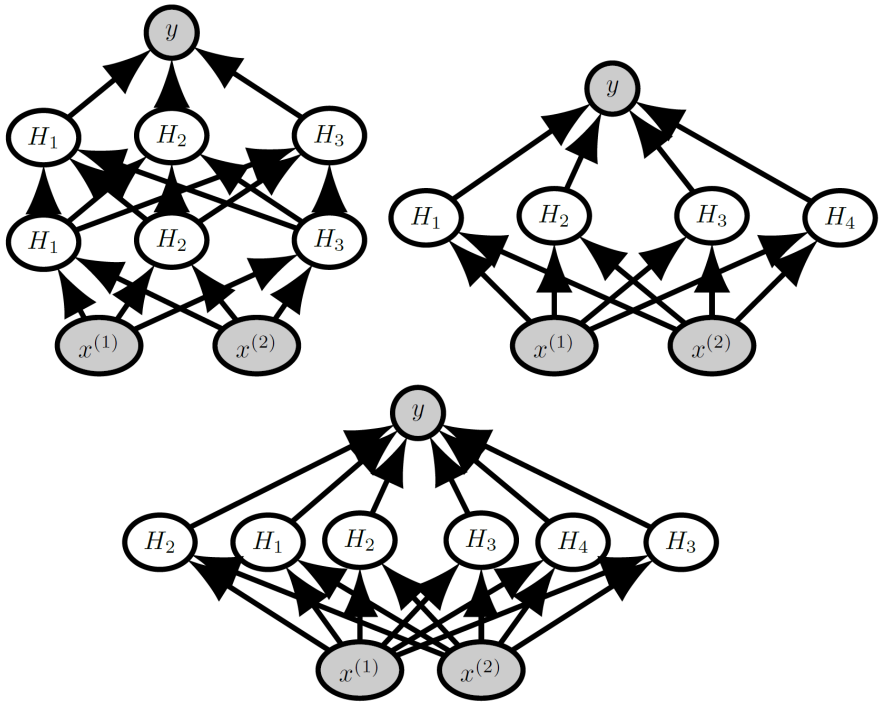


PFNs Enable Bayesian Deep Learning in a Forward Pass [ICLR 2022]

- Prior: weights of a given neural net
- Posterior predictive: Bayesian neural net
  - 10000x speedups over MCMC etc

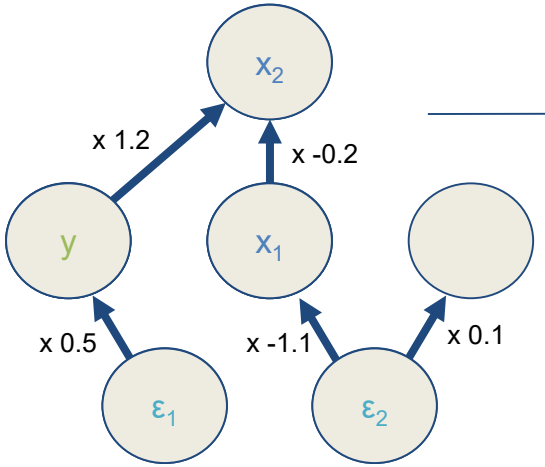


- Prior: different neural architectures & their weights
- Posterior predictive: "**Bayesian NAS**"
  - Not even possible with MCMC etc

