g-RIPS 2023 Fujitsu Group

Design and Development of Explainable AI with Wide Learning

Group Members:
Lily Ge (Northwestern University)
Yuki Kimura (Musashino University)
Takeshi Nakashima (Musashino University)
Molly Noel (Rensselaer Polytechnic Institute)

Academic Mentor:
Jesica Bauer (Rensselaer Polytechnic Institute)

Industry Mentor:
Dr. Hiroyuki Higuchi (Fujitsu Limited)

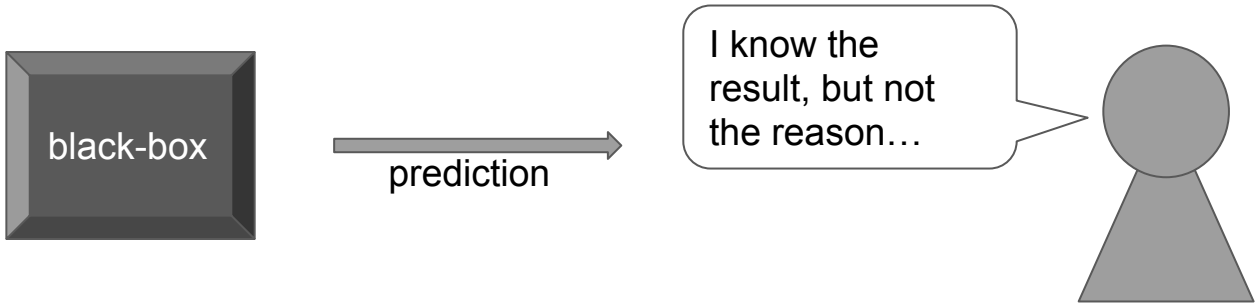
Outline

- Motivation
  - What is Explainable AI?
  - Explainability Approaches
- Wide Learning
- Our Approach
  - User selection LASSO
  - Model selection
    - Rademacher complexity
      - result
  - Interface Specification and Development
    - result
  - Pilot Survey for New Interface
    - result
- Conclusion

What is Explainable AI?

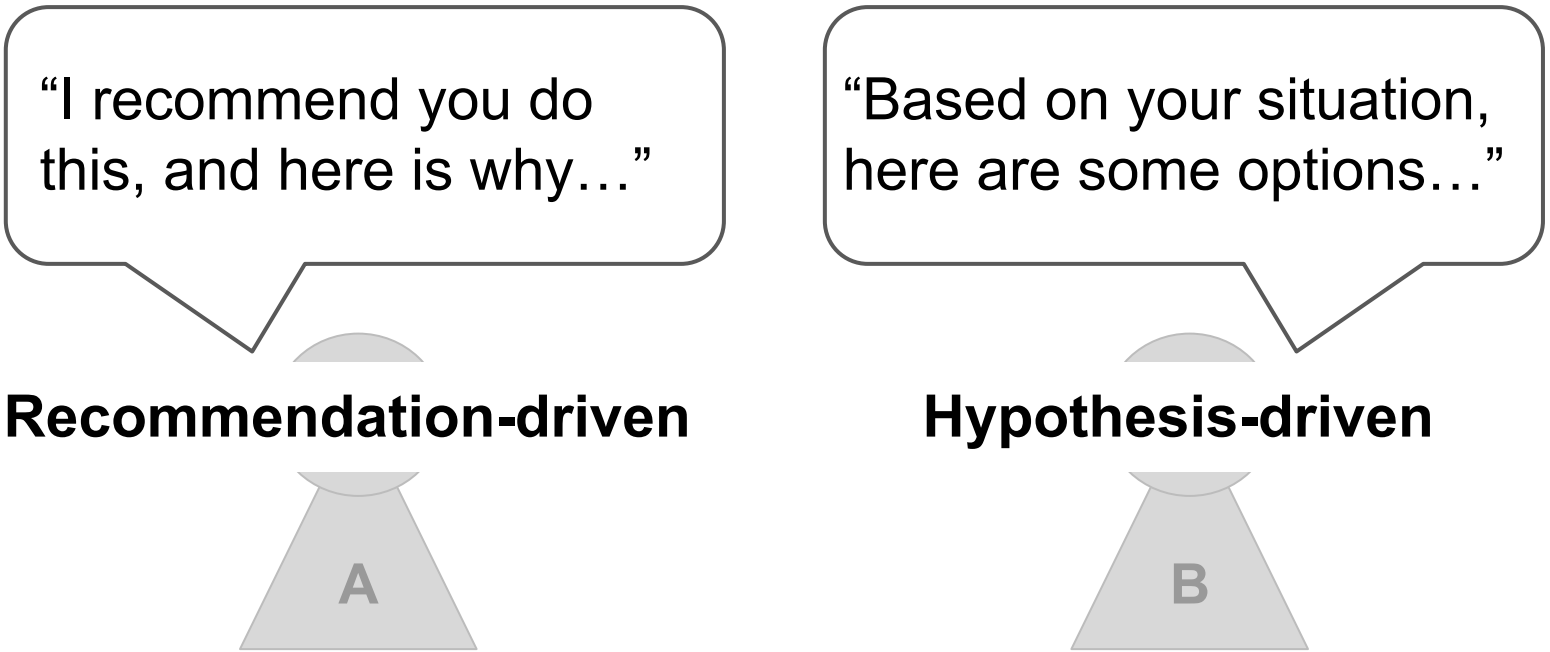
Modern Model

- Many modern AI predictions are based on black-box models/systems and lack explanatory power and conviction.
- This lack has a significant impact on industries such as finance and medicine, which are driven by trust.



