TabPFN: One Model to Rule Them All?

Qiong Zhang*
Renmin University of China

Yan Shuo Tan*
National University of Singapore

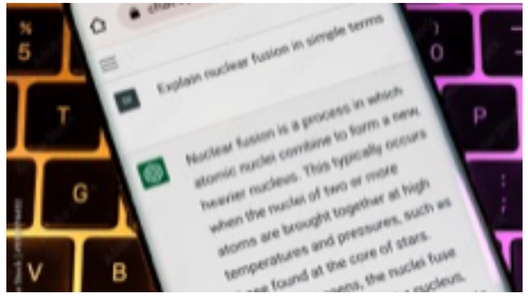
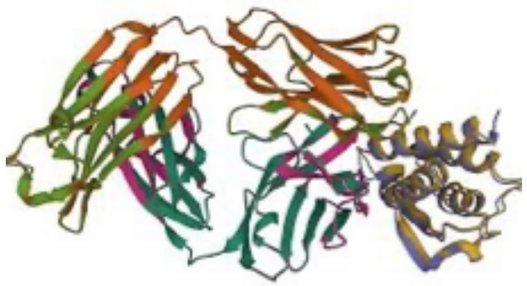
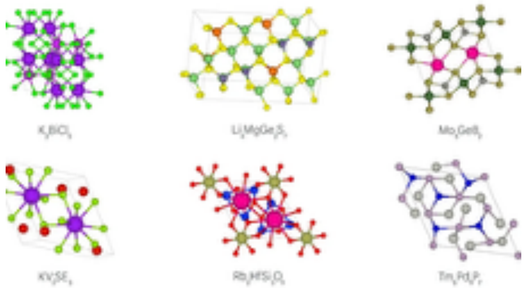
Qinglong Tian*
University of Waterloo

Pengfei Li
University of Waterloo

*Contributed equally

The deep learning revolution in data science

Deep learning has many successes, particularly in *computer vision* and *NLP*

	NLP/image generation	Biosciences	Material science
Past			
Present			
	Deepseek/ChatGPT	**AlphaFold**	**GNoME**

Old story: deep learning struggles with tabular data

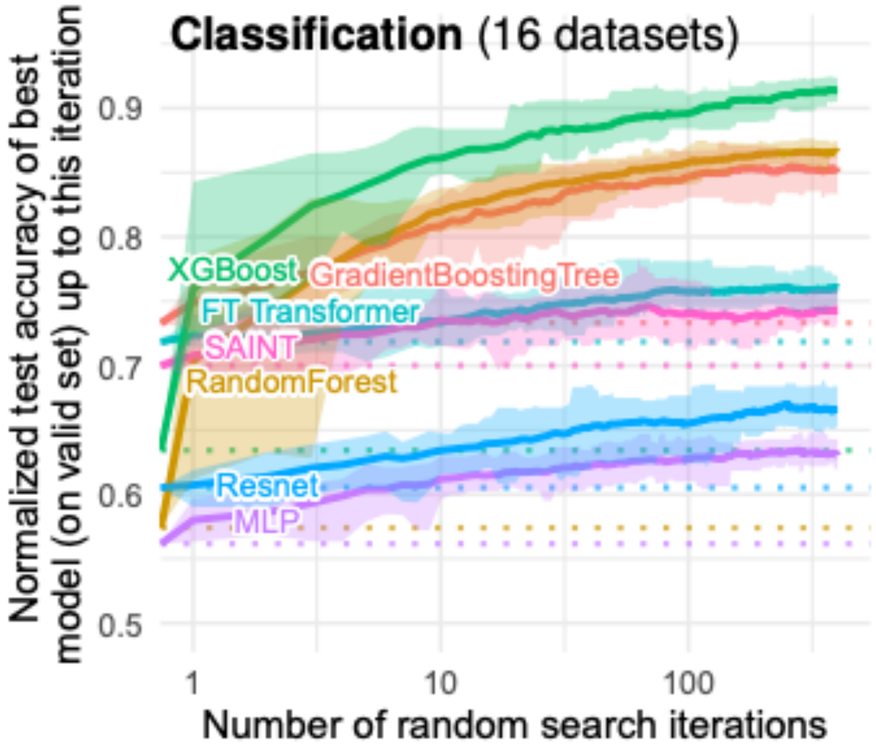
	0	1	2
0	1.5	21.0	'Mia'
1	4.0	35.0	'Lucas'
2	3.0	17.0	'Ang'
3	4.0	53.0	'Jia'

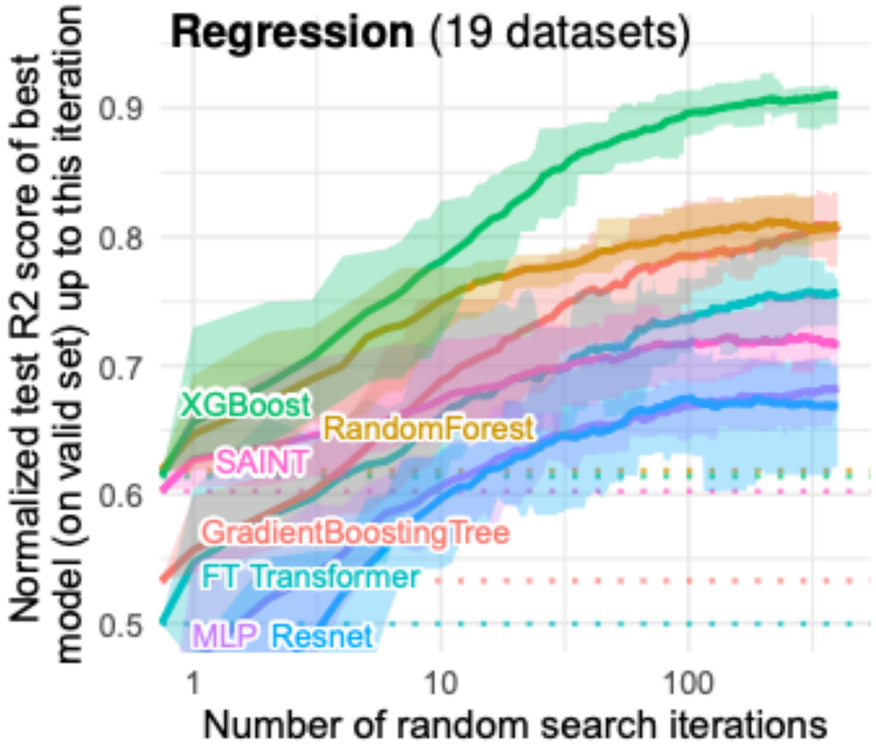
Numerical and string datatypes

Grinsztajn et al. (NeurIPS 2022) Why do tree-based models still outperform deep learning on typical tabular data?

Old story: deep learning struggles with tabular data

However, tree ensembles were gold standards for tabular data classification & regression





	0	1	2
0	1.5	21.0	'Mia'
1	4.0	35.0	'Lucas'
2	3.0	17.0	'Ang'
3	4.0	53.0	'Jia'

Numerical and string datatypes

Grinsztajn et al. (NeurIPS 2022) Why do tree-based models still outperform deep learning on typical tabular data?

A new breakthrough: deep learning outperforms tree ensembles

A new breakthrough: deep learning outperforms tree ensembles

Article

Accurate predictions on small data with a tabular foundation model

https://doi.org/10.1038/s41586-024-08328-6

Received: 17 May 2024

Accepted: 31 October 2024

Published online: 8 January 2025

Open access

Check for updates

Noah Hollmann[1,2,3,7]✉, Samuel Müller[1,7]✉, Lennart Purucker[1], Arjun Krishnakumar[1], Max Körfer[1], Shi Bin Hoo[1], Robin Tibor Schirrmeister[4,5] & Frank Hutter[1,3,6]✉

Tabular data, spreadsheets organized in rows and columns, are ubiquitous across scientific fields, from biomedicine to particle physics to economics and climate science[1,2]. The fundamental prediction task of filling in missing values of a label column based on the rest of the columns is essential for various applications as diverse as biomedical risk models, drug discovery and materials science. Although deep learning has revolutionized learning from raw data and led to numerous high-profile success stories[3–5], gradient-boosted decision trees[6–9] have dominated tabular data for the past 20 years. Here we present the Tabular Prior-data Fitted Network (TabPFN), a tabular foundation model that outperforms all previous methods on datasets with up to 10,000 samples by a wide margin, using substantially less training time. In 2.8 s, TabPFN outperforms an ensemble of the strongest baselines tuned for 4 h in a classification setting. As a generative transformer-based foundation model, this model also allows fine-tuning, data generation, density estimation and learning reusable embeddings. TabPFN is a learning algorithm that is itself learned across millions of synthetic datasets, demonstrating the power of this approach for algorithm development. By improving modelling abilities across diverse fields, TabPFN has the potential to accelerate scientific discovery and enhance important decision-making in various domains.

A new breakthrough: deep learning outperforms tree ensembles

Article

Accurate predictions on small data with a tabular foundation model

https://doi.org/10.1038/s41586-024-08328-6

Received: 17 May 2024

Accepted: 31 October 2024

Published online: 8 January 2025

Open access

Check for updates

Noah Hollmann[1,2,3,7]✉, Samuel Müller[1,7]✉, Lennart Purucker[1], Arjun Krishnakumar[1], Max Körfer[1], Shi Bin Hoo[1], Robin Tibor Schirrmeister[4,5] & Frank Hutter[1,3,6]✉

Tabular data, spreadsheets organized in rows and columns, are ubiquitous across scientific fields, from biomedicine to particle physics to economics and climate science[1,2]. The fundamental prediction task of filling in missing values of a label column based on the rest of the columns is essential for various applications as diverse as biomedical risk models, drug discovery and materials science. Although deep learning has revolutionized learning from raw data and led to numerous high-profile success stories[3–5], gradient-boosted decision trees[6–9] have dominated tabular data for the past 20 years. Here we present the Tabular Prior-data Fitted Network (TabPFN), a tabular foundation model that outperforms all previous methods on datasets with up to 10,000 samples by a wide margin, using substantially less training time. In 2.8 s, TabPFN outperforms an ensemble of the strongest baselines tuned for 4 h in a classification setting. As a generative transformer-based foundation model, this model also allows fine-tuning, data generation, density estimation and learning reusable embeddings. TabPFN is a learning algorithm that is itself learned across millions of synthetic datasets, demonstrating the power of this approach for algorithm development. By improving modelling abilities across diverse fields, TabPFN has the potential to accelerate scientific discovery and enhance important decision-making in various domains.