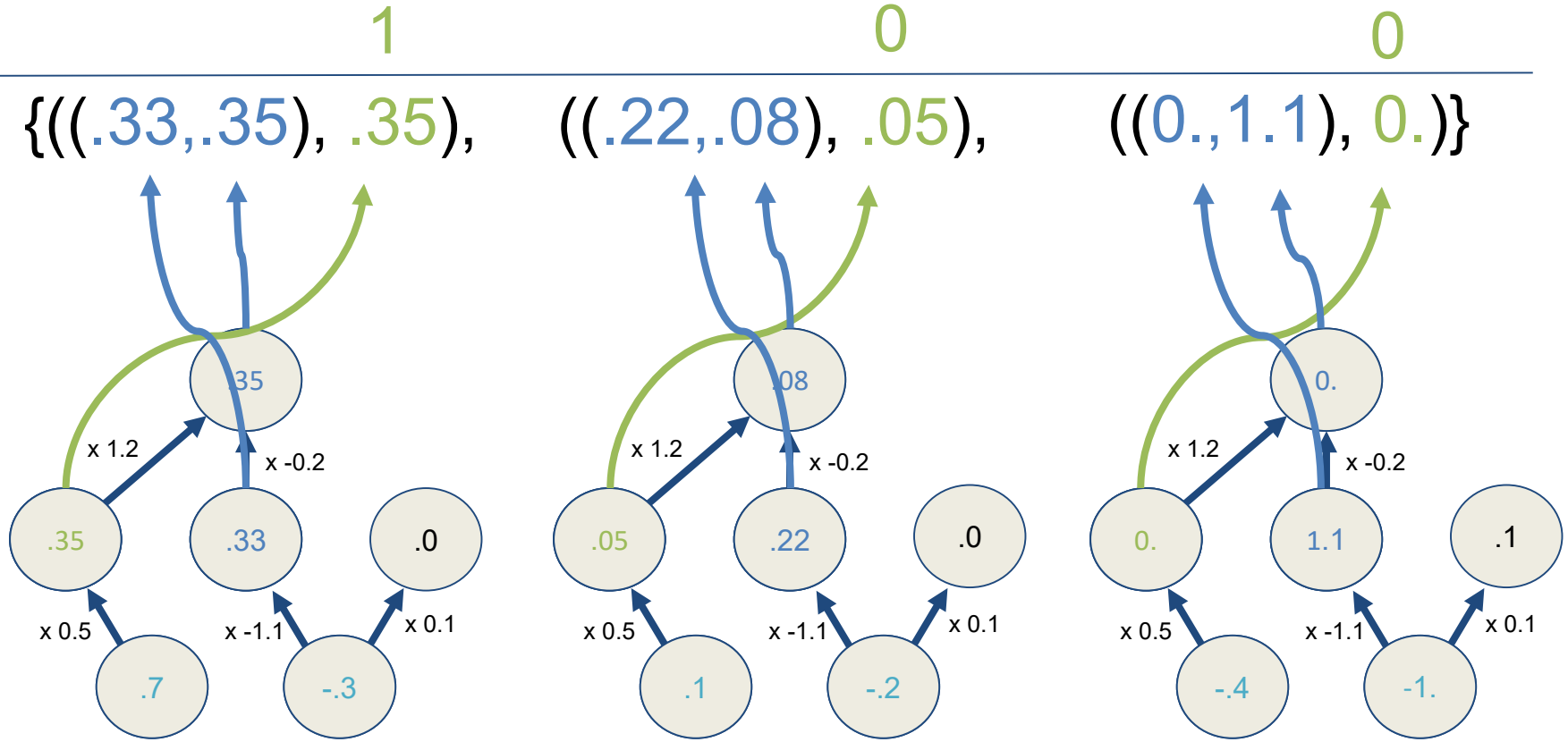


TabPFN Prior: Integrating Principles from Causality

Sample & initialize
a causal graph

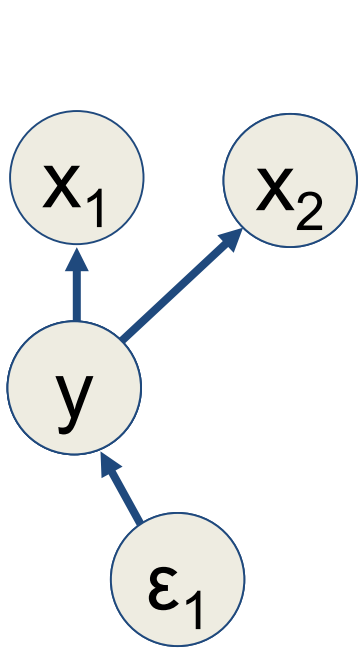


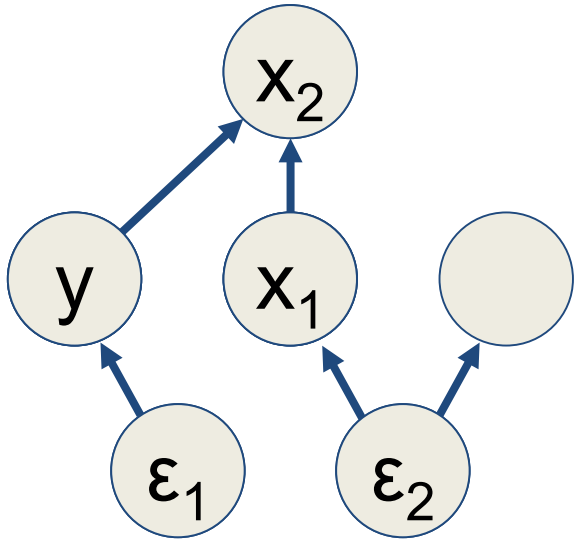
Build dataset: output >0.2?

