R2-Learn V2.0 User Manual

# R2-Learn V2.0 User Manual

19 October 2018

Table of Contents

* [1. Overview](#_overview)
  + [1.1. Machine learning](#_machine_learning)
  + [1.2. Machine learning with R2-Learn](#_machine_learning_with_r2_learn)
* [2. Getting started with R2-Learn](#_getting_started_with_r2_learn)
  + [2.1. Software requirements](#_software_requirements)
  + [2.2. Importing data into R2-Learn](#_importing_data_into_r2_learn)
    - [2.2.1. Importing data from a database](#_importing_data_from_a_database)
    - [2.2.2. Importing a local file](#_importing_a_local_file)
  + [2.3. Project home](#_project_home)
  + [2.4. Model deployment home page](#_model_deployment_home_page)
* [3. Starting a new project](#_starting_a_new_project)
  + [3.1. Create a project](#_create_a_project)
  + [3.2. Describe your business problem](#business-problem)
  + [3.3. Working with your data](#_working_with_your_data)
    - [3.3.1. Data Connect](#_uploading_your_data)
    - [3.3.2. Data Schema](#_cleaning_your_data)
    - [3.3.3. Data Quality](#_data_quality)
      * [Common Issues with Data](#_common_issues_with_data)
      * [Target Variable Quality](#_target_variable_quality)
      * [All Data Quality](#_all_data_quality)
  + [3.4. Modeling](#_modeling)
    - [3.4.1. Automatic Modeling](#_automatic_modeling)
    - [3.4.2. Advanced Modeling](#_advanced_modeling)
    - [3.4.3. Building your model](#_building_your_model)
      * [Model Selection](#_model_selection)
* [4. Deploying your models](#_deploying_your_models)
  + [4.1. Deployment](#_deployment)
    - [4.1.1. Predict with data source](#_predict_with_data_source)
    - [4.1.2. Predict with API](#_predict_with_api)
  + [4.2. Monitor your deployed models](#_monitor_your_deployed_models)
    - [4.2.1. Operation Monitor](#_operation_monitor)
    - 4.2.2. Performance Monitor
    - [4.2.3. Performance Status](#_performance_status)
* [Appendix A: Data Quality Fixes](#appendix-data-quality-fixes)
  + [A.1. Fixing outliers](#_fixing_outliers)
  + [A.2. Fixing missing values](#_fixing_missing_values)
  + A.3. Fixing data type mismatch
* [Appendix B: Advanced Modeling](#appendix-advanced-modeling)
  + [B.1. Advanced Variable Settings](#_advanced_variable_settings)
  + [B.2. Advanced Modeling Setting](#_advanced_modeling_setting)
    - [B.2.1. Create/Edit Model Setting By default](#_createedit_model_setting_by_default)
    - [B.2.2. Select Algorithm](#_select_algorithm)
    - [B.2.3. Set Max Model Ensemble Size](#_set_max_model_ensemble_size)
    - [B.2.4. Train Validation Holdout and Cross Validation](#_train_validation_holdout_and_cross_validation)
    - [B.2.5. Resampling Setting](#_resampling_setting)
    - [B.2.6. Set Measurement Metric](#_set_measurement_metric)
    - [B.2.7. Set Max Optimization Time](#_set_max_optimization_time)
    - [B.2.8. Random Seed](#_random_seed)
* [Appendix C: Model selection for Binary Classification problems](#_model_selection_for_binary_classification_problems)
  + [C.1. Simplified view](#_simplified_view)
  + [C.2. Advanced View](#_advanced_view)
    - [C.2.1. Top Section](#_top_section)
    - [C.2.2. Table of Models](#_table_of_models)
    - [C.2.3. Additional Model Details](#_additional_model_details)
* [Appendix D: Model selection for Regression problems](#_model_selection_for_regression_problems)
  + [D.1. Simplified View](#_simplified_view_2)
  + [D.2. Advanced View](#_advanced_view_2)
    - [D.2.1. Top Section](#_top_section_2)
    - [D.2.2. Table of Models](#_table_of_models_2)
    - [D.2.3. Additional Model Details](#_additional_model_details_2)

## [1. Overview](#_overview)

R2-Learn helps companies turn data into machine learning models quickly without needing AI (artificial intelligence) expertise. Built with cutting edge technology, R2-Learn uses AI to guide you through the process of creating, deploying, and keeping up-to-date custom machine learning models using data that you already have.

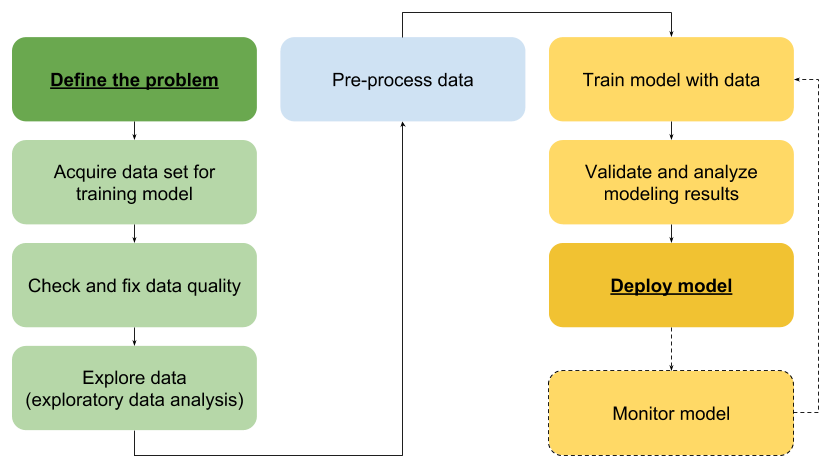
This section introduces the key concepts of machine learning and how they relate to R2-Learn.

### [1.1. Machine learning](#_machine_learning)

Machine learning is a branch of AI where computers are trained to learn from data using complex learning algorithms. With ever increasing computing power, machine learning is able to discover patterns and relations hidden in massive amounts of data that would be difficult for humans to identify and use. Machine learning models are being developed today to classify objects, detect anomalies, predict outcomes, and augment human capacity in general.

Businesses that use machine learning to help them make business decisions with data usually employ a machine learning workflow similar to the following:

1. **Define the problem**: Decide what problem you want to solve with data.
2. **Acquire dataset for the training model**: Get the data that you will use to train your machine learning models.
3. **Check and fix data quality**: Inspect the dataset for missing data and inconsistencies, and fix the dataset where appropriate to improve its quality.
4. **Explore data**: Explore your data to get a sense of what shape the data is, and identify significant patterns for further analysis.
5. **Pre-process data**: Modify the shape of your dataset so that it fits the parameters of the machine learning tools that you’ll be using to generate your model.
6. **Train model with data**: Feed your data into the machine learning tools that will generate your machine learning model for you.
7. **Validate and analyze modeling results**: Check that resulting model is able to make the predictions that you expect.
8. **Deploy model**: Deploying your model generally means making your model available for use. Usually, this takes the form of deploying the model as a REST API endpoint that you can send new input data to and receive a response containing the resulting prediction.
9. **Monitor model**: Once you have your machine learning model, it is important to monitor its performance to make sure that your predictions are always within an acceptable performance range. Machine learning models suffer from model drift over time, meaning that the predictions made with a machine learning model will become less accurate as it ages.

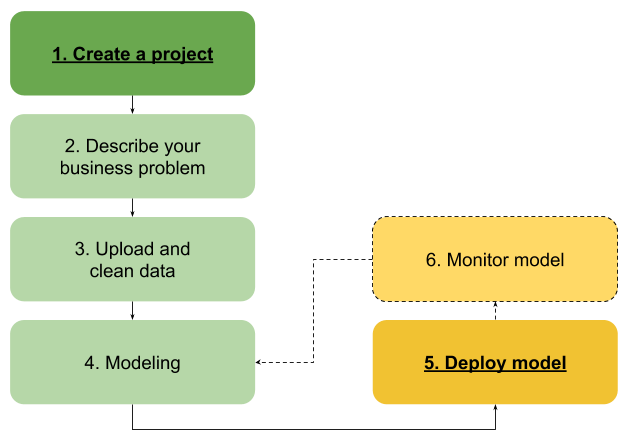


A well-trained data scientist is usually responsible for the workflow illustrated above. It has many steps and decision points. Making the correct decision at each point is critical to the quality of the machine learning model, which in turn determines how accurate predictions made with this model are. Data scientists today largely rely on their experience and intuition to find the best model for the task at hand, which can be very slow, tedious, expensive and flawed process.

### [1.2. Machine learning with R2-Learn](#_machine_learning_with_r2_learn)

R2-Learn makes it easier for businesses to work with machine learning by using AI to automate large portions of the standard machine learning workflow, reducing the amount of work and human-error that usually comes with the process.

The R2-Learn machine learning workflow is as follows:



1. [**Create a project**](#_create_a_project): Start a project in R2-Learn to begin building a machine learning model.
2. [**Describe your business problem**](#business-problem): Describe the business problem that you’re trying to solve. This helps document the intent of the project and its desired outcome.
3. [**Working with your data**](#_working_with_your_data): Upload your training dataset to R2-Learn. R2-Learn uses machine learning to assist you in checking and cleaning your dataset.
4. [**Modeling**](#_modeling): Once your dataset has been loaded into R2-Learn, you can select one of the following modeling modes to use:
   * [**Automatic Modeling**](#_automatic_modeling): R2-Learn automatically creates machine learning models based on the training data you’ve uploaded.
   * [**Advanced Modeling**](#_advanced_modeling): Advanced modeling allows users to override some default decisions made by R2-Learn and derive new variables to shape the resulting model such that it can better meet stated business goals.
5. [**Deployment**](#_deployment): Once you’ve created your machine learning model, R2-Learn automatically deploys the model on its servers and allows you to:
   * [**Predict with data source**](#_predict_with_data_source): Connect R2-Learn to your database or upload a CSV file containing your input data to run predictions on this data.
   * [**Predict with API**](#_predict_with_api): Use R2-Learn’s REST API to make predictions on your input data with your R2-Learn machine learning models.

Once deployed, you can monitor your model’s predictive performance and update models when they fall below a specified threshold.

## [2. Getting started with R2-Learn](#_getting_started_with_r2_learn)

This section of the manual helps you get started with your first R2-Learn project.

To get started with your first project, head on over to [Starting a new project](#_starting_a_new_project).

