Learning a Data-Driven Policy Network for Pre-Training Automated Feature Engineering

Liyao Li, Haobo Wang, Liangyu Zha, Qingyi Huang,
Sai Wu, Gang Chen, Junbo Zhao*

ICLR
International Conference On Learning Representations

浙江大學
ZHEJIANG UNIVERSITY

Motivation

Background of Tabular Data Prediction

columns = features or attributes of each sample

target

Tabular data:

RowID	Age	Gender	Weight (kg)	Height (m)	Heart disease
1	25	M	75	1.82	No
2	37	F	52	1.57	No
3	75	F	69	1.63	Yes
4	54	M	73	1.68	Yes
…	…	…	…	…	…
10000	69	M	88	1.75	Yes

rows = samples

Tabular data prediction usually is classification or regression task based on target value.

Common models to fit tabular data:

- Neural network (NN): MLP, Wide&Deep, TabNet, NODE, FT-Transformer
- Gradient Boosting Decision Tree (GBDT): Xgboost, LightGBM, CatBoost, Random Forest

Motivation

Background of Tabular Data Prediction

columns = features or attributes of each sample

target

Tabular data:

RowID	Age	Gender	Weight (kg)	Height (m)	Heart disease
1	25	M	75	1.82	No
2	37	F	52	1.57	No
3	75	F	69	1.63	Yes
4	54	M	73	1.68	Yes
…	…	…	…	…	…
10000	69	M	88	1.75	Yes

rows = samples

Recent works [1-3] show that GBDT generally <u>outperforms</u> NN on tabular data prediction.

GBDT is widely used in Kaggle competition and other business scenarios.

[1] Gorishniy, et al. "Revisiting deep learning models for tabular data." NIPS. 2021.

[2] Grinsztajn, et al. "Why do tree-based models still outperform deep learning on typical tabular data?." NIPS. 2022.

[3] Shwartz-Ziv, et al. "Tabular data: Deep learning is not all you need." Information Fusion. 2022.

Motivation

Background of Feature Engineering

When choosing GBDT, it is important to preprocess features since generating informative features and removing redundant ones could significantly improve fitting performance and enjoy high interpretability.

Feature Engineering (FE): feature generation and feature selection.

Common FE operations / actions:

- Unary: *abs, square, inverse, log, sqrt, power3*
- Binary: +, -, ×, ÷, *cross-combine*
- Selection: *delete*

Motivation

Background of Feature Engineering

For example, when dealing with a heart disease classification task, an experienced data scientist would generate a new feature named *body mass index* (BMI=Weight/Height^2) based on prior knowledge, as studies have shown a significant correlation between this index and the probability of developing heart disease.

Thus, this new feature can help the model fit better.

FE action sequence: Divide(*Weight*, Square(*Height*))

RowID	Age	Gender	Weight (kg)	Height (m)	Weight/Height^2	Heart disease
1	25	M	75	1.82	22.64	No
2	37	F	52	1.57	21.10	No
3	75	F	69	1.63	29.73	Yes
4	54	M	73	1.68	25.86	Yes
…	…	…	…	…	…	…
10000	69	M	88	1.75	28.73	Yes

Motivation

Background of Feature Engineering

Feature Engineering is traditionally conducted by human experts.

- They usually tends to investigate the data, such as analyzing its distribution, identifying the outliers, measuring the correlation between columns, etc., and then proposes an FE plan.
- They also can accumulate FE knowledge to accelerate decision-making when facing a new dataset.

Drawback:

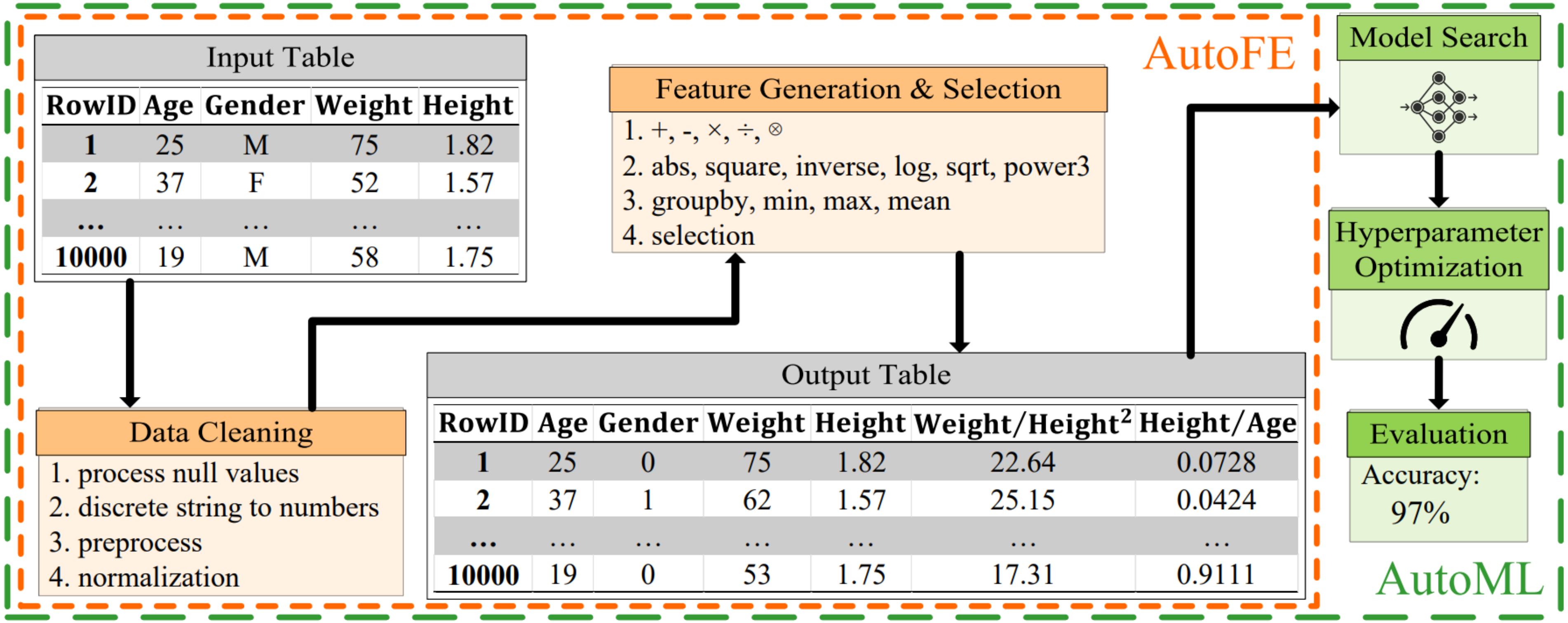
- labor intensive, needs lots of *trial-and-error*
- time-consuming