A new breakthrough: deep learning outperforms tree ensembles

Article

Accurate predictions on small data with a tabular foundation model

https://doi.org/10.1038/s41586-024-08328-6

Received: 17 May 2024

Accepted: 31 October 2024

Published online: 8 January 2025

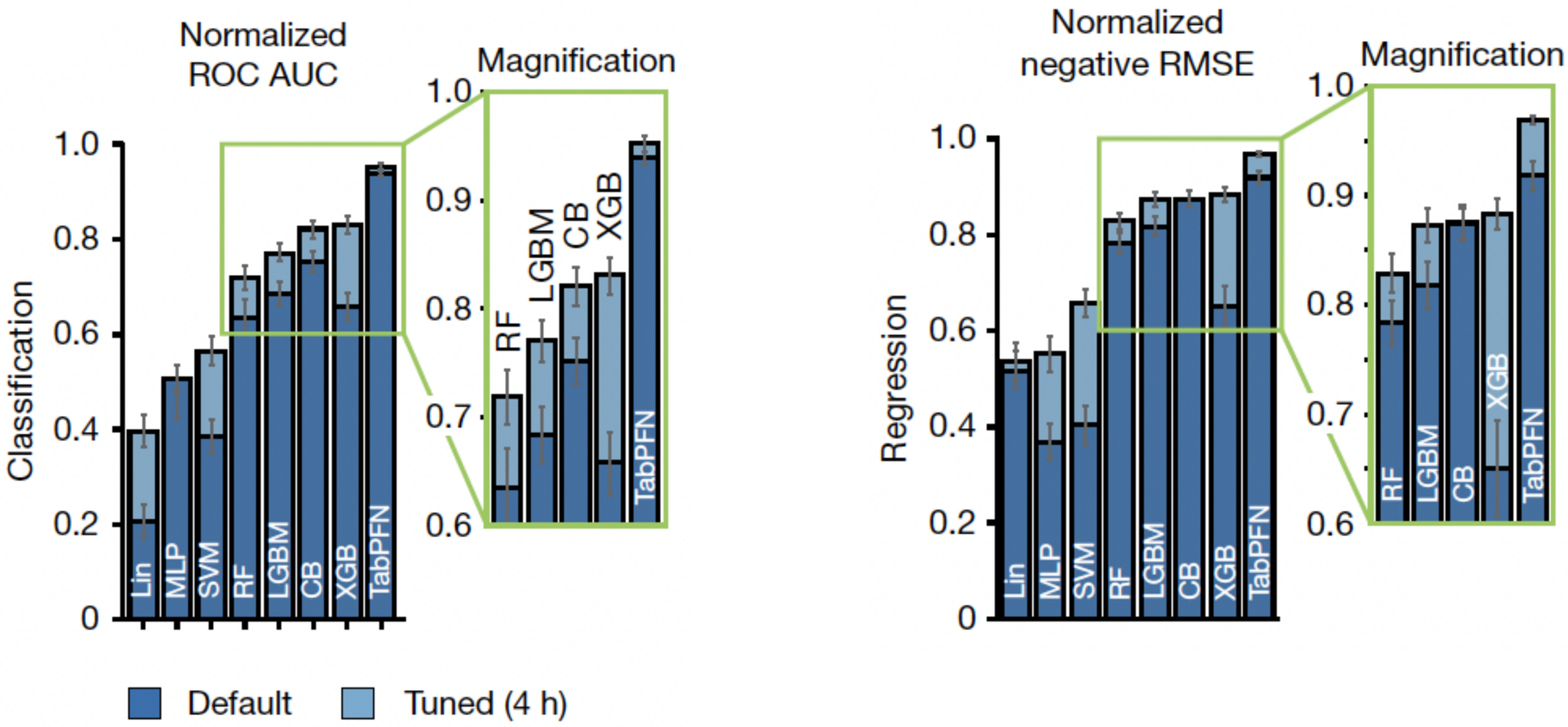
Open access

Check for updates

Noah Hollmann[1,2,3,7]✉, Samuel Müller[1,7]✉, Lennart Purucker[1], Arjun Krishnakumar[1], Max Körfer[1], Shi Bin Hoo[1], Robin Tibor Schirrmeister[4,5] & Frank Hutter[1,3,6]✉

Tabular data, spreadsheets organized in rows and columns, are ubiquitous across scientific fields, from biomedicine to particle physics to economics and climate science[1,2]. The fundamental prediction task of filling in missing values of a label column based on the rest of the columns is essential for various applications as diverse as biomedical risk models, drug discovery and materials science. Although deep learning has revolutionized learning from raw data and led to numerous high-profile success stories[3–5], gradient-boosted decision trees[6–9] have dominated tabular data for the past 20 years. Here we present the Tabular Prior-data Fitted Network (TabPFN), a tabular foundation model that outperforms all previous methods on datasets with up to 10,000 samples by a wide margin, using substantially less training time. In 2.8 s, TabPFN outperforms an ensemble of the strongest baselines tuned for 4 h in a classification setting. As a generative transformer-based foundation model, this model also allows fine-tuning, data generation, density estimation and learning reusable embeddings. TabPFN is a learning algorithm that is itself learned across millions of synthetic datasets, demonstrating the power of this approach for algorithm development. By improving modelling abilities across diverse fields, TabPFN has the potential to accelerate scientific discovery and enhance important decision-making in various domains.

A new breakthrough: deep learning outperforms tree ensembles



57 benchmark regression and classification datasets

TabPFN is a foundation model

Article

Accurate predictions on small data with a tabular foundation model

https://doi.org/10.1038/s41586-024-08328-6

Received: 17 May 2024

Accepted: 31 October 2024

Published online: 8 January 2025

Open access

Check for updates

Noah Hollmann[1,2,3,7]✉, Samuel Müller[1,7]✉, Lennart Purucker[1], Arjun Krishnakumar[1], Max Körfer[1], Shi Bin Hoo[1], Robin Tibor Schirrmeister[4,5] & Frank Hutter[1,3,6]✉

Tabular data, spreadsheets organized in rows and columns, are ubiquitous across scientific fields, from biomedicine to particle physics to economics and climate science[1,2]. The fundamental prediction task of filling in missing values of a label column based on the rest of the columns is essential for various applications as diverse as biomedical risk models, drug discovery and materials science. Although deep learning has revolutionized learning from raw data and led to numerous high-profile success stories[3–5], gradient-boosted decision trees[6–9] have dominated tabular data for the past 20 years. Here we present the Tabular Prior-data Fitted Network (TabPFN), a tabular foundation model that outperforms all previous methods on datasets with up to 10,000 samples by a wide margin, using substantially less training time. In 2.8 s, TabPFN outperforms an ensemble of the strongest baselines tuned for 4 h in a classification setting. As a generative transformer-based foundation model, this model also allows fine-tuning, data generation, density estimation and learning reusable embeddings. TabPFN is a learning algorithm that is itself learned across millions of synthetic datasets, demonstrating the power of this approach for algorithm development. By improving modelling abilities across diverse fields, TabPFN has the potential to accelerate scientific discovery and enhance important decision-making in various domains.

What is foundation model?

Foundation model: a machine learning or deep learning model trained on vast datasets so that it can be applied across a wide range of use cases

What is foundation model?

Foundation model: a machine learning or deep learning model trained on vast datasets so that it can be applied across a wide range of use cases

As a remedy, we introduce TabPFN, a foundation model for small- to medium-sized tabular data. This new supervised tabular learning method can be applied to any small- to moderate-sized dataset and yields dominant performance for datasets with up to 10,000 samples and 500 features. In a single forward pass, TabPFN significantly outperforms state-of-the-art baselines on our benchmarks, including gradient-boosted decision trees, even when these are allowed 4 h of tuning, a speedup of 5,140× (classification) and 3,000× (regression). Finally, we demonstrate various foundation model characteristics of TabPFN, including fine-tuning, generative abilities and density estimation.

What is foundation model?

Foundation model: a machine learning or deep learning model trained on vast datasets so that it can be applied across a wide range of use cases

As a remedy, we introduce TabPFN, a foundation model for small- to medium-sized tabular data. This new supervised tabular learning method can be applied to any small- to moderate-sized dataset and yields dominant performance for datasets with up to 10,000 samples and 500 features. In a single forward pass, TabPFN significantly outperforms state-of-the-art baselines on our benchmarks, including gradient-boosted decision trees, even when these are allowed 4 h of tuning, a speedup of 5,140× (classification) and 3,000× (regression). Finally, we demonstrate various foundation model characteristics of TabPFN, including fine-tuning, generative abilities and density estimation.

What is foundation model?

Foundation model: a machine learning or deep learning model trained on vast datasets so that it can be applied across a wide range of use cases

FM for NLP

GPT-4.5

What is foundation model?

Foundation model: a machine learning or deep learning model trained on vast datasets so that it can be applied across a wide range of use cases

FM for NLP

GPT-4.5

FM for tabular data

TabPFN

Goal of this talk

What is TabPFN?

Explain TabPFN from a statistical perspective

Is it good for statistical estimation?

Case studies: semi-supervised parameter estimation, heterogeneous treatment effects estimation, ~~prediction under covariate shift~~

Why does TabPFN work so well?

TabPFN's adaptivity on specific structures

How TabPFN works?

In-context learning

In-context learning (ICL): a technique where pre-trained LLMs learn new tasks from a few examples in the form of prompt, without requiring fine-tuning or parameter updates.

How TabPFN works?

In-context learning

In-context learning (ICL): a technique where pre-trained LLMs learn new tasks from a few examples in the form of prompt, without requiring fine-tuning or parameter updates.

北大->1898, 清华->1911，中科大->？

How TabPFN works?

In-context learning

In-context learning (ICL): a technique where pre-trained LLMs learn new tasks from a few examples in the form of prompt, without requiring fine-tuning or parameter updates.

北大->1898, 清华->1911，中科大->？

从这两组对应关系可以看出，模式可能是：

大学简称 -> 该大学的建校年份

How TabPFN works?