Islam, Sheikh Rabiul, William Eberle, Sheikh Khaled Ghafoor, and Mohiuddin Ahmed. "Explainable artificial intelligence approaches: A survey." arXiv preprint arXiv:2101.09429 (2021).

Explanation Approaches



Miller, Tim. "Explainable ai is dead, long live explainable ai! hypothesis-driven decision support." arXiv preprint arXiv:2302.12389 (2023).

“Hello, Wide Learning!” by Fujitsu Limited

- Wide Learning is a form of the hypothesis-driven approach
- We aim to enhance it and develop a more hypothesis-driven Explainable AI (XAI) tool

Combination of important items
Oviparous_No
Sized about the same as a cat?_Yes
Breathes using lungs_Yes ∧ Fins_Has
Wings_Does not have ∧ Eats meat_No ∧ Spine_Has
Teeth_Has ∧ Breathes using lungs_Yes
Flies_Does not ∧ Aquatic_No

How Can We Improve Wide Learning?

Pilot Survey

- Open-ended questions to elicit people's thoughts on the existing Wide Learning tool
- Anonymized and expected to take approximately 15 minutes to complete
- Distributed through the `#students-grips2023` Slack channel and received 10 responses
- Survey responses informed how we built the new user interface

Existing Interface of “Hello, Wide Learning!”

Training

1. Upload training data
2. Get important feature combinations generated by Wide Learning
3. Assign weights to the combinations

Testing

4. Upload testing data
5. Make predictions for testing data

Our focus

- Wide Learning efficiently identifies important feature combinations at Step 2
- Weights are assigned to these important combinations in Step 3 and are used to make predictions in Step 5
- We focus on making improvements to Steps 3-5, by addressing areas of confusion from the survey responses and incorporating hypothesis-driven learning

Improvement 1:
Feature Selection in LASSO

Logistic Lasso

- Logistic regression with L1 regularizer to assign weights to combinations from Step 2
- Lasso performs feature selection → some combinations are assigned weight 0
- Optimal weights are selected by solving the following problem:

$$\beta^\star = \arg\min_\beta \mathcal{L}(\beta) := f(\hat{y}, y) + \rho\|\beta\|_1$$

$$y_i \in \{0, 1\}, \text{ class label}$$

$$\hat{y}_i \in (0, 1), \text{ prediction based on } \beta$$

Robert Tibshirani.Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society: Series B (Methodological), 58(1):267–288, 1996.

User Input

- In Wide Learning and the other models considered, important feature combinations are selected by the model

- We include user selection of important combinations in our Lasso model to increase trustworthiness based on this paper:

  Satoshi Hara and Takanori Maehara. Finding alternate features in lasso. arXiv preprint arXiv:1611.05940, 2016.

User Input

- Lasso selects important feature combinations by assigning them a nonzero weight
- Feature combinations not included in the model are assigned weight zero
- User can select a feature combination with a zero weight to include in the model
- The optimal nonzero weight for this combination is found
- A feature combination included in the original model is assigned a zero weight to keep the same number of nonzero weights

Satoshi Hara and Takanori Maehara. Finding alternate features in lasso. arXiv preprint arXiv:1611.05940, 2016.

Finding Alternate Weights

Given the Lasso optimal solution β^*, we seek for whether there are any alternate feature x_j with $\beta_j^* = 0$ that can be replaced with a feature x_i selected by the Lasso (i.e., $\beta_i^* \neq 0$). We solve this problem by optimizing β_j in (1) while fixing as $\beta_i = 0$ and $\beta_k = \beta_k^* \, (k \neq i, j)$. The optimization problem can be expressed as

$$\beta_j^{(i)} = \underset{\beta_j}{\operatorname{argmin}} \, f(z^{(i)} + X_j \beta_j, y) + \rho |\beta_j|, \tag{2}$$

where X_j denotes the j-th column of X and $z^{(i)} = \sum_{k \neq i} X_k \beta_k^*$. If $\beta_j^{(i)} \neq 0$, the feature x_j can be an alternative of x_i. We note that the problem (2) is a univariate optimization problem, and can be solved easily, e.g., by using the proximal gradient method [5].

Satoshi Hara and Takanori Maehara. Finding alternate features in lasso. arXiv preprint arXiv:1611.05940, 2016.

User Selection Example

- Red betas are included in the model (nonzero), blue betas are not included in the model (zero)
- This example shows replacing feature combinations 2 and 3 with feature combination 1

$$\beta^{*} = \begin{bmatrix} \beta_1^{*} = 0 \\ \beta_2^{*} \neq 0 \\ \beta_3^{*} \neq 0 \\ \beta_4^{*} = 0 \end{bmatrix} \qquad \beta^{(2\to1)} = \begin{bmatrix} \beta_1^{(2\to1)} = \beta_1^{(2)} \\ \beta_2^{(2\to1)} = 0 \\ \beta_3^{(2\to1)} = \beta_3^{*} \\ \beta_4^{(2\to1)} = \beta_4^{*} \end{bmatrix} \qquad \beta^{(3\to1)} = \begin{bmatrix} \beta_1^{(3\to1)} = \beta_1^{(3)} \\ \beta_2^{(3\to1)} = \beta_2^{*} \\ \beta_3^{(3\to1)} = 0 \\ \beta_4^{(3\to1)} = \beta_4^{*} \end{bmatrix}$$

$$\mathcal{L}(\beta^{*}) \qquad \mathcal{L}(\beta^{(2\to1)}) \qquad \mathcal{L}(\beta^{(3\to1)})$$

Satoshi Hara and Takanori Maehara. Finding alternate features in lasso. arXiv preprint arXiv:1611.05940, 2016.

