PiML Training - Session 1

AI/ML Model Interpretability

Aijun Zhang, Ph.D.
Corporate Model Risk, Wells Fargo

Information Sharing at TD Bank – AI/ML Practice Forum | February 2, 2024

Disclaimer: This material represents the views of the presenter and does not necessarily reflect those of Wells Fargo.

Biographical Sketch

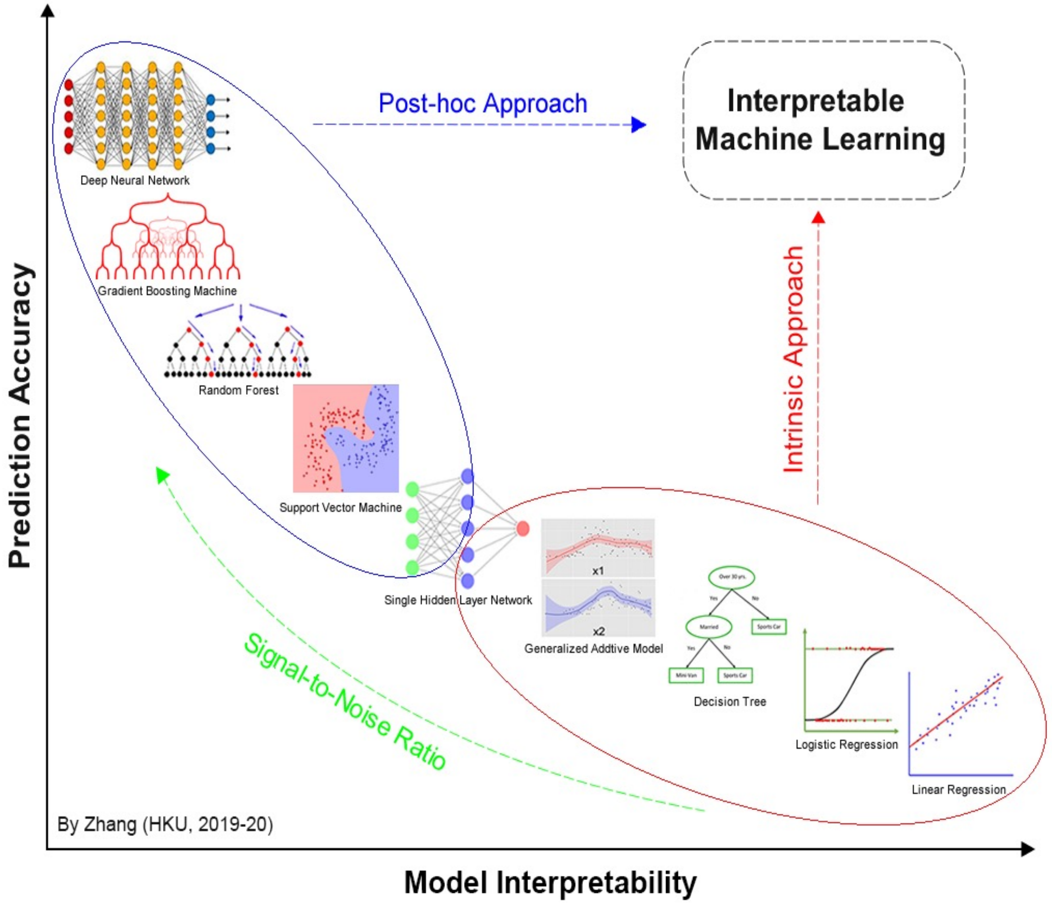
Aijun Zhang
SVP - Machine Learning &
Validation Engineering
Wells Fargo

- Aijun Zhang is a senior vice president, Head of Validation Engineering at Wells Fargo. He leads a machine learning & validation engineering team in Corporate Model Risk, responsible for PiML (Python interpretable machine learning) toolbox and VoD (Validation-on-Demand) platform.
- Aijun holds PhD degree in Statistics from University of Michigan at Ann Arbor, and he has 10+ years of experience working in financial risk management. Aijun was a former professor of statistics at University of Hong Kong. He has published ~40 papers in professional conferences and journals, with research topics in interpretable machine learning, data science and statistics.

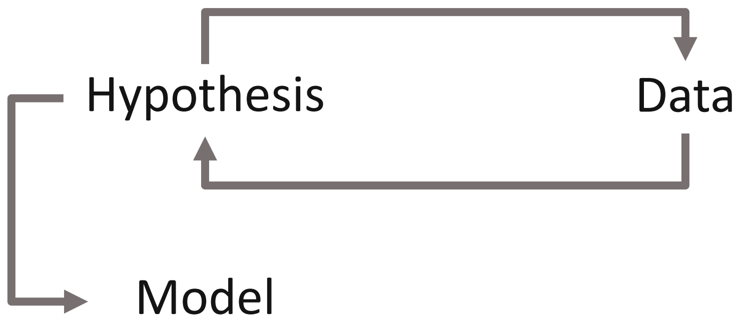
Outline

- **Introduction**
  - Interpretable machine learning
  - PiML toolbox
- **Machine Learning Interpretability**
  - Post-hoc explainability pitfalls
  - Inherent interpretability
  - FANOVA modeling framework
  - GAMI-Net and Interpretation
- **PiML User Guide and Examples**

Interpretable Machine Learning



Breiman (2001). Statistical modeling: The two cultures. *Statistical Science*.
Gunning (2017). Explainable Artificial Intelligence (XAI). *US DARPA Report.*

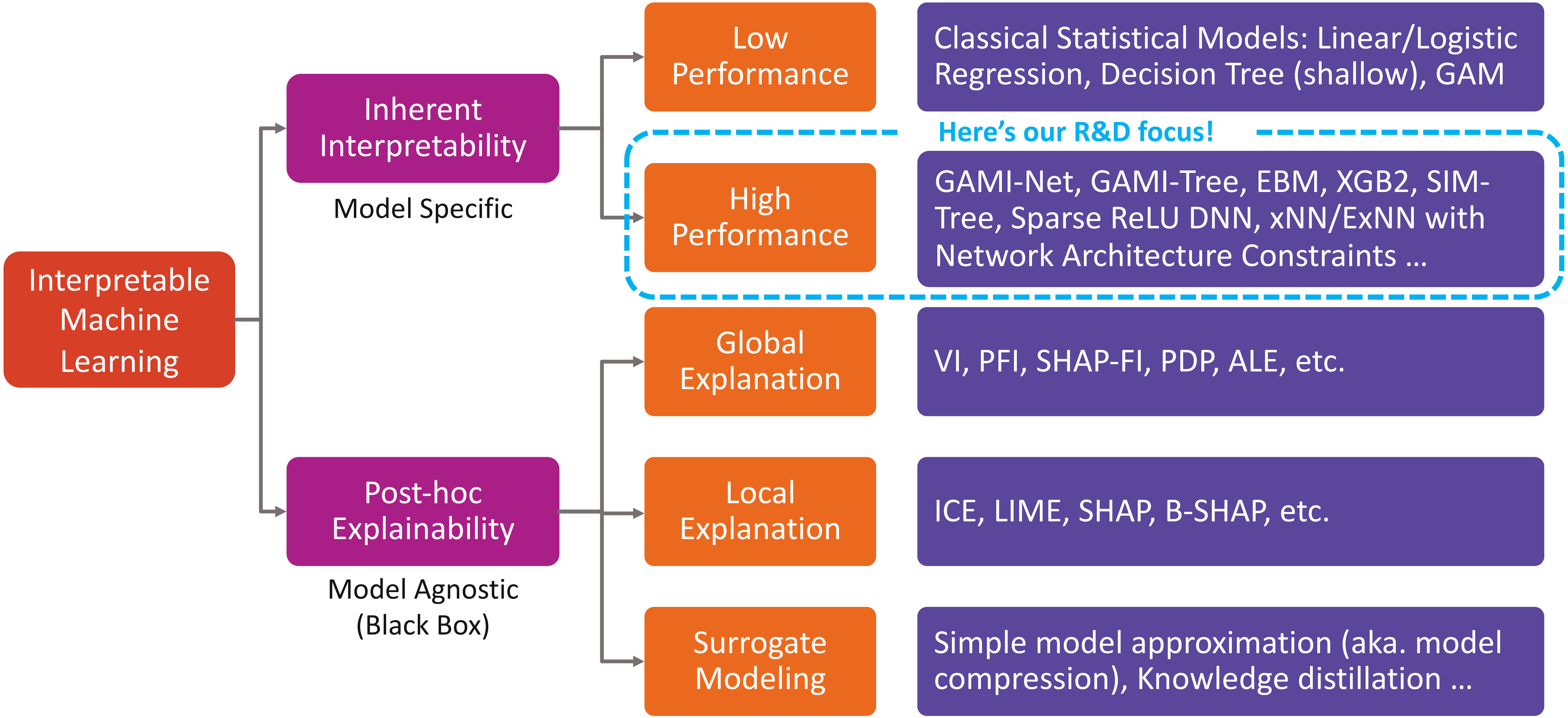


Last 20 years: modeling culture shift from data hypothesis to algorithmic prediction.

