g-RIPS Sendai 2023

Fujitsu Group

Final Report

Authors:
Lily Ge[1]
Yuki Kimura[2]
Takeshi Nakashima[2]
Molly Noel[3]

Mentors:
Jesica Bauer[+]
Dr. Hiroyuki Higuchi[*]

[1] Northwestern University
[2] Musashino University
[3] Rensselaer Polytechnic Institute
[+] Academic Mentor, Rensselaer Polytechnic Institute
[*] Industrial Mentor, Fujitsu Limited

Abstract

As Artificial Intelligence becomes more powerful, the need for clear reasoning behind important AI-supported decisions also grows rapidly. In an effort to reduce the gap between humans and AI, Fujitsu created Wide Learning, an explainable AI tool used for binary classification. Although it provides step-by-step information to help the users understand, initial evidence suggested that gaps remain and the presentation of information was not always intuitive. We propose several improvements, based on the pilot survey, to the existing Wide Learning tool that are designed to increase its convincingness, variety, and discoverability. To improve discoverability, we add explanations to the tool based on feedback from a pilot survey. For variety, we incorporate different types of models and present them to the user, which also supports the idea of the hypothesis-driven approach to explainability AI. To increase convincingness, we incorporate user feature selection and present the user with information about the complexity of each of the proposed models. These improvements are all incorporated into a new interactive user interface.

August 8, 2023

Contents

Table of Variables and Parameters

Variable	Definition
n	number of rows in a data set
p	number of columns, or features, in a data set
X	matrix of training data
$\mathcal{X}$	a subset of the training data, X
y	classification labels for the training data in X
$\mathcal{Y}$	a subset of the classification labels for the training data in $\mathcal{X}$
β	parameter to optimize in our objection function, e.g. $\mathcal{L}(\beta)$ for Lasso
β^*	optimal solution to the objective function
f	a loss function, e.g. $f(X, \beta, y)$ as the logistic loss function for Lasso
ρ	hyperparameter to control sparsity of β in Lasso
σ	the sigmoid function
r	an independent random variable from the Rademacher distribution
$\tilde{\beta}$	weights in the feature selection optimization problem
$\beta^{(i \to j)}$	new weights after performing feature selection (replacing feature i with feature j)
$\mathcal{G}$	$\mathcal{G} \subset \{f : X \to Y\}$
$\mathcal{S}$	$\mathcal{S} = \{x_1, ..., x_n\} \subset X$
$\hat{\mathfrak{R}}_S(\mathcal{G})$	$\hat{\mathfrak{R}}_S(\mathcal{G}) = \mathbb{E}_\sigma \left[\sup_{g \in \mathcal{G}} \frac{1}{n} \sum_{i=1}^{n} \sigma_i g(x_i) \right].$
$\mathfrak{R}_n(\mathcal{G})$	$\mathfrak{R}_n(\mathcal{G}) = \mathbb{E}_{S \sim D}[\hat{\mathfrak{R}}_S(\mathcal{G})].$
$\mathcal{Y}'$	$\mathcal{Y}' \subseteq \mathcal{Y}$
y'_i	$y'_i \in Y'$
m	$m = \lvert\{y'_1, ..., y'_m\}\rvert$
z_i	$z_i = \begin{cases} -y_i & (y_i \in \mathcal{Y}') \\ y_i & (y_i \notin \mathcal{Y}') \end{cases}$
$\hat{\mathfrak{C}}_{S,m}(\mathcal{G})$	$\hat{\mathfrak{C}}_{S,m}(\mathcal{G}) = \mathbb{E}_z \left[\sup_{g \in \mathcal{G}} \frac{1}{n} \sum_{i=1}^{n} z_i g(x_i) \right].$
$\mathfrak{C}_{n,m}(\mathcal{G})$	$\mathfrak{C}_{n,m}(\mathcal{G}) = \mathbb{E}_{S \sim D}[\hat{\mathfrak{C}}_S(\mathcal{G})].$

1 Introduction

Artificial Intelligence (AI) is capable of performing powerful tasks and has become prevalent in many domains (e.g., [9], [10], [11]). However, its implementation in some fields, such as finance and medicine, is limited due to a lack of user understanding and confidence in the construction of the model and the calculation of the final results. If a model lacks explainability, the users are left with no choice but to trust that the algorithm's output is correct—in other words, they are expected to rely on "black-boxes" for making decisions [12]. To address this issue, researchers in AI have teamed up with other fields such as Human-Computer Interaction to work toward the goal of better human-AI collaborations and investigating Explainable AI (XAI) [1].

One existing approach in XAI is Wide Learning developed by Fujitsu. By incorporating the process of discovery science, Wide Learning creates a more shallow neural network and provides users with step-by-step explanations [13]. When investigating this tool, we found that the explanations from Wide Learning are not always the most intuitive to understand. Additionally, we found that aspects that were more fully explained existed outside of the tool in areas which were more difficult to discover.

In this project, we aim to start with what Fujitsu has accomplished and further improve the explainability aspect of the Wide Learning tool by increasing the convincingness, variety, and discoverability of the existing tool. To do this, we identified and incorporated explainability techniques from literature (Section 2), investigated the existing tool and identified if it utilized these techniques (Section 3), created a pilot survey to curate information on the existing tool (Section 4), implemented new models to be used in the back-end of the Wide Learning interface to provide more variety to the users (Section 5), incorporated user feature selection into one of these models (Section 6), developed a new complexity measure to quantify a model's understandability to new users (Section 7), and combined all these results into a new user interface which we believe will be more beneficial to users of the Wide Learning Tool (Section 8).

2 Literature Review

To achieve our goal of developing an interface that supports the interpretability of AI models, we first discuss relevant works in the field of XAI.

Prior work has reviewed multiple goals for XAI, such as trustworthiness, causality, transferability, informativeness, confidence, fairness, accessibility, interactivity, and privacy awareness [5]. There also exist different explainability approaches. For instance, a review by Islam et al. discussed categories of explainability methods: intrinsically interpretable, model-agnostic, and example-based [14]. Intrinsically interpretable methods include linear / logistic regressions, decision trees; they are considered to not be "black box" models and are relatively more intuitive [14]. Model-agnostic methods (e.g., feature selection visualizations) are separated from the actual model itself and could be applicable across different models [14]. Example-based explanation methods provide explanations using selected data sets that can help explain the behavior of the model, and relevant methods include counterfactual, adversarial, prototypes, influential instances, and k-nearest neighbors model [14]. Broadly, explainability techniques can be grouped into recommendation-driven or hypothesis-driven approaches, which were discussed by Miller [23]. Miller argued that hypothesis-driven approaches can work better due to its alignment with human cognitive decision-making processes [23]. All of these approaches are working toward the goal of improving interpretability of AI models, which ultimately improves the overall usability of models in support of decision-making. We design and implement a hypothesis-driven explainability tool that builds off an existing tool (see Section 3.1), and is informed by existing approaches outlined in literature.

3 Background

In this section, we discuss the necessary background for our new Wide Learning tool. This includes the existing Wide Learning tool developed by Fujitsu, as well as the mathematical background for logistic regression and the Lasso method which are currently used in the existing back-end.

3.1 Wide Learning

Wide Learning, a tool originally developed by Fujitsu in 2018, incorporates discovery science and fast enumeration techniques to create a binary classification tool which explains the process of how the model works [13]. The Wide Learning tool allows the user to explore the process of learning from the training data and using the model on the test data. The process is broken into the following 5-step process:

1. The user is prompted to input a training data set and able to ensure that it was uploaded correctly.
2. Wide Learning analyzes the provided features and generates combinations of features which it determines are important to classification.
3. The user is presented with the trained model as well as the learned results and weights assigned to the features from Step 2.
4. The user is prompted to input a testing data set for use in the model and given the opportunity to ensure that it was uploaded correctly.
5. Wide Learning classifies the data set and provides some visuals to justify the final classification result.