In-context learning

In-context learning (ICL): a technique where pre-trained LLMs learn new tasks from a few examples in the form of prompt, without requiring fine-tuning or parameter updates.

北大->1898, 清华->1911，中科大->？

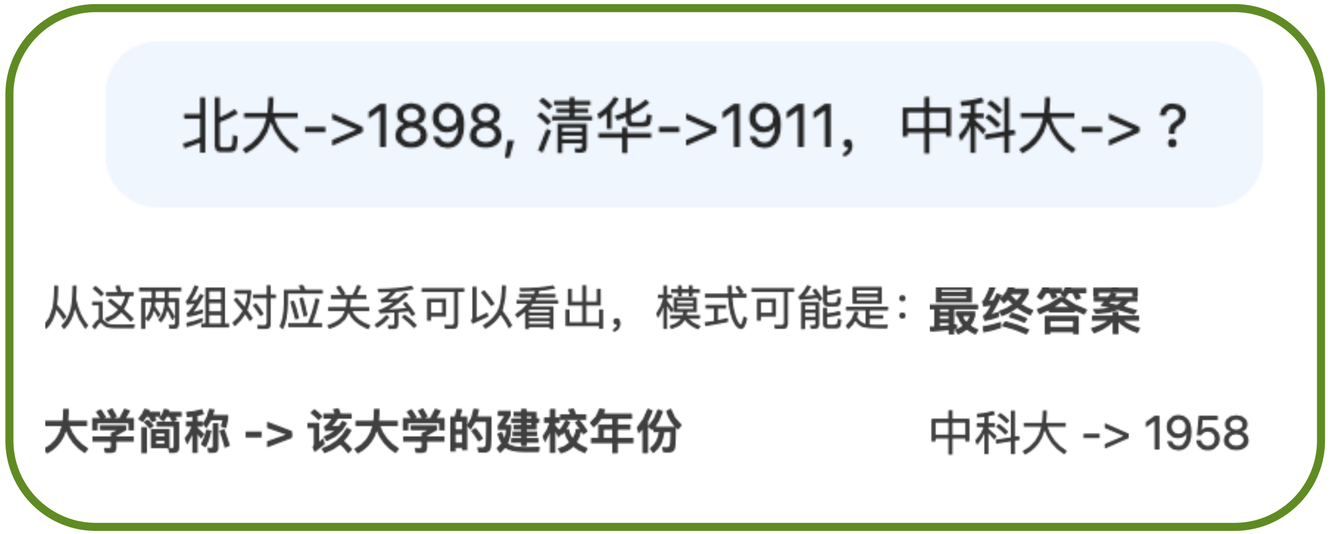
从这两组对应关系可以看出，模式可能是：**最终答案**

大学简称 -> 该大学的建校年份　　中科大 -> 1958

How TabPFN works?

In-context learning

In-context learning (ICL): a technique where pre-trained LLMs learn new tasks from a few examples in the form of prompt, without requiring fine-tuning or parameter updates.

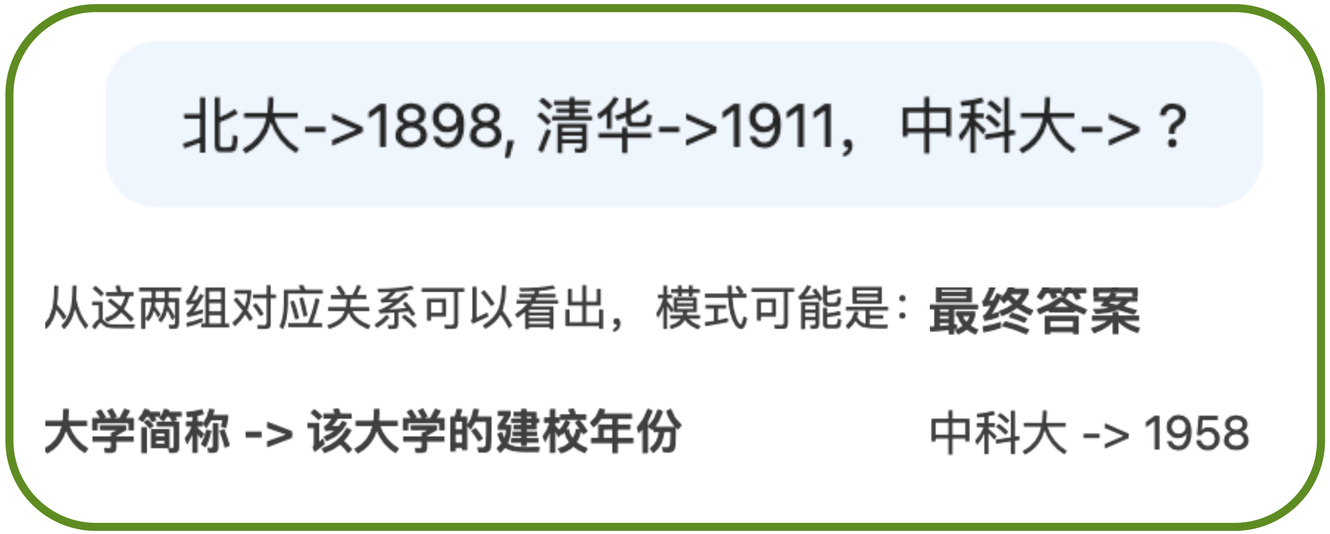


The parameters in the LLM do NOT change

How TabPFN works?

In-context learning

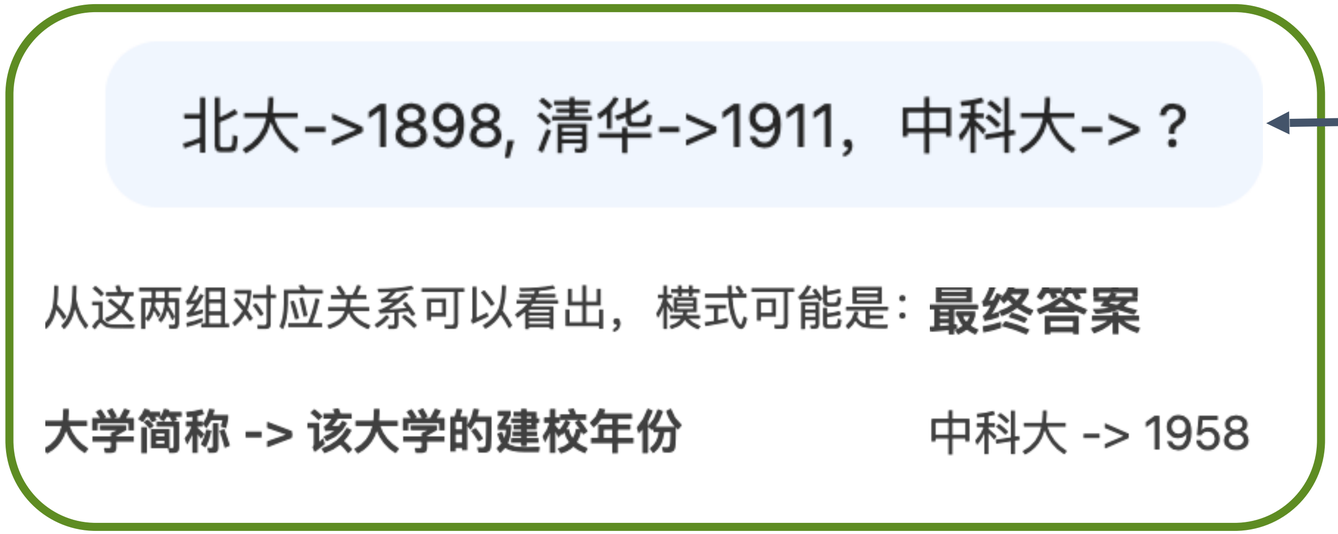
In-context learning (ICL): a technique where pre-trained LLMs learn new tasks from a few examples in the form of prompt, without requiring fine-tuning or parameter updates.



How TabPFN works?

In-context learning

In-context learning (ICL): a technique where pre-trained LLMs learn new tasks from a few examples in the form of prompt, without requiring fine-tuning or parameter updates.



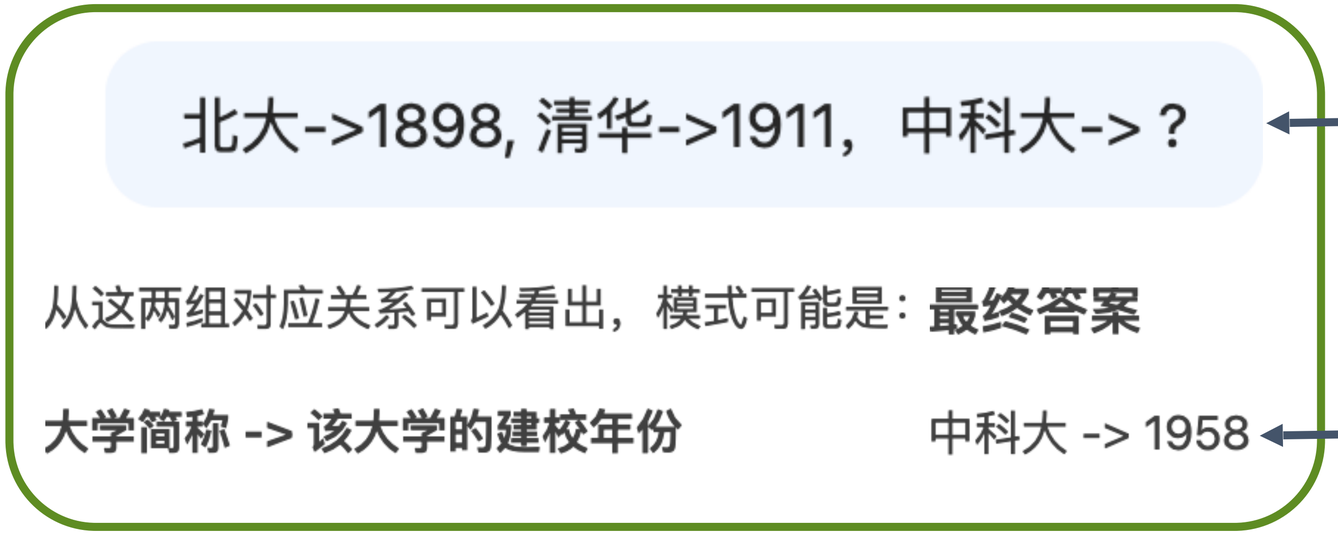
$$\{(X_i, Y_i)\}_{i=1}^{n}, X_{n+1}$$

New training set as examples in prompt

How TabPFN works?