Models are increasingly black box.

Data → Model

Interpretable Machine Learning: A Taxonomy



Interpretable Machine Learning

Inherent Interpretability

Model Specific

Post-hoc Explainability

Model Agnostic (Black Box)

Low Performance

Classical Statistical Models: Linear/Logistic Regression, Decision Tree (shallow), GAM

Here's our R&D focus!

High Performance

GAMI-Net, GAMI-Tree, EBM, XGB2, SIM-Tree, Sparse ReLU DNN, xNN/ExNN with Network Architecture Constraints …

Global Explanation

VI, PFI, SHAP-FI, PDP, ALE, etc.

Local Explanation

ICE, LIME, SHAP, B-SHAP, etc.

Surrogate Modeling

Simple model approximation (aka. model compression), Knowledge distillation …

PiML Toolbox Overview

PiML

An integrated Python toolbox for interpretable machine learning

- **PiML** (read π-ML) is a Python package for interpretable machine learning model development and testing.
- **Installation:** pip install piml (180K+ PyPI downloads)
- **PiML Github repo**: https://github.com/SelfExplainML/PiML-Toolbox
- **PiML User Guide** with lots of examples: https://selfexplainml.github.io/PiML-Toolbox/
- **PiML Tutorials in Medium** (recently launched): https://piml.medium.com/

📢 **May 4, 2022:** V0.1.0 is launched with low-code UI/UX.

🚀 **June 26, 2022:** V0.2.0 is released with high-code APIs.

🚀 **July 26, 2022:** V0.3.0 is released with classic statistical models.

🚀 **October 31, 2022:** V0.4.0 is released with enriched models and enhanced diagnostics.

🚀 **May 4, 2023:** V0.5.0 is released together with PiML user guide.

🎄 **December 1, 2023:** V0.6.0 is released with enhanced data handling and model analytics.

PiML Toolbox Overview

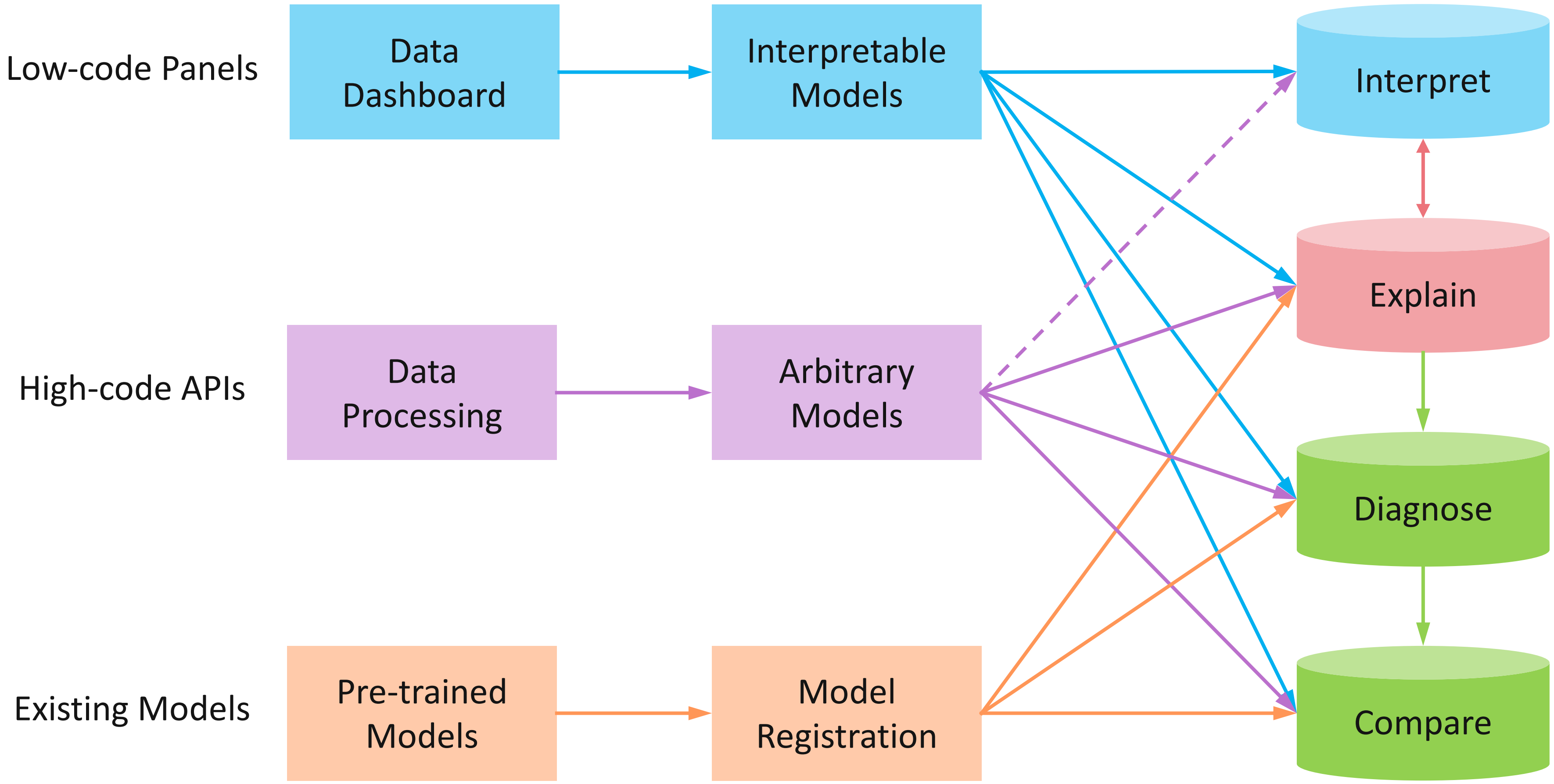
Model Development

- Data Exploration and Quality Check
- Inherently Interpretable ML Models
  - GLM, GAM, XGB1
  - XGB2, EBM, GAMI-Net, GAMI-Lin-Tree
- Locally Interpretable ML Models
  - Tree, Sparse ReLU Neural Networks
- Model-specific Interpretability
- Model-agnostic Explainability

Model Testing

- Model Diagnostics and Outcome Testing
  - Predictive Accuracy
  - Hyperparameter Turning
  - Weakness Detection
  - Reliability Test (Prediction Uncertainty)
  - Robustness Test
  - Resilience Test
  - Bias and Fairness
- Model Comparison and Benchmarking