Throughout the tool, there are explanatory techniques such as descriptive text, color-coded visualizations, and interactive features which allow users to explore the data and the classifications. However, our initial exploration identified certain aspects of the tool that may need more granular explanations for users from a more general audience. This may entail explicitly explaining how the combinations were chosen as important and the weights for the important combinations.

As our group was trained in the tool by Fujitsu Limited, we wanted to ensure that we had a more accurate reflection of how an unfamiliar user may interpret the interface. We decided to generate a pilot survey to distribute to other members in the Graduate-level Research in Industrial Projects for Students (g-RIPS), as their knowledge of mathematics would most likely reflect that of the typical user of the tool. We discuss this survey in more detail in Section 4.

3.2 Logistic Regression

The existing Wide Learning tool utilizes logistic regression as part of its classification technique, which is a statistical model primarily used for binary classification problems. That is,

$$y = \sum_{i=1}^{n} w_i x_i + b \quad i = 1, 2, ..., n \tag{3.1}$$

Then, it transforms the output to a classification using the sigmoid function

$$\sigma(y) = \frac{1}{1 + \exp(-y)} \tag{3.2}$$

where x is an element from the input data X, w is the the weights of the features, and b is the bias term. A value is classified with class 1 if the transformed value from the sigmoid function is greater than 0.5, and class 0 if it is less than or equal to 0.5.

By minimizing the loss function on the training data set, logistic regression learns the model's parameters by making predictions based on the input features, which in turn helps determine the feature importance. It utilizes ℓ_2-regularization to encourage sparsity of the model weights and calculates a weighted sum of input data features using a linear prediction model. ℓ_2-regularization adds the square of the absolute value of the coefficient to the objective function as a regularization term to control the coefficients from becoming too large [19].

3.3 Lasso Method

Tibshirani (1996) proposed the Least Absolute Shrinkage and Selection Operator (Lasso) as a way to select important model features using the following optimization problem [6]:

$$\arg\min_{\beta} \mathcal{L}(\beta) := f(X, \beta, y) + \rho\|\beta\|_1 \tag{3.3}$$

where f is a loss function, $X \in \mathbb{R}^{n\times p}$ is the input training data, $y \in \mathbb{R}^n$ is the vector of true outputs for the training data, ρ is a hyperparameter that determines the weight of the regularization term, and $\beta \in \mathbb{R}^{p+1}$ are the model parameters (one weight for each feature, p, and an intercept β_0). Here n is the number of training samples and p is the number of features. In this objective function, the first term represents the training loss based on the model prediction. The second term in the objective function is the regularization term, which encourages β to be sparse. The nonzero elements of the optimal β found by solving this Lasso optimization problem represent the weights on the important features that are extracted from the model [2].

We can equivalently pose the formulation by dividing the results by ρ to more clearly see the sparsity of β. Let $C = 1/\rho$. Therefore, a smaller value of C puts less weight on the logistic loss term of the function and a higher relative weight on the regularization term, which encourages β to be more sparse. For our implementation of Lasso, we choose C in order to keep the number of nonzero weights less than 100. In practice, we initialize the problem with $C = 18$. If that value results in a number of nonzero weights greater than 100, C is multiplied by 0.9 until the number of nonzero weights until the sparsity is less than or equal to 100. The Lasso objective function used in our implementation is defined as

$$\arg\min_{\beta} \mathcal{L}(\beta) := C[f(X, \beta, y)] + \|\beta\|_1 \tag{3.4}$$

For the case of Lasso logistic regression, the loss function f is defined as follows. Here, let $X[i, :]$ is the ith row of X, and $X[i, j]$ is the (i, j) entry of X. Then we can formulate our loss function as

$$f(X, \beta, y) := \sum_{i=1}^{n} -[y_i \log[\sigma(X[i, :], \beta)] + (1 - y_i) \log[1 - \sigma(X[i, :], \beta)]] \tag{3.5}$$

where σ is the sigmoid function that calculates the probability that a given data point has a label of 1.

This function is also used to make predictions on testing data points after the model has been trained. A probability greater than 0.5 leads to the testing data point to be classified with a label of 1, otherwise it is classified as 0. We can more explicitly state this as

$$\sigma(X[i, :], \beta) = P(y_i = 1|X[i, :], \beta) = \frac{1}{1 + \exp(-[\beta_0 + \beta_1 X[i, 1] + \beta_2 X[i, 2] + \cdots + \beta_n X[i, p]])} \tag{3.6}$$

By utilizing logistic regression and encouraging sparsity, we reduce the number of parameters from the initial problem and identify features which have a higher affect on the classification. While the underlying mathematics are more complicated, the results from the process are easier to understand for the user. Having less features naturally makes the problem easier to understand, as it lowers the dimensions of the problem. Additionally, the features which continue to exist within the problem due to their non-zero β should prove to be useful identifiers for prediction of future data sets even without the use of the tool.

4 Pilot Survey

In order to improve the Wide Learning tool, we first needed to better understand which aspects of the existing tool may benefit from additional explanations and which parts should be maintained. Therefore, we designed a pilot survey to collect data from people who have not yet interacted with the Wide Learning tool as extensively as our own team members.

4.1 Survey Structure

The survey was designed to take no more than 15 minutes in total and included screenshots from the existing tool to request feedback from the participants. Screenshots of the existing tool and the survey can be found in Appendix A. In general, the survey contained the following:

- Steps 1 (training) and 4 (testing) of the Wide Learning interface with the relevant training and testing data set already imported. We used the “Animals (mammals) classification” data set which was provided as an example for the tool.
- Results from Steps 2, 3, and 5 of the Wide Learning tool followed by open-ended questions to elicit people’s thoughts on the interface. Participants were then asked to rate how confident they were in their responses on a scale from one to five.
- Questions to identify if there are any confusions that need to be clarified in the existing interface.
- After people were asked to enter their responses, they were directed to the next page which included a summary of what the screenshot was showing and a question to ask them to describe whether their understanding aligned with our summary; if not, they were asked to respond with what they thought would have been helpful for them to understand.

4.2 Survey Results

We sent the original pilot survey in the `#students-grips2023` Slack channel with 16 potential participants and received 10 responses. These results were used to further inform how to design our interface to help improve interpretability. While overall participants seemed to have a general understanding of what the Wide Learning tool was displaying, there were several areas in the user interface that they thought were confusing.

Much of the confusion was stemmed from the “important feature combinations” which were displayed in Step 2 of the Wide Learning process. A screenshot of the tool at the time of writing is shown in Figure 1.