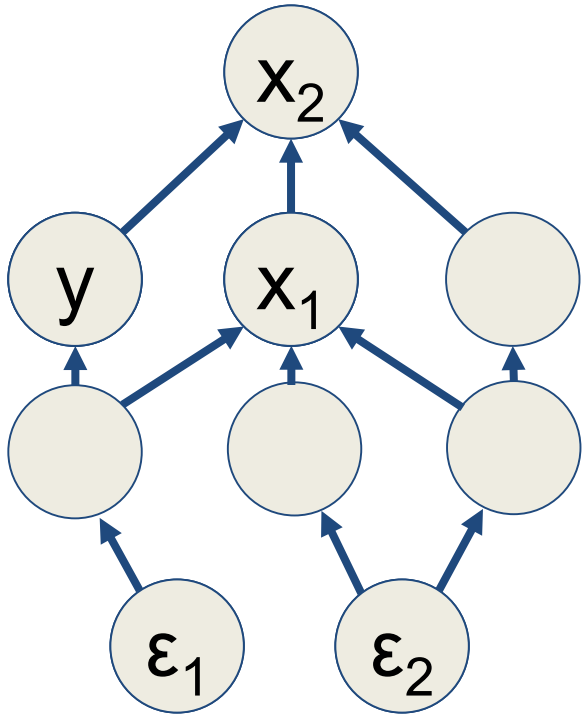


Sample noise
per example
& forward pass

TabPFN Prior: Simplicity Principle





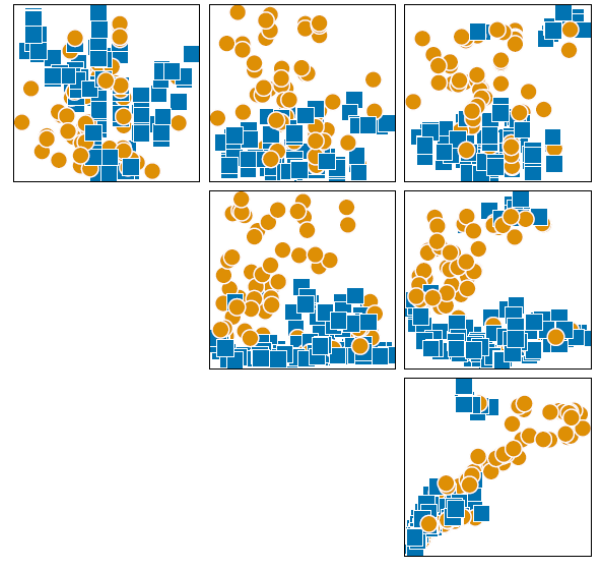


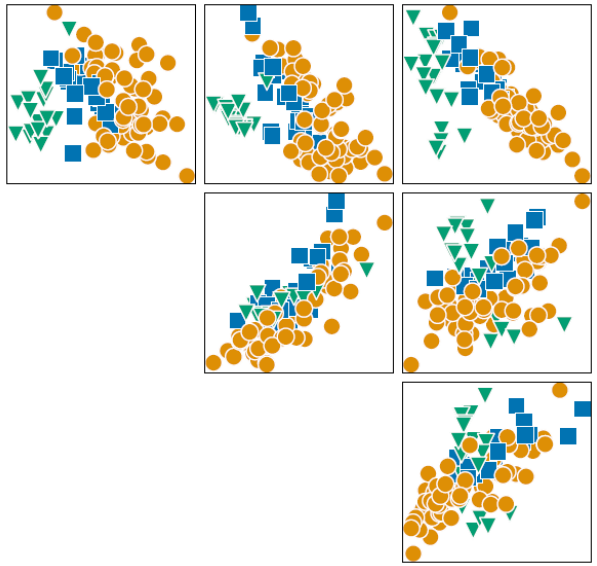
Prior likelihood

Graph Complexity

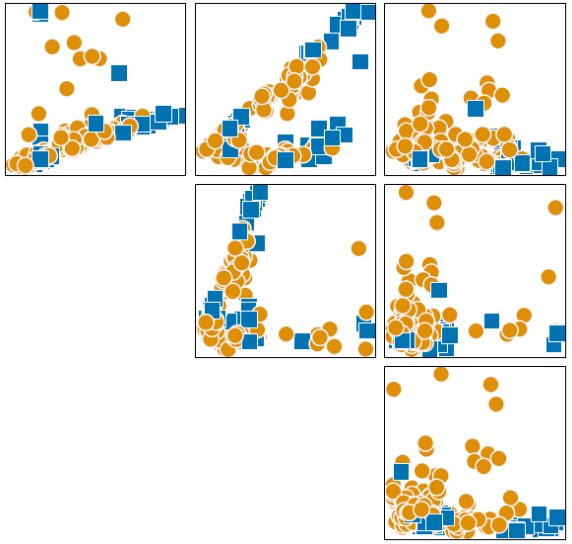
The generated datasets look similar to real ones