PiML Pipelines



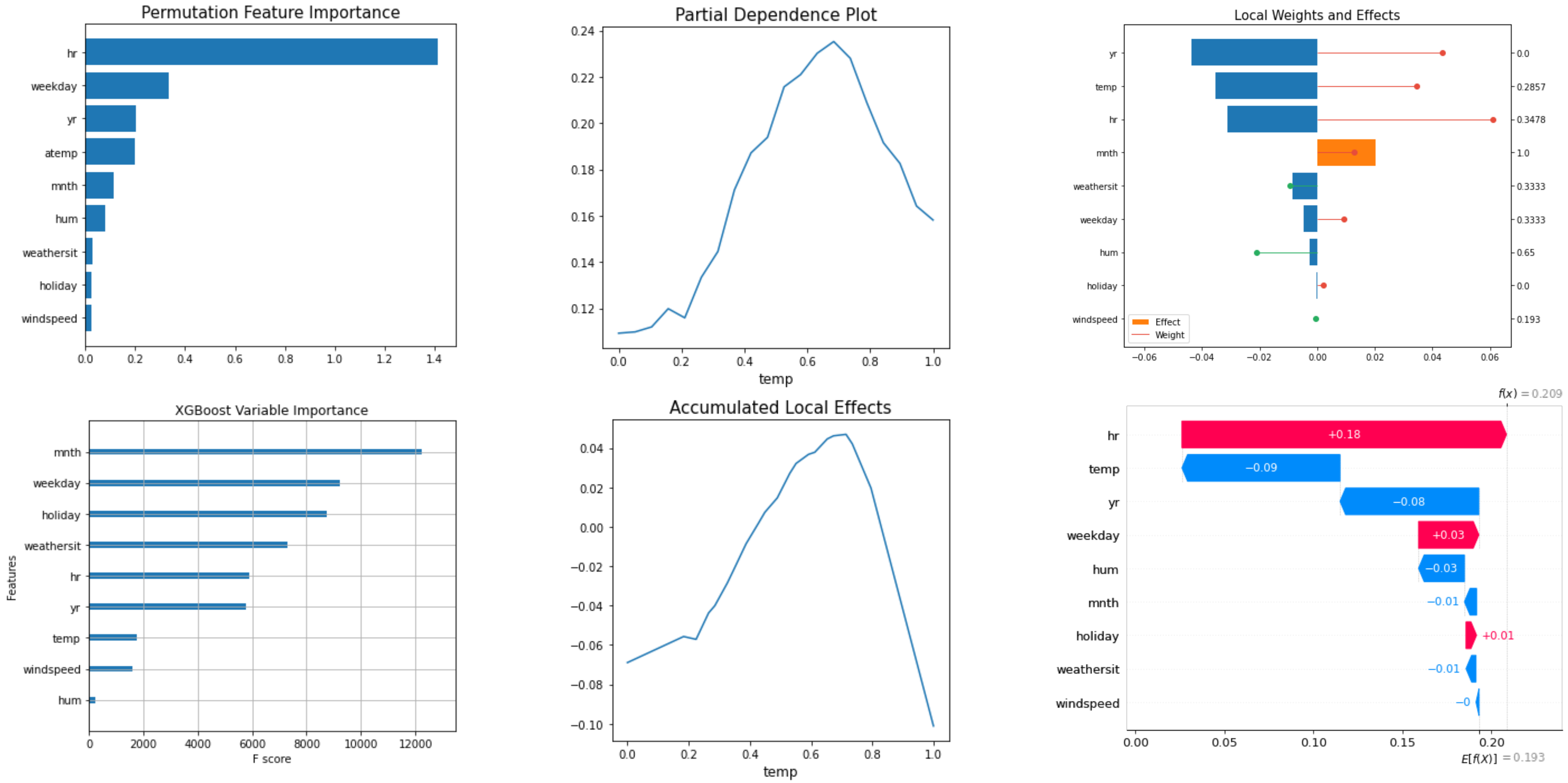
Outline

- **Introduction**
  - Interpretable machine learning
  - PiML toolbox
- **Machine Learning Interpretability**
  - Post-hoc explainability pitfalls
  - Inherent interpretability
  - FANOVA modeling framework
  - GAMI-Net and Interpretation
- **PiML User Guide and Examples**

Post-hoc Explainability Test

- **Post-hoc explainability test** is model-agnostic, i.e., it works for any pre-trained model.
  - Useful for explaining black-box models; but need to use with caution (there is no free lunch).
  - Post-hoc explainability tools sometimes have pitfalls, challenges and potential risks.
- **Local explainability tools** for explaining an individual prediction
  - ICE (Individual Conditional Expectation) plot
  - LIME (Local Interpretable Model-agnostic Explanations)
  - SHAP (SHapley Additive exPlanations)
- **Global explainability tools** for explaining the overall impact of features on model predictions
  - Examine relative importance of variables: **VI** (Variable Importance), **PFI** (Permutation Feature Importance), **SHAP-FI** (SHAP Feature Importance), **H-statistic** (Importance of two-factor interactions), etc.
  - Understand input-output relationships: 1D and 2D **PDP** (Partial Dependence Plot) and **ALE** (Accumulated Local Effects).

Post-hoc Explainability Pitfalls



PiML Demo: BikeSharing data fit by `XGBRegressor(max_depth=7, n_estimators=500)`

Post-hoc Explainability vs. Inherent Interpretability

- **Post-hoc explainability** is model agnostic, but there is no free lunch. According to Cynthia Rudin, use of auxiliary post-hoc explainers creates "double trouble" for black-box models.
- Various post-hoc explanation methods, including VI/FI, PDP, ALE, … (for global explainability) and LIME, SHAP, … (for local explainability), often produce results with disagreements.
- Lots of academic discussions about pitfalls, challenges and potential risks of using post-hoc explainers.
- This echoes CFPB Circular 2022-03 (May 26, 2022): Adverse action notification requirements in connection with credit decisions based on complex algorithms[1].

- **Inherent interpretability** is intrinsic to a model. It facilitates gist and intuitiveness for human insightful interpretation. It is important for evaluating a model's conceptual soundness.
- Model interpretability is a loosely defined concept and can be hardly quantified. Sudjianto and Zhang (2021)[2] proposed a qualitative rating assessment framework for ML model interpretability.
- Interpretable model design: a) interpretable feature selection and b) interpretable architecture constraints[3] such as additivity, sparsity, linearity, smoothness, monotonicity, visualizability, projection orthogonality, and segmentation degree.

[1] CFPB Circular 2022-03 Footnote 1: While some creditors may rely upon various post-hoc explanation methods, such explanations approximate models and creditors must still be able to validate the accuracy of those approximations, which may not be possible with less interpretable models. consumerfinance.gov

[2] Sudjianto and Zhang (2021): Designing Inherently Interpretable Machine Learning Models. arXiv: 2111.01743

[3] Yang, Zhang and Sudjianto (2021, IEEE TNNLS): Enhancing Explainability of Neural Networks through Architecture Constraints. arXiv: 1901.03838

Designing Inherently Interpretable Models

Model Characteristics	Gist for Interpretation
Additivity	Additive decomposition of feature effects tends to be more interpretable
Sparsity	Having fewer features or components tends to be more interpretable
Linearity	Linear or constant feature effects are easy to interpret
Smoothness	Continuous and smooth feature effects are relatively easy to interpret
Monotonicity	Sometimes increasing/decreasing effects are desired by expert knowledge
Visualizability	Direct visualization of feature effects facilitates diagnostics and interpretation
Projection	Sparse and near-orthogonal projection tends to be more interpretable
Segmentation	Having smaller number of segments (heterogeneous data) is more interpretable

[2] Sudjianto and Zhang (2021): Designing Inherently Interpretable Machine Learning Models. arXiv: 2111.01743

[3] Yang, Zhang and Sudjianto (2021, IEEE TNNLS): Enhancing Explainability of Neural Networks through Architecture Constraints. arXiv: 1901.03838

Inherently Interpretable FANOVA Models

- One effective way is to design inherently interpretable models by the functional ANOVA representation

$$g\big(\mathbb{E}(y|\boldsymbol{x})\big) = g_0 + \sum_{j} g_j(x_j) + \sum_{j<k} g_{jk}(x_j, x_k) + \sum_{j<k<l} g_{jkl}(x_j, x_k, x_l) + \cdots$$

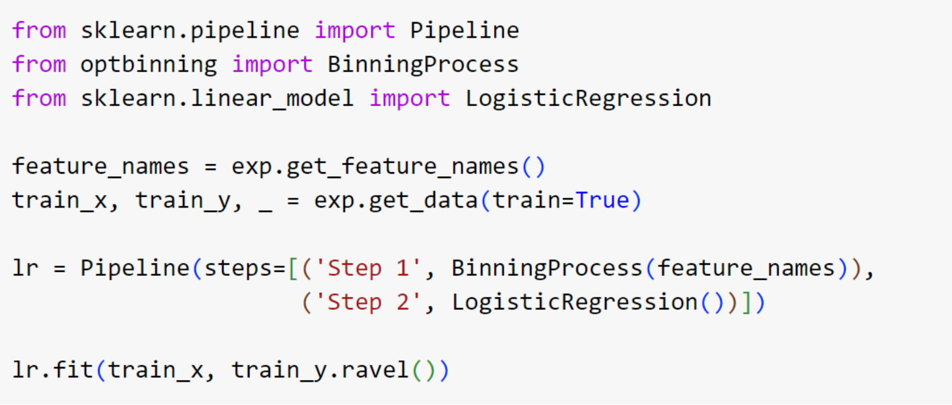
  It additively decomposes into the overall mean (i.e., intercept) g_0, main effects $g_j(x_j)$, two-factor interactions $g_{jk}(x_j, x_k)$, and higher-order interactions …

- GAM main-effect models: Binning Logistic, XGB1, GAM (estimated using Splines, etc.)

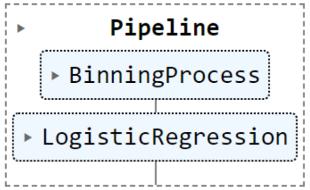
- GAMI main-effect plus two-factor-interaction models:
  - **EBM** (Nori, et al. 2019) → explainable boosting machine with shallow trees (piecewise constant)
  - **XGB2** (Lengerich, et al. 2020) → boosted trees of depth 2 with effect purification (piecewise constant)
  - **GAMI-Net** (Yang, Zhang and Sudjianto, 2021) → specialized neural nets (piecewise linear and continuous)
  - **GAMI-Lin-Tree** (Hu, Chen and Nair, 2022) → specialized boosted linear model-based trees (piecewise linear)

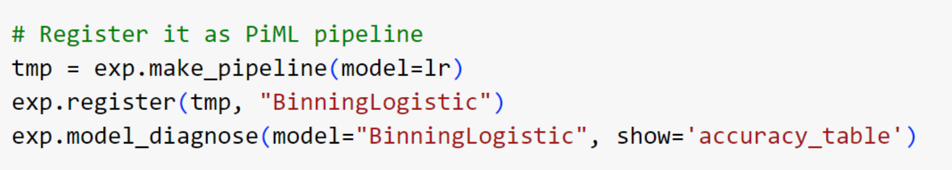
- **PiML Toolbox** integrates FANOVA-interpretable models and provides each model's inherent interpretability.