User Selection Example

- Example with four important feature combinations
- Optimal betas from Lasso

$$\boldsymbol{\beta}^* = \begin{bmatrix} \beta_1^* = 0 \\ \beta_2^* \neq 0 \\ \beta_3^* \neq 0 \\ \beta_4^* = 0 \end{bmatrix}$$

Satoshi Hara and Takanori Maehara. Finding alternate features in lasso. arXiv preprint arXiv:1611.05940, 2016.

User Selection Example

- β_2 and β_3 have nonzero weights, so they are included in the model

$$\boldsymbol{\beta}^* = \begin{bmatrix} \beta_1^* = 0 \\ \beta_2^* \neq 0 \\ \beta_3^* \neq 0 \\ \beta_4^* = 0 \end{bmatrix}$$

Satoshi Hara and Takanori Maehara. Finding alternate features in lasso. arXiv preprint arXiv:1611.05940, 2016.

User Selection Example

- β_1 and β_4 have weights of 0, so they are not included in the model

$$\boldsymbol{\beta}^* = \begin{bmatrix} \beta_1^* = 0 \\ \beta_2^* \neq 0 \\ \beta_3^* \neq 0 \\ \beta_4^* = 0 \end{bmatrix}$$

Satoshi Hara and Takanori Maehara. Finding alternate features in lasso. arXiv preprint arXiv:1611.05940, 2016.

User Selection Example

- Say the user wants to make sure β_1 is included in the model
- Need to find a new optimal nonzero β_1

$$\boldsymbol{\beta}^* = \begin{bmatrix} \beta_1^* = 0 \\ \beta_2^* \neq 0 \\ \beta_3^* \neq 0 \\ \beta_4^* = 0 \end{bmatrix}$$

Satoshi Hara and Takanori Maehara. Finding alternate features in lasso. arXiv preprint arXiv:1611.05940, 2016.

Option 1 - Replace β_1 with β_2

- Find new optimal value of β_1

$$\beta^{(2\to1)} = \begin{bmatrix} \beta_1^{(2)} \\ 0 \\ \beta_3^* \\ \beta_4^* \end{bmatrix}$$

Satoshi Hara and Takanori Maehara. Finding alternate features in lasso. arXiv preprint arXiv:1611.05940, 2016.

Option 1 - Replace β_1 with β_2

- Find new optimal value of β_1
- Set β_2 to 0

$$\beta^{(2\to1)} = \begin{bmatrix} \beta_1^{(2)} \\ 0 \\ \beta_3^* \\ \beta_4^* \end{bmatrix}$$

Satoshi Hara and Takanori Maehara. Finding alternate features in lasso. arXiv preprint arXiv:1611.05940, 2016.

Option 1 - Replace β_1 with β_2

- Find new optimal value of β_1
- Set β_2 to 0
- Keep β_3 and β_4 the same

$$\beta^{(2\to 1)} = \begin{bmatrix} \beta_1^{(2)} \\ 0 \\ \beta_3^* \\ \beta_4^* \end{bmatrix}$$

Satoshi Hara and Takanori Maehara. Finding alternate features in lasso. arXiv preprint arXiv:1611.05940, 2016.

Option 2 - Replace β_1 with β_3

- Find new optimal value of β_1
- Set β_3 to 0
- Keep β_2 and β_4 the same

$$\beta^{(3\to 1)} = \begin{bmatrix} \beta_1^{(3)} \\ \beta_2^* \\ 0 \\ \beta_4^* \end{bmatrix}$$

Satoshi Hara and Takanori Maehara. Finding alternate features in lasso. arXiv preprint arXiv:1611.05940, 2016.

Choosing the Best New Solution

- Compare the objective values of the new solutions to the original β^*

$$\beta^* = \arg\min_{\beta} \mathcal{L}(\beta) := f(\hat{y}, y) + \rho\|\beta\|_1$$

$$y_i \in \{0, 1\}, \text{ class label}$$

$$\hat{y}_i \in (0, 1), \text{ prediction based on } \beta$$

$\mathcal{L}(\beta^*)$ $\mathcal{L}(\beta^{(2\to1)})$ $\mathcal{L}(\beta^{(3\to1)})$

Improvement 2:
More Result Options

Additional Models to Select

We can improve the convincingness of our model results by providing other models to compare, such as:

- Logistic regression
- Perceptron
- Decision Trees
- Random Forest
- Support Vector Machines (SVM)
- Gaussian Naive Bayes

Reasons for selecting six models

・Hello Wide Learning targets binary classification problems.

・Six models are well-known methods in the field of classification.

・Six models can be executed using sklearn.