In-context learning

In-context learning (ICL): a technique where pre-trained LLMs learn new tasks from a few examples in the form of prompt, without requiring fine-tuning or parameter updates.



$\{(X_i, Y_i)\}_{i=1}^{n}, X_{n+1}$

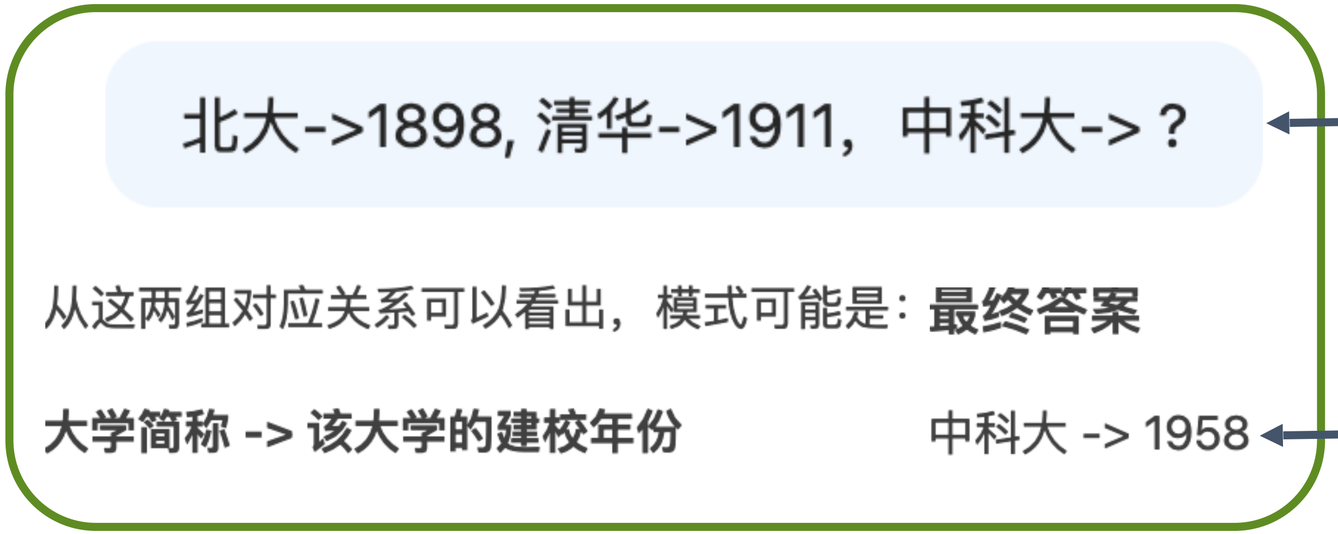
New training set as examples in prompt

$\hat{Y}_{n+1}$

How TabPFN works?

In-context learning

In-context learning (ICL): a technique where pre-trained LLMs learn new tasks from a few examples in the form of prompt, without requiring fine-tuning or parameter updates.



$\{(X_i, Y_i)\}_{i=1}^{n}, X_{n+1}$

New training set as examples in prompt

$\hat{Y}_{n+1}$

The parameters in the TabPFN do NOT change

TabPFN's ICL in statistical language

Amortized Bayesian inference

$$\mathcal{D} = (X_1, Y_1, \ldots, X_n, Y_n) \sim \pi^* \in \Pi$$

TabPFN's ICL in statistical language

Amortized Bayesian inference

$$\mathcal{D} = (X_1, Y_1, \ldots, X_n, Y_n) \sim \pi^* \in \boxed{\Pi}$$ prior p

TabPFN's ICL in statistical language

Amortized Bayesian inference

$$\mathcal{D} = (X_1, Y_1, \ldots, X_n, Y_n) \sim \pi^* \in \boxed{\Pi}$$ **prior p**

The posterior predictive distribution for Y_{n+1} given $X_{n+1} = x$ is

$$p(y|x, \mathcal{D}) = \int_{\Pi} \pi(y|x) p(\pi|\mathcal{D})\, d\pi$$

TabPFN's ICL in statistical language

Amortized Bayesian inference

$$\mathcal{D} = (X_1, Y_1, \ldots, X_n, Y_n) \sim \pi^* \in \boxed{\Pi}$$ **prior p**

The posterior predictive distribution for Y_{n+1} given $X_{n+1} = x$ is

$$\boxed{p(y|x, \mathcal{D})} = \int_\Pi \pi(y|x) p(\pi|\mathcal{D})\, d\pi$$

train a model for approximation

TabPFN's ICL in statistical language

Amortized Bayesian inference

$$\mathcal{D} = (X_1, Y_1, \ldots, X_n, Y_n) \sim \pi^* \in \boxed{\Pi}$$ prior p

The posterior predictive distribution for Y_{n+1} given $X_{n+1} = x$ is

$$\boxed{p(y|x, \mathcal{D})} = \int_\Pi \pi(y|x) p(\pi|\mathcal{D})\, d\pi$$

train a model for approximation

Key innovation

Traditional Bayesian approaches (e.g., MCMC, Variational inference): approximate

$$x \mapsto p(y|x, \mathcal{D})$$

for a **fixed dataset** $\mathcal{D}$

TabPFN: approximate the map **jointly** over x and $\mathcal{D}$

$$(\mathcal{D}, x) \mapsto p(y|x, \mathcal{D})$$

No per-dataset training is required

TabPFN's ICL in statistical language

Amortized Bayesian inference

$$\mathcal{D} = (X_1, Y_1, \ldots, X_n, Y_n) \sim \pi^* \in \boxed{\Pi}$$ **prior p**

The posterior predictive distribution for Y_{n+1} given $X_{n+1} = x$ is

$$\boxed{p(y|x, \mathcal{D})} = \int_{\Pi} \pi(y|x) p(\pi|\mathcal{D})\, d\pi$$

train a model for approximation

Prior fitted network
1. Specifically designed transformer (7M parameters)
2. Trained purely on synthetic datasets (130M datasets)

Key innovation

Traditional Bayesian approaches (e.g., MCMC, Variational inference): approximate

$$x \mapsto p(y|x, \mathcal{D})$$

for a **fixed dataset** $\mathcal{D}$

TabPFN: approximate the map **jointly** over x and $\mathcal{D}$

$$(\mathcal{D}, x) \mapsto p(y|x, \mathcal{D})$$

No per-dataset training is required

Case studies

1 **Semi-supervised parameter estimation**

2 **Heterogeneous treatment effects estimation**

3 **Prediction under covariate shift**

Case studies

We compare with SOTA (recently published in top statistics journal) using their experimental setting.