### [2.1. Software requirements](#_software_requirements)

R2-Learn is a web application that can be run in your browser.

We recommend that you use:

* Google Chrome version 65 and newer.

### [2.2. Importing data into R2-Learn](#_importing_data_into_r2_learn)

You can import data into R2-Learn for:

* [Building machine learning models](#_working_with_your_data), and
* [Making predictions](#_deployment) on the imported data.

R2-Learn supports the following methods for importing data:

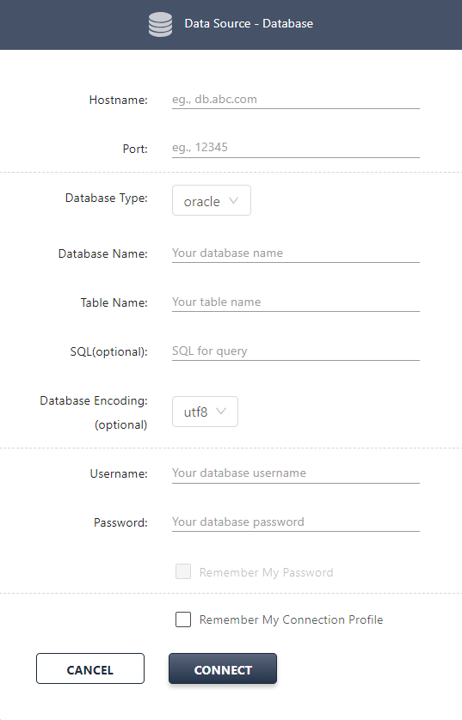
* [Importing data from a database](#_importing_data_from_a_database)
* [Importing a local file](#_importing_a_local_file)

#### [2.2.1. Importing data from a database](#_importing_data_from_a_database)

R2-Learn supports importing data from the following databases:

* Oracle Database 11G
* MySQL

Your database must support ODBC 11.2 connections.



#### [2.2.2. Importing a local file](#_importing_a_local_file)

You can import a UTF-8 encoded CSV file containing your training or input data into R2-Learn.

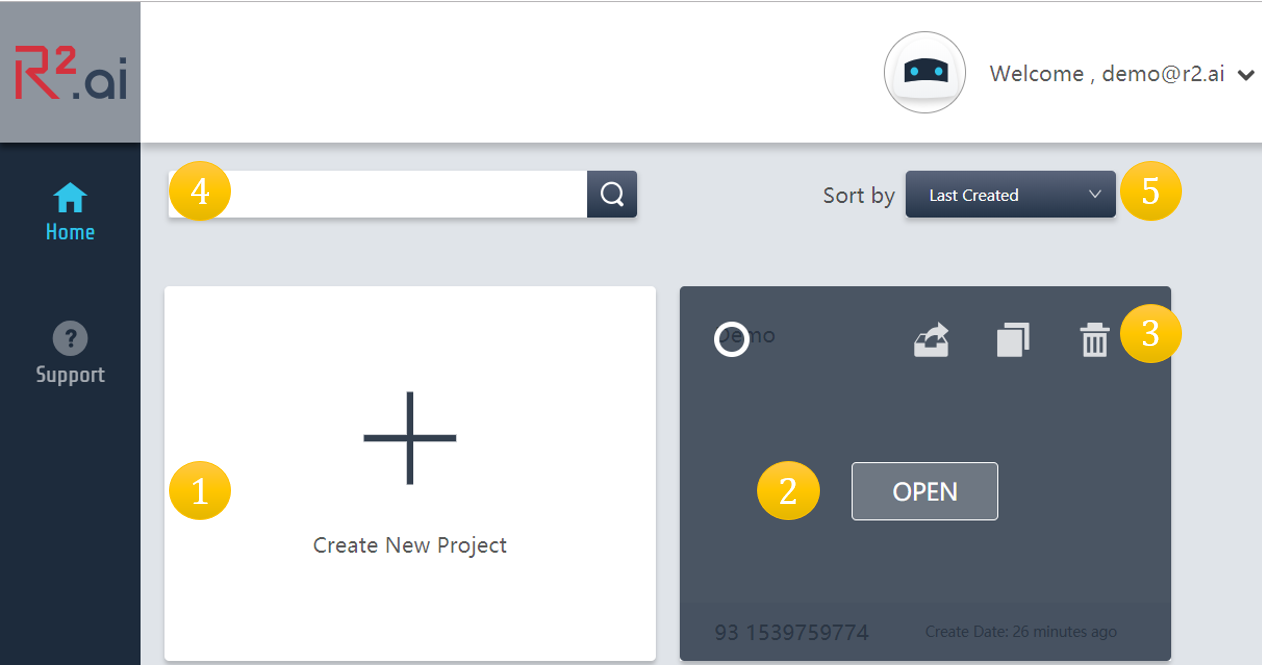
Your CSV file must:

* Have a header row.
* Be UTF-8 encoded.

|  |  |
| --- | --- |
| Note | If you are importing a CSV file to be used for [model deployment](#_deployment) or to make predictions on, that CSV file must fulfill the following:   * It must contain all the variables required by the deployed model, and * Names of variables contained in the CSV file must be the same as the variable names that are used in the training dataset. |

### [2.3. Project home](#_project_home)

When you sign in to R2-Learn, you’ll land on the project home page:



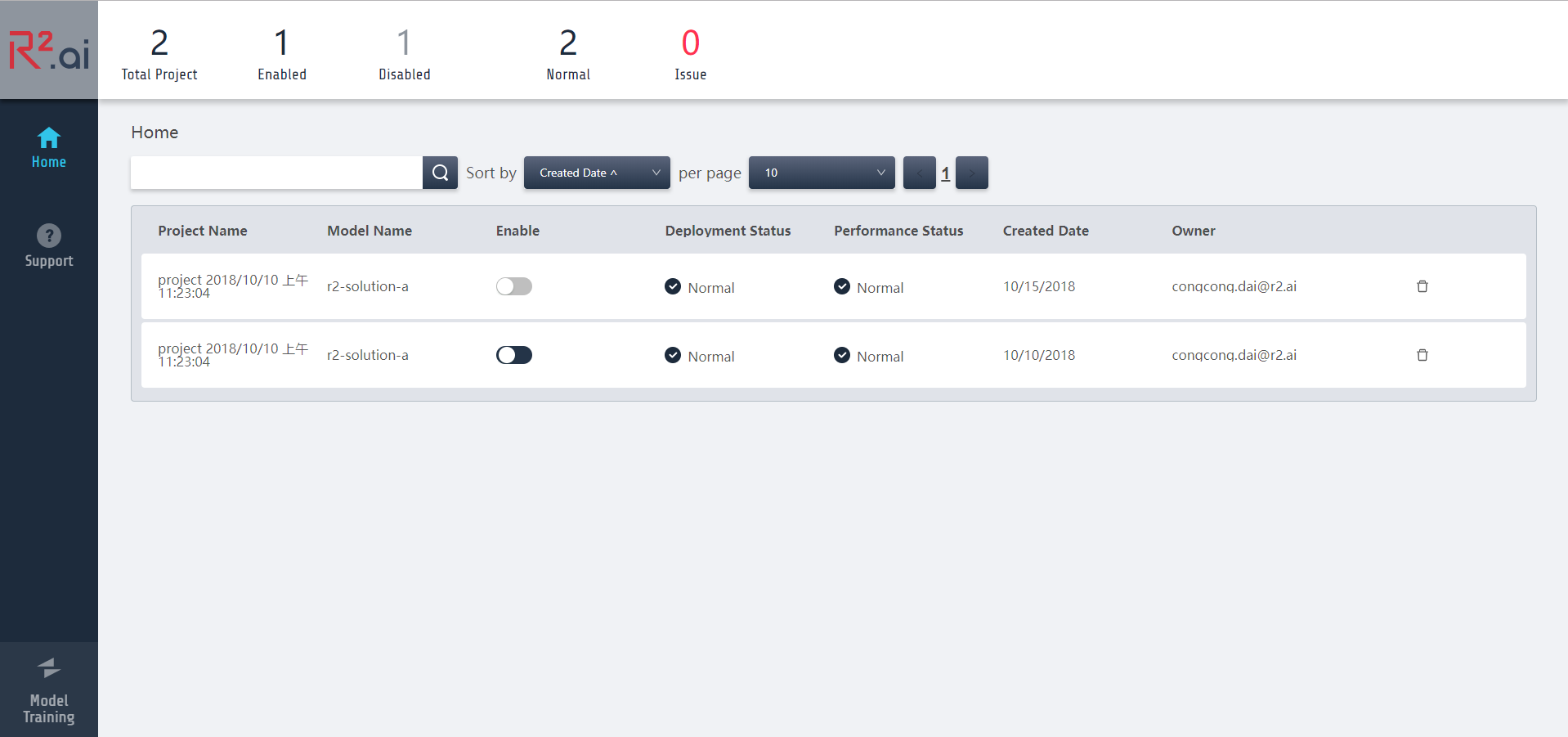
On the project home page, you can:

1. **Create** new projects.
2. **Open** and **edit** an existing project.
3. **Delete** a project.
4. **Search** for a project.
5. **Sort** projects.

Click [**Deploy**](#_model_deployment_home_page) in the navigation menu at the bottom of the side bar to go to the model deployment home page.