Logistic Regression

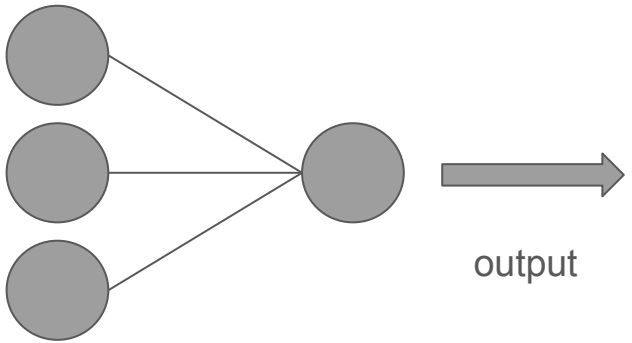
- A statistical model primarily used for binary classification
- it calculates using a linear prediction model
- The analysis is performed by passing the values through a sigmoid function

$$y = \sum_{i=1}^{n} w_i x_i + b \ (i = 1, 2, ..., n)$$: linear prediction model

$$\sigma(y) = \frac{1}{1 + \exp(-y)}$$: sigmoid function

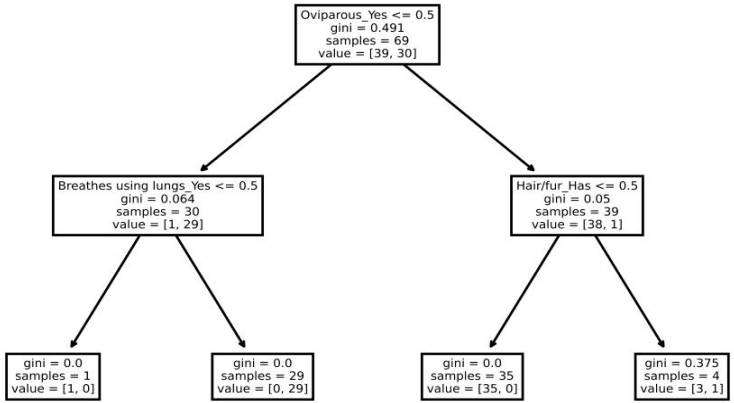
Perceptron

- A simple linear classification algorithm
- calculates the weighted sum of input data features and classifies them into two categories



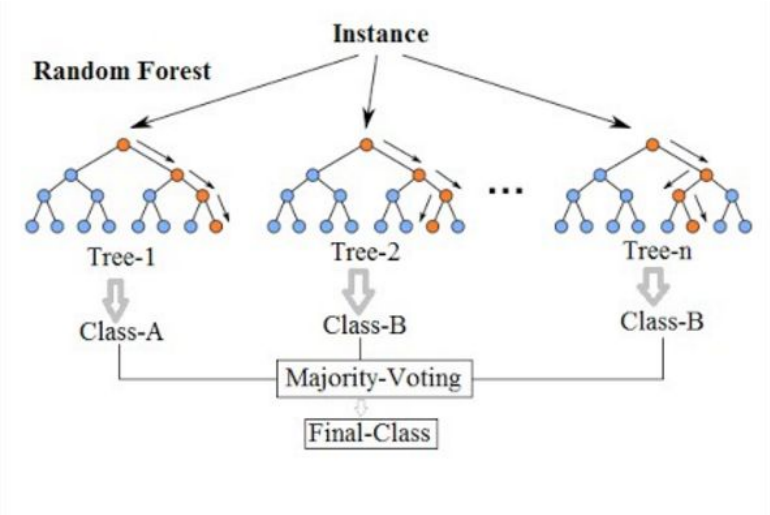
Decision Tree

- An analysis that creates a tree diagram from data
- Classifies data based on “yes” or “no” information



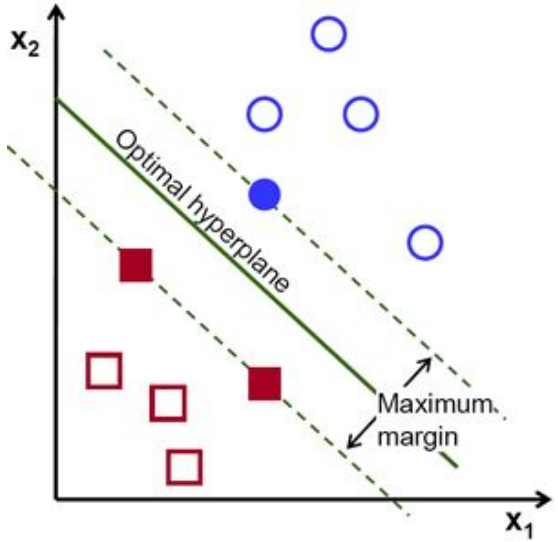
Random Forest

- Combines the outputs of multiple decision trees to produce a single result
- Allows for highly accurate predictions by obtaining majority voting



Support Vector Machine (SVM)

- Well-known algorithm in machine learning
- High generalizability with a small amount of supervised data



Gaussian Naive Bayes

- Machine learning algorithm used for binary classification problems
- Assume that the characteristics of each class follow a Gaussian distribution
- Predicting new classes of data based on calculated probabilities by learning the mean and variance of each class from training data

One of our approach

- Add “model selection”
  - we thought that this proposal would solve the “variety” problem required by Fujitsu.

- Furthermore, we added the complexity of the model.
  - Improving the interpretability of the models

Improvement 3: Explaining the Models and Calculating Complexity

Simple = Better? → Rademacher Complexity

- Rademacher complexity is a measure that quantifies the complexity of a function set.
- Rademacher complexity can be measured for most predictive models such as Lasso, decision trees, and SVM.
- Our goal is to use Rademacher complexity to find models that are not complex. Simple = Better!

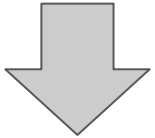
$$\hat{\mathfrak{R}}_S(\mathcal{G}) = \mathbb{E}_\sigma \left[\sup_{g \in \mathcal{G}} \frac{1}{n} \sum_{i=1}^{n} \sigma_i g(x_i) \right] \qquad S = \{x_1, \cdots, x_n\} \in \mathcal{X}$$

	feature_1	feature_2	feature_3	feature_4
Name1	1	-1	1	-1
Name2	1	1	-1	1
Name3	1	-1	1	-1
Name4	-1	1	1	1

.....