Motivation

Background of Automated Feature Engineering

Automated Feature Engineering has emerged as a crucial component of AutoML pipeline.

Most AutoFE methods are based on <u>reinforcement learning</u>, inspired by neural architecture search from AutoML.



Input Table

RowID	Age	Gender	Weight	Height
1	25	M	75	1.82
2	37	F	52	1.57
…	…	…	…	…
10000	19	M	58	1.75

Output Table

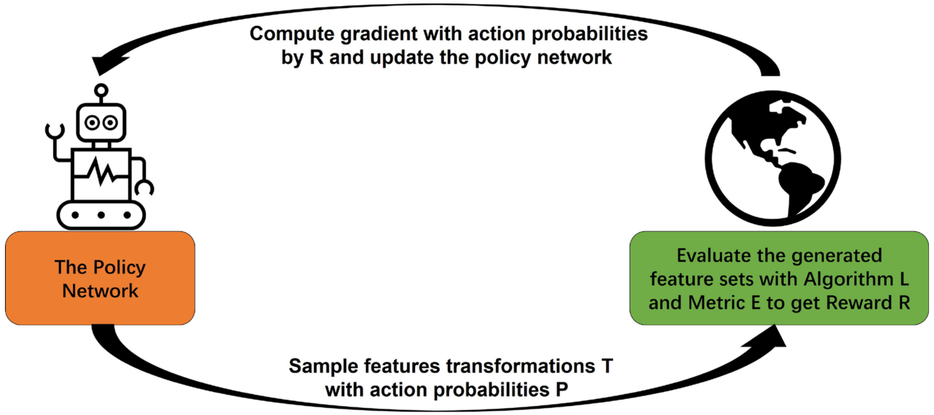
RowID	Age	Gender	Weight	Height	Weight/Height^2	Height/Age
1	25	0	75	1.82	22.64	0.0728
2	37	1	62	1.57	25.15	0.0424
…	…	…	…	…	…	…
10000	19	0	53	1.75	17.31	0.9111

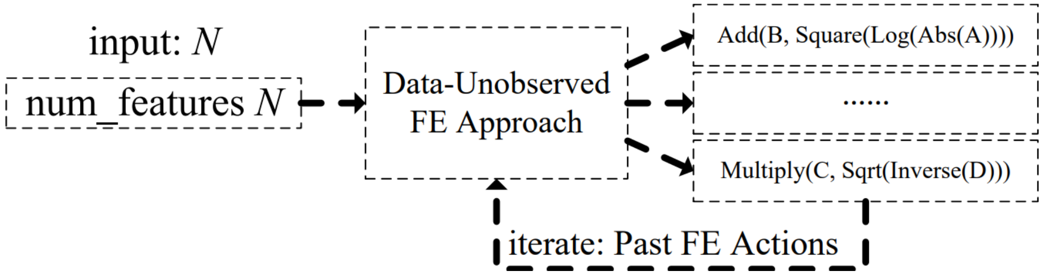
The purpose (or objective function) of feature engineering is defined as Equation 1. Given a dataset $\mathbf{D} = (\mathbf{X}, \mathbf{Y})$ with a set of original features $\mathbf{X}$ and the target $\mathbf{Y}$, search a sequence of transformation actions $\mathcal{T}$ to derive a transformed feature set $\hat{\mathbf{X}_{\mathcal{T}}}$, which maximizes the cross-validation performance $\mathbf{E}(\mathbf{L}(\hat{\mathbf{X}_{\mathcal{T}}}, \mathbf{Y}))$ for a given algorithm $\mathbf{L}$ and a metric $\mathbf{E}$.

$$\mathcal{T} = \arg\max_{\mathcal{T}} \mathbf{E}(\mathbf{L}(\hat{\mathbf{X}_{\mathcal{T}}}, \mathbf{Y})) \quad (1)$$

Existing AutoFE methods [1-3]:

- Generally initialized by the total number of features *N*.
- Embedded the past FE actions as the *state* in their MDP.
- Learning the mapping from one FE action sequence to another with better performance.