### [2.4. Model deployment home page](#_model_deployment_home_page)

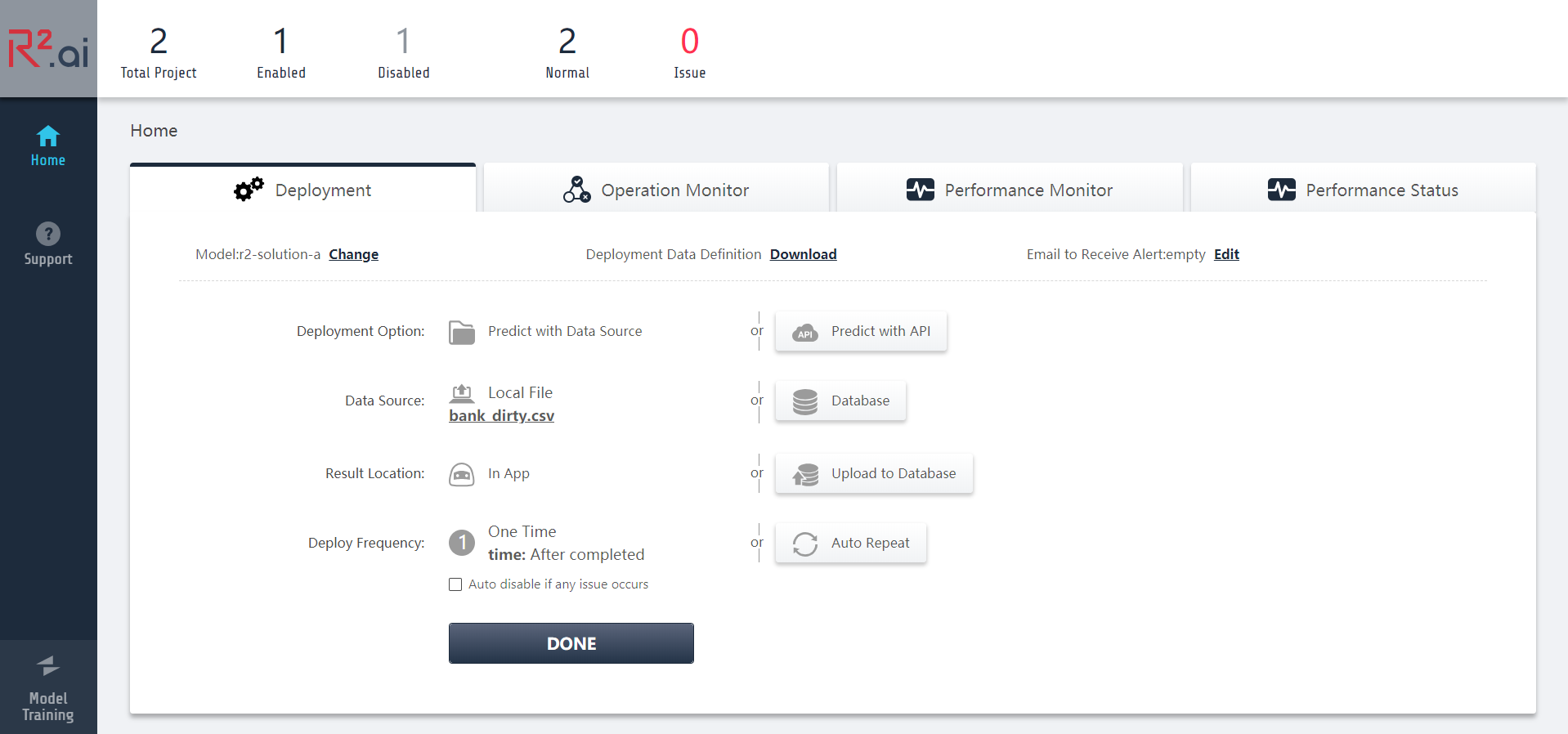
The model deployment home page shows all the projects you’ve created, and the deployment/validation status of these projects:



**Deployment/validation status**: Shows the current status of a deployment or validation task. Possible values are:

* **Running**: A task is currently running.
* **Idle**: Project has no tasks that are running.
* **Issue**: Project has run into an issue while running a task. Please create a new case.
* **Canceled**: Case under a project is canceled.

Click on a project to open it:



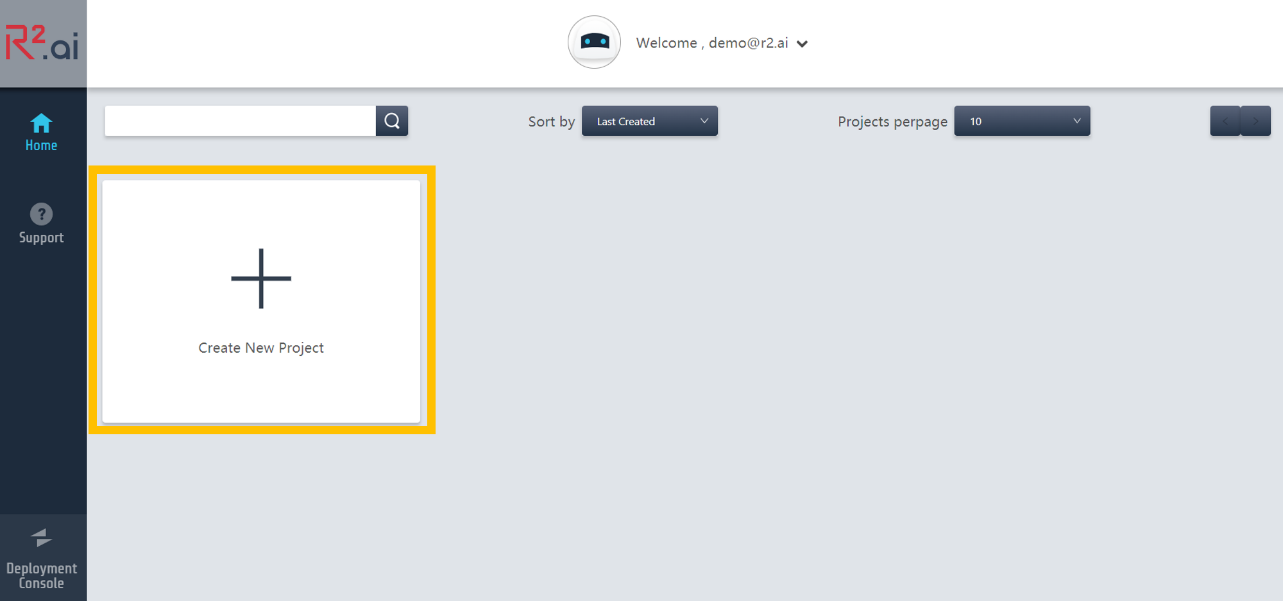
For more information on how to deploy models, see [Deploying your models](#_deploying_your_models).

## [3. Starting a new project](#_starting_a_new_project)

This section will walk you through how to start a new machine learning project, using R2-Learn’s automatic modeling features.

### [3.1. Create a project](#_create_a_project)

When you first sign in on R2-Learn, you’ll be greeted with an empty project home page. Click on **Create New Project** to start a new project.



You’ll be brought to the **Project** section, where you’ll be asked to:

* **Name your project.** Each project is given a default name of project <DD/MM/YYY, HH:MM:SS>. Modify the project name to suit your needs.
* **Describe your project (optional).** Enter a general project description. This helps document the high-level intent and scope of the project for future reference. Low-level specifics of the project should be documented as the [**Problem Statement**](#business-problem) in the **Business Problem** section that follows.

Once done, click **Continue**.

### [3.2. Describe your business problem](#business-problem)

In the **Problem** section, you’ll be entering your project’s **Problem Description** and **Problem Type**.

The project’s **Problem Description** (optional) asks you for a **Problem Statement** and a **Business Value** to be documented. Both of these are not used directly to building your machine learning model, but are important for keeping track of projects and making sure that business units that eventually take over the projects are made aware of the kinds of problems that this project’s models are intended to solve.

The **Problem Type** asks you to specify the type of predictions you expect the project’s models to come up with:

* **True or False (Binary Classification)**: The project’s models should predict if an event is likely to happen or not. For example, if a customer will purchase a product or not; or if someone is likely to develop Type 2 diabetes or not.
* **Continuous Values (Regression)**: To predict a continuous/numeric value. For example, how much a product would cost given a set of variables.

Once you’ve set your **Business Problem** and your **Problem Type**, click **Continue** to move on to [loading your data](#_working_with_your_data) into R2-Learn.

### [3.3. Working with your data](#_working_with_your_data)

Now, we’ll start working with your data. The **Data** section walks you through:

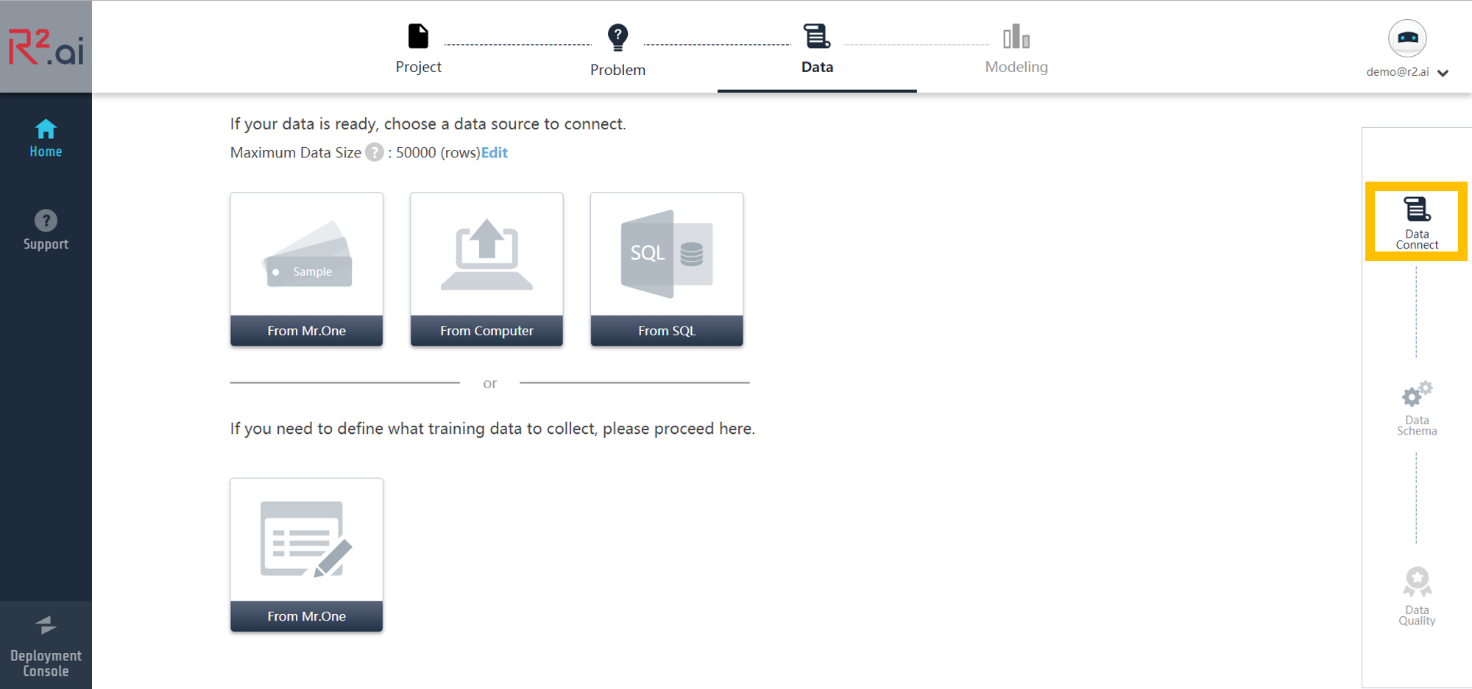
1. [Uploading your data](#_uploading_your_data) in the **Data Connect** tab.
2. [Cleaning your data](#_cleaning_your_data) in the **Data Schema** tab
3. Control your data quality **in the Data Quality tab**

Your training dataset needs to contain as much information as possible that is related to the study goal. Good datasets are thorough, focused on the study goal, and contain as large a sample size as possible. If possible, have a domain expert verify that your training data is suitable for the [business problem](#Describe%20your%20business%0Aproblem). Once you have a good dataset, R2-Learn takes care of the rest.