1 **Semi-supervised parameter estimation**

2 **Heterogeneous treatment effects estimation**

3 **Prediction under covariate shift**

Semi-supervised learning

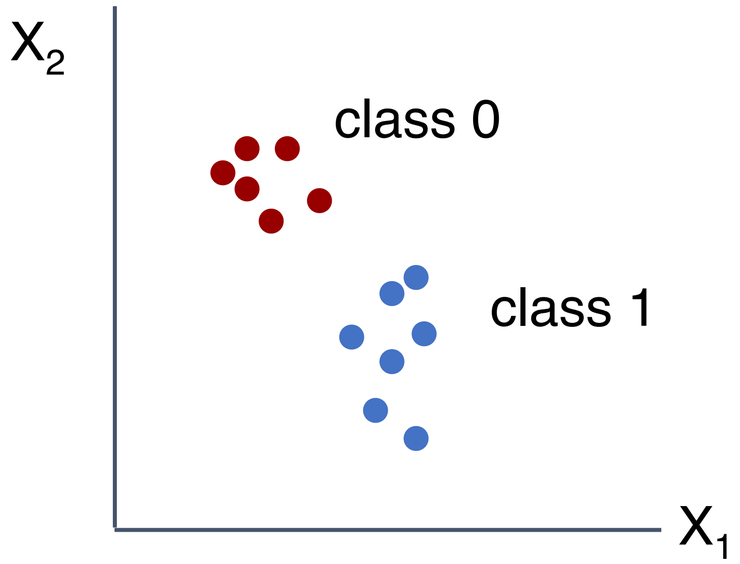
Problem & motivation

Labeled data is limited; unlabeled data is plentiful

Semi-supervised learning

Problem & motivation

Labeled data is limited; unlabeled data is plentiful

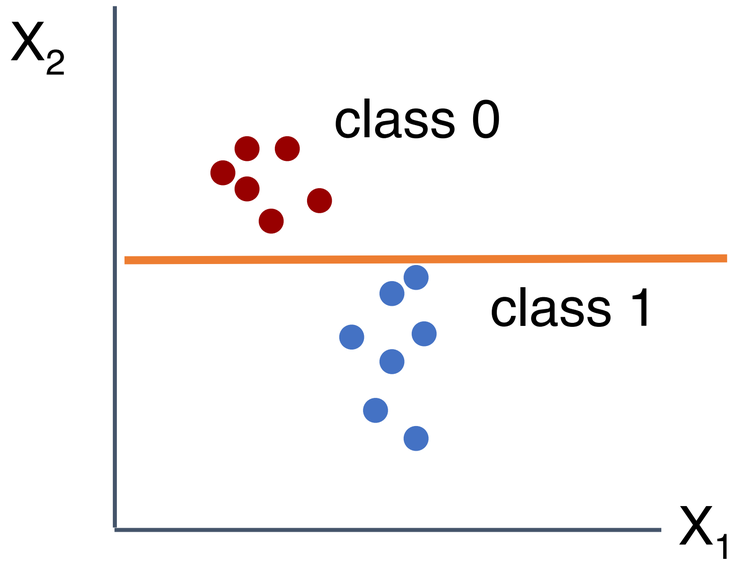


n labeled samples $\mathcal{D}_L = \{(X_i, Y_i)\}_{i=1}^{n}$

Semi-supervised learning

Problem & motivation

Labeled data is limited; unlabeled data is plentiful

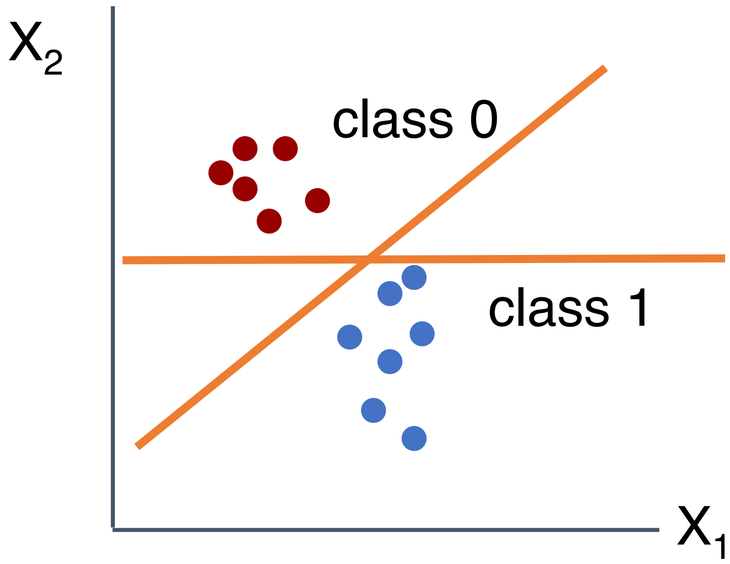


n labeled samples $\mathcal{D}_L = \{(X_i, Y_i)\}_{i=1}^{n}$

Semi-supervised learning

Problem & motivation

Labeled data is limited; unlabeled data is plentiful

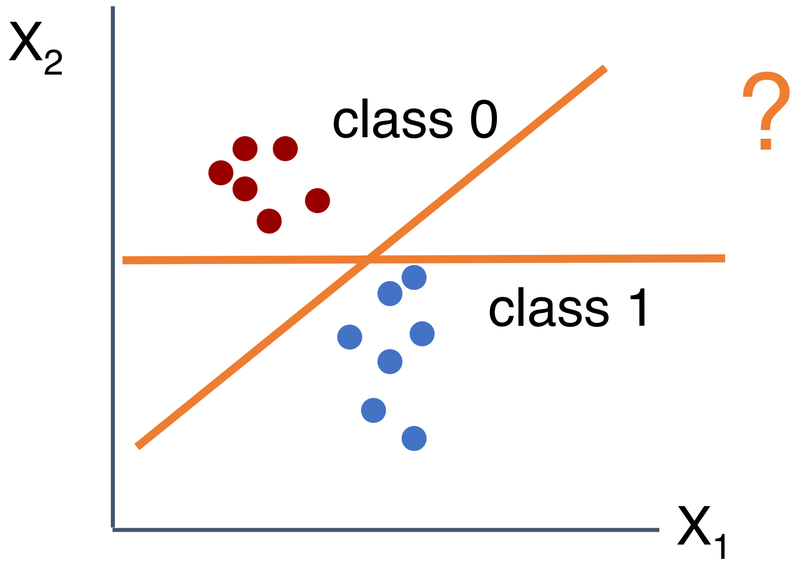


n labeled samples $\mathcal{D}_L = \{(X_i, Y_i)\}_{i=1}^{n}$

Semi-supervised learning

Problem & motivation

Labeled data is limited; unlabeled data is plentiful

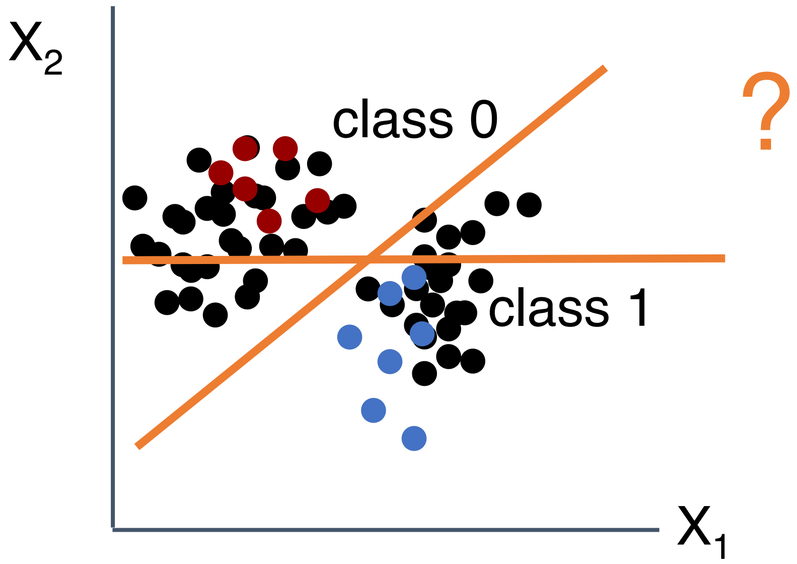


n labeled samples $\mathcal{D}_L = \{(X_i, Y_i)\}_{i=1}^{n}$

Semi-supervised learning

Problem & motivation

Labeled data is limited; unlabeled data is plentiful

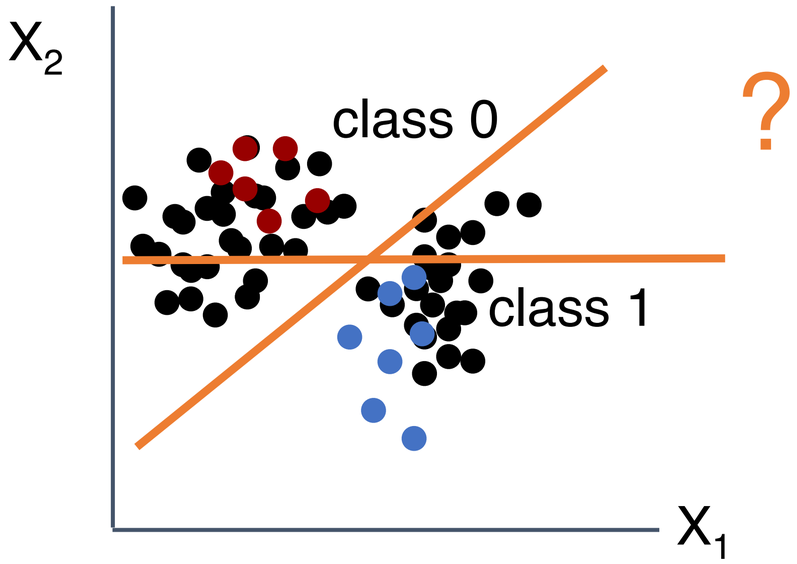


n labeled samples $\mathcal{D}_L = \{(X_i, Y_i)\}_{i=1}^{n}$

Semi-supervised learning

Problem & motivation

Labeled data is limited; unlabeled data is plentiful



?

n labeled samples $\mathcal{D}_L = \{(X_i, Y_i)\}_{i=1}^{n}$

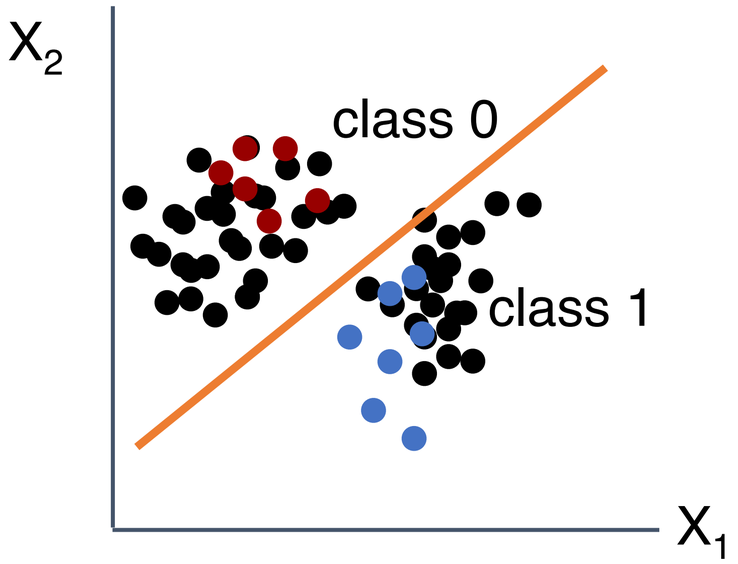
m unlabeled samples $\mathcal{D}_U = \{X_i\}_{i=n+1}^{n+m}$

$$m \gg n$$

Semi-supervised learning

Problem & motivation

Labeled data is limited; unlabeled data is plentiful



n labeled samples $\mathcal{D}_L = \{(X_i, Y_i)\}_{i=1}^{n}$

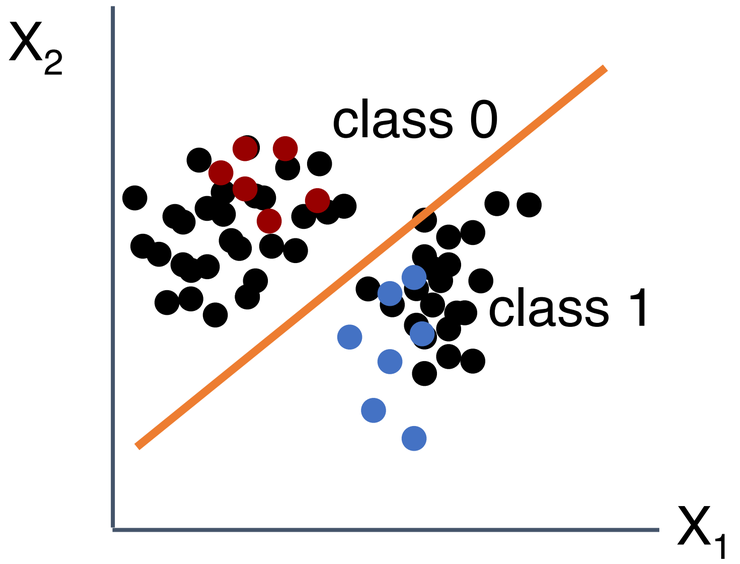
m unlabeled samples $\mathcal{D}_U = \{X_i\}_{i=n+1}^{n+m}$

$$m \gg n$$

Semi-supervised learning

Problem & motivation

Labeled data is limited; unlabeled data is plentiful



True parameter

$$\theta^* = \text{argmin}_{\theta \in \Theta} \mathbb{E}_{P_{X,Y}} \{ l(\theta; X, Y) \}$$