Binning Logistic vs. XGB1

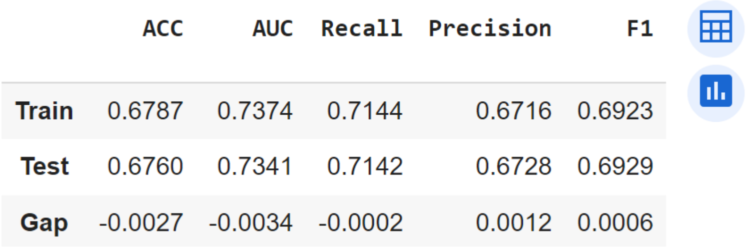


```python
from sklearn.pipeline import Pipeline
from optbinning import BinningProcess
from sklearn.linear_model import LogisticRegression

feature_names = exp.get_feature_names()
train_x, train_y, _ = exp.get_data(train=True)

lr = Pipeline(steps=[('Step 1', BinningProcess(feature_names)),
                     ('Step 2', LogisticRegression())])

lr.fit(train_x, train_y.ravel())
```

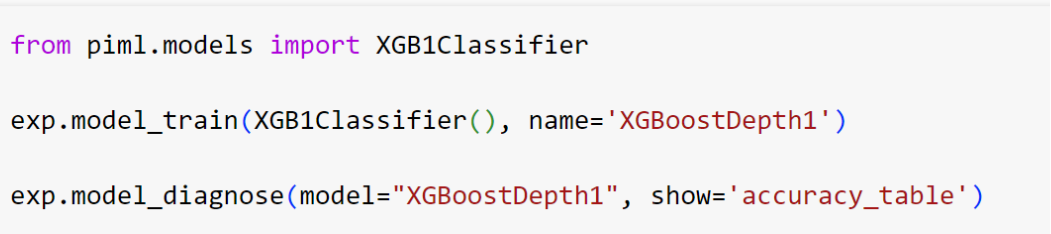




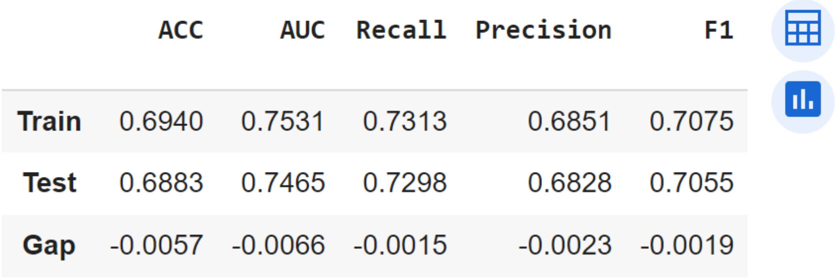
```python
# Register it as PiML pipeline
tmp = exp.make_pipeline(model=lr)
exp.register(tmp, "BinningLogistic")
exp.model_diagnose(model="BinningLogistic", show='accuracy_table')
```



	ACC	AUC	Recall	Precision	F1
Train	0.6787	0.7374	0.7144	0.6716	0.6923
Test	0.6760	0.7341	0.7142	0.6728	0.6929
Gap	-0.0027	-0.0034	-0.0002	0.0012	0.0006



```python
from piml.models import XGB1Classifier

exp.model_train(XGB1Classifier(), name='XGBoostDepth1')

exp.model_diagnose(model="XGBoostDepth1", show='accuracy_table')
```



	ACC	AUC	Recall	Precision	F1
Train	0.6940	0.7531	0.7313	0.6851	0.7075
Test	0.6883	0.7465	0.7298	0.6828	0.7055
Gap	-0.0057	-0.0066	-0.0015	-0.0023	-0.0019

- Binning Logistic is a GAM main effect model with piecewise constant basis functions (feature engineering). It performs manual binning one variable at a time.
- XGB1 is also a GAM main effect model of the same type. It performs automated binning jointly for all variables.
- Both GAM models are inherently interpretable, easy to quantify feature importance and draw main effect plots.

XGB1, XGB2 and Beyond

- **Proposition:** A depth-K tree-ensemble can be reformulated to an FANOVA model with main effects and k-way interactions with $k \leq K$.

- Examples: XGB1 is GAM with main effects; XGB2 is GAMI with main effects plus two-factor interactions.

- Three-step unwrapping technique for tree ensembles (e.g., RF, GBDT, XGBoost, LightGBM, CatBoost):

  1. **Aggregation:** all leaf nodes with the same set of k distinct split variables sum up to a raw k-way interaction.
  2. **Purification:** recursively cascade effects from high-order interactions to lower-order ones to obtain a unique FANOVA representation subject to hierarchical orthogonality constraints (Lengerich, et al., 2020).
  3. **Attribution:** quantify the importance of purified effects either locally (for a sample) or globally (for a dataset).

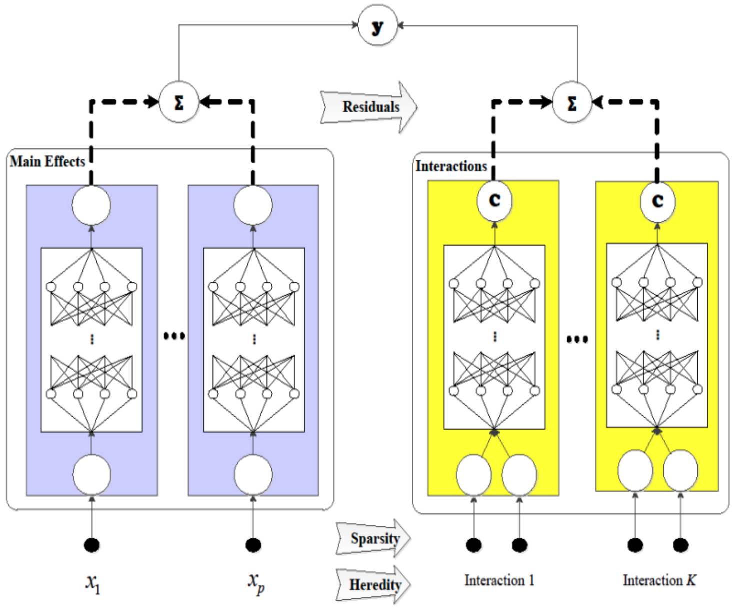
- Strategies to enhance model (e.g., XGBoost) interpretability without sacrificing model performance

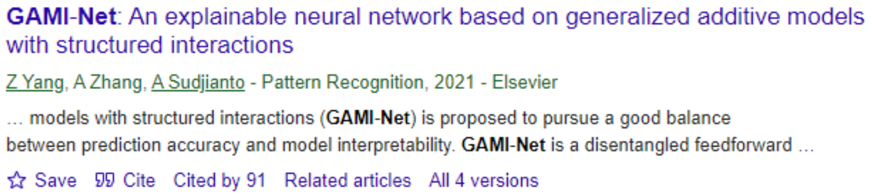
  - XGB hyperparameters: max_tree_depth, max_bins, candidate interactions, monotonicity, L1/L2 regularization, etc.
  - Pruning of purified effects: effect selection by L1 regularization, forward and backward selection with early stopping
  - Other strategies such as post-hoc smoothing of purified effects, local flattening, and boundary effect adjustment.

GAMI-Net and Interpretability Constraints

- **GAMI-Net** (Yang, Zhang and Sudjianto, 2021) is an FANOVA-interpretable model using neural networks.
- **Three-stage training algorithm:**
  - Stage 1: train the main effect subnetworks and **prune** the trivial ones by validation performance.
  - Stage 2: train pairwise interactions on residuals, by
    - Select candidate interactions by heredity constraint;
    - Evaluate their scores (by FAST) and select top-K interactions;
    - Train the selected two-way interaction subnetworks;
    - Prune trivial interactions by validation performance.
  - Stage 3: retrain main effects and interactions simultaneously for fine-tuning network parameter.

$$g(E(y|\boldsymbol{x})) = \mu + \sum h_j(x_j) + \sum f_{jk}(x_j, x_k)$$




GAMI-Net: An explainable neural network based on generalized additive models with structured interactions

Z Yang, A Zhang, A Sudjianto - Pattern Recognition, 2021 - Elsevier

… models with structured interactions (**GAMI-Net**) is proposed to pursue a good balance between prediction accuracy and model interpretability. **GAMI-Net** is a disentangled feedforward …

☆ Save 99 Cite Cited by 91 Related articles All 4 versions

GAMI-Net and Interpretability Constraints

$$g(E(y|\boldsymbol{x})) = \mu + \sum h_j(x_j) + \sum f_{jk}(x_j, x_k)$$

GAMI-Net incorporates the following constraints inherently.