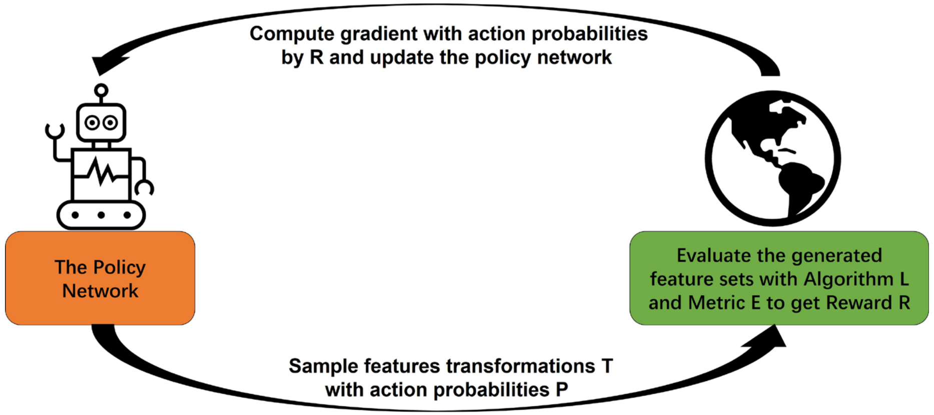


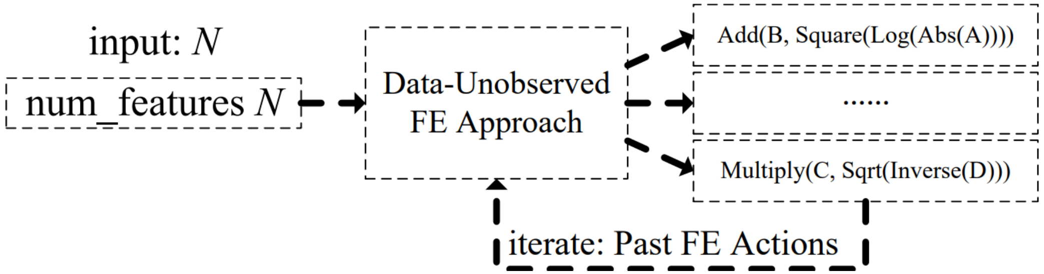
[1] Khurana, et al. "Feature engineering for predictive modeling using reinforcement learning." AAAI. 2018.
[2] Chen, et al. "Neural feature search: A neural architecture for automated feature engineering." ICDM. 2019.
[3] Zhu, et al. "DIFER: differentiable automated feature engineering." International Conference on Automated Machine Learning. 2022.

Limitations of existing AutoFE methods [1-3] :

Deviating from how human experts cope with the data

- Generate features via a data-unobserved way, unhelpful for understanding the data.
- Lack transferability, unfeasible to borrow knowledge from previous training experience to speed up the exploration process when facing a completely new dataset.

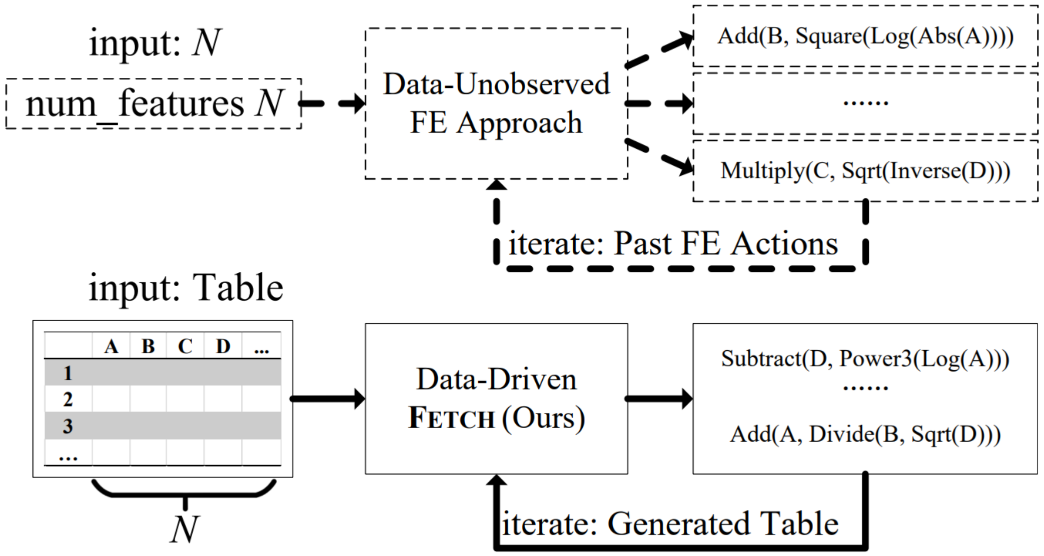




[1] Khurana, et al. "Feature engineering for predictive modeling using reinforcement learning." AAAI. 2018.

[2] Chen, et al. "Neural feature search: A neural architecture for automated feature engineering." ICDM. 2019.

[3] Zhu, et al. "DIFER: differentiable automated feature engineering." International Conference on Automated Machine Learning. 2022.