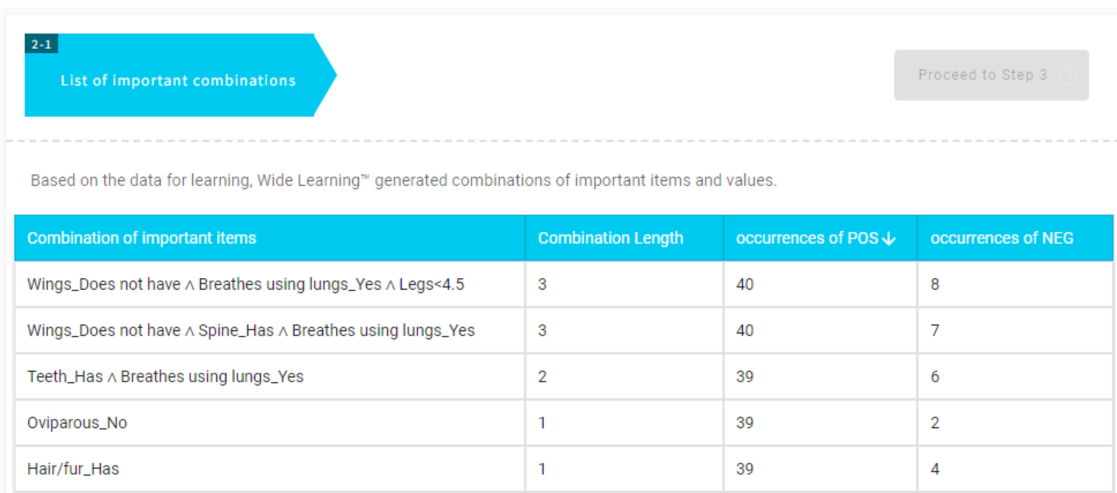


Figure 1: A screenshot of Step 2 of the Wide Learning tool at the time of writing this document.

The first area of confusion regarded the table headings: **Combinations of important items**, **Combination Length**, **occurrences of POS**, and **occurrences of NEG**. Some participants thought that the **occurrence of POS** column meant the number of training samples that had that combination, rather than the number of training samples that had that combination that belonged to the determined “positive” class. Additionally, some survey respondents did not understand the formatting for the **Combinations of important items** column. For example, `Teeth_Has` $\wedge$ `Breathes using lungs_Yes`.

The second major area of confusion regarded Step 3 of the Wide Learning tool (shown in Figure 2).

Figure 2: A screenshot of Step 3 of the Wide Learning tool at the time of writing this document.

This table displayed a column **importance (weight)** but did not explain how this weight was calculated or what it correlated to even when users interacted with the "[Intermediate users] Check the evaluation values of the learning results here" dialog box.

With the results of this survey in mind, we wanted to ensure our future interface provides more context for these areas of confusion. Additionally, we hope to expand the interface to provide more convincingness, variety, and discoverability to further enhance the user experience. The following sections discuss individual aspects which we have developed to increase the value of the Wide Learning tool Section 8 showing how each piece was implemented into a cohesive product.

5 Increasing Variety: More Modeling Options

It is common for an explanation for a judgement to not always be unique. In fact, allowing a user to see multiple perspectives can help increase their trust in the final presented result. The existing interface for Wide Learning only presents the user with one model, meaning the user is not presented any variety. Therefore, we hoped to implement some new models to execute on the back-end to provide the user with more options to get their results and additionally ensure that they are always getting the most accurate prediction.

In this section, we describe additional models that are incorporated in our proposed interface with the intention to be to provide the best of these options to the user and allow them to investigate the results of the other models. Due to time constraits these models were all implemented using Python's `scikit-learn` library unless otherwise noted [15].

5.1 Decision Tree

Decision Trees are one of the machine learning algorithms commonly used in supervised learning. They employ a tree-like branching structure to classify or regress data. At each stage, the data is split based on specific features, forming new groups or child nodes. Eventually, the data reaches leaf nodes where it is assigned to specific classes, thus performing the classification [16]. The advantage of decision trees lies in their interpretability, as the tree's structure allows for the visual inspection of feature importance. Feature importance is computed using the Mean Decrease Impurity (MDI) method, which evaluates how much each feature can separate the data by considering the impurity within nodes . When constructing the decision tree, the importance of each feature is estimated based on how much it reduces impurity within the nodes. A higher numerical value indicates a more important feature. However, it might be challenging to determine whether the importance is positive or negative. Nonetheless, by considering the magnitude of the numerical values, one can still understand the relative importance of features .

5.2 Random Forest

Random Forest is a machine learning method that combines multiple decision trees to make predictions. Ensemble learning, which is used in Random Forest, involves combining multiple machine learning models to achieve better performance than individual models. When determining feature importance in Random Forest, the average reduction in impurity calculated from multiple decision trees is used [17].

5.3 Perceptron

Perceptron is a simple linear classification algorithm used for 2-class classification problems. Similar to logistic regression, it calculates the weighted sum of input data features and classifies them into two categories based on whether the value exceeds a certain threshold. However, it is only applicable to linearly separable data, and the boundary is determined by a straight line. Furthermore, like logistic regression, the perceptron determines the feature importance by training its model parameters using the training data [20].

5.4 Support Vector Machine (SVM)

Support Vector Machine(SVM) is a machine learning model used for classification problems. SVM aims to find a hyperplane in the feature space that best separates two classes. The goal is to find a hyperplane that maximizes the distance (margin) between data points. Additionally, the distance from the data points within the margin to the hyperplane should also be maximized [18]. By following these steps, we can discover the hyperplane. In addition, in the linear SVM model of scikit-learn, weights are assigned to each feature to find a hyperplane that separates the data. The computation of SVM weights is based on the equation of the separating hyperplane. During the training process of linear SVM, the objective function is minimized to find the optimal separating hyperplane. In this process, a regularization term is introduced to determine the weights. This allows us to understand the importance and influence of each feature.

5.5 Gaussian Naive Bayes

Gaussian Naive Bayes is a machine learning algorithm used for binary classification problems. It assumes that the features of each class follow a Gaussian distribution and uses Bayes' theorem to calculate the conditional probability. By learning the mean and variance of each class from the training data, it can predict the class of new data based on the calculated probabilities. However, since it uses a probability model and it is difficult to directly interpret how specific features influence predictions, it is considered a black-box model [21]. Therefore, if this model is chosen during model selection, it lacks interpretability, and it would be necessary to add post-hoc explanations. However, at the moment, we have not yet considered an approach for addressing this aspect.

6 Increasing Convincingness: Implementing Feature Selection

A result which is accepted by the user is considered to be "convincing". One such way that we can increase the tool's convincingness is to allow the user to have input in what features are included in the model. Being able to directly interact with the model and prioritize information which may be important in practice could increase this metric.

The optimal features selected by Lasso might not be the only important features [2]. The paper by Hara and Maehara (2016) introduces a way to replace certain features selected by Lasso with user selected features. This can increase the trustworthiness of the model, as it allows the user to make sure that the features that they believe are important are included in the model [2]. This paper also introduces a way to calculate a score between two features, to determine if they are similar enough to have one replace the other with a minimal decrease in accuracy. We apply this technique to the feature selection in the new Wide Learning interface in order to increase the model's trustworthiness. The following is a description of the process developed by Hara and Maehara (2016) [2] that we apply to our Logistic Lasso model:

The original Lasso model performs feature selection by assigning nonzero weights to some features and zero weights to others. The features that are assigned zero weights are not included in the final model. The process by Hara and Maehara (2016) allows users to select features that were not included in the original model. Once this feature has been selected, a new nonzero weight is assigned to the feature. To keep the number of nonzero weights the same as the original model, when one feature is added to the model by changing its weight from zero to a new nonzero value, another feature is taken out of the model by reassigning it a weight of zero. The optimal model weights selected by Lasso are denoted by the vector $\beta^* \in \mathbb{R}^{p+1}$, where p is the number of feature combinations generated by Wide Learning. Note that β_0 is the intercept term that does not correspond to any of the model weights, and is not eligible to be added to the model or taken out of the model.