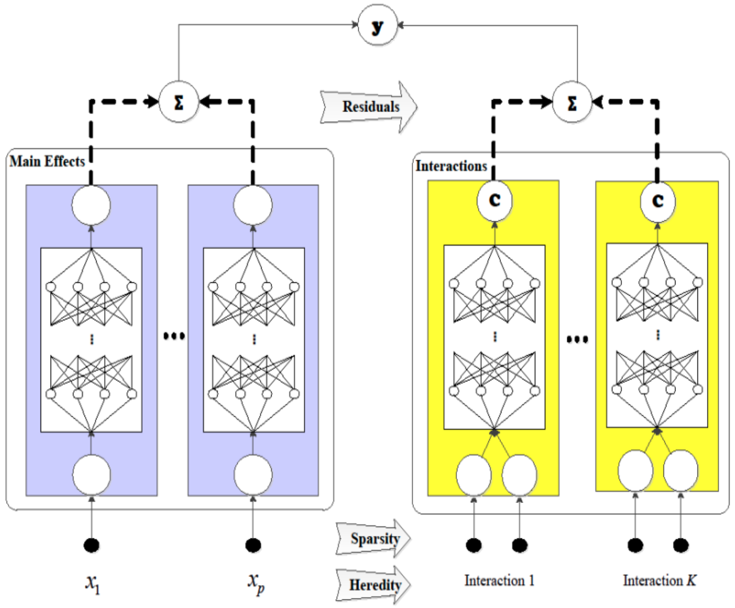
- **Sparsity**: select only the most important main effects and pairwise interactions.
- **Heredity**: a pairwise interaction is selected only if at least one (or both) of its parent main effects is selected.
- **Marginal Clarity**: enforce the pairwise interactions to be nearly orthogonal to the main effects, by imposing penalty

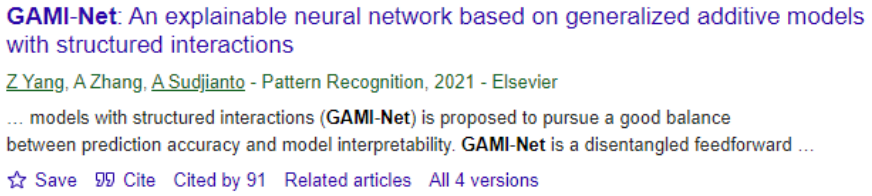
$$\Omega(h_j, f_{jk}) = \left| \frac{1}{n} \sum h_j(x_j) f_{jk}(x_j, x_k) \right|$$

- **Monotonicity**: certain features can be constrained to be monotonic increasing or decreasing, by imposing penalty

$$\Omega(x_j) = \max\left\{-\frac{\partial g}{\partial x_j}, 0\right\} \text{(if inceasing) or } \max\left\{\frac{\partial g}{\partial x_j}, 0\right\} \text{(if deceasing)}$$

PS: There is also a TensorFlow-lattice version of GAMI-Net with hard monotone constraints.





GAMI-Net: An explainable neural network based on generalized additive models with structured interactions

Z Yang, A Zhang, A Sudjianto - Pattern Recognition, 2021 - Elsevier

… models with structured interactions (**GAMI-Net**) is proposed to pursue a good balance between prediction accuracy and model interpretability. **GAMI-Net** is a disentangled feedforward …

Effect Importance and Feature Importance

- In GAMI-Net, each **effect importance** (before normalization) is given by

$$D(h_j) = \frac{1}{n-1}\sum_{i=1}^{n} h_j^2(x_{ij}), \qquad D(f_{jk}) = \frac{1}{n-1}\sum_{i=1}^{n} f_{jk}^2(x_{ij}, x_{ik})$$

- For prediction at $\boldsymbol{x}_i$, the **local feature importance** is given by

$$\phi_j(x_{ij}) = h_j(x_{ij}) + \frac{1}{2}\sum_{j \neq k} f_{jk}(x_{ij}, x_{ik})$$

- For GAMI-Net (or EBM), the **global feature importance** is given by

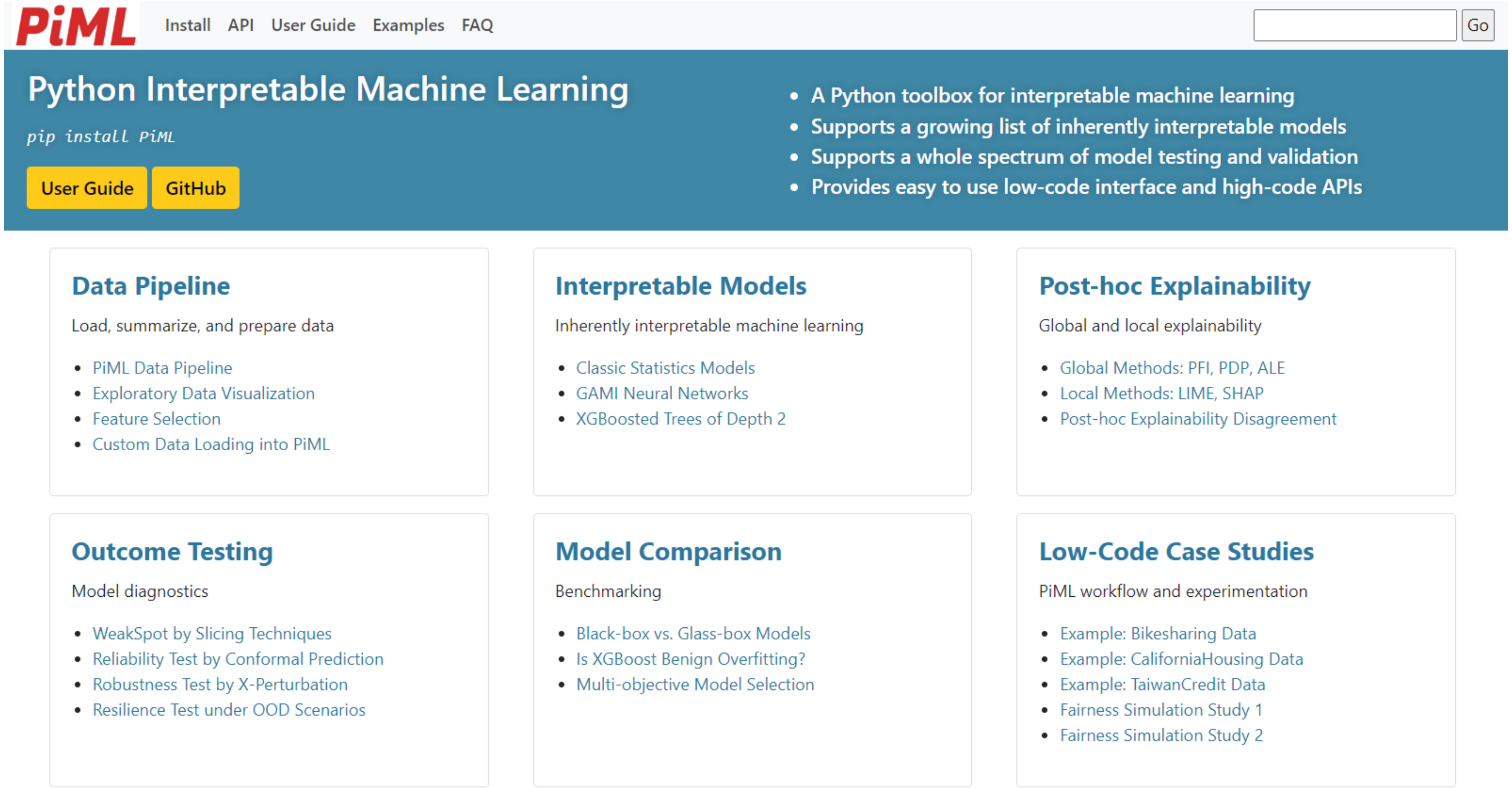
$$\mathrm{FI}(x_j) = \frac{1}{n-1}\sum_{i=1}^{n}\left(\phi_j(x_{ij}) - \overline{\phi_j}\right)^2$$

- The effect can be visualized by a line plot (for main effect) or heatmap (for pairwise interaction).

Outline

- **Introduction**
  - Interpretable machine learning
  - PiML toolbox
- **Machine Learning Interpretability**
  - Post-hoc explainability pitfalls
  - Inherent interpretability
  - FANOVA modeling framework
  - GAMI-Net and Interpretation
- **PiML User Guide and Examples**

PiML Docs and Examples



Python Interpretable Machine Learning

pip install PiML

- A Python toolbox for interpretable machine learning
- Supports a growing list of inherently interpretable models
- Supports a whole spectrum of model testing and validation
- Provides easy to use low-code interface and high-code APIs

Data Pipeline

Load, summarize, and prepare data

- PiML Data Pipeline
- Exploratory Data Visualization
- Feature Selection
- Custom Data Loading into PiML

Interpretable Models

Inherently interpretable machine learning

- Classic Statistics Models
- GAMI Neural Networks
- XGBoosted Trees of Depth 2

Post-hoc Explainability

Global and local explainability

- Global Methods: PFI, PDP, ALE
- Local Methods: LIME, SHAP
- Post-hoc Explainability Disagreement

Outcome Testing

Model diagnostics

- WeakSpot by Slicing Techniques
- Reliability Test by Conformal Prediction
- Robustness Test by X-Perturbation
- Resilience Test under OOD Scenarios

Model Comparison

Benchmarking

- Black-box vs. Glass-box Models
- Is XGBoost Benign Overfitting?
- Multi-objective Model Selection