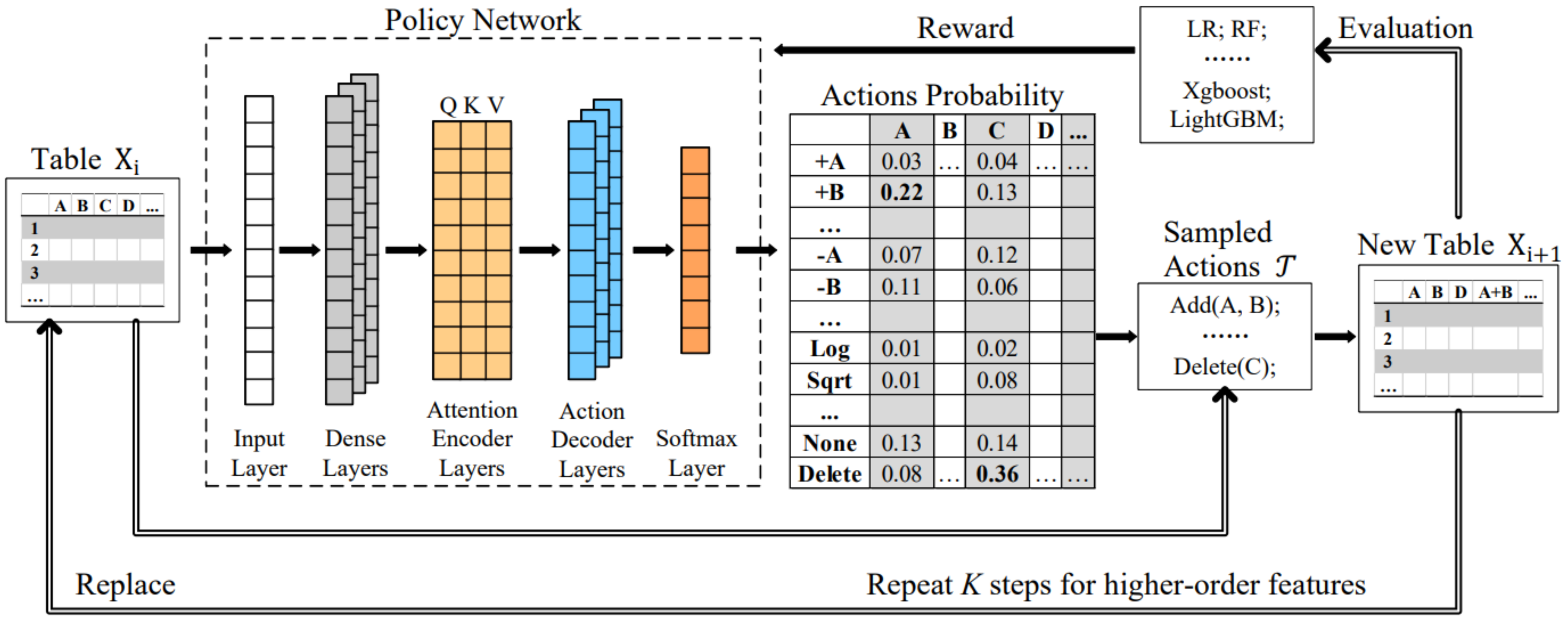


To emulate the human experts:

- ➢ We propose a data-driven AutoFE framework for both classification and regression tasks, dubbed *FETCH*. It learns how to map raw data to suitable FE actions sequence. Empirical results show its on-par or superior performances to the previous state-of-the-art methods.
- ➢ For the first time, we characterize a transferability principle for AutoFE. It reflects how much knowledge or experience a trained policy may be able to accumulate to enable the exploration of unseen datasets, which is also linked to the across-datasets *pre-training* paradigm.

Proposed *FETCH*

Overall Model Architecture

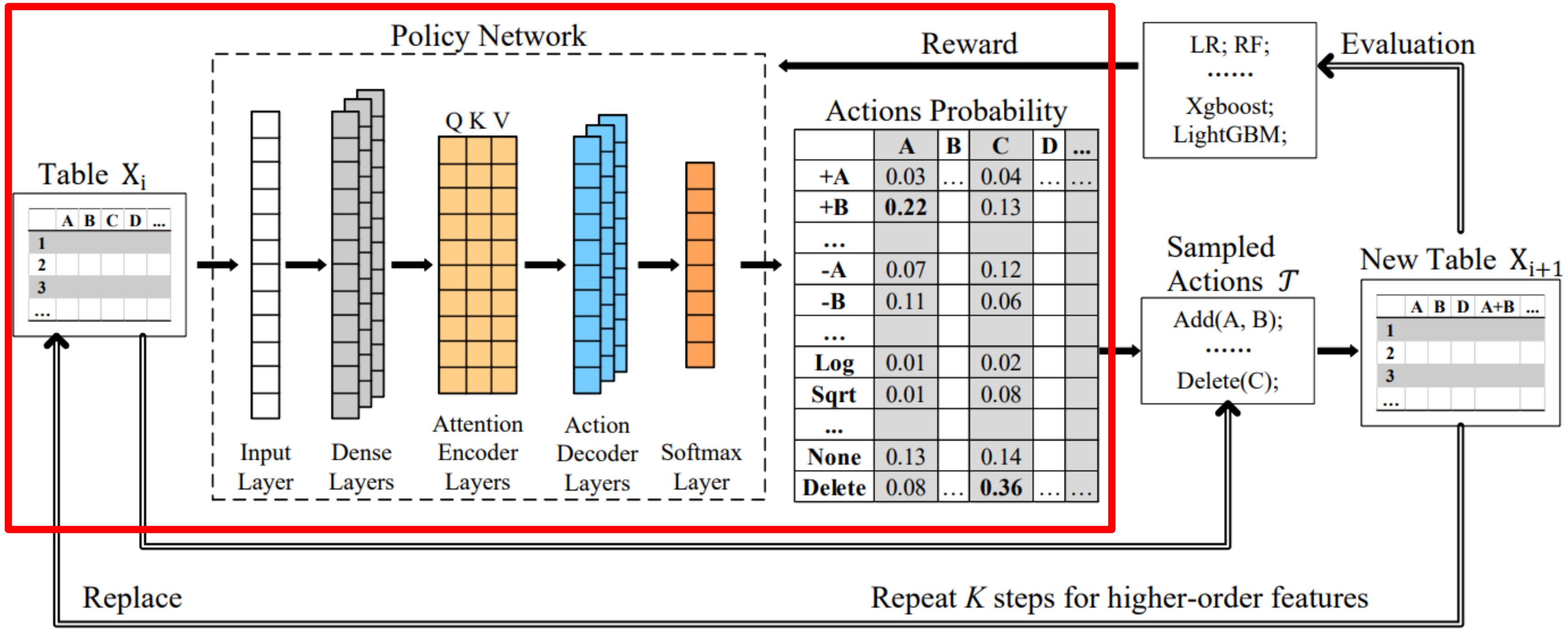


	A	B	C	D	...
+A	0.03	...	0.04	...	...
+B	**0.22**		0.13		
...					
-A	0.07		0.12		
-B	0.11		0.06		
...					
Log	0.01		0.02		
Sqrt	0.01		0.08		
...					
None	0.13		0.14		
Delete	0.08	...	**0.36**	...	...