Synthetic datasets

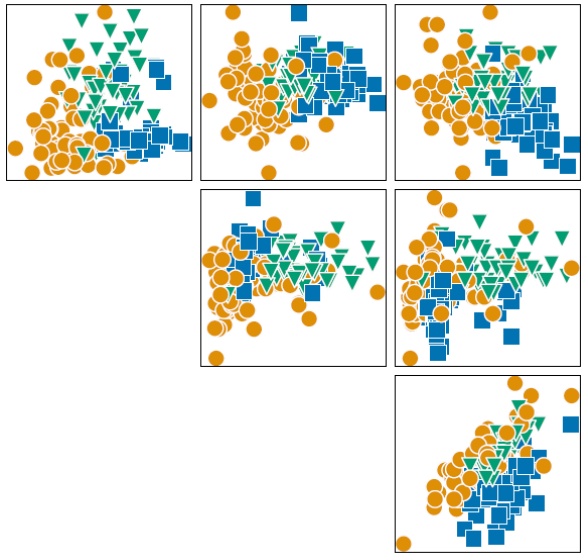




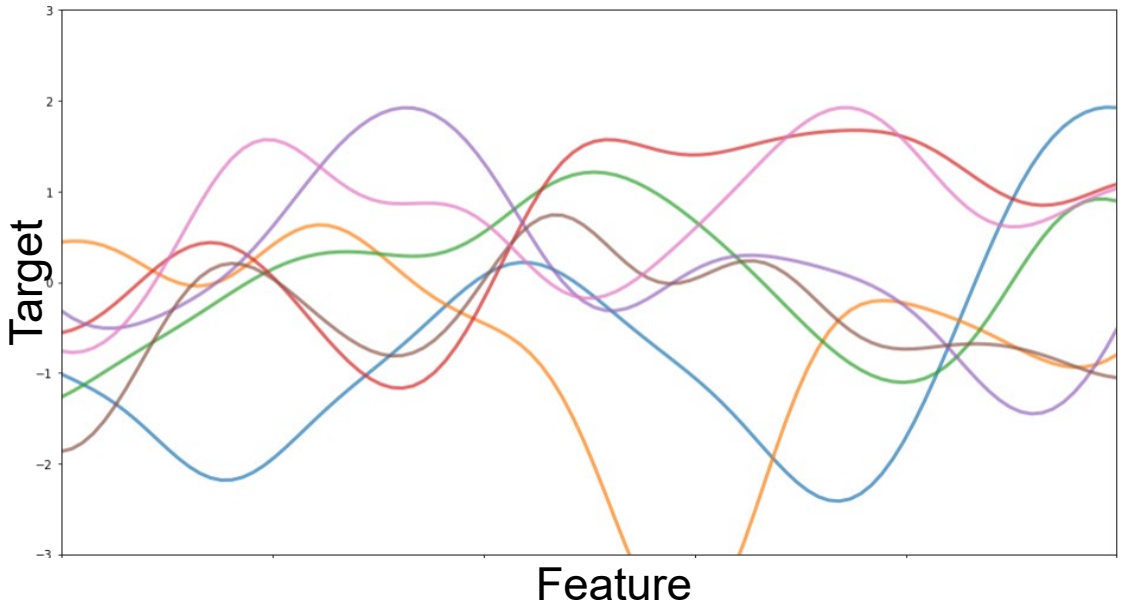
Parkinsons dataset

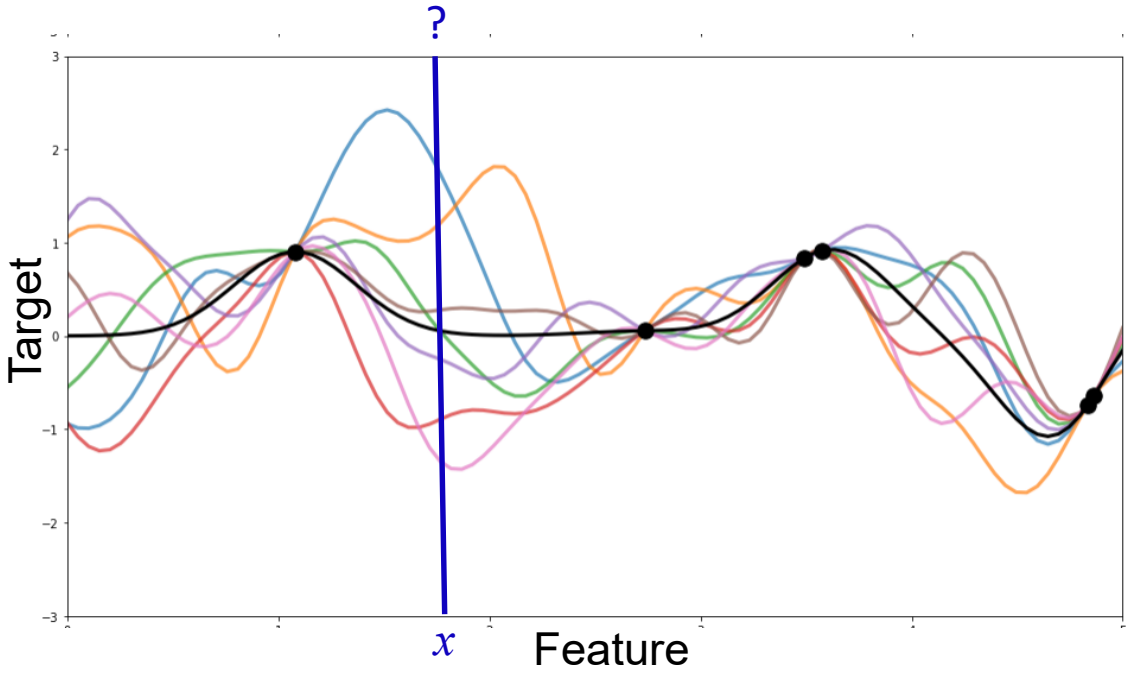


Wine dataset