n labeled samples $\mathcal{D}_L = \{(X_i, Y_i)\}_{i=1}^{n}$

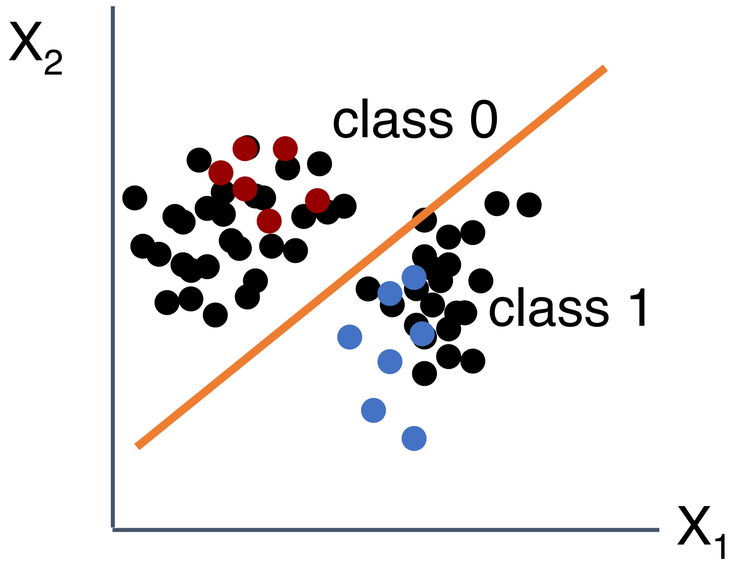
m unlabeled samples $\mathcal{D}_U = \{X_i\}_{i=n+1}^{n+m}$

$$m \gg n$$

Semi-supervised learning

Problem & motivation

Labeled data is limited; unlabeled data is plentiful



True parameter

$$\theta^* = \text{argmin}_{\theta \in \Theta} \mathbb{E}_{P_{X,Y}} \{ l(\theta; X, Y) \}$$

true distribution of the data

n labeled samples $\mathcal{D}_L = \{(X_i, Y_i)\}_{i=1}^{n}$

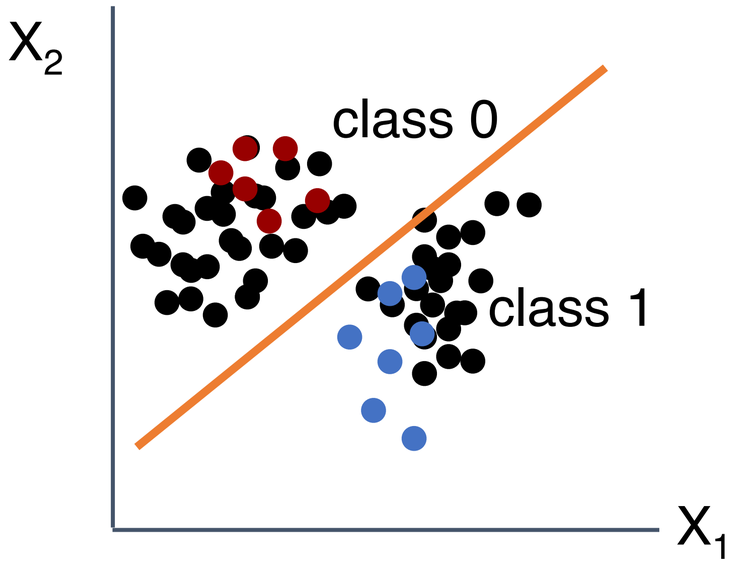
m unlabeled samples $\mathcal{D}_U = \{X_i\}_{i=n+1}^{n+m}$

$$m \gg n$$

Semi-supervised learning

Problem & motivation

Labeled data is limited; unlabeled data is plentiful



True parameter

self-specified loss function

$$\theta^* = \text{argmin}_{\theta \in \Theta} \mathbb{E}_{P_{X,Y}} \{ l(\theta; X, Y) \}$$

true distribution of the data

n labeled samples $\mathcal{D}_L = \{(X_i, Y_i)\}_{i=1}^{n}$

m unlabeled samples $\mathcal{D}_U = \{X_i\}_{i=n+1}^{n+m}$

$$m \gg n$$

Semi-supervised learning

Methods

Semi-supervised learning

Methods

- **SLZ (Song, Lin and Zhou. 2023 JASA) SOTA**

Semi-supervised learning

Methods

- **SLZ (Song, Lin and Zhou. 2023 JASA)** SOTA
- **Empirical risk minimizer (ERM)**

$$\hat{\theta}_{\mathrm{ERM}}(\mathcal{D}) = \arg\min_{\theta} \sum_{(X,Y)\in\mathcal{D}} l(\theta; X, Y)$$

Semi-supervised learning

Methods

- **SLZ (Song, Lin and Zhou. 2023 JASA)** SOTA
- **Empirical risk minimizer (ERM)**

$$\hat{\theta}_{\mathrm{ERM}}(\mathcal{D}) = \arg\min_\theta \sum_{(X,Y)\in\mathcal{D}} l(\theta; X, Y)$$

- **TabPFN-based methods**

1. Use TabPFN to impute labels: $\hat{\mathcal{D}}_L = \{(X_i, \hat{Y}_i)\}_{i=1}^{n}$ $\hat{\mathcal{D}}_U = \{(X_i, \hat{Y}_i)\}_{i=n+1}^{n+m}$

Semi-supervised learning

Methods

- **SLZ (Song, Lin and Zhou. 2023 JASA)** SOTA
- **Empirical risk minimizer (ERM)**

$$\hat{\theta}_{\mathrm{ERM}}(\mathcal{D}) = \arg\min_{\theta} \sum_{(X,Y)\in\mathcal{D}} l(\theta; X, Y)$$

- **TabPFN-based methods**

1. Use TabPFN to impute labels: $\hat{\mathcal{D}}_L = \{(X_i, \hat{Y}_i)\}_{i=1}^{n}$ $\hat{\mathcal{D}}_U = \{(X_i, \hat{Y}_i)\}_{i=n+1}^{n+m}$
2. **TabPFN-I:** $\hat{\theta}_{\mathrm{ERM}}(\hat{\mathcal{D}}_L \cup \hat{\mathcal{D}}_U)$

Semi-supervised learning

Methods

- **SLZ (Song, Lin and Zhou. 2023 JASA)** SOTA
- **Empirical risk minimizer (ERM)**

$$\hat{\theta}_{\mathrm{ERM}}(\mathcal{D}) = \arg\min_{\theta} \sum_{(X,Y)\in\mathcal{D}} l(\theta; X, Y)$$

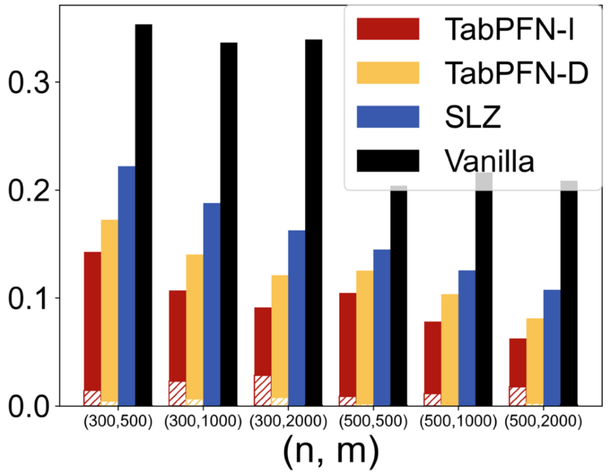
- **TabPFN-based methods**

1. Use TabPFN to impute labels: $\hat{\mathcal{D}}_L = \{(X_i, \hat{Y}_i)\}_{i=1}^{n}$ $\hat{\mathcal{D}}_U = \{(X_i, \hat{Y}_i)\}_{i=n+1}^{n+m}$
2. **TabPFN-I:** $\hat{\theta}_{\mathrm{ERM}}(\hat{\mathcal{D}}_L \cup \hat{\mathcal{D}}_U)$
3. **TabPFN-D(Angelopoulos et al. 2023 Science):**
   - estimate the imputation bias by $\Delta = \hat{\theta}_{\mathrm{ERM}}(\hat{\mathcal{D}}_L) - \hat{\theta}_{\mathrm{ERM}}(\mathcal{D}_L)$
   - return the estimator $\hat{\theta}_{\mathrm{ERM}}(\hat{\mathcal{D}}_L \cup \hat{\mathcal{D}}_U) - \Delta$