Proposed *FETCH*

Overall Model Architecture

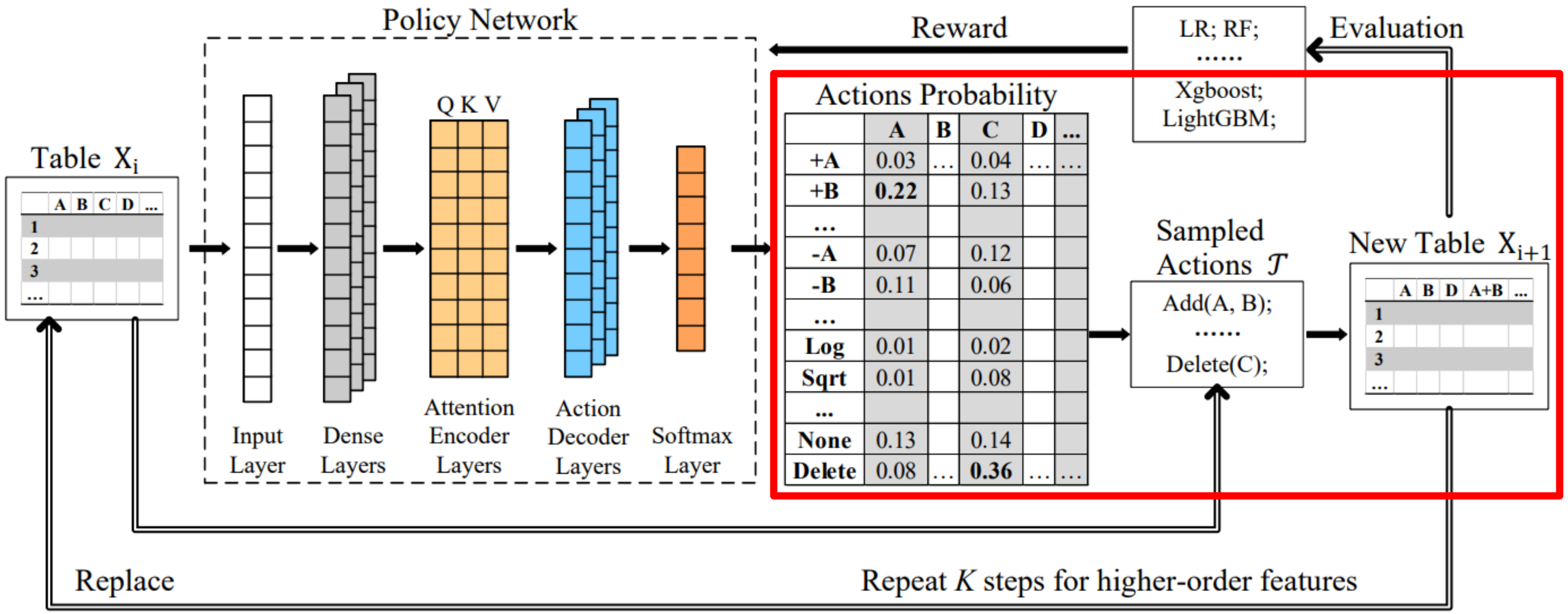
- A table **Xi** is input to the attention-based policy network as the *state*.
- The policy network learns the representation of each column in the table and maps it to the probability of selecting feature engineering actions (like +A, Log, Delete…).



Proposed *FETCH*

Overall Model Architecture

- Sample a set of feature engineering actions, which specifies which columns to operate on and generates a new table $\mathbf{X}_{i+1}$ containing new features.