Label
1
-1
1
1

Original Data Set



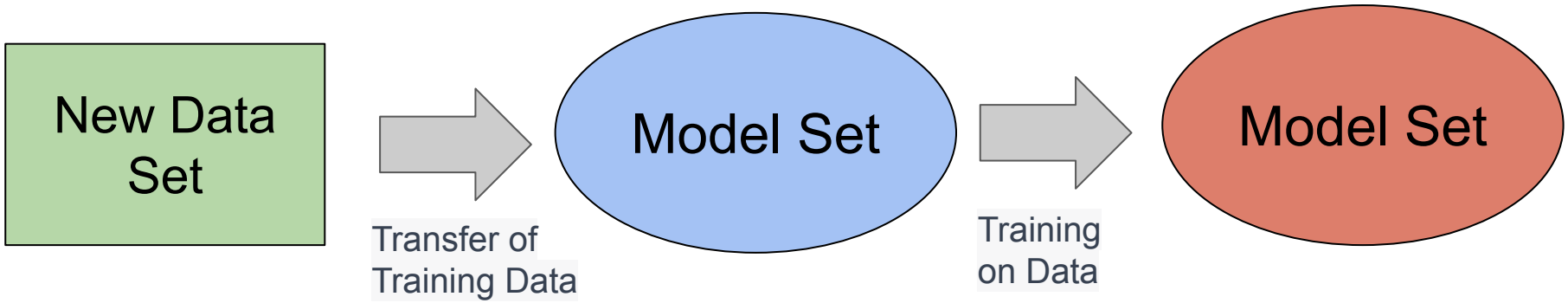
Randomly re-labelled

	feature_1	feature_2	feature_3	feature_4
Name1	1	-1	1	-1
Name2	1	1	-1	1
Name3	1	-1	1	-1
Name4	-1	1	1	1

.....

Random Label
1
1
1
-1

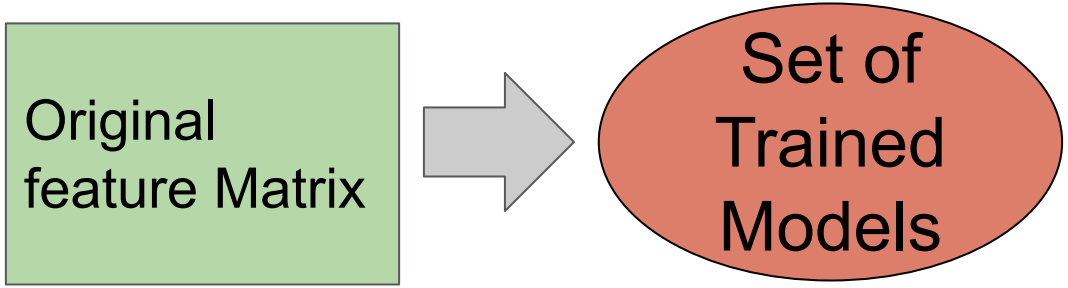
New Data Set



New dataset with randomly re-labelled labels and original feature matrix

Set of Untrained Models

Make a Set of Trained Models



Predict random labels using the original feature matrix.

We obtained a matrix of predictions for each model

Model_1	Model_2	Model_3	Model_4
1	1	1	-1
1	1	1	1
1	-1	1	1
-1	-1	-1	-1

.....

n

Random Label
1
1
1
-1

σ

Model_1	Model_2	Model_3	Model_4
1	1	1	-1
1	1	1	1
1	-1	1	1
-1	-1	-1	-1

.....

X

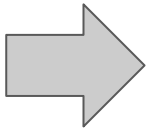
Let the random labels be the vector σ, the feature matrix of the model be the matrix X, and the number of rows be n.

Random Label
1
1
1
-1

n

Model_1	Model_2	Model_3	Model_4
1	1	1	-1
1	1	1	1
1	-1	1	1
-1	-1	-1	-1

$$\frac{\sigma^T X}{n}$$



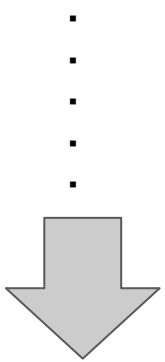
Model_1	Model_2	Model_3	Model_4
1	0.5	1	0.5

……

Calculate the product of σ and X to determine the percentage of random labels that could be predicted.

	Model_1	Model_2	Model_3	Model_4
$\sigma^{(1)}$	1	0.5	1	0.5
$\sigma^{(2)}$	0.5	0.5	0.5	0
$\sigma^{(3)}$	1	1	1	0.5
$\sigma^{(4)}$	0.5	0	-0.5	1

……



average

Rademacher	0.72	0.65	0.32	0.44

……

Generate a random label σ many times and average a series of calculations.