Say the user decides to select feature j to add to the model, with corresponding weight $\beta_j^* = 0$. To maintain sparsity, let i be an index such that $\beta_i^* \neq 0$; this will be the feature that is removed from the model by setting $\beta_i = 0$. Then, define $\tilde{\beta}$ such that $\tilde{\beta}_i = 0$, $\tilde{\beta}_j \neq 0$ and $\tilde{\beta}_k = \beta^*$ where $k \neq i, j$. The new nonzero value of β_j is found by solving the following optimization problem:

$$\beta_j^{(i)} = \arg\min_{\tilde{\beta}_j} C[f(X, \tilde{\beta}, y)] + |\tilde{\beta}_j| \tag{6.1}$$

where $\beta_j^{(i)}$ is the optimal β_j for the above problem. This is the same objective function as the original Lasso problem, however we are only optimizing with respect to β_j. This optimization problem is solved using the proximal gradient descent method with a constant step size.

The updated model weights after adding β_j to the model and taking β_i out of the model are defined as $\beta^{(i\to j)}$, where $\beta_i^{(i\to j)} = 0$, $\beta_j^{(i\to j)} = \beta_j^{(i)}$, and $\beta_k^{(i\to j)} = \beta_k^*, \forall k \neq i, j$

We know $\beta_j^{(i)} = 0$ when $C|\nabla f(X, \tilde{\beta}, y)| \leq 1$ based on the optimality condition defined in [2], where $\nabla f(X, \tilde{\beta}, y)$ is the gradient of f with respect to β_j. Since the goal of this optimization problem is to find a new optimal nonzero value of β_j when β_i is set to 0, if this optimal value $\beta_j^{(i)}$ is 0, then β_i cannot be replaced with β_j in the model.

For a given β_j to be inserted into the model, any $\beta_i \neq 0$ where $\beta_j^{(i)} \neq 0$ is a candidate for the feature combination that can be removed from the model. In order to choose which β_i is the best feature to be removed, we compare the original Lasso objective value for all of the potential new solutions $\beta^{(i\to j)}$. Whichever solution leads to an objective value $\mathcal{L}(\beta^{(i\to j)})$ that is closest to the original objective value $\mathcal{L}(\beta^*)$ is the best new vector of weights. If all $\beta_j^{(i)} = 0$, then β_j cannot be inserted into the model. We use this method to perform user feature selection for Lasso in our user interface.

The code to perform this user feature selection is implemented in Python. The original Lasso coefficients β^* are found using the scikit-learn library for Logistic Regression with the L1 regularizer and the liblinear solver [15]. PyTorch is used in our implementation of the proximal gradient descent method used to find $\beta_j^{(i)}$.

7 Increasing Convincingness: Measuring Model Complexity

In addition to allowing user input, convincingness could also be determined by how complicated the models used are. However, we found that some of the existing metrics for complexity failed to account for information which we felt was valuable to reflect the user experience. Therefore, we generated a new metric which reflects the complexity of the model to the user and which allows them to select a model which may be more applicable and understandable for their usage.

7.1 Rademacher Complexity

We define a hypothesis set from the collection of models discussed in Section 5 and Section 6. This hypothesis set can respond to any labeling, which means that it is possible for additional noise to also be learned. We examine the complexity of the model using the concept as it is treated in machine learning using Rademacher complexity. That is, a model which is able to easily follow random labels is more complex and therefore sacrifices its explanatory power. As Wide Learning does binary classification, we redefine Rademacher complexity by consulting [22].

Let the input space X be mapped to the label space Y such that $Y = \{-1, 1\}$. The set of classifiers that map the input space to our label space is denoted by $\mathcal{G} \subset \{f : X \to Y\}$. The set of input points is denoted by $\mathcal{X} = \{x_1, \ldots, x_n\} \subseteq X$ and the corresponding labels are $\{y_1, \ldots y_n\} \subseteq Y$, where $y_i \in Y \forall i$ where n represents the total number of data points or samples in our data set. Let $\{r_1, \ldots, r_n\}$ be independent random variables from the Rademacher distribution. That is, r_i is either -1 or 1 with equal probability.

Then, the empirical Rademacher complexity is defined as

$$\hat{\mathfrak{R}}_{\mathcal{X}}(\mathcal{G}) = \mathbb{E}_r \left[\sup_{g \in \mathcal{G}} \frac{1}{n} \sum_{i=1}^{n} r_i g(x_i) \right] \tag{7.1}$$

which indicates an intuitive explanation. When $r_i g(x_i) = -1$, meaning that r_i is 1 and $g(x_i)$ is -1, or r_i is -1 and $g(x_i)$ is 1, $g(x_i)$ failed to make a correct prediction. Conversely, if $r_i g(x_i) = 1$, it means that $g(x_i)$ successfully carried out the forecast. The sum of these tells us how well the model $g(x_i)$ is able to follow the random labels r_i and the randomly labelled x_i. The Rademacher complexity is then shown.

However, in proposing this method, we found an interesting result. When the input point $\mathcal{X} = \{x_1, ..., x_n\}$ is a random variable following some distribution D, the Rademacher complexity of $\mathcal{G}$ can change dependent on the chosen distribution D. This means that depending on our initialization of our random labels, we could get a complexity measure which is inaccurate.

Therefore, we can redefine our Rademacher complexity as

$$\mathfrak{R}_n(\mathcal{G}) = \mathbb{E}_{\mathcal{X} \sim D}[\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G})] \tag{7.2}$$

The reason why the Rademacher complexity takes the expected value of the empirical Rademacher complexity is that the value of the empirical Rademacher complexity depends on the data. In other words, taking the expected value of the empirical Rademacher complexity for a set of input points $\mathcal{X}$ is the definition of Rademacher complexity.

7.2 Verification of Model Complexity

As mentioned in the previous section, the Rademacher complexity of our models is dependent on the distribution chosen for our random labels. This could result in our interface suggesting a model which is only perceived to be less complex due to a favorable initialization.

Our question is: if two models have the same Rademacher complexity for a given data size, are they similarly tracked for a given label? When the Rademacher complexity of a model that can follow some random noise but not completely random labels is equal to that of a model that cannot follow some random noise, it seems that they should not be treated as equivalent models.

Therefore, we propose a method to assess the models ability to follow label clutter.

The operation of adding noise to the training data seems to be called "noise injection". However, we couldn't find a specific name for the method of adding noise to the labels of the training data and measuring the complexity of the model. We denote this new complexity measure as $\hat{\mathfrak{C}}$.

Let $\mathcal{Y}' \subseteq \mathcal{Y}$ such that $|\{y'_1, ..., y'_m\}| = m$ with $0 \leq m \leq n$. For every element $y_i \in \mathcal{Y}$, we define a new element z_i as

$$z_i = \left\{ \begin{array}{ll} -y_i & (y_i \in \mathcal{Y}') \\ y_i & (y_i \notin \mathcal{Y}') \end{array} \right\} \tag{7.3}$$

Then, the definition of complexity $\hat{\mathfrak{C}}$ is

$$\hat{\mathfrak{C}}_{\mathcal{X},m}(\mathcal{G}) = \mathbb{E}_z \left[\sup_{g \in \mathcal{G}} \frac{1}{n} \sum_{i=1}^{n} z_i g(x_i) \right] \tag{7.4}$$

This new measure allows us to arbitrarily select n real labels that a certain data has without duplication and then invert the labels, i.e. prepare new labels that are -1 times more real labels. The complexity $\hat{\mathfrak{C}}$ is a measure of the average conformity of the function set $\mathcal{G}$ to the labels for such labels. When $m = 0$ or $m = n$, it is exactly the same as or exactly opposite to the actual label. On the other hand, when $m = n/2$, half of the labels are random.