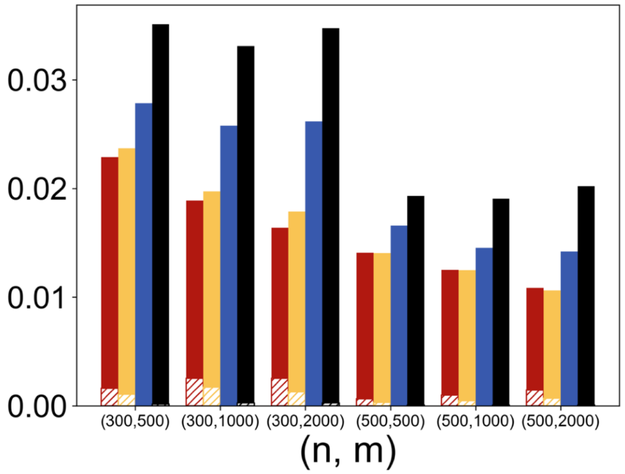
Semi-supervised learning

Empirical results

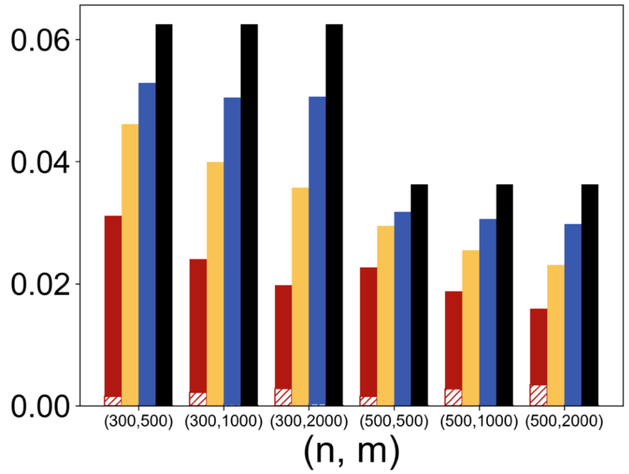
Variance Bias^2



Linear regression



Logistic regression



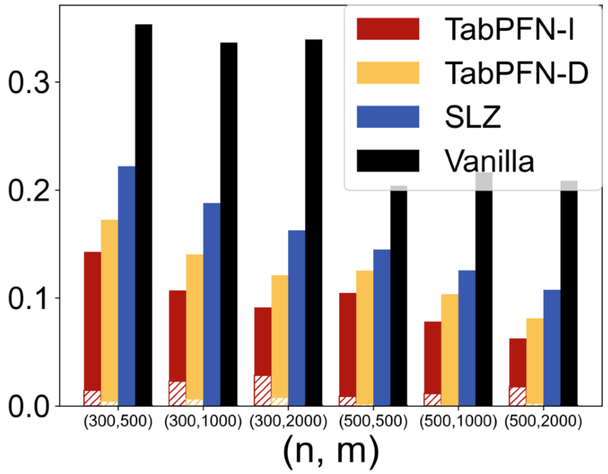
Quantile regression

Semi-supervised learning

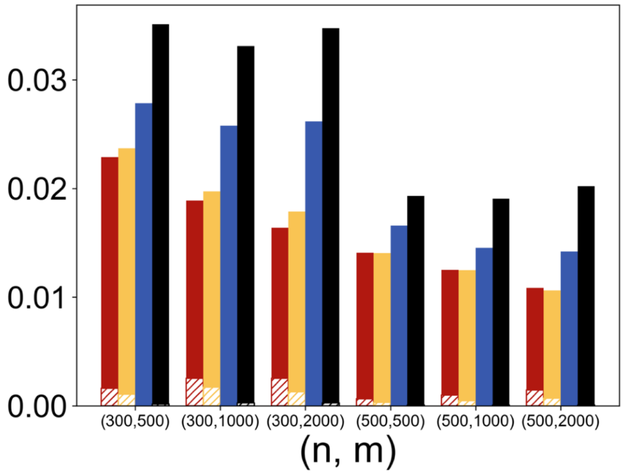
Empirical results

TabPFN beats SOTA in estimating regression coefficients when model is misspecified

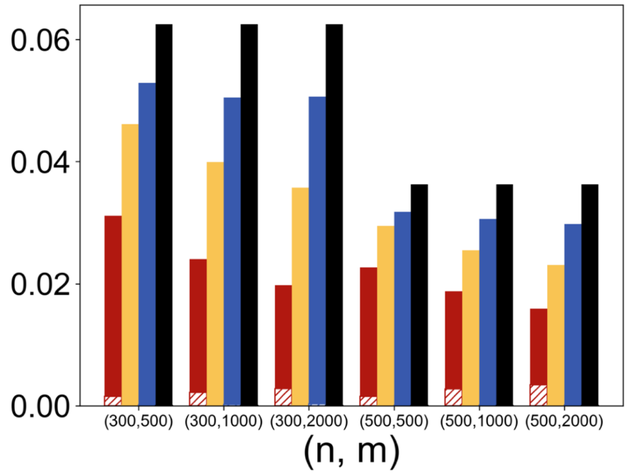
Variance Bias^2



Linear regression



Logistic regression



Quantile regression

Heterogeneous treatment effects estimation

Problem & motivation

Heterogeneous treatment effects estimation

Problem & motivation

Heterogeneous treatment effects estimation

Problem & motivation

Heterogeneous treatment effects estimation

Problem & motivation

Heterogeneous treatment effects (HTE): how different groups of individuals benefit differentially from the same treatment

Heterogeneous treatment effects estimation

Problem & motivation

Heterogeneous treatment effects (HTE): how different groups of individuals benefit differentially from the same treatment

Heterogeneous treatment effects estimation

Problem & motivation

Heterogeneous treatment effects (HTE): how different groups of individuals benefit differentially from the same treatment

Potential outcome framework:

- Binary treatment: $T_i \in \{0, 1\}$
- Potential outcomes: $Y_i(0), Y_i(1)$ (control/treatment), $Y_i^{\text{obs}} = Y_i(T_i)$
- Average treatment effect (ATE): $\tau = \mathbb{E}\{Y(1) - Y(0)\}$
- Conditional average treatment effect (CATE):

$$\tau(x) = \mathbb{E}\{Y(1) - Y(0) \mid X = x\}$$

Heterogeneous treatment effects estimation

Problem & motivation

Heterogeneous treatment effects (HTE): how different groups of individuals benefit differentially from the same treatment

Potential outcome framework:

- Binary treatment: $T_i \in \{0, 1\}$
- Potential outcomes: $Y_i(0), Y_i(1)$ (control/treatment), $Y_i^{\text{obs}} = Y_i(T_i)$
- Average treatment effect (ATE): $\tau = \mathbb{E}\{Y(1) - Y(0)\}$
- Conditional average treatment effect (CATE):

$$\tau(x) = \mathbb{E}\{Y(1) - Y(0) \mid X = x\}$$

Goal: Estimate this function

CATE estimators

Two “nuisance” parameters:

- Response functions: $\mu_j(x) = \mathbb{E}\{Y(j) \mid X = x\}, \quad j = 0, 1$
- Propensity score: $e(x) = \mathbb{P}\{T = 1 \mid X = x\}$

CATE estimators

Two “nuisance” parameters:

- Response functions: $\mu_j(x) = \mathbb{E}\{Y(j) \mid X = x\}, \quad j = 0, 1$
- Propensity score: $e(x) = \mathbb{P}\{T = 1 \mid X = x\}$

Fact:

$$\tau(x) = \mu_1(x) - \mu_0(x)$$

$$\tau(x) = \mathbb{E}\left\{ \frac{T \cdot Y}{e(x)} - \frac{(1-T) \cdot Y}{1 - e(x)} \middle| X = x \right\}$$

CATE estimators

Two “nuisance” parameters:

- Response functions: $\mu_j(x) = \mathbb{E}\{Y(j) \mid X = x\}, \quad j = 0, 1$
- Propensity score: $e(x) = \mathbb{P}\{T = 1 \mid X = x\}$

Fact:

$$\tau(x) = \mu_1(x) - \mu_0(x)$$

$$\tau(x) = \mathbb{E}\left\{ \frac{T \cdot Y}{e(x)} - \frac{(1-T) \cdot Y}{1 - e(x)} \middle| X = x \right\}$$

Idea:

CATE estimators

Two "nuisance" parameters:

- Response functions: $\mu_j(x) = \mathbb{E}\{Y(j) \mid X = x\}, \quad j = 0, 1$
- Propensity score: $e(x) = \mathbb{P}\{T = 1 \mid X = x\}$

Fact:

$$\tau(x) = \mu_1(x) - \mu_0(x)$$

$$\tau(x) = \mathbb{E}\left\{ \frac{T \cdot Y}{e(x)} - \frac{(1-T) \cdot Y}{1 - e(x)} \middle| X = x \right\}$$

Idea:

- Step 1: Estimate $e(x)$ and/or $\mu_j(x)$ using regression and classification algorithms

CATE estimators

Two "nuisance" parameters:

- Response functions: $\mu_j(x) = \mathbb{E}\{Y(j) \mid X = x\}, \quad j = 0, 1$
- Propensity score: $e(x) = \mathbb{P}\{T = 1 \mid X = x\}$

Fact:

$$\tau(x) = \mu_1(x) - \mu_0(x)$$

$$\tau(x) = \mathbb{E}\left\{ \frac{T \cdot Y}{e(x)} - \frac{(1-T) \cdot Y}{1 - e(x)} \middle| X = x \right\}$$

Idea:

- Step 1: Estimate $e(x)$ and/or $\mu_j(x)$ using regression and classification algorithms
- Step 2: Combine (in some way) to get estimate for $\tau(x)$

CATE estimators

Two "nuisance" parameters:

- Response functions: $\mu_j(x) = \mathbb{E}\{Y(j) \mid X = x\}, \quad j = 0, 1$
- Propensity score: $e(x) = \mathbb{P}\{T = 1 \mid X = x\}$

Fact:

$$\tau(x) = \mu_1(x) - \mu_0(x)$$

$$\tau(x) = \mathbb{E}\left\{ \frac{T \cdot Y}{e(x)} - \frac{(1-T) \cdot Y}{1 - e(x)} \middle| X = x \right\}$$

Idea: **Metalearner framework (Kunzel et al., 2019)**

- Step 1: Estimate $e(x)$ and/or $\mu_j(x)$ using regression and classification algorithms — **base learners**
- Step 2: Combine (in some way) to get estimate for $\tau(x)$ — **meta-learners**

CATE estimators

S-learner

1. Form an estimator $\hat{\mu}(t, x)$ for the function
$$(t, x) \mapsto \mu_t(x)$$
by jointly regressing Y_i^{obs} on T_i and X_i
1. S-learner is

$$\hat{\tau}_S(x) = \hat{\mu}(1, x) - \hat{\mu}(0, x)$$

DR-learner SOTA

1. Form estimates $\hat{\mu}_j(x)$ for $\mu_j(x)$, and $\hat{e}(x)$ for propensity score
2. Compute doubly robust pseudo-outcomes $\tilde{Y}_i$
3. Estimate CATE by regressing $\tilde{Y}_i$ on X_i

$$\tilde{Y}_i = \frac{T_i(Y_i - \hat{\mu}_1(X_i))}{\hat{e}(X_i)} - \frac{(1-T_i)(Y_i - \hat{\mu}_0(X_i))}{1-\hat{e}(X_i)} + \hat{\mu}_1(X_i) - \hat{\mu}_0(X_i)$$

CATE estimators

S-learner

1. Form an estimator $\hat{\mu}(t, x)$ for the function
$$(t, x) \mapsto \mu_t(x)$$
by jointly regressing Y_i^{obs} on T_i and X_i
1. S-learner is

$$\hat{\tau}_S(x) = \hat{\mu}(1, x) - \hat{\mu}(0, x)$$

DR-learner SOTA

1. Form estimates $\hat{\mu}_j(x)$ for $\mu_j(x)$, and $\hat{e}(x)$ for propensity score
2. Compute doubly robust pseudo-outcomes $\tilde{Y}_i$
3. Estimate CATE by regressing $\tilde{Y}_i$ on X_i

$$\tilde{Y}_i = \frac{T_i(Y_i - \hat{\mu}_1(X_i))}{\hat{e}(X_i)} - \frac{(1-T_i)(Y_i - \hat{\mu}_0(X_i))}{1-\hat{e}(X_i)} + \hat{\mu}_1(X_i) - \hat{\mu}_0(X_i)$$