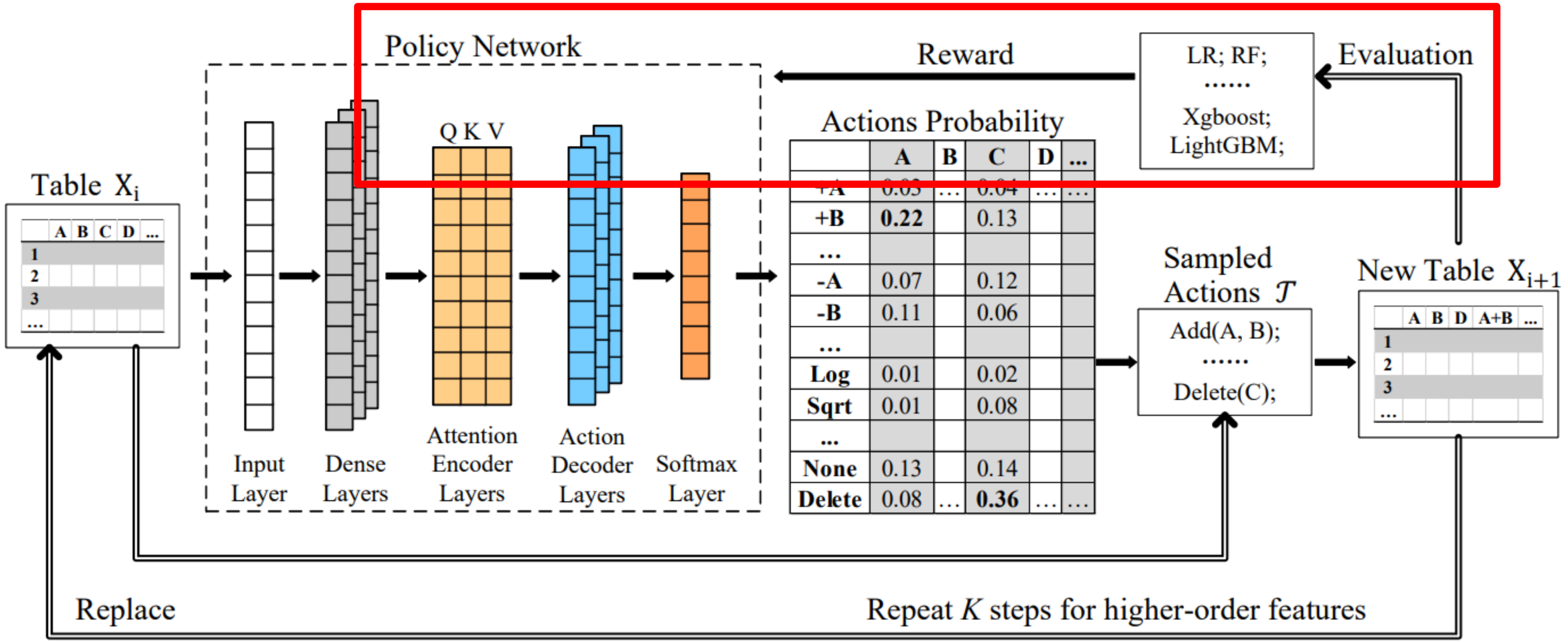


	A	B	C	D	...
+A	0.03	...	0.04	...	...
+B	**0.22**		0.13		
...					
-A	0.07		0.12		
-B	0.11		0.06		
...					
Log	0.01		0.02		
Sqrt	0.01		0.08		
...					
None	0.13		0.14		
Delete	0.08	...	**0.36**	...	...

Proposed *FETCH*

Overall Model Architecture

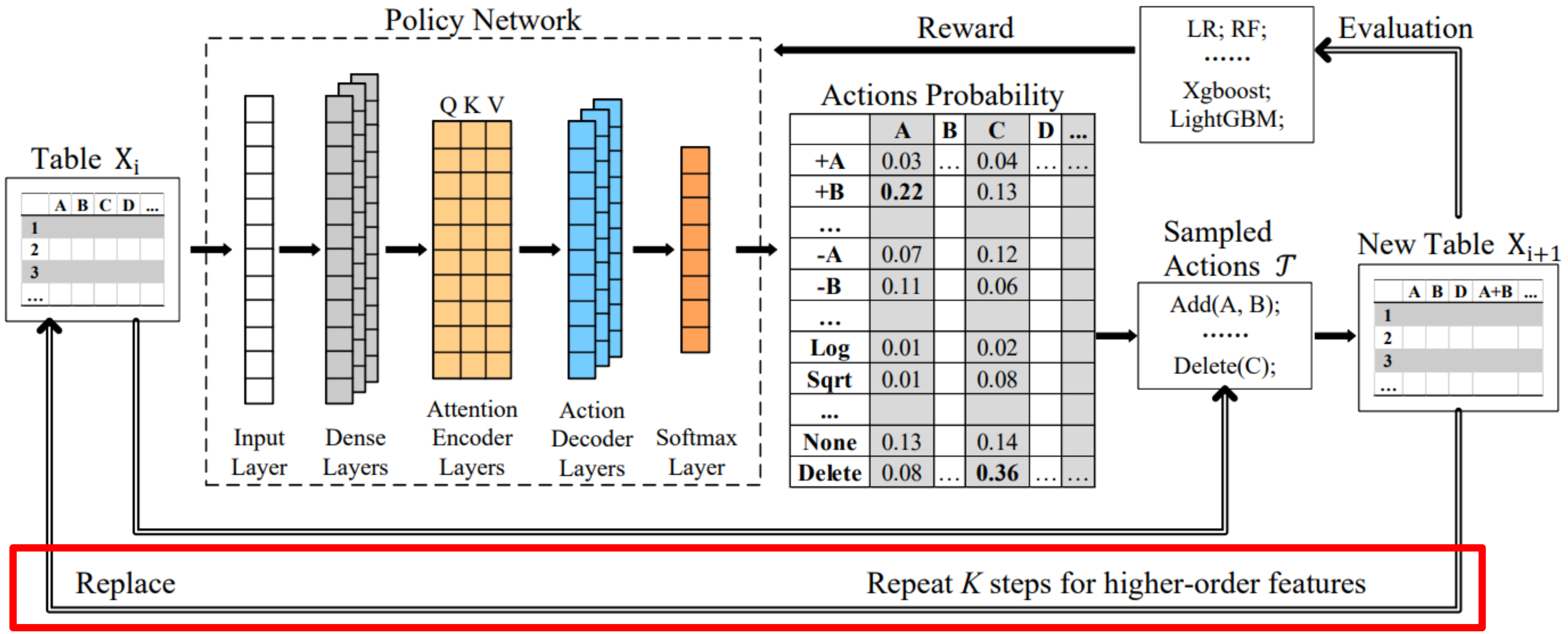
- The new table **Xi+1** is input to pre-defined ML algorithm for cross-validation evaluation.
- The evaluation score is used as *reward* to update policy network through Proximal Policy Optimization (PPO).



Proposed *FETCH*

Overall Model Architecture

- To generate high-order features, the original input **Xi** will be replaced by the newly generated feature set **Xi+1**.



Proposed *FETCH*

Transferability via Pre-Training

- For the first time, our work attempts to use the pre-training of the AutoFE policy network to achieve transfer learning on multiple tabular datasets.

- We pre-train the policy network on some large-scale tabular datasets one by one and then fine-tune it on a smaller target dataset to generate effective features for the target dataset.

- Our experiments show that the pre-trained policy network can significantly reduce the searching epoch number and achieve better performance than training from scratch on the target dataset.

Proposed *FETCH*

MDP Setup

State: Input or post-transformation tabular data $\mathbf{X}$. (the core innovation of our MDP setup)

FE Action Space:

- For numeric features
  - The unary operation: abs, square, inverse, log, sqrt, power3
  - The binary operation: +, -, ×, ÷
- For categorical features: binning, cross-combine
- Feature selection: delete, terminate

Reward Function:

$$\mathcal{R}(\mathbf{X}_i) = \bar{E}(\mathbf{X}_i) + E_{\text{diff}}(\mathbf{X}_i)$$

where $\bar{E}$ represents the average performance obtained from k-fold cross-validation and

$$E_{\text{diff}}(\mathbf{X}_i) = \sum_k \min(0, E_k(\mathbf{X}_i) - \bar{E}(\mathbf{X}_{i-1}))$$

Goals:

- Effectiveness of our data-driven MDP setup, compared with previous SOTA methods like DIFER, NFS.
- Validation of transferability via pre-training.
- Other ablation studies.