Low-Code Case Studies

PiML workflow and experimentation

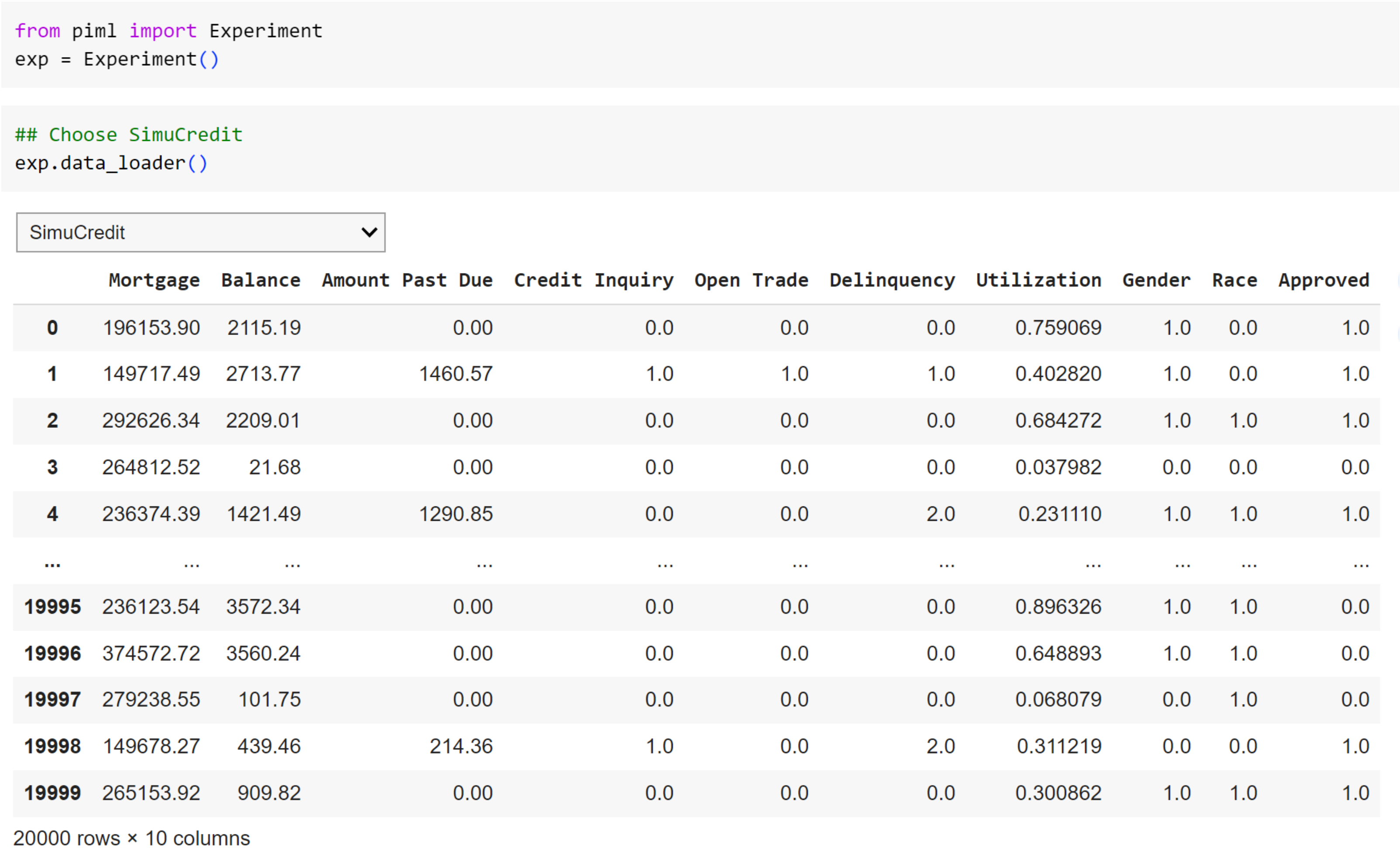
- Example: Bikesharing Data
- Example: CaliforniaHousing Data
- Example: TaiwanCredit Data
- Fairness Simulation Study 1
- Fairness Simulation Study 2

https://selfexplainml.github.io/PiML-Toolbox

SimuCredit Data from PiML

An educational synthetic credit decisioning dataset with

- **Credit features**
  - Mortgage size
  - Balance of credit account
  - Amount Past Due
  - # Credit Inquiry
  - # Open Trade
  - Delinquency status
  - Utilization rate
- **Demographic features**
  - Race
  - Gender
- **Binary Response**
  - 0/1 approved



```python
from piml import Experiment
exp = Experiment()
```

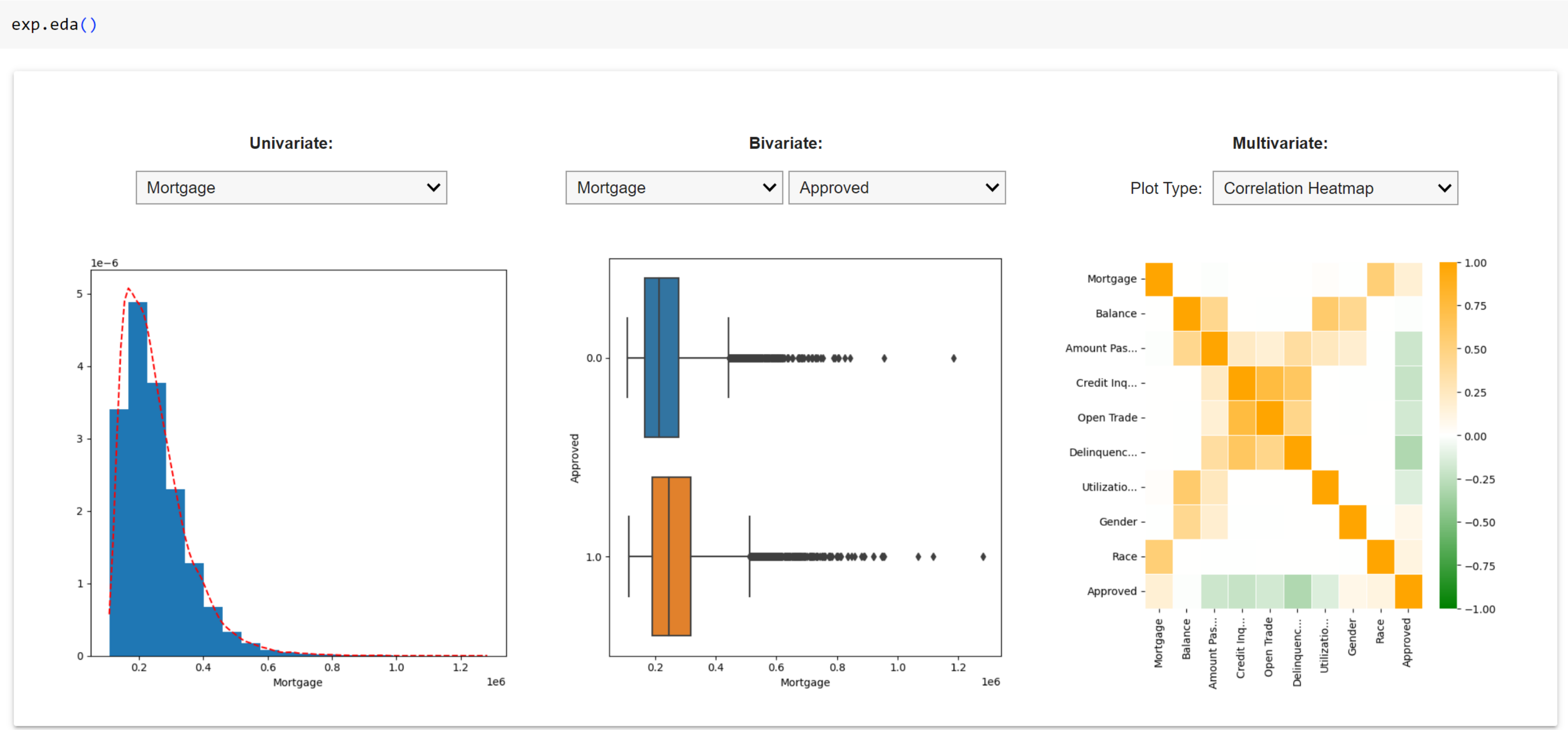
```python
## Choose SimuCredit
exp.data_loader()
```

SimuCredit ⌄

	Mortgage	Balance	Amount Past Due	Credit Inquiry	Open Trade	Delinquency	Utilization	Gender	Race	Approved
0	196153.90	2115.19	0.00	0.0	0.0	0.0	0.759069	1.0	0.0	1.0
1	149717.49	2713.77	1460.57	1.0	1.0	1.0	0.402820	1.0	0.0	1.0
2	292626.34	2209.01	0.00	0.0	0.0	0.0	0.684272	1.0	1.0	1.0
3	264812.52	21.68	0.00	0.0	0.0	0.0	0.037982	0.0	0.0	0.0
4	236374.39	1421.49	1290.85	0.0	0.0	2.0	0.231110	1.0	1.0	1.0
...	...	...	...	...	...	...	...	...	...	...
19995	236123.54	3572.34	0.00	0.0	0.0	0.0	0.896326	1.0	1.0	0.0
19996	374572.72	3560.24	0.00	0.0	0.0	0.0	0.648893	1.0	1.0	0.0
19997	279238.55	101.75	0.00	0.0	0.0	0.0	0.068079	0.0	1.0	0.0
19998	149678.27	439.46	214.36	1.0	0.0	2.0	0.311219	0.0	0.0	1.0
19999	265153.92	909.82	0.00	0.0	0.0	0.0	0.300862	1.0	1.0	1.0