Relation to Bayesian Supervised Learning





Prior over functions parameterized by latents t

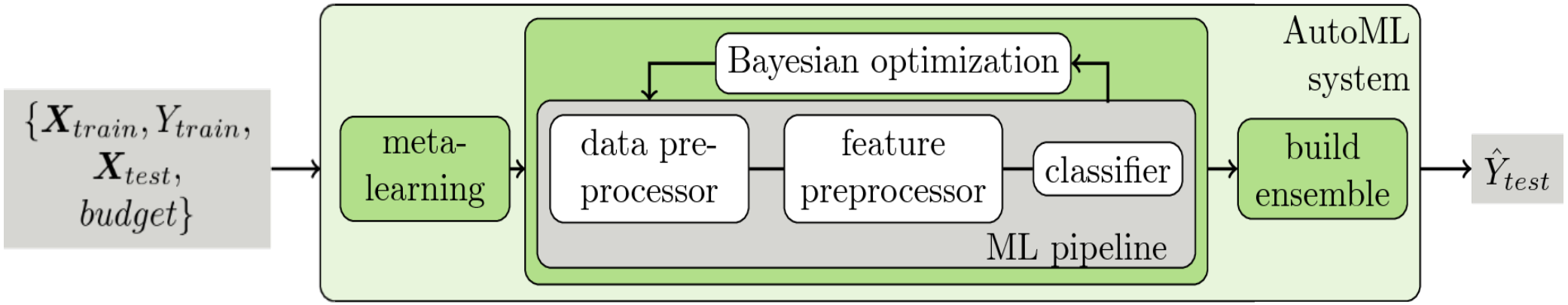
- Noise values, graph structure, weights, activation functions, etc