Once the user loads data via R2-Learn’s data connecting interface, R2-Learn detects the data types, checks the data quality and fixes the minor issues automatically or prompts the user to fix them manually.

#### [3.3.1. Data Connect](#_uploading_your_data)

The **Data Connect** tab allows you to select what dataset to use for your machine learning model.



Here, you’re loading your training dataset into R2-Learn. You can either:

* Connect a [database](#_importing_data_from_a_database), or
* Upload a [local file](#_importing_a_local_file).

If you’re just getting to know R2-Learn, you can use one of the sample datasets provided instead.

After loading your dataset, you can **change the maximum number of rows** or records that R2-Learn will use to develop your machine learning model.

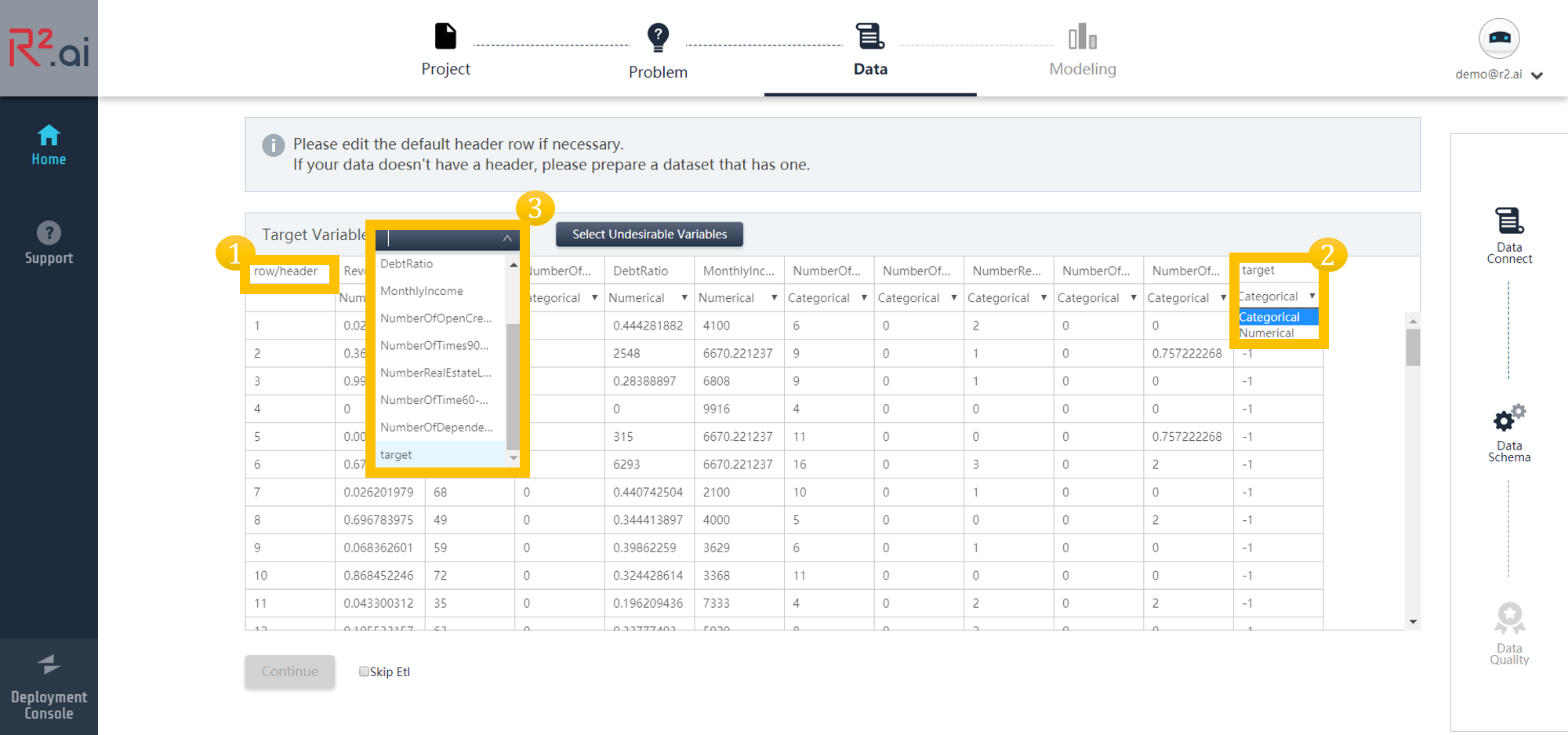
|  |  |
| --- | --- |
| Important | The dataset that you load into R2-Learn must have a header row. |

#### [3.3.2. Data Schema](#_cleaning_your_data)

Once you’ve loaded your training dataset, R2-Learn displays a sample of your data in the **Data Schema** tab for you to inspect. Here, you can:

* **1. Edit your header row** if necessary.
* **2. Verify the Data Type** assigned to each column in your dataset
* **3. Select Your Target Variable**

|  |  |
| --- | --- |
| Important | You must make sure that **all** column headers each have a **unique value** before R2-Learn will allow you to proceed. |



When you’ve finished verifying and editing your header row values, R2-Learn asks you to inspect and verify the **Data Type** assigned to each column in your dataset. If the data type for a column is incorrect, select the correct data type from the dropdown menu for that column to change it.

There are two possible data types for each column:

* **Numerical**: Data in column should be treated as a series of numerical values.
* **Categorical**: Data in column should be treated as distinct categories.

|  |  |
| --- | --- |
| Important | Accurately identifying the data type contained in each column will significantly affect your machine learning model. Consult the owner of the dataset if you are unsure how to classify the type of data the columns contain. |

R2-Learn attempts to automatically identify the data types of each column depending on where you’ve loaded your data from:

* If you’ve loaded your dataset by [connecting to a database](#_importing_data_from_a_database), data type information for each column is captured from the database and displayed here.
* If you’ve loaded your dataset by [uploading a local file](#Importing%20data%20from%20a%20local%20file), data type information is automatically inferred for each column and displayed here.

When you’ve finished verifying and editing your column data types, select **Target Variable** that we want our machine learning model to predict. The target variable should contain a data type that matches the [**Problem Type**](#business-problem-type) you’ve set when determining your [Business Problem](#business-problem).

Click on the **Target Variable** dropdown menu and select a variable or column header to set as the target variable for your machine learning model.

You can also click on **Select Undesirable Variables** and remove columns or variables from the dataset used to train your machine learning model. You should consider removing variables that have a fixed one-to-one mapping to the target variable, such as:

* ID numbers.
* Names of people.
* Product names.
* Variables directly derived from the target variable.
* Variables that the target variable is directly derived from.

For example, in a dataset for training a model to predict the yearly income of a person, we can safely remove a variable that contains the monthly income of a person. Columns are dimmed in the main display when removed from your model.

When you’ve selected your target variable and removed any unwanted columns or variables from your dataset, click **Continue** to move on to [Data Quality](#_data_quality).

#### [3.3.3. Data Quality](#_data_quality)

When you get to the **Data Quality** tab, R2-Learn attempts to detect possible issues that would prevent accurate modeling. It checks for data quality issues at two levels:

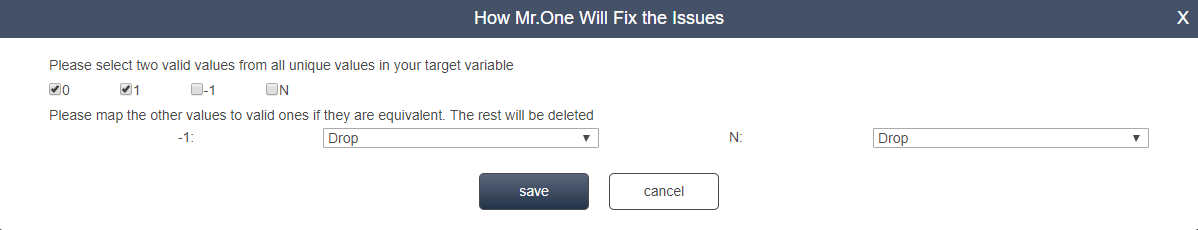
* 1. [Target Variable Quality](#_target_variable_quality): Here, R2-Learn analyzes and flags any issues that it encounters with your target variable.
* 2. [All Data Quality](#_all_data_quality): After R2-Learn has resolved issues with your target variable, it then examines the rest of your dataset for any possible issues.

图片包含 屏幕截图

已生成极高可信度的说明

##### [Common Issues with Data](#_common_issues_with_data)

Two possible issue types that R2-Learn identifies and fixes are:

* **Your Target Variable has more than 2 Unique Values**: When working with a True or False (Binary Classification) [business problem type](#business-problem-type), your target variable must have only 2 different unique values (typically, "true" and "false").
* 
* **Data Type Mismatch**: When a variable’s data type doesn’t match what R2-Learn expects, it issues a **Data Type Mismatch** warning. For example, R2-Learn issues a **Data Type Mismatch** warning if it thinks that a variable should be a numerical data type but contains text values instead.
* **Missing Value**: Missing values in your dataset can be either due to an error in data collection or that a data point contains a null value.

|  |  |
| --- | --- |
| * Note | * A null value represents an absence of data ("no value"), and is different from "zero value". |

* **Outlier**: When working with continuous values, a data point is considered an outlier if is outside the expected range for the given dataset. It is important to inspect data points that are outliers, and determine if they are genuine deviations from the trend shown in the data, or bad data points.