If an input point $\mathcal{X}$ of this complexity follows a certain distribution D, $\mathfrak{C}$ is defined as

$$\mathfrak{C}_{n,m}(\mathcal{G}) = \mathbb{E}_{\mathcal{X}\sim D}[\hat{\mathfrak{C}}_{\mathcal{X}}(\mathcal{G})] \tag{7.5}$$

Then, let us compare the correspondence between Rademacher Complexity and Complexity $\mathfrak{C}$.

The set of possible values of m is M, i.e., $M = \{m \in \mathbb{Z} : 0 \leq m \leq n\}$. The range that the error between the random label generated from the random variable r_i and the real label can take is equal to M. Consider the probability that the error between a random label and a real label can be k. The probability that any one random label is equal to the real label is 1/2. The error between the random label and the real label is k, given that the size of the label is n. So, the probability that the error occurs exactly k times in n trials, i.e., it follows a binomial distribution. We denote such a given probability as $P_{(M=k)}$. Since the expected value of the binomial distribution can be expressed as $n/2$, the Rademacher complexity is approximately

$$\mathfrak{R}_n(\mathcal{G}) = \mathfrak{C}_{n,\frac{n}{2}}(\mathcal{G}) \quad \text{or more strictly}$$

$$\mathfrak{R}_n(\mathcal{G}) = \sum_{i=1}^{n} P_{(M=i)}\mathfrak{C}_{n,i}(\mathcal{G})$$

Figure 3 visualizes how much the model is unable to follow a given label as the randomness of the label increases. Using the traditional definition of Rademacher complexity, the Perceptron and the 2-way decision tree have a Rademacher complexity of 0.304 and 0.281 respectively. However, from the figure, we see that the Perceptron is less trackable than the 2-way decision tree as the label clutter increases. In this way, we believe that this metric may allow us to more consistently find a simple model than the traditional definition.

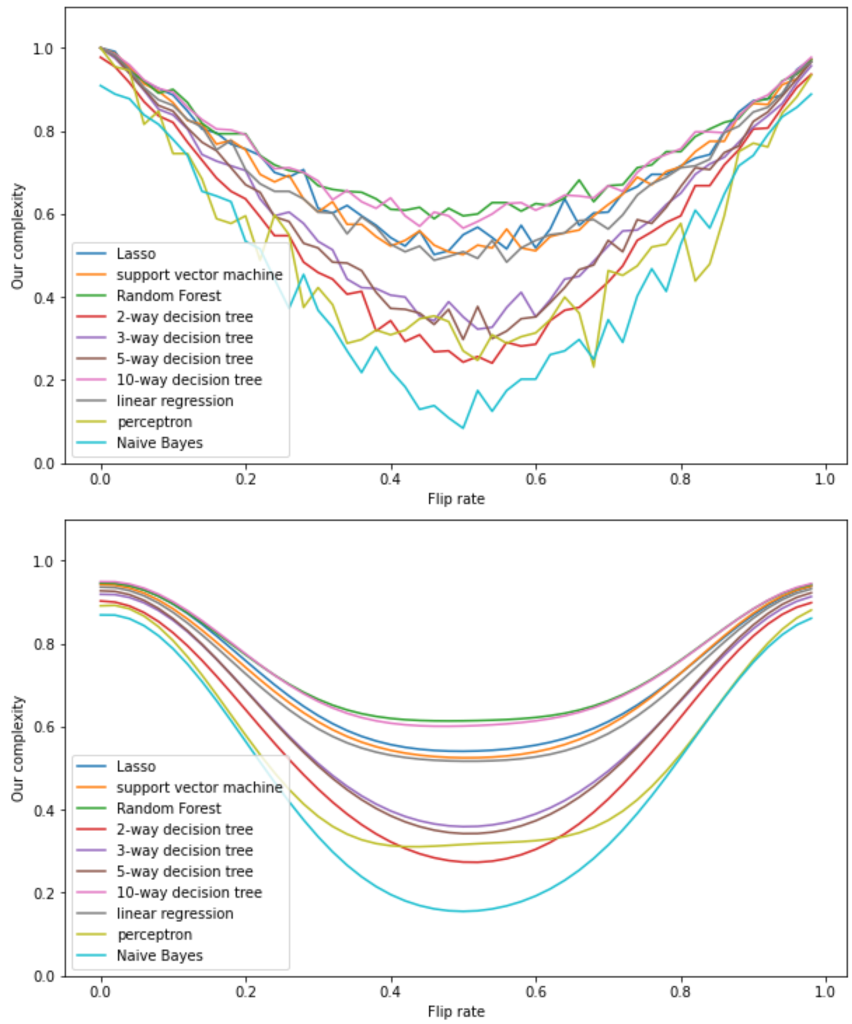


Figure 3: The graphs above are for various models, varying the complexity $\hat{\mathfrak{C}}_{\mathcal{X},m}$ m from 0 to n in the animal data set.The flip rate on the x-axis corresponds to n/m. The graph below is a Fourier inverse transform after removing the high-frequency spectrum obtained by Fourier transform.

8 Increasing Discoverability: Interface Improvements

As discussed in Section 4, the existing interface was confusing to users because of lack of explanation throughout the tool. While additional explanations existed elsewhere on the website, making it difficult to find and not accessible while interacting with the tool.

To address these issues identified from the initial pilot survey, we added additional explanations in the forms of paragraphs explaining the underlying models which were being used. Additionally, we added hover text to the columns to describe exactly what the columns were representing.

We also reformatted the combinations generated in Step 2 so that they are more human-readable. That is, instead of `[feature]_[classification]`, we did `[classification] [feature]`. For example, `Teeth_Has` $\wedge$ `Breathes using lungs_Yes` became `"Has Teeth" and "Breathes using lungs"`.

By making these changes, we believe that users will have an increased discoverability with the tool, as they are now able investigate without going to external sources. We also have ensured that the information in the previous sections also follow these new guidelines to continue to be useful for the users.

9 New Wide Learning Tool

We improved the front-end interface of the system that also incorporates user input. We iterate on the design of the interface using the data collected from the pilot survey as well as our own experiences.

For the development of the interface, we used programming languages Python and Javascript. The back-end of the interface is supported by Flask, and the front-end is supported with React JS. For version control, we used Github to store the up-to-date version of our code; see the repository here.

The models and data are retrieved from the back-end for the user to interact with. The development of the new interface was done on a local server. For instance, every time the page loads, the back-end code will generate the data needed to be displayed to the user, which will be fetched using a 'GET' request in the Application Programming Interfaces (API) call. Once the user inputs a desired feature or uploads a data set, the API would send a 'POST' request to the back-end, which includes the data from user that will get processed. The details of the new interface are outlined in Section 9, which are separated into 8 steps (see Appendix B for demo videos).

The original Wide Learning tool breaks down the process into five steps, as described in Section 3.1. The technique used from step 1 to step 2 in Wide Learning is highly effective and efficient at identifying important combinations of features in the data [13]. Therefore we will continue to use this same technique for our new Wide Learning interface.

The area of the current Wide Learning process that we changed was the process by which weights are assigned to feature combinations between step 2 and step 3.

The existing Wide Learning tool only presents one model to the user that is used to make predictions on the test data. However, many types of models exist that can effectively perform this binary classification, including those described in Section 5. Our new user interface presents the user with different options for models. Allowing the user to have some choice can increase the trustworthiness of the model [2].