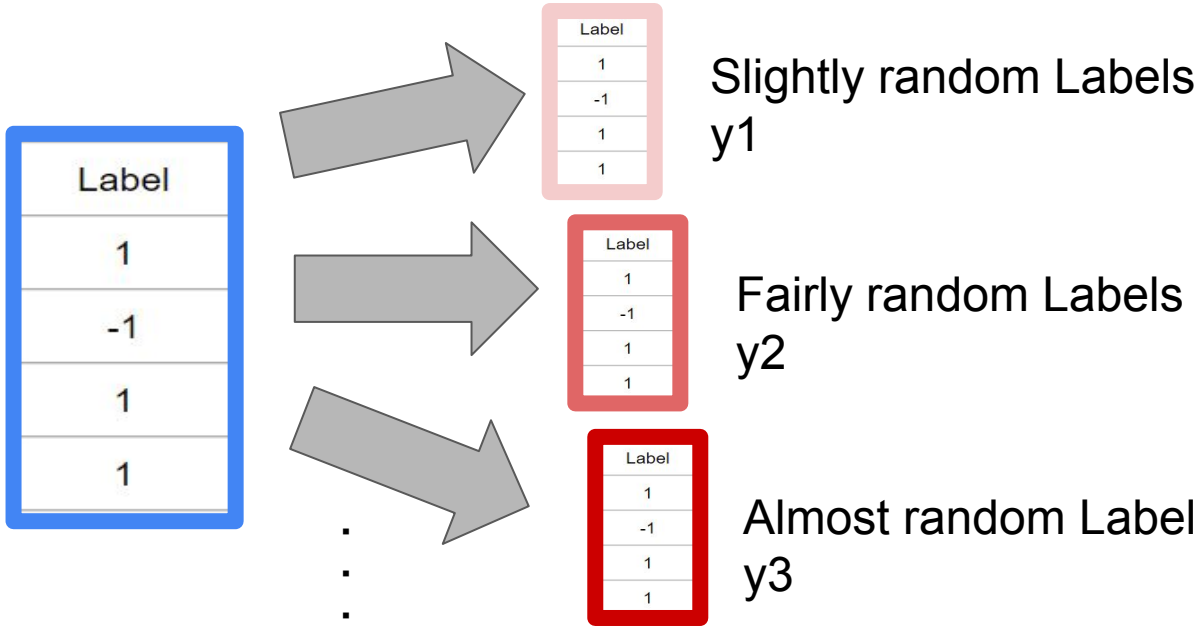
It is called the empirical Rademacher complexity.

- By measuring the ability to learn to random labels, the potential complexity of the model could have been measured.

- However, we felt the need to evaluate the model in more detail and prepared a new metric.

Our metric shows how well the model predicts as the randomness of the labels increases.

This allows us to examine how much more random the model becomes less predictable.