Posterior $p(t|D) = \frac{p(D|t)p(t)}{\int p(D|t)dt}$

Posterior predictive distribution $p(y|x, D) = \int p(y|x, t)p(t|D)dt$

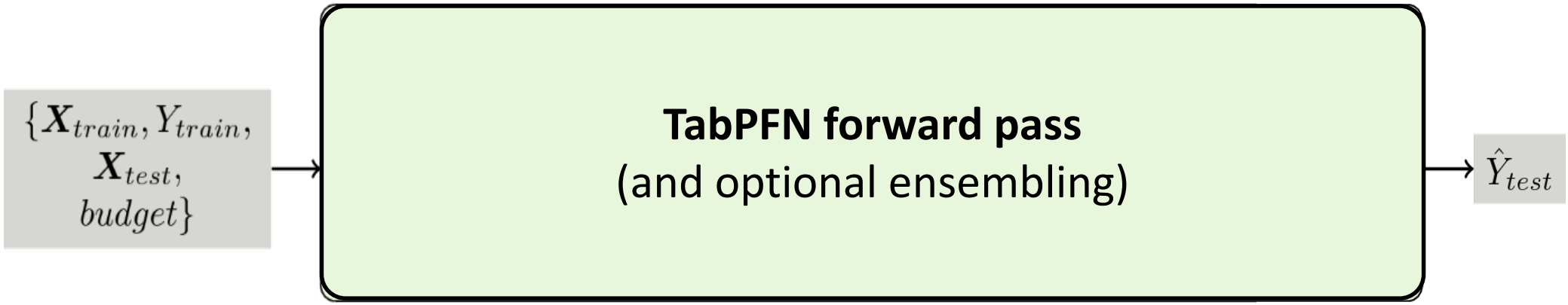
Simplifying AutoML for Real-time Training

- AutoML pipeline



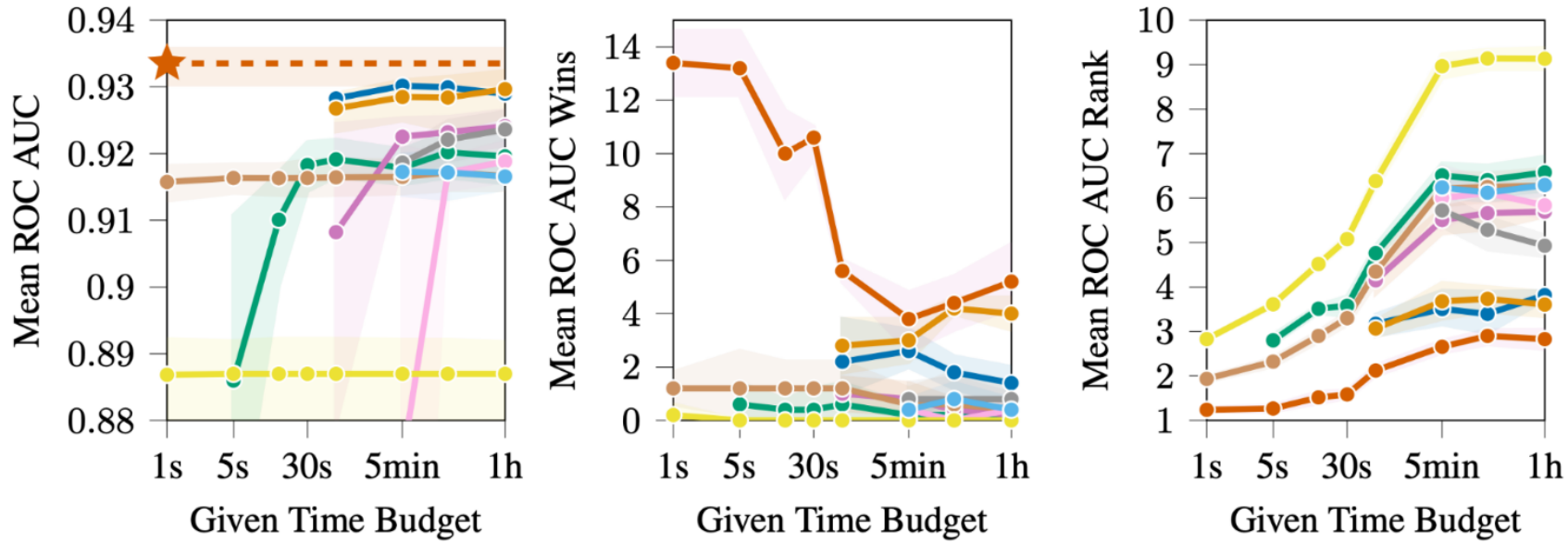
Simplifying AutoML for Real-time Training