The new interface performs the classification task as follows:

1. The user uploads their training data.
2. Important combinations of features are generated and shown to the user, using the same technique from the original Wide Learning tool. The training data is used to create a binary matrix based on these important feature combinations.
3. This binary matrix is used to train all of the Lasso, SVM, Decision Tree, Random Forest, Logistic Regression, Gaussian Naive Bayes, and Perceptron models as discussed in Section 5.
4. Training accuracy and complexity scores are calculated for each model following the metric shown in Section 7.
5. The user is then shown the model with the highest training accuracy. If multiple models have the same training accuracy, the one with the lowest complexity score is shown.
6. After seeing this best model, the user is presented with a list of the other models, along with their corresponding training accuracies and complexity scores. The user can then select one of these models if they are not satisfied with the original model.
7. If the user selects Lasso, they will also be presented with an option to modify the Lasso model by selecting feature combinations to include in the model, as described in Section 6.
   (a) Once the user selects a feature combination, it is inserted into the model and another feature is taken out of the model.
   (b) The updated weights are shown to the user.
   (c) At this point, they can either select another feature combination to add to the model, or accept the current model.
8. Once the user has decided which model they want to use, the testing phase begins. The user uploads their testing data, then predictions are made based on the model that was selected by the user. The user will be able to see the predictions for the other types of models as well.

The videos of the current state of the interface can be found in Appendix B.

We believe that this new interface improves upon the existing Wide Learning tool in all requested aspects from Fujitsu Limited by improving the tool's variety, convincingness, and discoverability. However, we need to assure that these metrics were actually met per their guidelines.

10 New Interface Survey

We used our first pilot survey to get feedback on the existing Wide Learning tool. The survey responses provided valuable information that helped inform the creation of our new Wide Learning user interface. To evaluate the clarity and explainability of this new Wide Learning workflow and user interface, we designed a similar pilot survey for the new tool. The new user interface is more interactive and certain aspects have hover explanations. Therefore, we used video clips that demonstrate how the interface works. In the survey, short video clips are shown for each step that demonstrate what the new Wide Learning tool is doing (see Appendix B for demo videos). After watching the video for each step in our user interface, the survey respondent is asked to describe what they think is happening in that step. They are also asked to describe any aspects that they found to be particularly confusing or clear. Next, they are asked to rate how confident they are in their description on a scale from 1 to 5. At the end of the survey, they are asked to comment on the clarity of the interface as a whole. They are also asked to provide any final comments or suggestions.

10.1 Survey Results

The pilot survey for the new user interface described in Section 10 was distributed to several Fujitsu employees. The survey was anonymized and was expected to take 15 minutes to complete. We received 5 responses. These respondents were already familiar with the existing Wide Learning tool When asked to rate their familiarity with Wide Learning on a scale from 1 to 5 (1 being Never Used it Before and 5 being Expert), two respondents selected 4 and three respondents selected 5. When the respondents were asked to describe what was happening at each step and to comment on aspects that they found confusing or clear, the responses varied from person to person. At each step, some people found that it was clear, while others identified areas that they found confusing. Unlike our original pilot survey, there was much less of a consensus as to which aspects were clear or confusing. However, these survey results still provide valuable feedback that can be used to make further improvements to the new user interface. For the last questions, where respondents were asked to comment on the overall flow of the interface and give any additional feedback. Overall, the feedback was positive and people said that they were able to understand the flow of the user interface.

11 Discussion and Future Work

While this project made several improvements to the existing Wide Learning tool, there are still several directions that could be explored to further improve explainability in aspects such as convincingness, variety, and discoverability. As described in the previous section, the feedback from the new pilot survey can be incorporated into future iterations of the interface. This could include adding additional explanations or making improvements to the existing ones.

The results from the survey on the new interface suggest that certain parts needed better representations, such as the graphs showing the change in complexity values (Figure 3), even more so for non-domain experts. Out of the 5 responses we received, 4 respondents expressed some confusion on how to use the graphs, which suggest that there could be better ways, perhaps simpler or more scaffolded ways, to present the complexity data. One future direction of research could be to explore and evaluate different representations to help with understanding.

In the last step of the new interface, we currently show the predictions from the set of models. Another future direction would be to show the user a more direct comparison of the prediction results from the different models for each testing data point. If the user could see that the majority of the models considered made the same prediction for a given data point, that could increasing the convincingness of those predictions.

While developing and testing our new user interface, we used the datasets provided in the Wide Learning Website (mainly the animal data for mammal classification). For further development of this tool, more datasets should be tested, including datasets not provided by the Wide Learning Website.

As with the nature of development of applications, many iterations are necessary to ensure the quality of the tool. We hope our iteration of the interface can serve as a foundation to future iterations of improvements.

12 Conclusion

We have created a new version of the Wide Learning tool based on hypothesis-driven AI. Rather than presenting the user with one model to perform classification, we train multiple models to present to the user. First, we recommend which model is best, and then present the alternative models. We provide the training accuracy and the Rademacher complexity for each model so the user can make an informed decision. If the user selects the Lasso model, they are able to select feature combinations that they believe are important to include in the model. By giving the user some control in which models are being used, this could increase their trust in the model [2]. We also include additional explanations in our interface that were informed by the results of our initial pilot survey, to increase the explainability of the Wide Learning tool.

References

[1] Q Vera Liao and Kush R Varshney. Human-centered explainable ai (xai): From algorithms to user experiences. *arXiv preprint arXiv:2110.10790*, 2021.

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

[3] Kristopher Kyle and Scott A Crossley. Automatically assessing lexical sophistication: Indices, tools, findings, and application. *Tesol Quarterly*, 49(4):757–786, 2015.

[4] Christoph Molnar, Giuseppe Casalicchio, and Bernd Bischl. Quantifying model complexity via functional decomposition for better post-hoc interpretability. In *Machine Learning and Knowledge Discovery in Databases: International Workshops of ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part I*, pages 193–204. Springer, 2020.

[5] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador Garcia, Sergio Gil-Lopez, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115, 2020.

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

[7] Hiroaki Iwashita, Takuya Takagi, Hirofumi Suzuki, Keisuke Goto, Kotaro Ohori, and Hiroki Arimura. Efficient constrained pattern mining using dynamic item ordering for explainable classification. *arXiv preprint arXiv:2004.08015*, 2020.

[8] Su-In Lee, Honglak Lee, Pieter Abbeel, and Andrew Y Ng. Efficient l~ 1 regularized logistic regression. In *Aaai*, volume 6, pages 401–408, 2006.

[9] Mohammed Yousef Shaheen. Applications of artificial intelligence (ai) in healthcare: A review. *ScienceOpen Preprints*, 2021.

[10] Harry Surden. Artificial intelligence and law: An overview. *Georgia State University Law Review*, 35:19–22, 2019.

[11] Muhammad Nazrul Islam, Toki Tahmid Inan, Suzzana Rafi, Syeda Sabrina Akter, Iqbal H. Sarker, and A. K. M. Najmul Islam. A systematic review on the use of ai and ml for fighting the covid-19 pandemic. *IEEE Transactions on Artificial Intelligence*, 1(3):258–270, 2020.

[12] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: A survey on explainable artificial intelligence (xai). *IEEE Access*, 6:52138–52160, 2018.

[13] Kotaro Ohori, Tatsuya Asai, Hiroaki Iwashita, Keisuke Goto, Junichi Shigezumi, Takuya Takagi, Yuri Nakao, and Hirokazu Anai. Wide learning technology to provide trust through knowledge discovery. *FUJITSU SCIENTIFIC & TECHNICAL JOURNAL*, 56(1):46–51, 2020.

[14] Sheikh Rabiul Islam, William Eberle, Sheikh Khaled Ghafoor, and Mohiuddin Ahmed. Explainable artificial intelligence approaches: A survey. *arXiv preprint arXiv:2101.09429*, 2021.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[16] Lior Rokach and Oded Maimon. Top-down induction of decision trees classifiers-a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 35(4):476–487, 2005.

[17] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[18] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20:273–297, 1995.