20000 rows × 10 columns

SimuCredit Data Exploration by PiML



```
exp.eda()
```

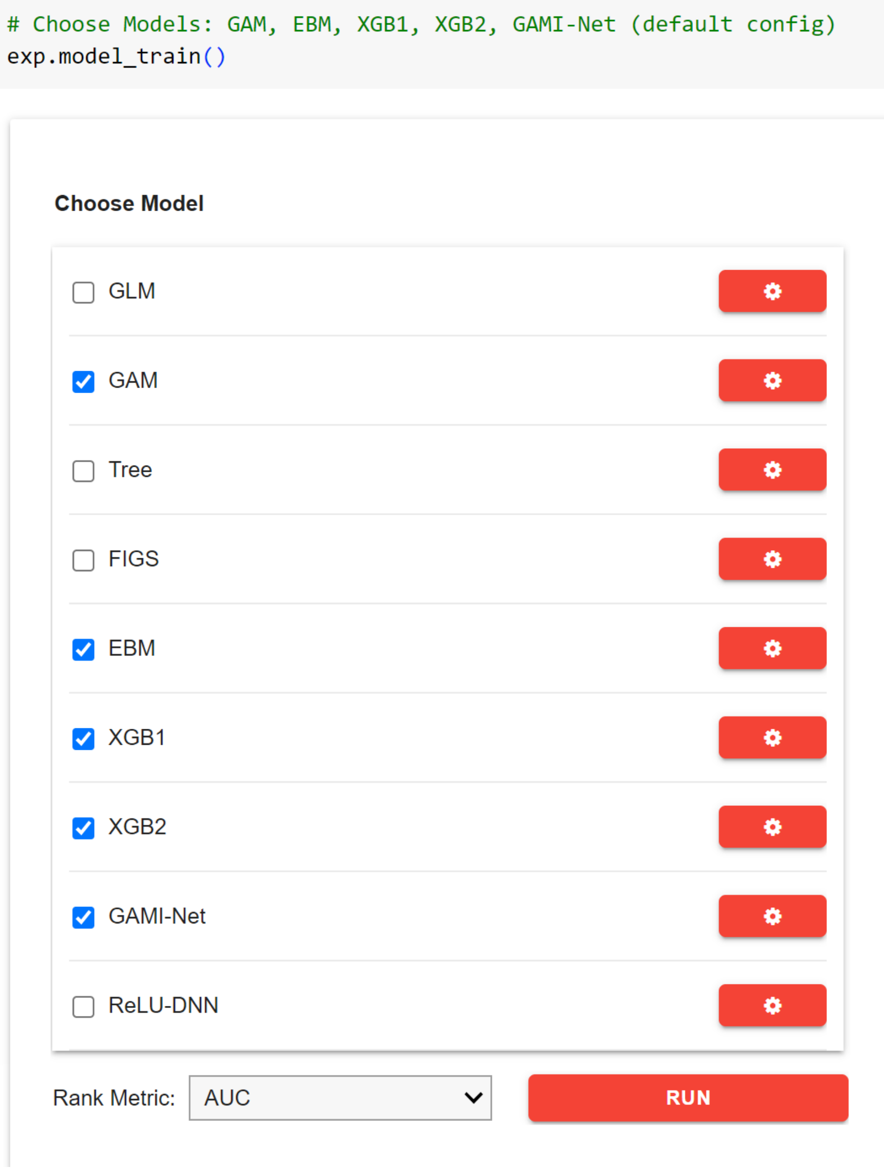
Univariate: Mortgage

Bivariate: Mortgage, Approved

Multivariate: Plot Type: Correlation Heatmap

- Prepare data by removal of “Gender” and “Race” and train-test split (various split methods) …

FANOVA Models: Performance Leaderboard



```
# Choose Models: GAM, EBM, XGB1, XGB2, GAMI-Net (default config)
exp.model_train()
```

Choose Model

□ GLM ⚙

☑ GAM ⚙

□ Tree ⚙

□ FIGS ⚙

☑ EBM ⚙

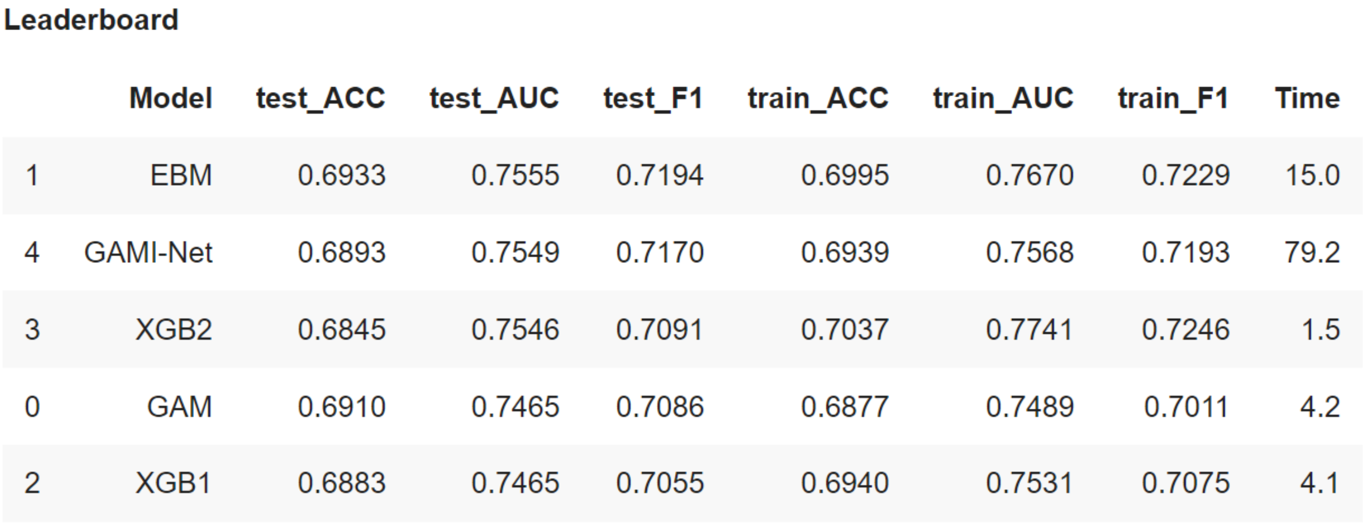
☑ XGB1 ⚙

☑ XGB2 ⚙

☑ GAMI-Net ⚙

□ ReLU-DNN ⚙

Rank Metric: AUC

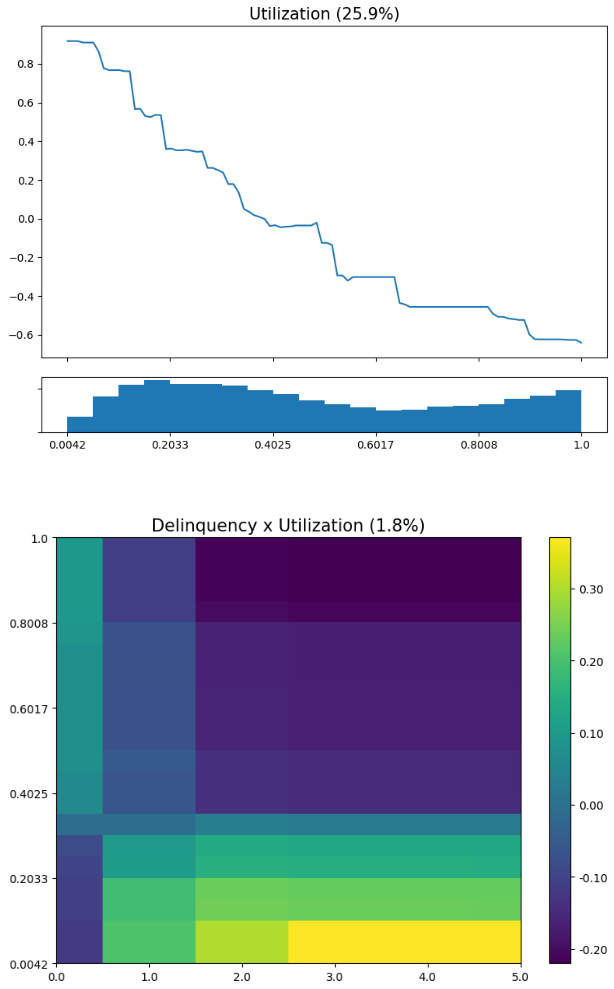


Leaderboard

	Model	test_ACC	test_AUC	test_F1	train_ACC	train_AUC	train_F1	Time
1	EBM	0.6933	0.7555	0.7194	0.6995	0.7670	0.7229	15.0
4	GAMI-Net	0.6893	0.7549	0.7170	0.6939	0.7568	0.7193	79.2
3	XGB2	0.6845	0.7546	0.7091	0.7037	0.7741	0.7246	1.5
0	GAM	0.6910	0.7465	0.7086	0.6877	0.7489	0.7011	4.2
2	XGB1	0.6883	0.7465	0.7055	0.6940	0.7531	0.7075	4.1