- TabPFN pipeline



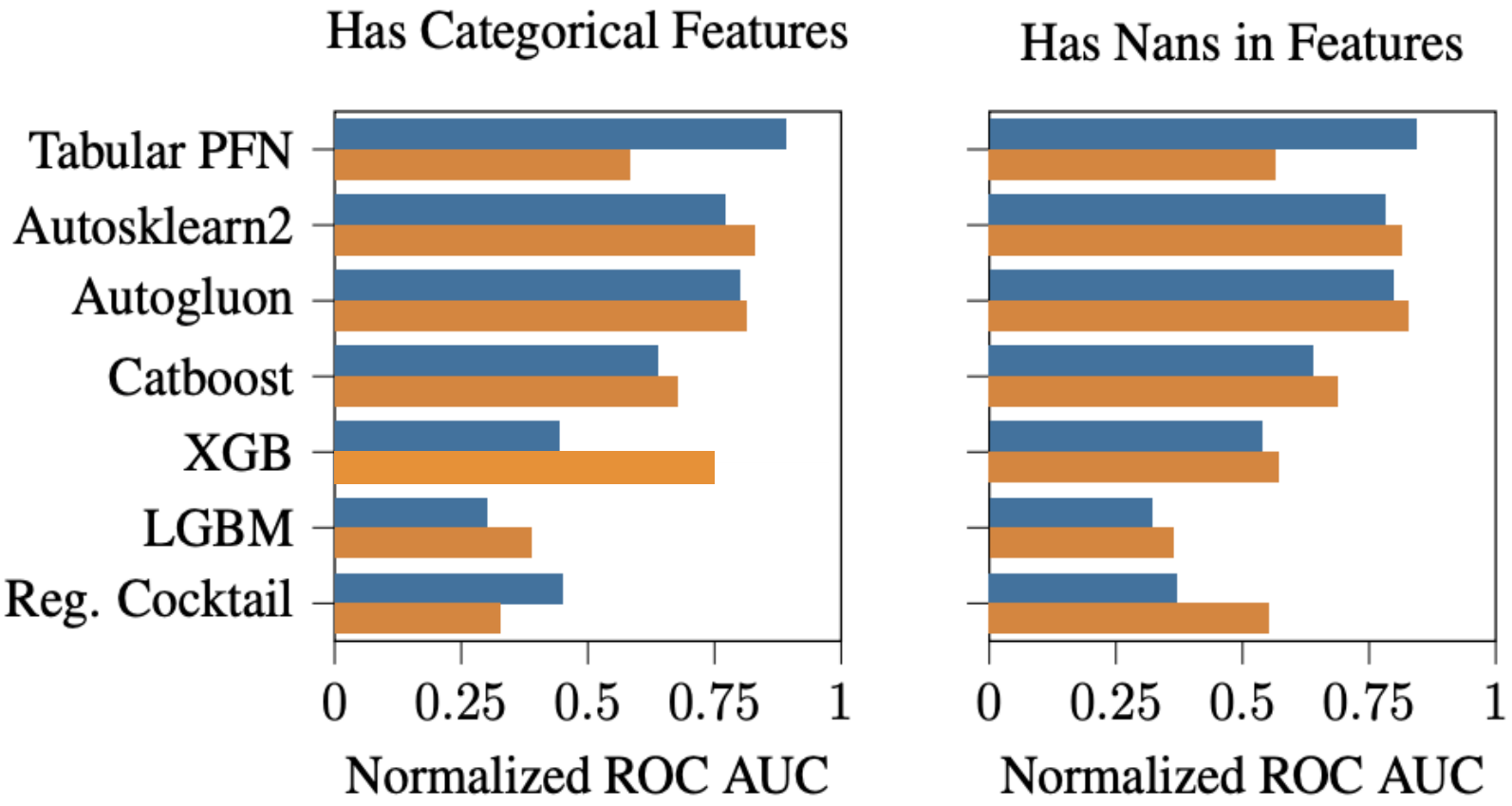
Results

OpenML-CC18 suite subset with < 1000 examples, numerical features & no missing values