[19] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. *Applied logistic regression*, volume 398. John Wiley & Sons, 2013.

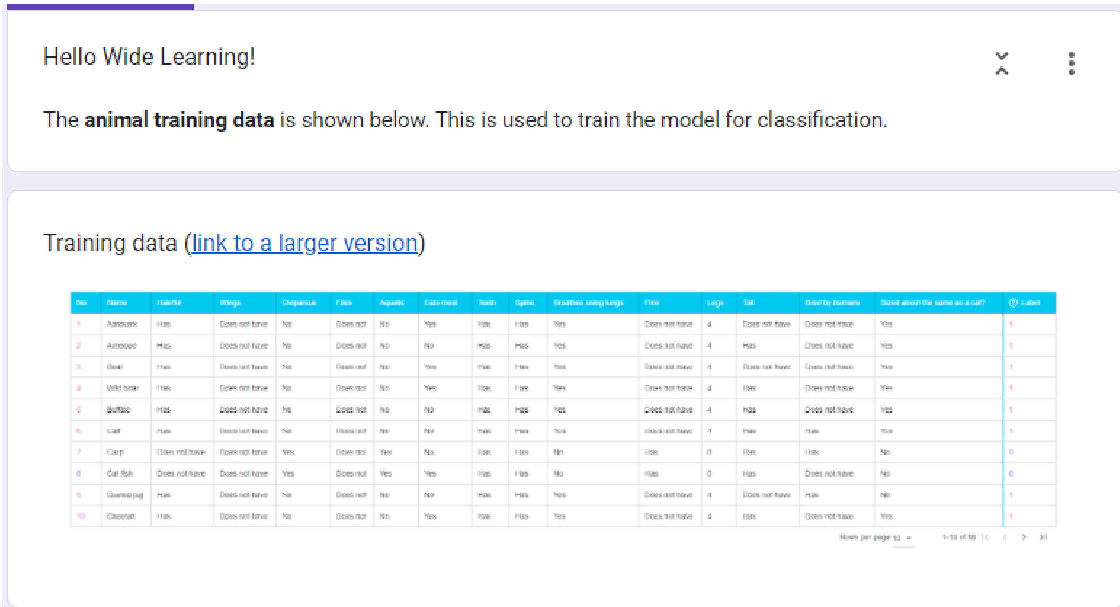
[20] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[21] Andrew Ng and Michael Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems*, 14, 2001.

[22] Takafumi Kanamori. Statistical learning theory. 2015.

[23] Tim Miller. Explainable ai is dead, long live explainable ai! hypothesis-driven decision support using evaluative ai. In *Proceedings of the 2023 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '23, page 333‐342, New York, NY, USA, 2023. Association for Computing Machinery.

A Pilot Survey Visualization



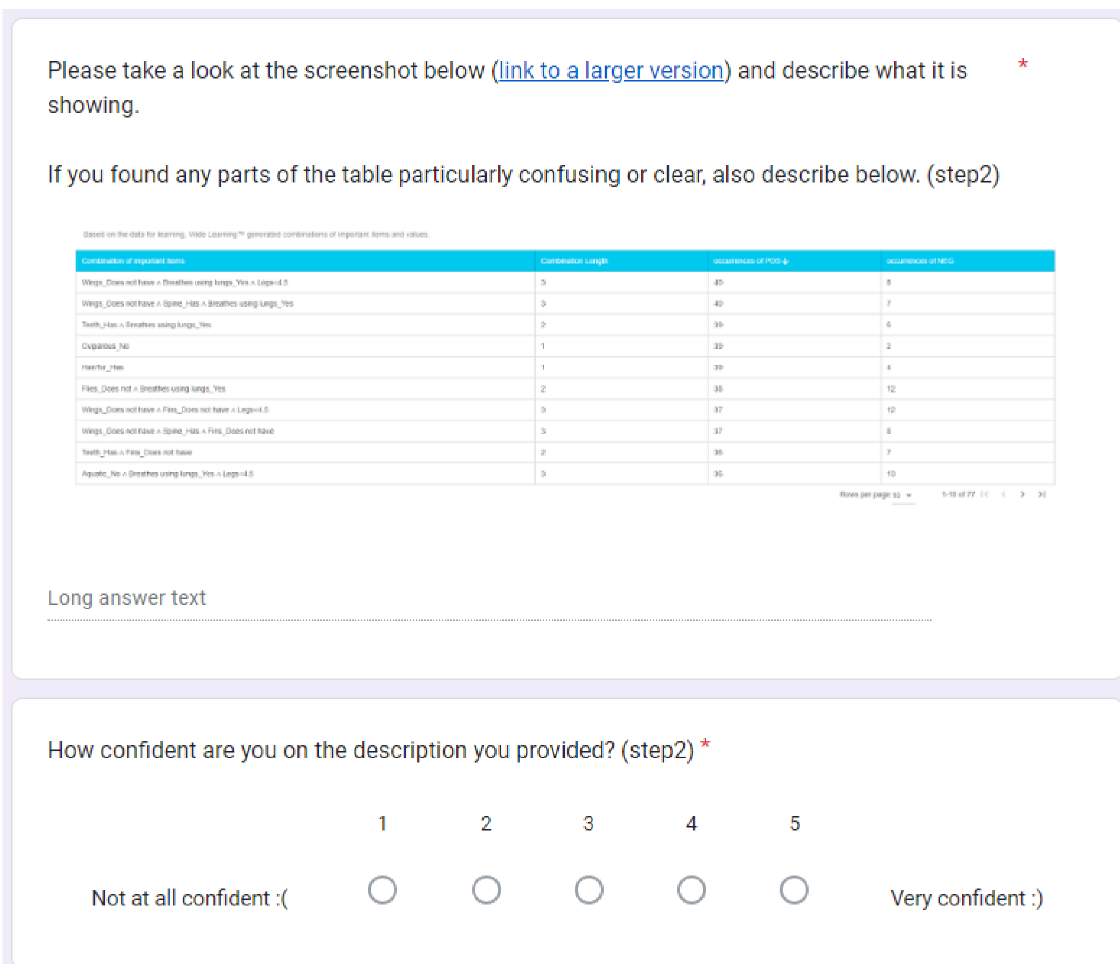
Hello Wide Learning!

The **animal training data** is shown below. This is used to train the model for classification.

Training data (link to a larger version)

No	Name	Hair/fur	Wings	Oviparous	Flies	Aquatic	Eats meat	Teeth	Spine	Breathes using lungs	Fins	Legs	Tail	Bred by humans	Sized about the same as a cat?	Label
1	Aardvark	Has	Does not have	No	Does not	No	Yes	Has	Has	Yes	Does not have	4	Does not have	Does not have	Yes	1
2	Antelope	Has	Does not have	No	Does not	No	No	Has	Has	Yes	Does not have	4	Has	Does not have	Yes	1
3	Bear	Has	Does not have	No	Does not	No	Yes	Has	Has	Yes	Does not have	4	Does not have	Does not have	Yes	1
4	Wild boar	Has	Does not have	No	Does not	No	Yes	Has	Has	Yes	Does not have	4	Has	Does not have	Yes	1
5	Buffalo	Has	Does not have	No	Does not	No	No	Has	Has	Yes	Does not have	4	Has	Does not have	Yes	1
6	Calf	Has	Does not have	No	Does not	No	No	Has	Has	Yes	Does not have	4	Has	Has	Yes	1
7	Carp	Does not have	Does not have	Yes	Does not	Yes	No	Has	Has	No	Has	0	Has	Has	No	0
8	Cat fish	Does not have	Does not have	Yes	Does not	Yes	Yes	Has	Has	No	Has	0	Has	Does not have	No	0
9	Guinea pig	Has	Does not have	No	Does not	No	No	Has	Has	Yes	Does not have	4	Does not have	Has	No	1
10	Cheetah	Has	Does not have	No	Does not	No	Yes	Has	Has	Yes	Does not have	4	Has	Does not have	Yes	1

Rows per page 10 1-10 of 88



Please take a look at the screenshot below (link to a larger version) and describe what it is showing. *

If you found any parts of the table particularly confusing or clear, also describe below. (step2)

Based on the data for learning, Wide Learning™ generated combinations of important items and values.