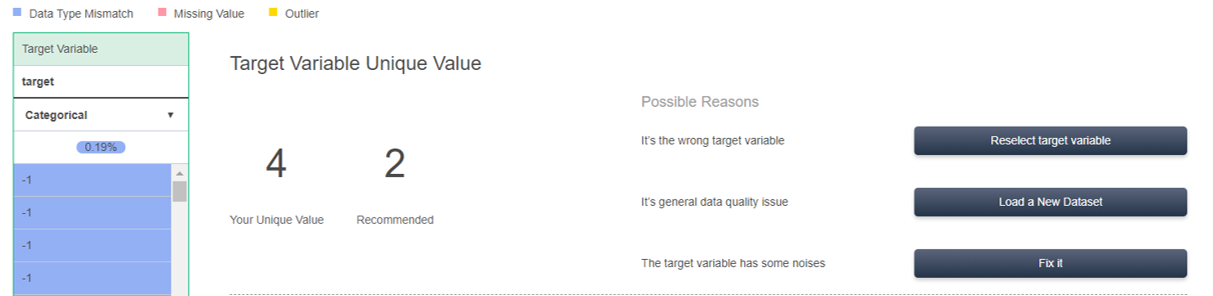
A list of available fixes for these issues can be found in [Appendix: Data Quality Fixes](#appendix-data-quality-fixes).

##### [Target Variable Quality](#_target_variable_quality)

In the **Target Variable Quality** section, R2-Learn identifies and attempts to fix any issues it finds with your target variable.

###### 

When R2-Learn finds an issue that it can automatically fix, it displays these issues in the **Data Quality** tab:

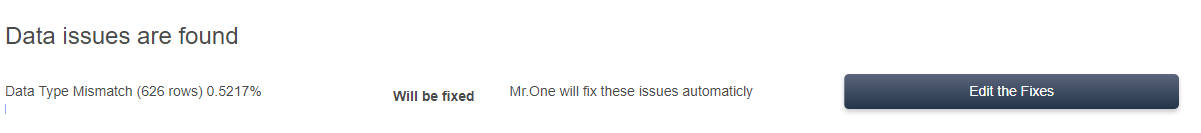


Doing nothing to have R2-Learn automatically fix issues with your target variable, or manually fix it by:

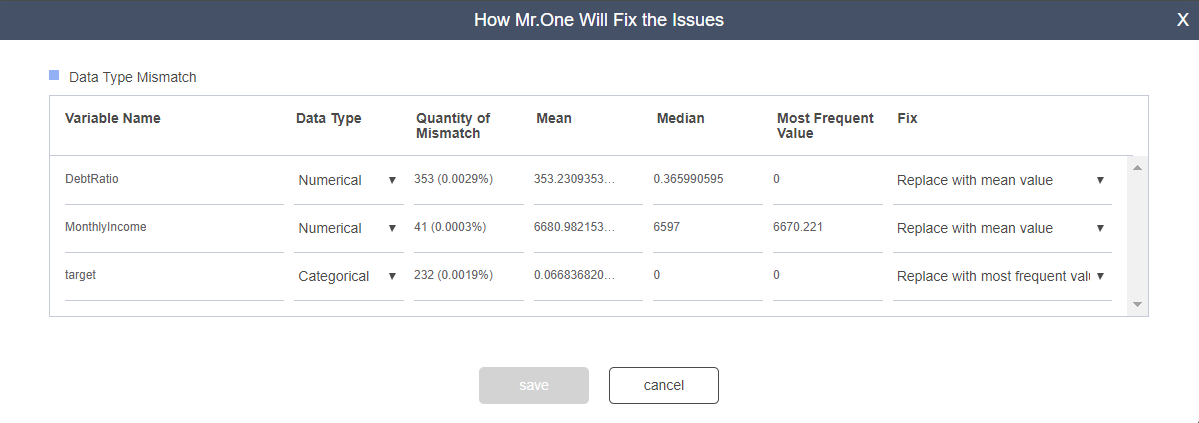
* Click **Fix it** to modify how R2-Learn fixes issues in your target variable. See [Appendix: Data Quality Fixes](#appendix-data-quality-fixes) for a list of common fixes available.
* Click **Reselect Target Variable** to go back to the [Target Variable](#_target_variable) tab and select a new target variable.
* Click **Load a New Dataset** to go back to the [Data Connect](#_uploading_your_data) tab and select a fresh dataset to work with. (To cancel loading a new dataset, click on the **Data Quality** tab.)

##### [All Data Quality](#_all_data_quality)

Once you’ve resolved all issues with your target variable, R2-Learn then examines the quality of all the independent variables in your dataset, and displays a summary of its findings here.



R2-Learn automatically provides default fixes. To override the default fixes, you can select one of the following:

* Click **Load a New Dataset** mentioned in Target Variable Quality to go back to the [Data Connect](#_uploading_your_data) tab and select a fresh dataset to work with. (To cancel loading a new dataset, click on the **Data Quality** tab.)
* Select **Edit the Fixes**. This brings up a dialog box that allows you to choose how you want to fix your data.  
    
  See [Appendix: Data Quality Fixes](#appendix-data-quality-fixes) for a list of common fixes available.

|  |  |
| --- | --- |
| Note | Saving the changes to your data may take a minute or two. Please be patient. |

Once you’re done with verifying and fixing the quality of your data, click **Continue** to start [Modeling](#_modeling) your data.

### [3.4. Modeling](#_modeling)

You’re ready to start building your machine learning model!

You can choose to build your machine learning model using:

* [**Automatic Modeling**](#_automatic_modeling), or
* [**Advanced Modeling**](#_advanced_modeling)

#### [3.4.1. Automatic Modeling](#_automatic_modeling)

**Automatic Modeling** automatically builds your machine learning model for you, based on the training dataset that you’ve loaded and cleaned in the steps leading up to this point.

Click **Automatic Modeling** and R2-Learn will start building your model for you.

#### [3.4.2. Advanced Modeling](#_advanced_modeling)

Selecting **Advanced Modeling** gives you more control over the modeling process.

For information on Advanced modeling, see [Appendix: Advanced Modeling](#appendix-advanced-modeling).

#### [3.4.3. Building your model](#_building_your_model)

Once you’ve selected and finished configuring your modeling mode, R2-Learn will start building your machine learning model.

There is a possibility that the model training process will fail if R2-Learn is unable to train a sufficiently performant machine learning model from the given training dataset. If model training fails, you have to either reconfigure your project to fix dataset quality issues, or choose a new or larger dataset.

##### [Model Selection](#_model_selection)

After your model training completes, R2-Learn displays a summary of your model training results in the **Model Selection** section. You’ll be shown different charts, depending on the [**Problem Statement**](#business-problem) that you’ve set:

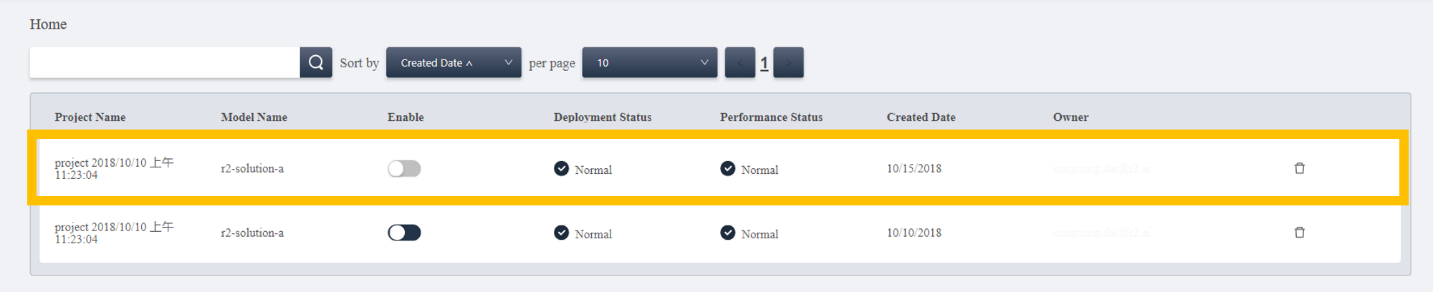
* [Appendix: Model selection for Binary Classification problems](#_model_selection_for_binary_classification_problems)
* [Appendix: Model selection for Regression problems](#_model_selection_for_regression_problems)

Once you’re done selecting your model, click [**Deploy**](#Deploy%20Model) **the Model** to deploy your model.

## [4. Deploying your models](#_deploying_your_models)

Deploying your models allow you to make predictions on a new set of input data. This usually involves publishing a REST API endpoint to which you can send data to and receive predictions from.

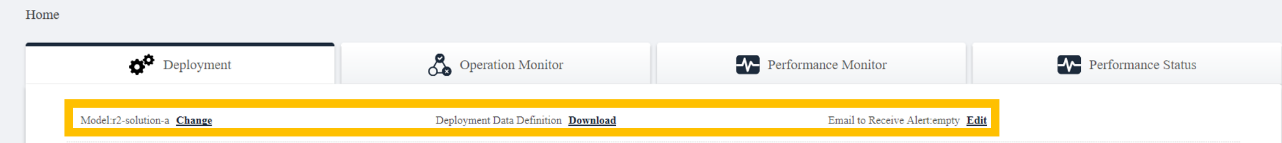
R2-Learn handles model deployment for you automatically. Once you’ve successfully created a machine learning model, you can access the deployed model by clicking on its containing project:



In an open project, you have the following tabs:

* [Deployment](#_deployment)
* [Operation Monitor](#_operation_monitor)
* [Performance Monitor](#_performance_setting)
* [Performance Status](#_performance_status)

In addition, you can access the following options:



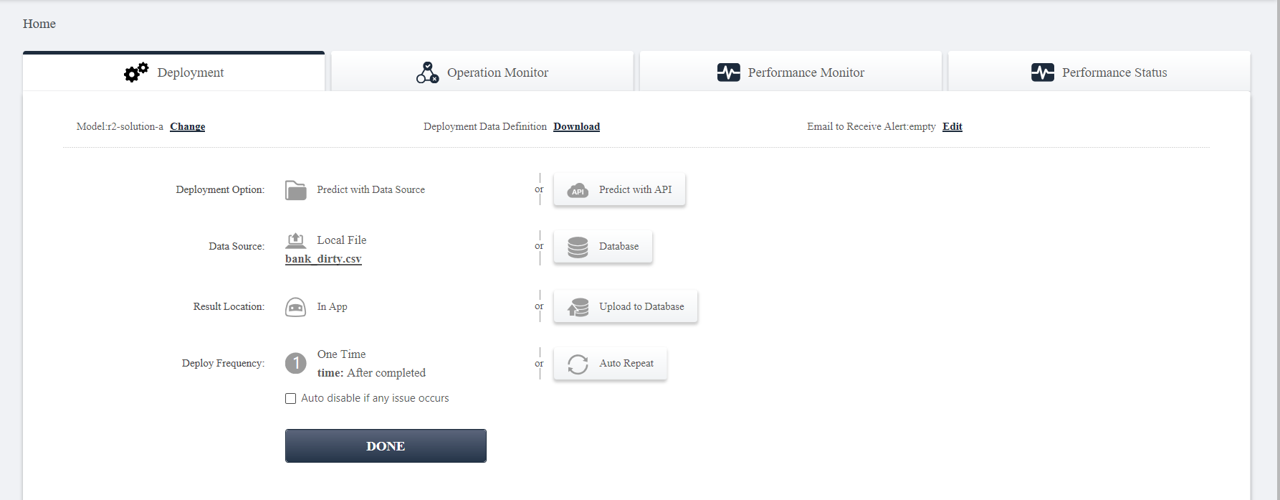
* **The model being used**: Select a model to run a deployment or other operations on.
* **Deployment Data Definition**: Click **Download** to download a text file containing a comma-separated list of the variables used by your model.
* **Email to Receive Alert**: Click to enter an email address to send deployment-related alerts to.