Current emphasis:

- Choice of meta-learner more important than base learner
- DR-learner achieves optimal asymptotic rates (under smoothness assumptions)

CATE estimation

Experimental design

Followed Foster and Syrgkanis (2023 AOS) setting

$$T_i \sim \text{Bern}(e(X_i)), \quad X_i \sim U([-0.5, 0.5]^6)$$

$$Y_i \sim \tau(X_i)(T_i - 1/2) + b(X_i) + \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0, \sigma^2)$$

Setup A: complicated $e(x)$ and $b(x)$, but a relatively simple $\tau(x)$
Setup B: large $b(x)$ resulting in substantial confounding, discontinuous $\tau(x)$

Base-learner: estimators for the regression functions (Auto-ML or TabPFN)

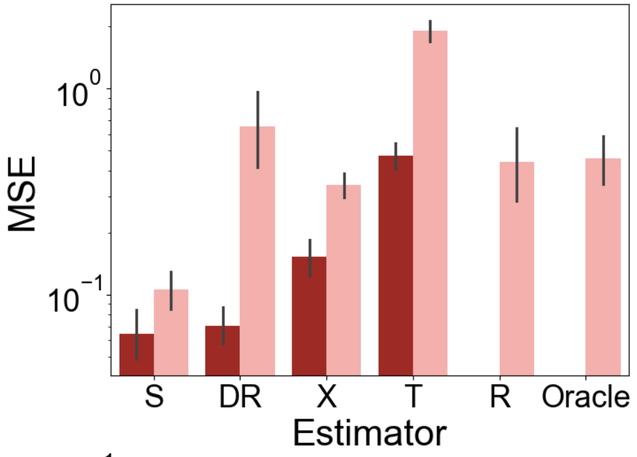
CATE estimation

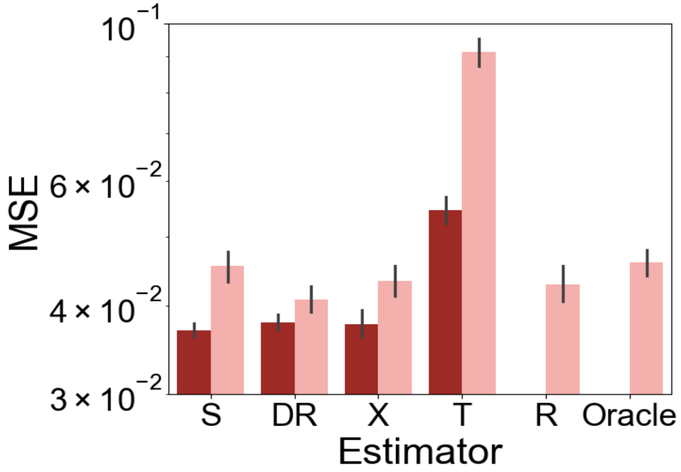
Results

small-sample high variance

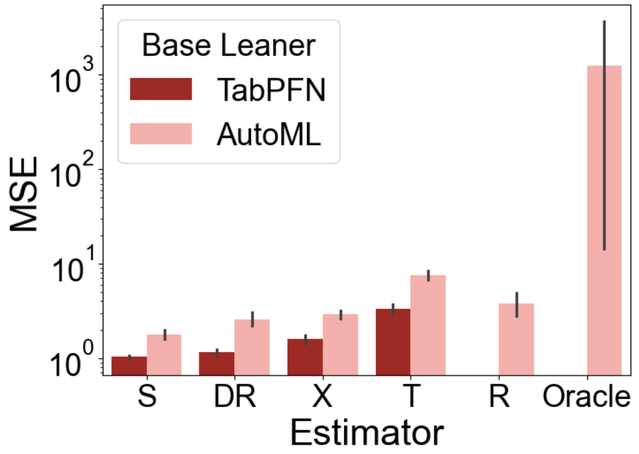
large-sample low variance

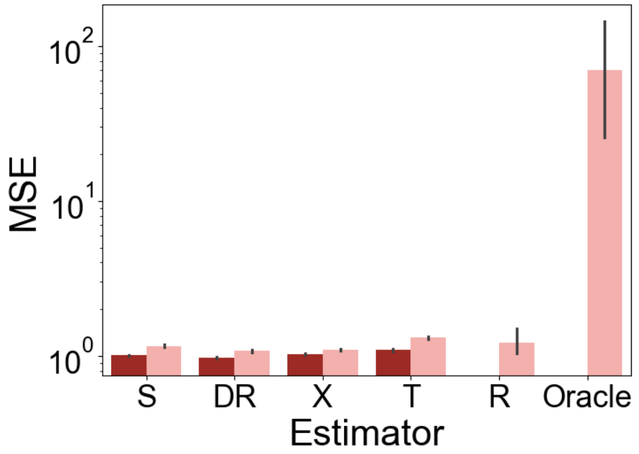
Setup A





Setup B





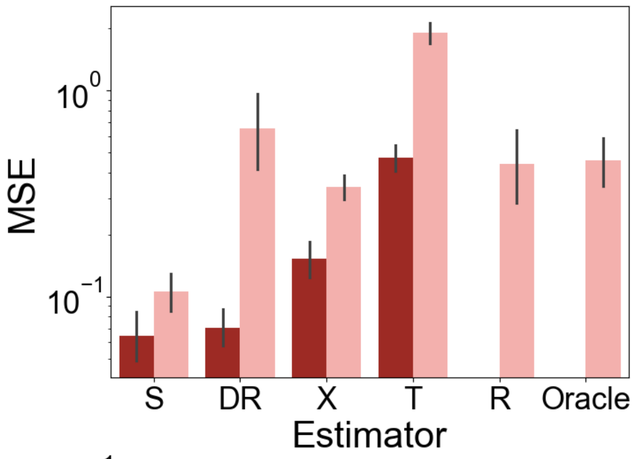
CATE estimation

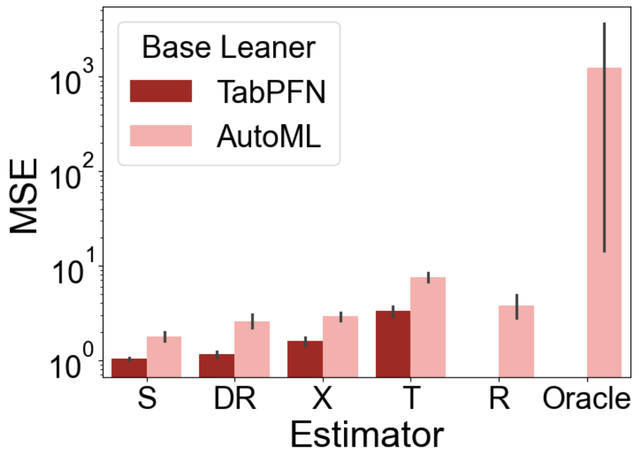
Results

TabPFN > AutoML; S-Learner (TabPFN) wins without double robustness; base learner choice is important

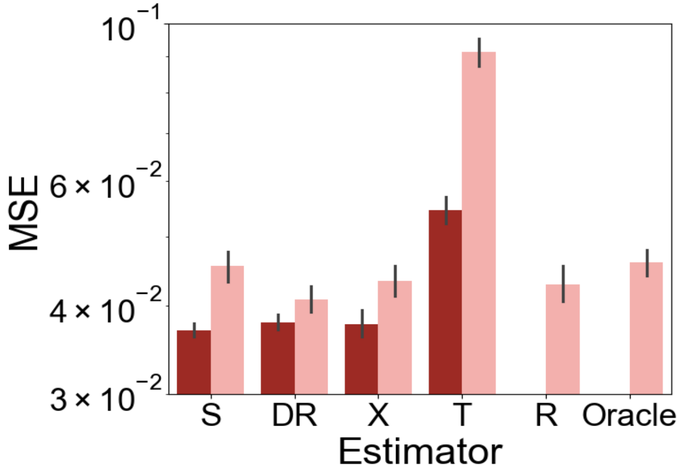
Setup A | **Setup B**

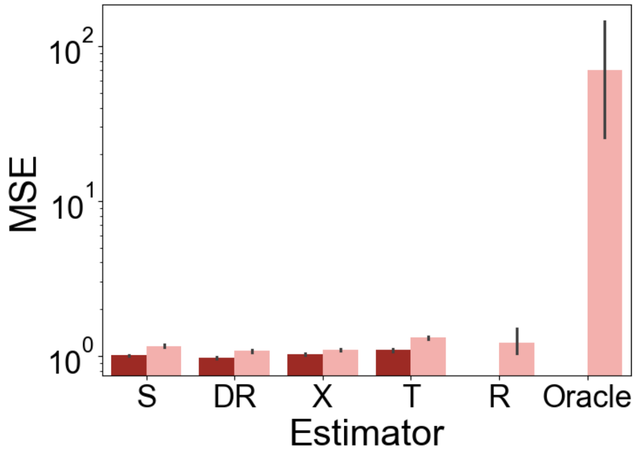
small-sample high variance





large-sample low variance





Why does TabPFN work well?

TabPFN adapts flexibly to various functional structures and can leverage parametric structures, even outperforming specialized methods.

Why does TabPFN work well?

TabPFN adapts flexibly to various functional structures and can leverage parametric structures, even outperforming specialized methods.

1. **Flexible but distinct from GPs:** Like Gaussian processes, TabPFN adapts to any function in low dimensions, but differs in extrapolation, ease of step-function fitting, and handling rapid variations.

1. **Sparsity-dependent performance:** Outperforms LASSO in high-sparsity/low-SNR regimes but lags in dense linear regression (≥20 features).

1. **Robust & efficient classifier:** Robust to label noise and also efficient..

Paper link

Summary

1. TabPFN achieves significantly better estimation accuracy than the comparator methods.

2. TabPFN is NOT a perfect model.

3. The ideas of “in-context learning” and “foundation model” have the potential to supersede existing modeling approaches on a wide range of statistical tasks

4. **We should engage with this opportunity!**