Datasets:

- 27 datasets including 11 regression (R) tasks and 16 classifications (C) tasks.
- Publicly published on OpenML, UCI repository and Kaggle.

Metric:

- Classification tasks: F1-score
- Regression tasks: $(1 - (\text{relative absolute error}))$
- Both are higher the better. ↑

Experiments

Effectiveness

- FETCH achieves state-of-the-art performance on 25 out of 27 datasets overall and gets a close second place in the remaining datasets.
- FETCH still has a great advantage, with 18 out of 27 datasets in total performing better than 2 AutoML methods.

Table 1: Effectiveness comparison of FETCH with other AutoFE and AutoML methods. **Bold** indicates superior results amongst AutoFE methods. Note that AutoML methods focus on model search instead of feature engineering.

Dataset	C/R	Instances\Features	AutoFE Methods							AutoML Methods	
			Base	Random	DFS	AutoFeat	NFS	DIFER	FETCH	AutoSklearn	AutoGluon
Airfoil	R	1503\5	0.5068	0.6211	0.6003	0.5955	0.6226	0.6125	**0.6463**	0.5151	0.5083
BikeShare DC	R	10886\11	0.9880	0.9989	0.9990	0.9891	0.9991	0.9995	**0.9997**	0.9911	0.9967
House King County	R	21613\19	0.6843	0.6838	0.6908	0.6917	0.6934	0.6948	**0.7475**	0.7005	0.7442
Housing Boston	R	506\13	0.4641	0.4788	0.4708	0.4703	0.4977	0.5072	**0.5224**	0.4403	0.4857
Openml_586	R	1000\25	0.6564	0.6646	0.7188	0.7178	0.7223	0.6946	**0.7671**	0.7297	0.7904
Openml_589	R	1000\25	0.6395	0.6285	0.6956	0.7278	0.7165	0.6789	**0.7562**	0.7183	0.7998
Openml_607	R	1000\50	0.6363	0.6392	0.6815	0.6499	0.6485	0.6564	**0.7404**	0.7265	0.7694
Openml_616	R	500\50	0.5605	0.5834	0.5807	0.5927	0.5856	0.5982	**0.6749**	0.6618	0.6743
Openml_618	R	1000\50	0.6351	0.6277	0.6848	0.6374	0.6461	0.6553	**0.7351**	0.7198	0.7520
Openml_620	R	1000\25	0.6309	0.6288	0.6528	0.6574	0.6943	0.7262	**0.7506**	0.7199	0.7855
Openml_637	R	500\50	0.5160	0.5478	0.5105	0.5763	0.5739	0.6006	**0.6453**	0.6416	0.6742
Adult Income	C	48842\14	0.8478	0.8485	0.8502	0.8483	0.8497	**0.8584**	0.8537	0.8629	0.8738
Amazon Employee	C	32769\9	0.9450	0.9442	0.9451	0.9453	0.9461	0.9474	**0.9479**	0.9471	0.9473
Credit Default	C	30000\25	0.8044	0.8089	0.8056	0.8086	0.8101	0.8108	**0.8114**	0.8194	0.8214
Credit_a	C	690\6	0.8362	0.8665	0.8216	0.8581	0.8695	0.8638	**0.8754**	0.8623	0.8377
Fertility	C	100\9	0.8700	0.8947	0.7900	0.8910	**0.9189**	0.8800	0.8900	0.8400	0.8800
German Credit	C	1001\24	0.7390	0.7738	0.7490	0.7600	0.7786	0.7730	**0.7910**	0.7460	0.7590
Hepatitis	C	155\12	0.8258	0.8639	0.8516	0.8677	0.8766	0.8839	**0.9290**	0.8065	0.7871
Ionosphere	C	351\34	0.9237	0.9514	0.9373	0.9286	0.9543	0.9515	**0.9716**	0.8194	0.8214
Lymphography	C	690\6	0.8315	0.8480	0.8113	0.8453	0.8614	0.8827	**0.9260**	0.8418	0.8522
Megawatt1	C	4900\12	0.8655	0.8706	0.8813	0.8893	0.9167	0.9089	**0.9209**	0.8853	0.8850
Messidor Features	C	1150\19	0.6594	0.7026	0.7089	0.7359	0.7417	0.7541	**0.7689**	0.7402	0.7255
PimaIndian	C	768\8	0.7566	0.7609	0.7540	0.7643	0.7784	0.7839	**0.7969**	0.7462	0.7631
SpamBase	C	4601\57	0.9154	0.9211	0.9198	0.9237	0.9341	0.9372	**0.9405**	0.9272	0.9042
SpectF	C	267\44	0.7751	0.8221	0.8125	0.8331	0.8608	0.8538	**0.8838**	0.7828	0.7010
Wine Quality Red	C	999\12	0.5597	0.5774	0.5422	0.5641	0.5814	0.5779	**0.6042**	0.5804	0.5729
Wine Quality White	C	4900\12	0.4976	0.5046	0.4855	0.5023	0.5111	0.5153	**0.5235**	0.5376	0.5259

Experiments

Transferability

- Pre-training approach can be effective in improving scores and finding more appropriate feature engineering actions faster.
- Pre-training on data-driven FETCH can accumulate and transfer prior knowledge to unobserved datasets and improve FE efficacy more effectively.