Random Label
1
1
1
-1

Model_1	Model_2	Model_3	Model_4
1	1	1	-1
1	1	1	1
1	-1	1	1
-1	-1	-1	-1

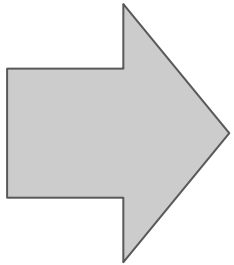
$$\frac{y_1^T X_1}{n}$$

Model_1	Model_2	Model_3	Model_4
1	0.5	1	0.5

Random Label
1
1
1
-1

Model_1	Model_2	Model_3	Model_4
1	1	1	-1
1	1	1	1
1	-1	1	1
-1	-1	-1	-1

$$\frac{y_2^T X_2}{n}$$



Model_1	Model_2	Model_3	Model_4
1	0.5	1	0.5

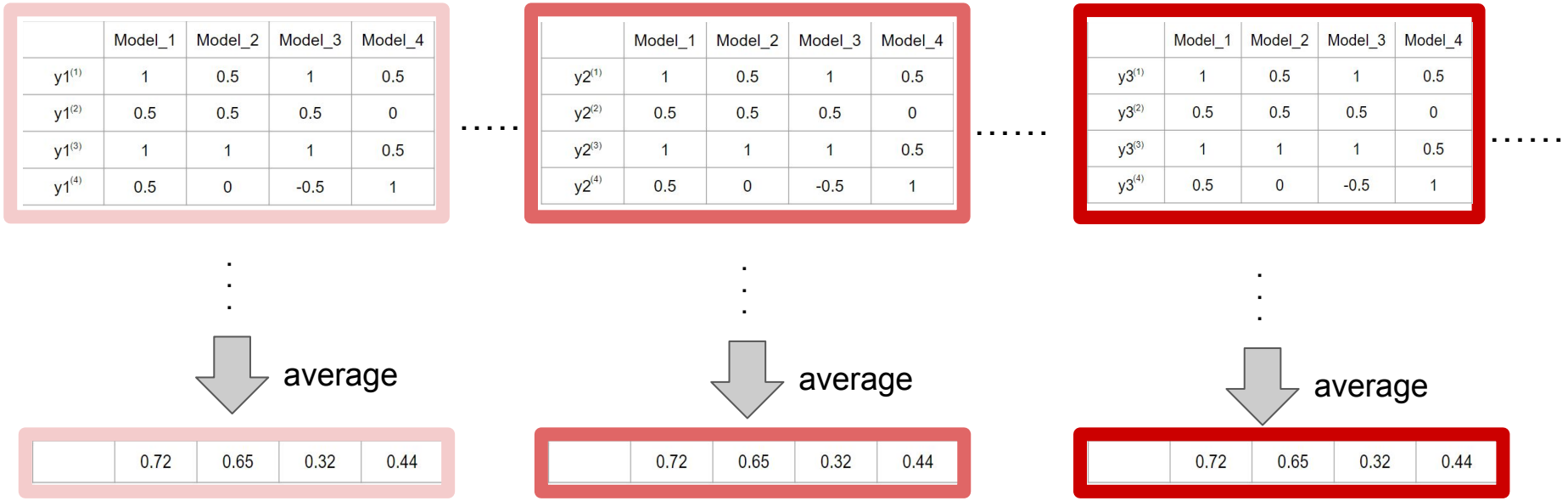
Random Label
1
1
1
-1

Model_1	Model_2	Model_3	Model_4
1	1	1	-1
1	1	1	1
1	-1	1	1
-1	-1	-1	-1

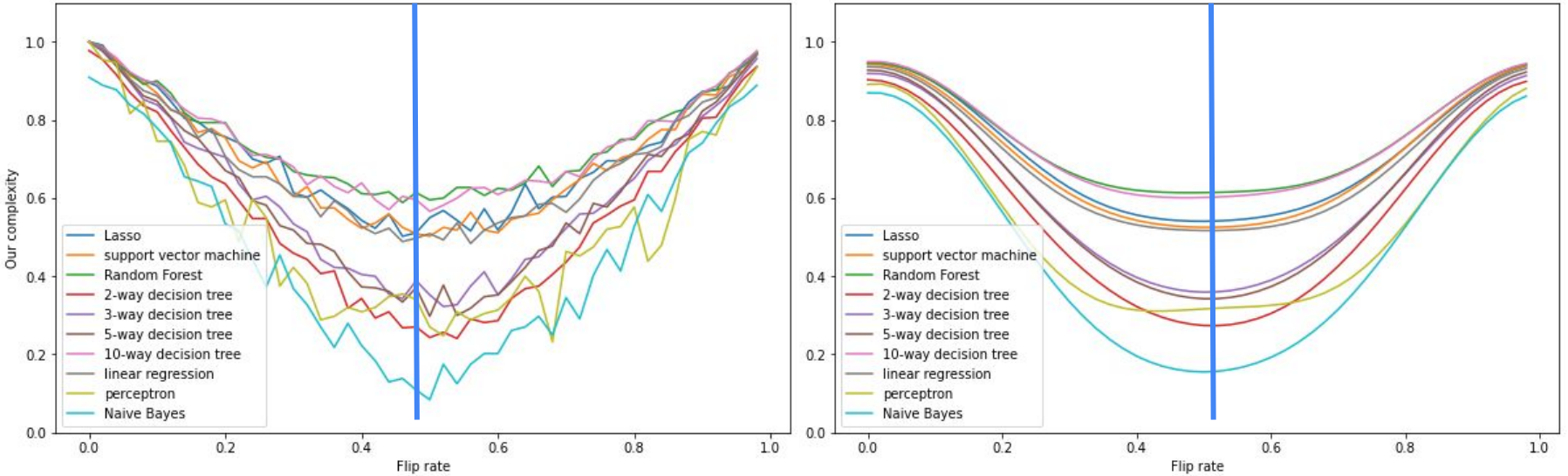
$$\frac{y_3^T X_3}{n}$$

Model_1	Model_2	Model_3	Model_4
1	0.5	1	0.5

For each label, the same process is used as when the Rademacher complexity is calculated.



The process of determining the label for each certain clutter is repeated many times and the average is taken.



The axis is blue in the middle, the labels are completely random.

This graph shows how much the model loses predictive accuracy for random labels as the randomness of the labels increases.

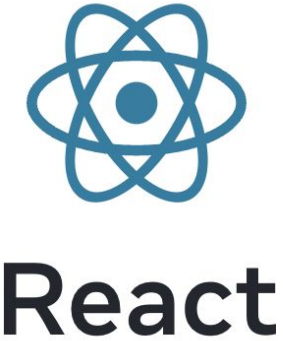
Complexity Summary

- The potential complexity of the model was evaluated by looking at the model's follow-up to random labels by Rademacher complexity.

- Our new method is a more detailed analysis of Rademacher complexity, which will lead to the discovery of a simpler model by referring to the graph.