### [4.1. Deployment](#_deployment)

When you open a project in the [Model deployment home page](#_model_deployment_home_page), you’ll be brought to the deployment tab. Here, you can select how you want to use your deployed machine learning models:

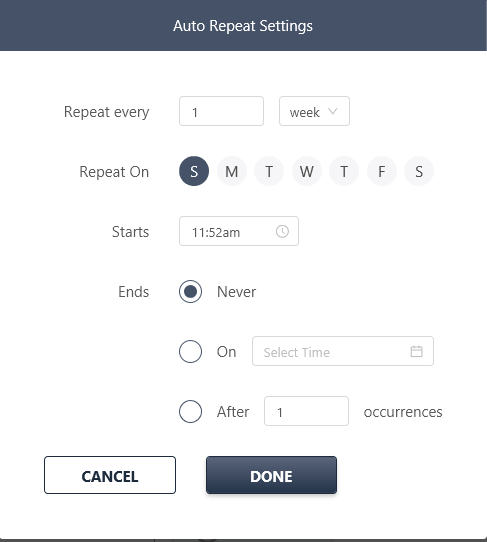
* [**Predict with data source**](#_predict_with_data_source): Predict using data from a local file, or from a database.
* [**Predict with API**](#_predict_with_api): Predict by sending data to R2-Learn using R2-Learn’s REST API.

|  |  |
| --- | --- |
| Important | Imported data must have the same variables as the machine learning model used. To download a file containing a list of variables required in your imported dataset, click **Download** next to **Deployment Data Definition**. |



#### [4.1.1. Predict with data source](#_predict_with_data_source)

To make a prediction on data imported from a data source:

1. Click on **Predict with Data Source**.
2. Select the data source to import data from:
   * **Database**: Provide connection details to a database that contains data you want to run predictions on. For more information, see [Importing data from a database](#_importing_data_from_a_database).
   * **Local File**: Uploads a file to R2-Learn that contains data you want to run predictions on. For more information, see [Importing a local file](#_importing_a_local_file).
3. Select **Result Location**, where the **Result Location** is where your model’s predictions are saved:
   * **In App**: Saves and displays the results in R2-Learn.
   * **Upload to Database**: Writes the results to a given database.
4. Select **Deploy Frequency**. This tells R2-Learn how to schedule the deployment:
   * **One Time**: Sets your deployment to run a prediction on the data source once.
   * **Auto Repeat**: Sets your deployment to run predictions on the data source according to a set schedule.
   * 
   * **Auto disable if any issue occurs**: Select this option to stop the deployment if R2-Learn encounters an issue.

#### [4.1.2. Predict with API](#_predict_with_api)

When you deploy your machine learning model in R2-Learn, a REST API endpoint is automatically generated for you to make requests to.

To make a prediction with R2-Learn’s REST API, you need:

* An R2-Learn API KEY.
* Your R2-Learn user name.
* Your R2-Learn project name. For example, project 11/09/2018, 18:04:13.
* Your R2-Learn model name. For example, Ridge1.auto.09.11.2018\_18:05:24.

To run a prediction using R2-Learn’s REST API with cURL and save the results to output.csv, run the following command in your terminal:

curl -F 'data=@path/to/local/file' http://service2.newa-tech.com/api/user\_name/project\_name/model\_name/api\_key -o output.csv

You will need to write the URL encoded version of your project name and model name into the request. For example, project 11/09/2018, 18:04:13 as a URL encoded string is project%2011%2F09%2F2018%2C%2018%3A04%3A13, and Ridge1.auto.09.11.2018\_18:05:24 as a URL encoded string is Ridge1.auto.09.11.2018\_18%3A05%3A24.

Your resulting cURL API request should look something like this:

curl -F 'data=@/home/r2user/input\_data.csv' http://service2.newa-tech.com/api/r2user/project%2011%2F09%2F2018%2C%2018%3A04%3A13/Ridge1.auto.09.11.2018\_18%3A05%3A24/apikey912ec803b2ce49e4a -o /home/r2user/output.csv

You can also write the API request in Python:

# API code  
import requests  
# Make predictions on your data  
data = {"file": open('/path/to/your/data.csv', 'rb')}  
Response = requests.post('http://service2.newa-tech.com/api/<user\_name>/<project\_namr>/<model\_name>/<api\_key>', data)

### [4.2. Monitor your deployed models](#_monitor_your_deployed_models)

You can monitor and fine-tune the performance of your deployed models in the following tabs:

* [Operation Monitor](#_operation_monitor)
* [Performance Monitor](#_performance_setting)
* [Performance Status](#_performance_status)

#### [4.2.1. Operation Monitor](#_operation_monitor)

This page displays information about any ongoing or completed operations for your models. For each operation, the operation monitor shows:

* **Model Name**: The name of the model being run on the input data.
* **Start Time**: Time at which the operation started.
* **End Time**: Time at which the operation ended.
* **Deployment Style**: The method by which R2-Learn connects to the input data: through a database connection, an uploaded local file, or through an API request.
* **Status**: Status of the operation.
* **Results**: Result of the operation.

In addition, the following options are available for each operation:

* **Download Result**: After an operation completes, you can download the result dataset. The results are appended with an extra column named "<target\_variable\_name>\_pred" that contains the predicted value for each data point.
* **Cancel Ongoing Deployment**: Stops an ongoing operation.

#### [4.2.2. Performance Monitor](#_performance_setting)

Here, you can upload a validation dataset to run with a model on to validate the performance of that model.

You can connect R2-Learn to a validation dataset to run your model on by:

* [Importing data from a database](#_importing_data_from_a_database)
* [Importing a local file](#_importing_a_local_file)

The following options are available:

* **Measurement Metric**: Select the measurement metric to use when running your model on the validation dataset. You can select one of the performance metrics based on your preference.
  + For Binary Classification models: Select either Accuracy or AUC as your measurement metric.
  + For Regression models: Select either RMSE or R² as your measurement metric.
* **Metric Threshold**: This is the threshold which you can set for notifications of the performance of your deployed data. If the performance of the deployed data is worse than this threshold, R2-Learn displays an alert.
* **Deploy Frequency**: You can schedule repeating operations here.
* **Auto Disable if any issue occurs**: Enabling this option sets the operation to terminate if it encounters any issues.

#### [4.2.3. Performance Status](#_performance_status)

Here, you can monitor the status of any ongoing or completed operations. Displayed here is the following information:

* **Model Name**: The name of the model being run on the input data.
* **Start Time**: Time at which the operation started.
* **End Time**: Time at which the operation ended.
* **Performance**: Displays model performance.
  + For Regression models, it displays your model’s R² and RMSE.
  + For Binary Classification models, it displays your model’s AUC and Accuracy.
* **Threshold**: The threshold you’ve set under [Performance Setting](#_performance_setting)
* **Results**: After an operation completes, you can download the result dataset. The results are appended with an extra column named "<target\_variable\_name>\_pred" that contains the predicted value for each data point.

## [Appendix A: Data Quality Fixes](#appendix-data-quality-fixes)

Some of the issues with data quality that R2-Learn may flag are:

* **Data Type Mismatch**: When a variable’s data type doesn’t match what R2-Learn expects given what it has inferred from the dataset, it issues a **Data Type Mismatch** warning. For example, if R2-Learn expects a variable to be categorical, but is labeled as numerical.
* **Missing Value**: Missing values in your data can be either due to an error in data collection or a null value.

|  |  |
| --- | --- |
| * Note | * A null value represents an absence of data ("no value"), and is different from "zero value". |

* **Outlier**: The data point is outside the expected range for the given dataset. It is important to inspect data points that are outliers, and determine if they are genuine deviations from the trend shown in the data or bad data points.

In the [Data Quality](#_data_quality) tab, R2-Learn helps you fix these issues with the following tools:

* [Fixing outliers](#_fixing_outliers)
* [Fixing missing values](#_fixing_missing_values)
* Fixing data type mismatch

### [A.1. Fixing outliers](#_fixing_outliers)

* **Valid Range Editing**
  + With our rule engine, our system automatically extrapolates from your dataset a reasonable range for a variable, and treats values outside the range as outliers. Editing the Valid Range manually expands or contracts the range of values that R2-Learn will accept for that variable.
* **Replace with Boundaries**
  + Replace outlier values with the boundary value.
  + Applicable only to numerical variables.
  + For example, if a column has values [1, 50, 60, 70, 1000] and has a reasonable range of (40, 80), selecting **Replace with Boundaries** will replace the outliers [1] and [1000] with the respective boundary values, giving us the values [40, 50, 60, 70, 80].
* **Do Nothing**
  + Leave the outlier value as is.
  + Applicable only to numerical variables.
* **Replace with mean value**
  + Replace the outlier value with a value that is the average of all the values within the valid range in that column.
  + Applicable to only numerical variables.
  + For example, if a column has values [1, 50, 60, 70, 1000] with a valid range of (40,80) then the outlier values [1] and [1000] will be replaced with the value [60], giving us a resulting dataset of [60, 50, 60, 70, 60].
* **Replace with median value**
  + Replace the outlier value with a value that is the median of all the values within the valid range in that column.
  + Applicable to only numerical variables.
  + For example, if a column has values [1, 6, 5, 8, 50] with a valid range of (3,10), then the outlier values [1] and [50] will be replaced with the value [6], giving us a resulting dataset of [6, 6, 5, 8, 6].
* **Drop the Rows**
  + Removes the row that contains the outlier value.
* **Replace with 0**
  + Replace the outlier value with 0.
  + Applicable only to numerical variables.