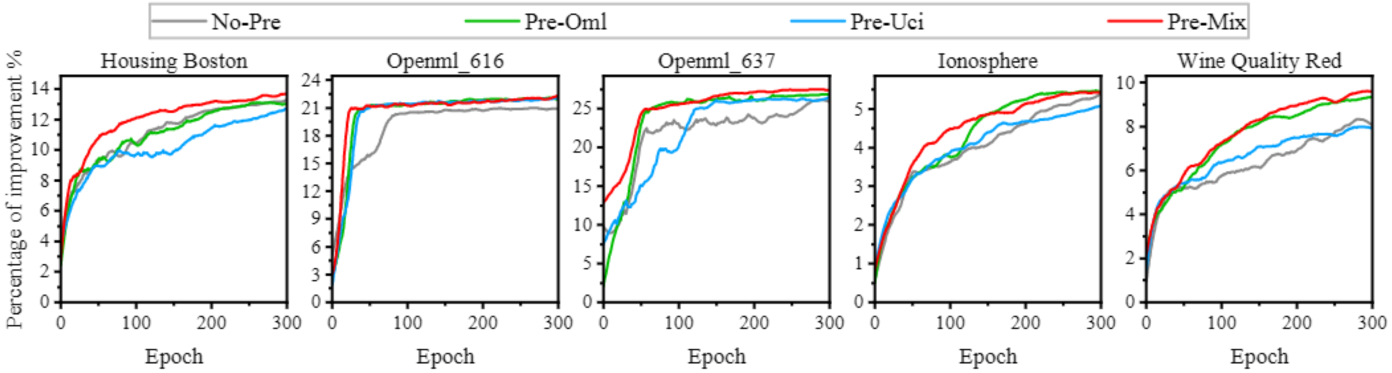


Figure 4: Transferability comparison of improvement (%) on 5 datasets under the model from scratch (No-Pre) and 3 kinds of pre-training models (Pre-Oml, Pre-Uci, and large-scale Pre-Mix).

Pre-trained Datasets	5 from OpenML	5 from UCI repo.
No-Pre		
Pre-Oml	✓	
Pre-Uci		✓
Pre-Mix	✓	✓

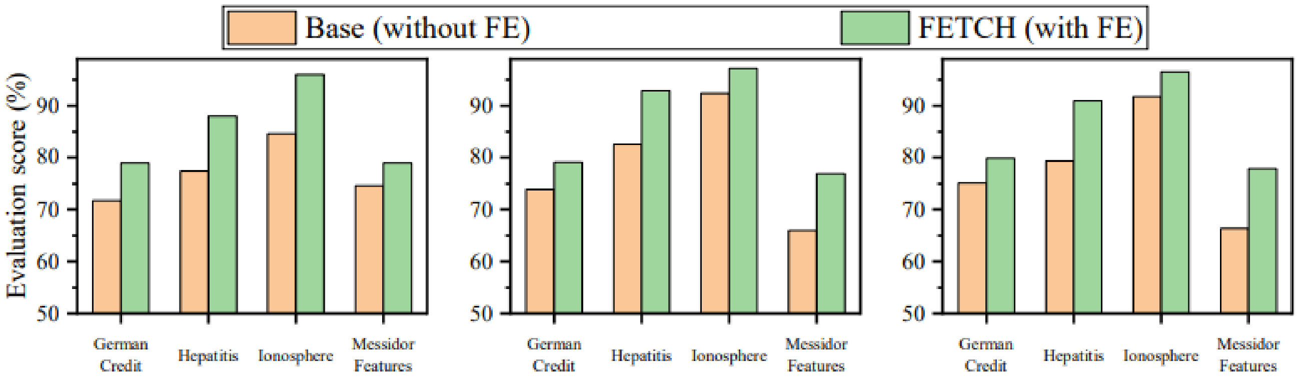
Table 2: Transferability comparison of the best pre-trained model (Pre-Best) and non-pre-trained one (No-Pre). See text for details.

Dataset	No-Pre	Pre-Best	Max-Diff (%)	$\text{Epoch}_{Max-Diff}$
Housing Boston	0.5224	**0.5357**	2.31	123
Openml_616	0.6749	**0.6942**	4.67	27
Openml_637	0.6453	**0.6631**	5.51	26
Ionosphere	0.9716	**0.9864**	1.12	148
Wine Quality Red	0.6042	**0.6207**	2.59	152

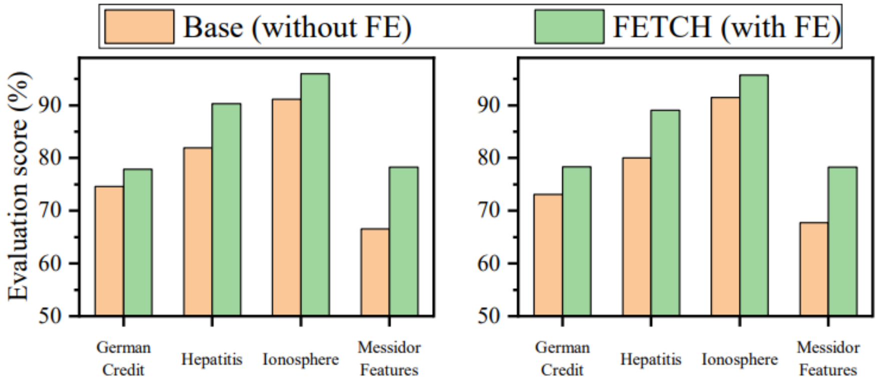
Experiments

Flexibility toward Model Choices

➢ FETCH has the effect of promoting the fitting performance for all these ML models.



(a) Logistic Regression (b) Random Forest (c) Xgboost



(a) LightGBM (b) Catboost

Summary

- We propose a <u>data-driven</u> AutoFE framework for both classification and regression tasks, dubbed *FETCH*. It learns how to map raw data to suitable FE actions sequence. Empirical results show its on-par or superior performances to the previous state-of-the-art methods.

- For the first time, we characterize a <u>transferability</u> principle for AutoFE. It reflects how much knowledge or experience a trained policy may be able to accumulate to enable the exploration of unseen datasets, which is also linked to the across-datasets *pre-training* paradigm.