Combination of important items	Combination Length	occurrences of POS ↓	occurrences of NEG
Wings_Does not have ∧ Breathes using lungs_Yes ∧ Legs<4.5	3	40	8
Wings_Does not have ∧ Spine_Has ∧ Breathes using lungs_Yes	3	40	7
Teeth_Has ∧ Breathes using lungs_Yes	2	39	6
Oviparous_No	1	39	2
Hair/fur_Has	1	39	4
Flies_Does not ∧ Breathes using lungs_Yes	2	38	12
Wings_Does not have ∧ Fins_Does not have ∧ Legs<4.5	3	37	12
Wings_Does not have ∧ Spine_Has ∧ Fins_Does not have	3	37	8
Teeth_Has ∧ Fins_Does not have	2	36	7
Aquatic_No ∧ Breathes using lungs_Yes ∧ Legs<4.5	3	35	10

Rows per page 10 1-10 of 77

Long answer text

How confident are you on the description you provided? (step2) *

	1	2	3	4	5	
Not at all confident :(	○	○	○	○	○	Very confident :)

Here is our summary of Step 2

Wide Learning generates all possible combinations of variables. The combinations in this table are the most important combinations of features to determine whether or not an animal is a mammal. The importance of a combination is measured by the length of the combination and the number of positive and negative occurrences for the animals in the training dataset that have that combination of features. This table (link to a larger version) only shows feature combinations with a maximum length of three, however, important feature combinations that are longer than three may exist.

Step 2

Based on the data for learning, Wide Learning™ generated combinations of important items and values.

Combination of important items	Combination Length	occurrences of POS↓	occurrences of NEG
Wings_Does not have ∧ Breathes using lungs_Yes ∧ Legs<4.5	3	40	8
Wings_Does not have ∧ Spine_Has ∧ Breathes using lungs_Yes	3	40	7
Teeth_Has ∧ Breathes using lungs_Yes	2	39	6
Oviparous_No	1	39	2
Hair/fur_Has	1	39	4
Flies_Does not ∧ Breathes using lungs_Yes	2	38	12
Wings_Does not have ∧ Fins_Does not have ∧ Legs<4.5	3	37	12
Wings_Does not have ∧ Spine_Has ∧ Fins_Does not have	3	37	8
Teeth_Has ∧ Fins_Does not have	2	36	7
Aquatic_No ∧ Breathes using lungs_Yes ∧ Legs<4.5	3	36	13

Rows per page 10 1-10 of 77

If your understanding did not align with our summary, describe what information would have been helpful to better understand this step. *

Type "N/A" if your understanding did align with our summary. (step2)

Long answer text

Please take a look at the screenshots below (link to a larger version) and describe what it is showing. *

If you found any parts of the table or chart particularly confusing or clear, also describe below. (step3-1)

Long answer text

How confident are you on the description you provided? (step3-1) *

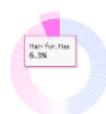
	1	2	3	4	5	
Not at all confident :(	○	○	○	○	○	Very confident :)

Here is our summary of Step 3

From the results of Step 2, important feature combinations are listed, and Wide Learning assigns a weight to each of the listed combinations. Combinations with positive weights (link to a larger positive version) are important for identifying the positive class (mammal), and combinations with negative weights (link to a larger negative version) are important for identifying the negative class (not a mammal). The pie chart shows the relative sizes of each of the weights.

Step 3 (positive)

The importance (weight) is calculated to indicate how important
the "combinations of important items" that occurred in Step 2 is in determining whether the data is 0 (NEG) or 1 (POS).



POS NEG

Combination of important items	Importance (weight) ↓
Hair/fur_Has	0.91
Oviparous_No	0.88
Sized about the same as a cat?_Yes	0.47
Wings_Does not have ∧ Spine_Has ∧ Breathes using lungs_Yes	0.38
Wings_Does not have ∧ Breathes using lungs_Yes ∧ Legs<4.5	0.36
Teeth_Has ∧ Breathes using lungs_Yes	0.35
Breathes using lungs_Yes ∧ Fins_Has	0.29
Eats meat_No ∧ Teeth_Has	0.29
Wings_Does not have ∧ Breathes using lungs_Yes ∧ Legs<3.0	0.25
Wings_Does not have ∧ Aquatic_No ∧ Legs<4.5	0.24

Rows per page: 10 1-10 of 50

Step 4

Here is the test data (link to a larger version) with new animals that we are going to classify.
Click "Next" to see the results.

Step 4

No.	Name	Hair/fur	Wings	Oviparous	Flies	Aquatic	Eats meat	Teeth	Spine	Breathes using lungs	Fins	Legs	Tail	Bred by humans	Sized about the same as a cat?
1	Vulture	Does not have	Has	Yes	Does	No	Yes	Does not have	Has	Yes	Does not have	2	Has	Does not have	Yes
2	Dolphi	Does not have	Does not have	No	Does not	Yes	Yes	Has	Has	Yes	Has	0	Has	Does not have	Yes
3	Penguin	Does not have	Has	Yes	Does not	Yes	Yes	Does not have	Has	Yes	Does not have	2	Has	Does not have	Yes
4	Platypus	Has	Does not have	Yes	Does not	Yes	Yes	Does not have	Has	Yes	Does not have	4	Has	Does not have	Yes
5	Worm	Does not have	Does not have	Yes	Does not	No	No	Does not have	Does not have	Yes	Does not have	0	Does not have	Does not have	No

Rows per page: 10 1-5 of 5

Please take a look at the screenshots below (link to a larger version) and describe what it is showing. *

If you found any parts of the table or chart particularly confusing or clear, also describe below. (step5-1)

Long answer text

How confident are you on the description you provided? (step5-1) *

	1	2	3	4	5	
Not at all confident :(	○	○	○	○	○	Very confident :)

Please take a look at the screenshots below (link to a larger version) and describe what it is showing. *

If you found any parts of the table or chart particularly confusing or clear, also describe below. (step5-2)

Long answer text

How confident are you on the description you provided? (step5-2) *

	1	2	3	4	5	
Not at all confident :(	○	○	○	○	○	Very confident :)

Here is our summary of Step 5

The prediction score for each animal on the left table in the screenshots below is the probability that the animal belongs to the positive class (mammal). This is calculated using the weights from Step 3. These two screenshots show the details for the prediction for a dolphin. The table on the right in the first screenshot (positive, [link to a larger version]) shows the list of important positive feature combinations that a dolphin has, along with their corresponding weights. The table on the right in the second screenshot (negative, [link to a larger version]) shows the list of important negative feature combinations that a dolphin has, along with their corresponding weights. The pie chart shows the relative sizes of each of the weights for a dolphin.

Step 5 (positive)

B Our New Interface Workflow

Step 1: Upload Training Data
Step 2: Get Important Combinations
Step 3: Learned Weights
Step 4 (part 1): Set of Possible Models
Step 4 (part 2): Set of Possible Models (Results)
Step 5 and 6: User select feature and display weights
Step 7: Upload Testing Data
Step 8: Predictions from the Possible Models