### [A.2. Fixing missing values](#_fixing_missing_values)

* **Replace with most frequent value**
  + Replace the missing value with the value that appears the most frequently for that column.
  + Applicable to only categorical variables.
  + For example, if a column has values [1, 2, 5, 3, 2], then all missing values will be set to category 2.
* **Replace with a new unique value**
  + Replace the missing value with a new unique value that will function as a new category.
  + Applicable to only categorical variables.
* **Replace with mean value**
  + Replace the missing values with the average of the remaining values in the column.
  + Applicable to only numerical variables.
  + For example, if a column has the following non-missing values [4, 6, 8], then all missing values will be replaced with the value 6.
* **Replace with median value**
  + Replace missing values with the median of the remaining values in the column
  + Applicable to only numerical variables
  + For example, if a column has the following non-missing values [6, 5, 8], then all missing values will be replaced with the value 6.
* **Drop the rows**
  + Removes the row that contains the missing value.
* **Replace with 0**
  + Replaces the missing values with 0.
  + Applicable only to numerical variables.
* **Why Missing?**
  + Alternatively, you can ask R2-Learn to fix the missing values for you. Select one of the following three options to tell R2-Learn why the value is missing, and let R2-Learn automatically fix the missing value:
    - "I don’t know"
    - "Left on blank on purpose"
    - "Failed to collect data"

### A.3. Fixing data type mismatch

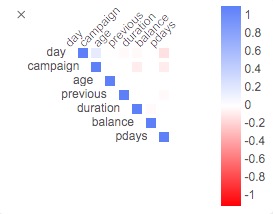
* **Replace with mean value**
  + Replace mismatched data values with the average of the remaining values in the column.
  + Applicable to only numerical variables.
  + For example, if a column has the following values [4, 6, 8], then all mismatched values will be replaced with the value 6.
* **Delete the row**
  + Removes the row that contains the mismatched value.
* **Replace with min value**
  + Replace mismatched data values with the minimum of the remaining values in the column.
  + Applicable to only numerical variables.
  + For example, if a column has the following values [4, 6, 8], then all mismatched values will be replaced with the value 4.
* **Replace with max value**
  + Replace mismatched data values with the maximum of the remaining values in the column.
  + Applicable to only numerical variables.
  + For example, if a column has the following values [4, 6, 8], then all mismatched values will be replaced with the value 8.
* **Replace with max+1 value**
  + Replace mismatched data values with the maximum+1 of the remaining values in the column.
  + Applicable to only numerical variables.
  + For example, if a column has the following values [4, 6, 8], then all mismatched values will be replaced with the value 9.
* **Replace with most frequent value**
  + Replace the mismatched data values with the value that appears the most frequently for that column.
  + Applicable to only categorical variables.
  + For example, if a column has values [1, 2, 5, 3, 2], then all mismatched values will be set to category 2.
* **Replace with median value**
  + Replace mismatched values with the median of the remaining values in the column
  + Applicable to only numerical variables
  + For example, if a column has the following values [6, 5, 8], then all mismatched values will be replaced with the value 6.

## [Appendix B: Advanced Modeling](#appendix-advanced-modeling)

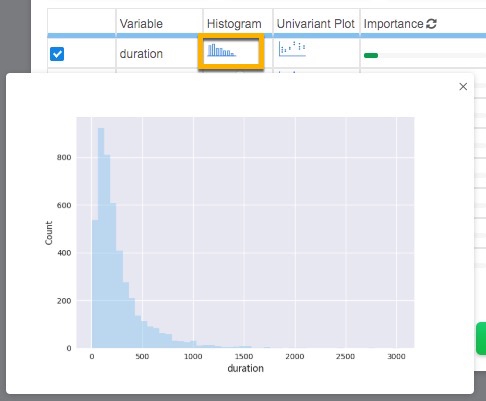
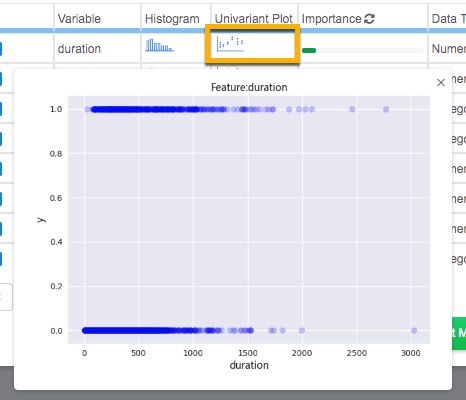
When you select **Advanced Modeling**, you can configure the following settings:

### [B.1. Advanced Variable Settings](#_advanced_variable_settings)

Under this section, you can modify the variables in your data to alter your model. You can change the following in **Advanced Variable Settings**:

* **Selecting/Deselecting Variables**: You can select additional variables or remove variables from your dataset. After your machine learning model has been built, you can see how the change in your variable selection affects your model in the [Model Selection](#_model_selection) section.
* **Check Correlation Matrix**: Click **Check Correlation Matrix** to display a small graph that shows the correlational strength between 15 variables with the strongest correlational relationships in your dataset. The stronger the correlation, the cooler (closer to blue) the color. Cool colors (shades of blue) represent a positive correlation, while warm colors (shades of red) represents a negative correlation.
* 
* **Create New Variable**: Click **Create New Variable** to add a new variable to your model. This new variable can be a combination of existing variables, or an operation performed on an existing variable. To create a new variable:
  1. Click on **Create New Variable**.
  2. Enter a unique name for your new variable.
  3. Click on the field after the equal ("=") sign to open a dropdown menu listing all possible operations. You can perform the following operations on your existing variables to create your new variable:
     + Groupby and compute
     + Combine Variables
     + Compare Values
     + Accumulate Values
     + log()
     + Other operators: +,-,\*,/

|  |  |
| --- | --- |
| Note | For more information on each of these operations, click on **See Tips** when selecting the operation. |

* **Histogram/Bar Chart**: Clicking on a variable’s "Histogram" displays a graph that shows the distribution of all the values for the variable. Numerical variables are shown as a histogram while categorical variables are shown as a bar chart.
* 
* **Univariant Plot**: Clicking on a variable’s "Univariant Plot" displays a graph that shows a distribution of the values for that variable.
* 
* **Variable Importance**: The variable importance section represents the statistical significance of a predictor for the target variable; the higher the "Importance" of a variable, the more a change in this variable would affect the predicted target variable. Variables displayed in **Advance Variable Settings** are by default sorted from highest to lowest "Importance".

### [B.2. Advanced Modeling Setting](#_advanced_modeling_setting)

Here you are able to configure other settings for model building:

#### [B.2.1. Create/Edit Model Setting By default](#_createedit_model_setting_by_default)

When you click on **Advanced Modeling**, R2-Learn creates a new "Model Setting". In the **Advanced Modeling Settings** tab, your new model setting is:

1. Given the default name "custom.<MM.DD.YYY\_HH:MM:SS>"
2. Sets the {select} field to this new model setting.

You can build on a previously defined "Model Setting" by selecting it in the {select} field. Change the name in the "Name Your Model Settings" field to save your modified model settings as a new model setting.

#### [B.2.2. Select Algorithm](#_select_algorithm)

By default, R2-Learn builds models for each project using all available classifier algorithms. You can remove classifiers from this list to reduce the amount of time modeling takes. The following is a table of classifiers and the problem types they are compatible with:

|  |  |  |
| --- | --- | --- |
| Algorithm | Binary Classification | Regression |
| Adaboost | ✓ | ✓ |
| ARD (Automatic Relevance Determination) Regression |  | ✓ |
| Bernoulli Naive Bayes | ✓ |  |
| Decision Tree | ✓ | ✓ |
| Extra Trees | ✓ | ✓ |
| Gaussian Naive Bayes | ✓ |  |
| Gaussian Process |  | ✓ |
| Gradient Boosting | ✓ | ✓ |
| K Nearest Neighbor | ✓ | ✓ |
| Linear Discriminant Analysis | ✓ |  |
| Multinomial Naive Bayes | ✓ |  |
| Passive Aggressive | ✓ |  |
| Quadratic Discriminant Analysis | ✓ |  |
| Random Forest | ✓ | ✓ |
| SGD (Stochastic Gradient Descent) | ✓ | ✓ |
| Logistic Regression | ✓ |  |
| Ridge Regression |  | ✓ |

#### [B.2.3. Set Max Model Ensemble Size](#_set_max_model_ensemble_size)

This is the maximum number of sub-models you R2-Learn that can be combined to form one machine learning model, or an ensemble model For example, if the **Max Model Ensemble Size** is set to 3, then the resulting ensemble can be made up of a maximum number of 3 sub-models.

The actual number of sub-models in the resulting ensemble would depend on the allotted [training time](#Set%20Max%20Training%20Time).

#### [B.2.4. Train Validation Holdout and Cross Validation](#_train_validation_holdout_and_cross_validation)

You can choose to build your machine learning models using:

* **Train Validation Holdout**: This constructs a machine learning model by partitioning your training dataset into the three subsets:
  + **Training set**: This is used to build the machine learning model.
  + **Validation set**: This is used to tune the hyper parameters of classifiers for better accuracy.
  + **Holdout set**: This set of data helps assess the performance of the final model; it does not play any role in model building, but is used for validating the built model.
  + You are able to set the percentage of your dataset to use for each subset by dragging the circle at the intersection of any two sets.
* **K-fold Cross Validation Holdout**: This constructs a machine learning model by:
  1. Partitioning your training dataset into 'k' number of "folds", or 'k' number of equally sized subsamples.
  2. Running a modeling process where each of the 'k' subsamples is in turn used as the validation set while the remaining subsamples are used as training set.
  3. This process is repeated for each of the 'k' subsamples, giving us 'k' number of models.
  4. We then find the average of the 'k' number of models to attain a single model.
  + This method takes much longer than the **Train Validation Holdout**. You are able to set the number of folds and the holdout percentage.