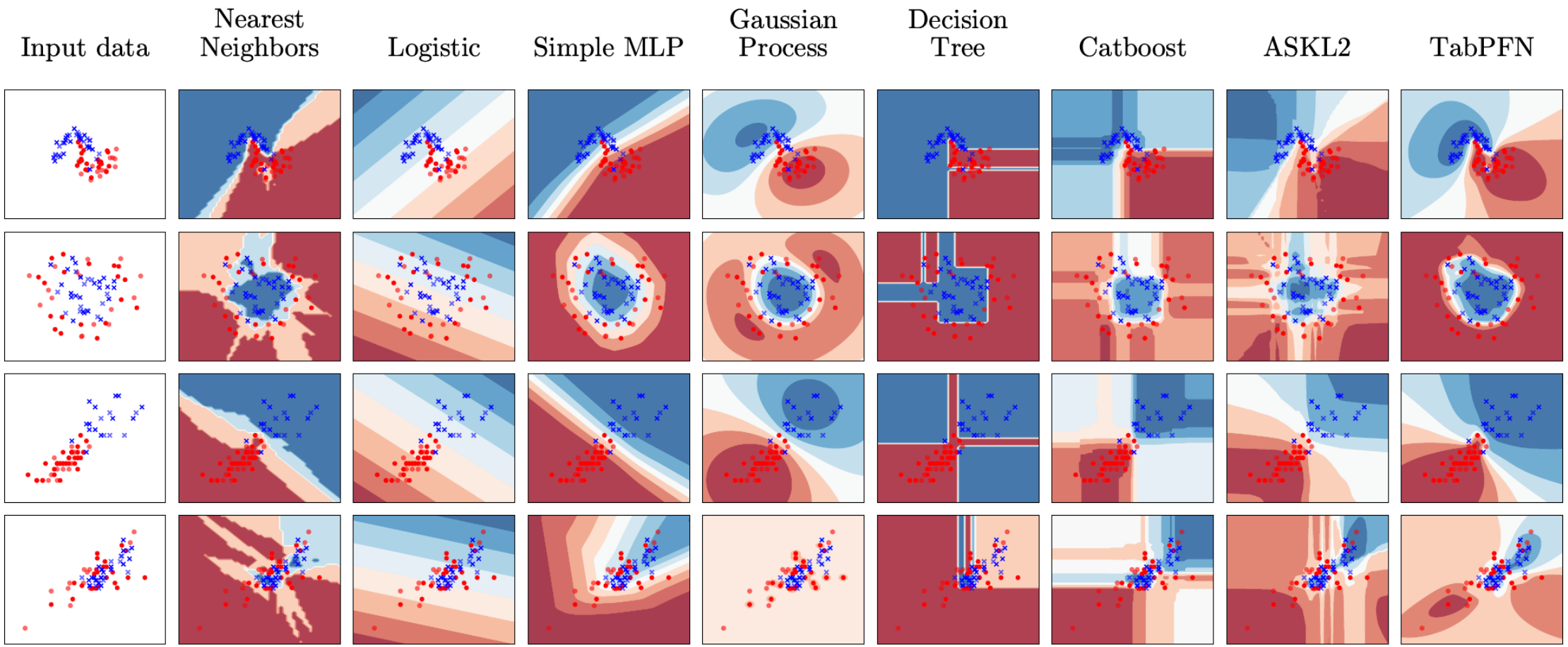
Results confirmed on 67 additional datasets

Limitations (other than size)



Evaluation on a total of 180 datasets

TabPFN Makes Smooth, Intuitive Predictions



What Does This Mean For AutoML?

- The **first of many AutoML foundation models to come**
- Is standard AutoML rendered unnecessary?
  - No! This simply shakes up the space of base-level algorithms
  - This is meta-learning and as such anyways part of AutoML
  - AutoML systems should simply include TabPFN

- TabPFN is **as green as AutoML will ever get** ☺
- TabPFN's speed **enables true user interaction**
- TabPFN is **user-centric & data-centric, not model-centric & ML expert-centric**
  - No more need for the user to know anything about XGBoost etc & their hyperparameters

Conclusions

- TabPFN is **fully learned**, based on a **synthetic data-generating mechanism**
- TabPFN computes **posterior Bayesian inference** for the given prior
  - In our prior: elements of causality & simplicity

- **Excellent performance** on **small datasets**
  - Up to 1000 data points, purely numerical, no missing values
  - Training + prediction costs less than 1s
  - Better predictions on average than any other ML / AutoML method in 1h

- We will found a **startup** on TabPFN
  - Please talk to me to share your advice, or if you're interested