“Hello, New Learning!”
Current User Interface

New Interface with Our Algorithms



Back-End

Python Flask

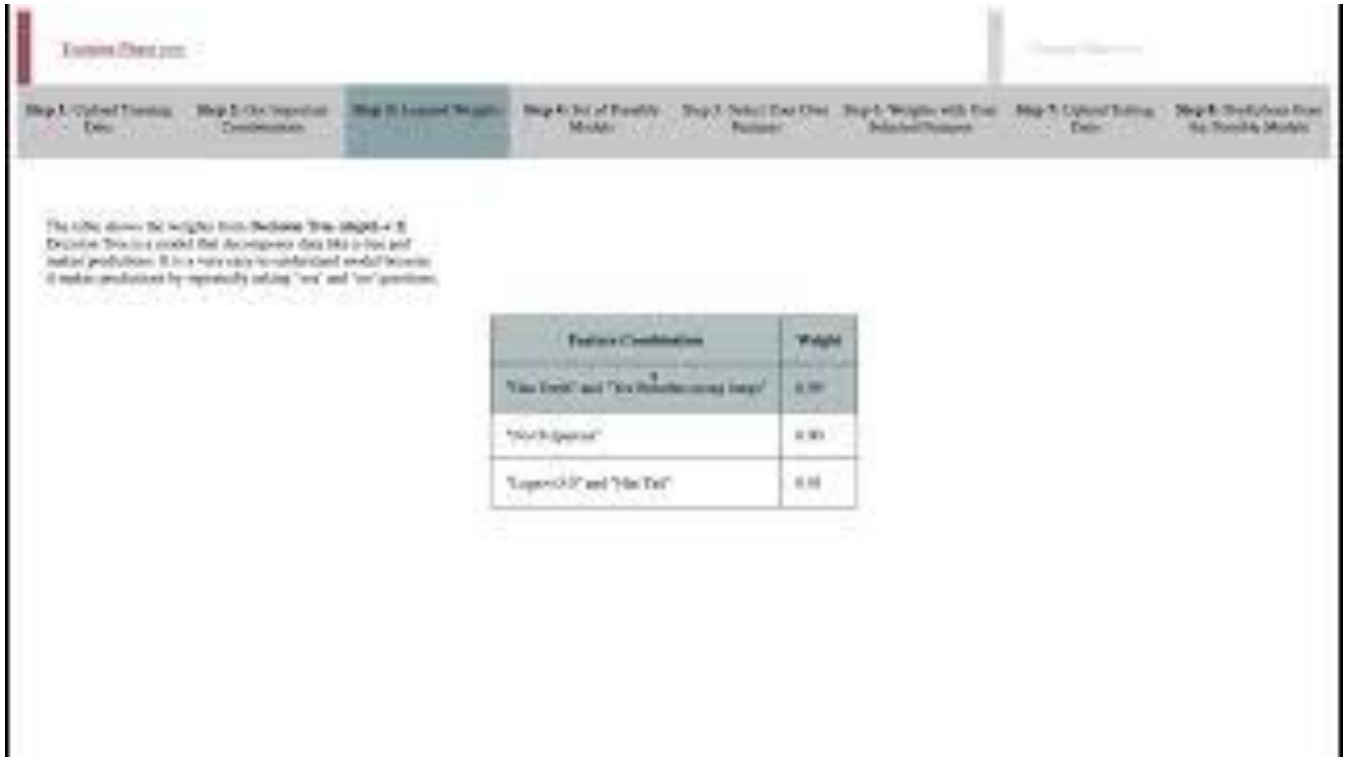
Front-End

React JS

Demo of New User Interface

Demo of New User Interface

Demo of New User Interface

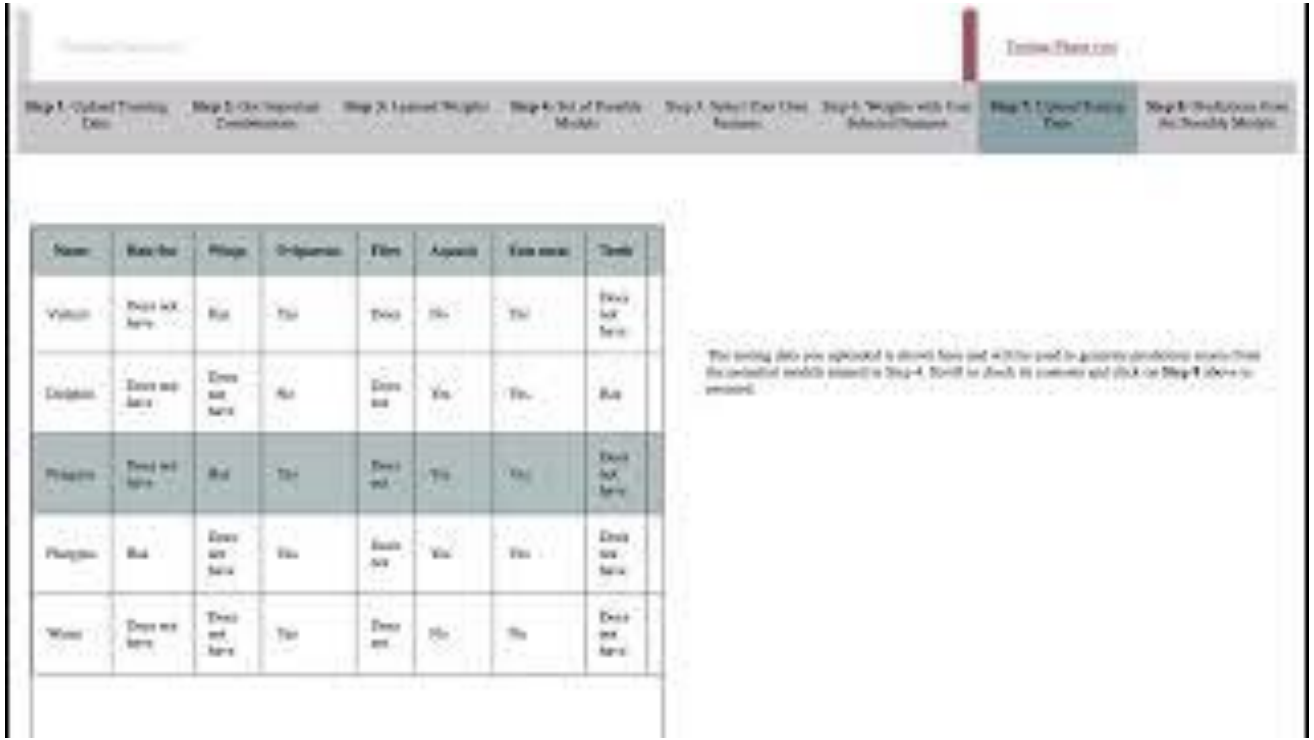


Demo of New User Interface

Demo of New User Interface

Demo of New User Interface

Demo of New User Interface



Demo of New User Interface

Conclusion

- Our version of Wide Learning considers many different models while making predictions
- The user can decide which model they prefer based on accuracy and complexity
- The user also has the option to select important feature combinations for the Lasso model
- The new user interface incorporates additional explanations to address feedback from the pilot survey
- A new survey has been sent out to evaluate the new workflow and interface we developed

Thank You!