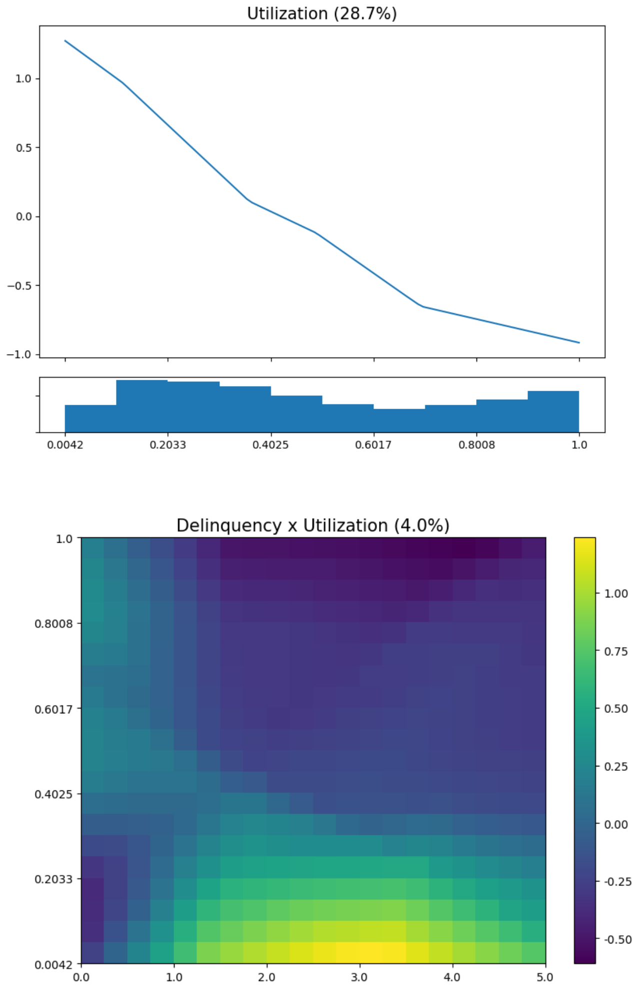
FANOVA Models: Model Interpretability

EBM


Utilization (25.9%)

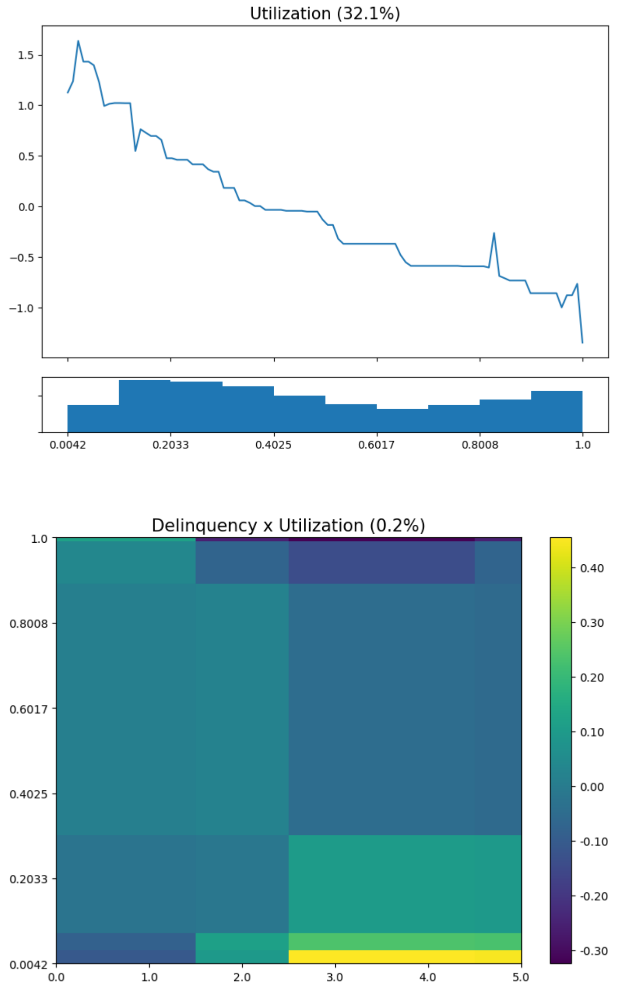
Delinquency x Utilization (1.8%)

GAMI-Net


Utilization (28.7%)

Delinquency x Utilization (4.0%)

XGB2


Utilization (32.1%)

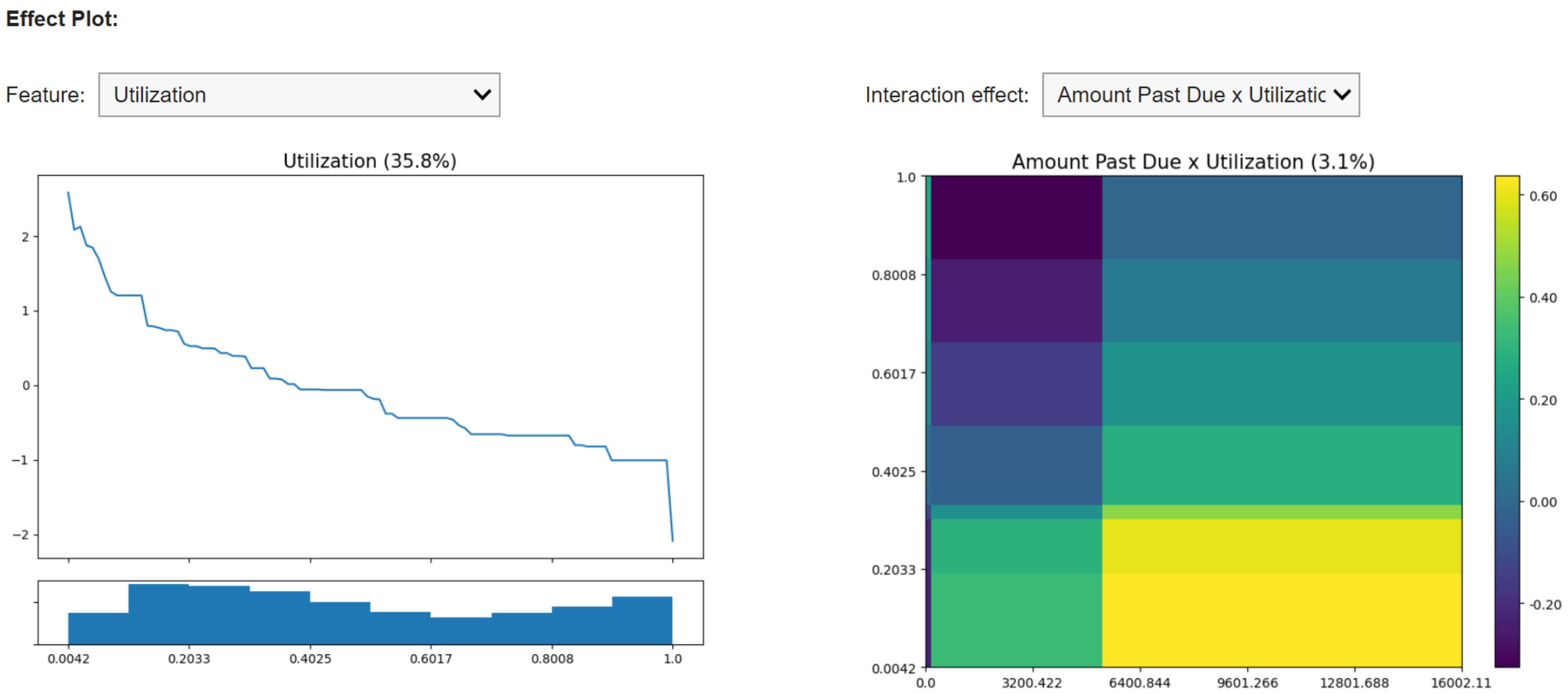
Delinquency x Utilization (0.2%)

Monotone Constraints

- Rerun “exp.model_train()” for XGB2 with monotone constraints:

```
# Increasing = "Mortgage", "Balance"]
# Decreasing = "Utilization", "Delinquency", "Credit Inquiry", "Open Trade", "Amount Past Due"
```

- Prediction performance may not sacrifice, while model interpretability gets enhanced.



Effect Plot:

Feature: Utilization

Interaction effect: Amount Past Due x Utilizatic

Thank you

Aijun Zhang, Ph.D.

Email: Aijun.Zhang@wellsfargo.com

LinkedIn: https://www.linkedin.com/in/ajzhang/