#### [B.2.5. Resampling Setting](#_resampling_setting)

|  |  |
| --- | --- |
| Note | Only available for Binary Classification models. |

If the target variable shows an uneven distribution of sample sizes for each outcome, then you have an unbalanced dataset.

R2-Learn can balance out your dataset by **upsampling** or **downsampling** your training dataset. For both upsampling and downsampling, you can adjust the ratio of one category to another.

#### [B.2.6. Set Measurement Metric](#_set_measurement_metric)

You can change the measurement metric used to judge the effectiveness of the machine learning model.

For Binary Classification models, you can select:

* **AUC (area under the curve)** (default)
* **Accuracy**
* **F1**

For Regression models, you can select:

* **MSE (mean square error)** (default)
* **R²**

#### [B.2.7. Set Max Optimization Time](#_set_max_optimization_time)

You can set the maximum optimization time for optimizing the model building parameters. The longer the time allotted for optimization, the better the performance of the resulting machine learning model. For larger training datasets, we recommend you to set a higher maximum optimization time.

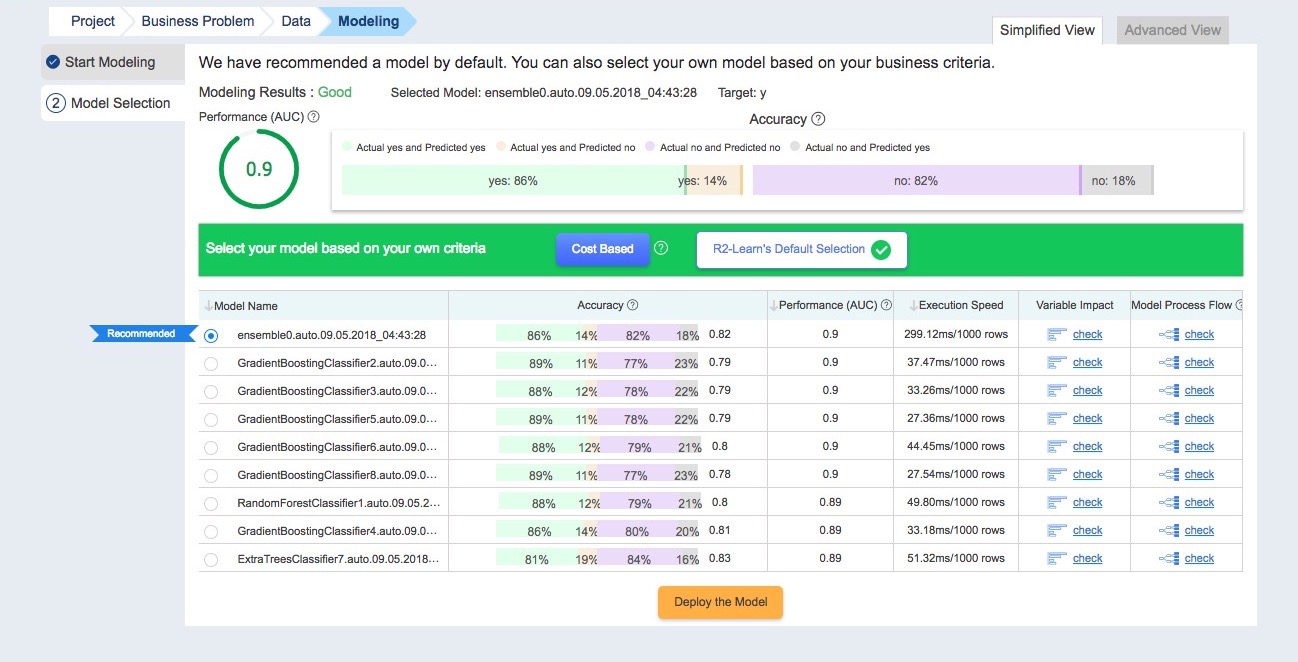
For any dataset, you should set an optimization time of at least 9 minutes.

#### [B.2.8. Random Seed](#_random_seed)

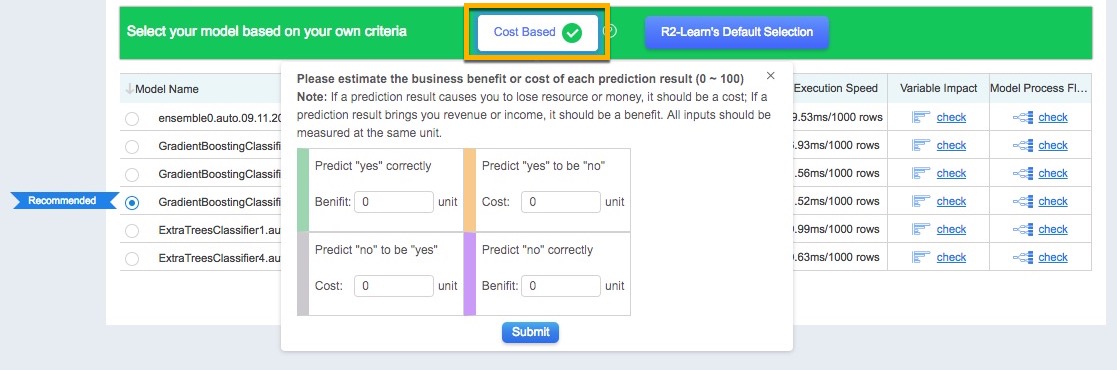
This seeds any random numbers generated while training your machine learning models. Setting the seed allows you to reproduce your modeling process by using the same random seed to build similar models.

## [Appendix C: Model selection for Binary Classification problems](#_model_selection_for_binary_classification_problems)

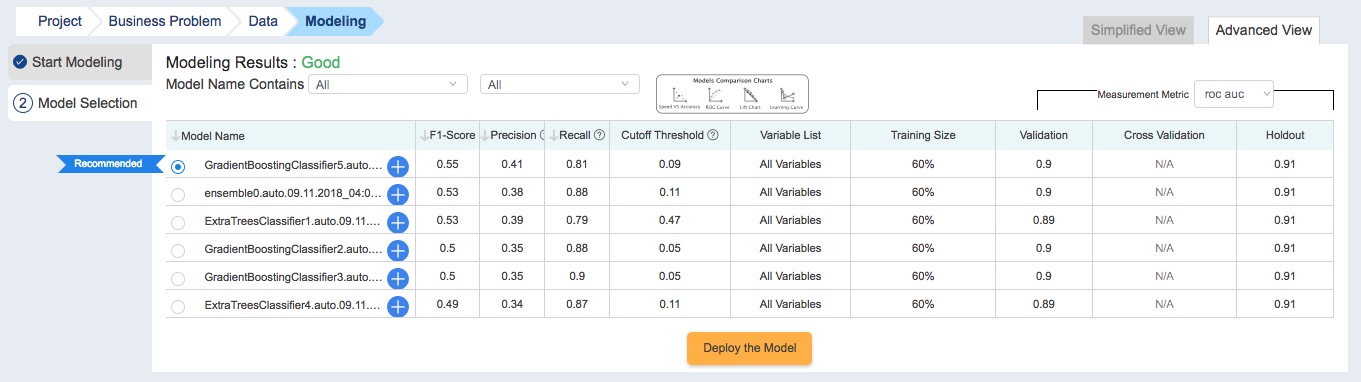
### [C.1. Simplified view](#_simplified_view)



When in the **Model Selection** tab for Binary Classification problems, the following is displayed:

* **Modeling Results**: A general grade indicating the general performance levels of the trained models.
* **Selected Model**: Your currently selected machine learning model. This is selected from the list of trained machine learning models displayed below.
* **Target**: The [Target Variable](#_target_variable) that is being predicted by the machine learning model.
* **Performance**: Shows the performance of the selected model. This is usually AUC (area under the curve) for Binary Classification models.
* **Accuracy**: Shows the accuracy of the predictions made by the selected model.
* **Select your model based on your own criteria**: R2-Learn automatically selects a model that it thinks is the most suitable for your business problem from the list of trained models. You modify the means by which it picks its recommendation by selecting one of the following options:
  + **R2-Learn’s Default Recommendation**: R2-Learn makes a recommendation based on a balance between execution time and performance.
  + **Cost Based**: Here, you can quantify the business costs incurred for false positives (Type I errors) and false negatives (Type II errors), and the benefits gained from correct positive and negative predictions. R2-Learn will recommend a model to use based on this data.
  + 
* **Table of trained models**: This section also shows a list of models that have been built with the training data. The listed models each have a:
  + **Model Name**: The name of this model. This is usually "<algorithm\_name>.<model\_setting\_name>", where <model\_setting\_name> is automatically determined if you have selected [Automatic Modeling](#_automatic_modeling), or is set when configuring your [Advanced Modeling Setting](#_advanced_modeling_setting).
  + **Accuracy**: Shows the proportion of predictions the model makes that match actual outcomes. The higher this value is, the more likely a prediction the model makes is correct.
  + **Performance**: Shows the performance metric for this model. This is usually AUC (area under the curve) for Binary Classification models, and MSE (mean square error) for Regression models.
  + **Execution Speed**: The amount of time the model will take to process a given number of rows in a dataset.
  + **Variable Impact**: Click on **check** to see the impact each of the model’s variables has on the target variable. Displays the impacts each variable in the model has on the predicted target variable. The higher this value is, the larger the effect a change in this variable has on the outcome predicted by the model.
  + **Model Process Flow**: Click on **check** to display a step-by-step breakdown on how this model is built.

### [C.2. Advanced View](#_advanced_view)



Users can use the **Advanced View** to view more detailed information about the models available for deployment.

#### [C.2.1. Top Section](#_top_section)

The top of the section displays the following:

* **Modeling Results**: A general grade indicating the general performance levels of the trained models.
* **Model Name Contains**: Allows you to filter the model name by algorithm and by Model Setting name.
* **Model Comparison Charts**: Charts the performance of each model for the following performance comparisons:
  + **Speed v.s. Accuracy**: Charts the speed v.s. accuracy tradeoff for each model.
  + **Lifts Chart**: The Lift Chart is used to determine the effectiveness of a model by showing the ratio between results obtained with and without using the predictive model.
  + **ROC Curves**: Charts the diagnostic ability of the binary classifier as its threshold varies.

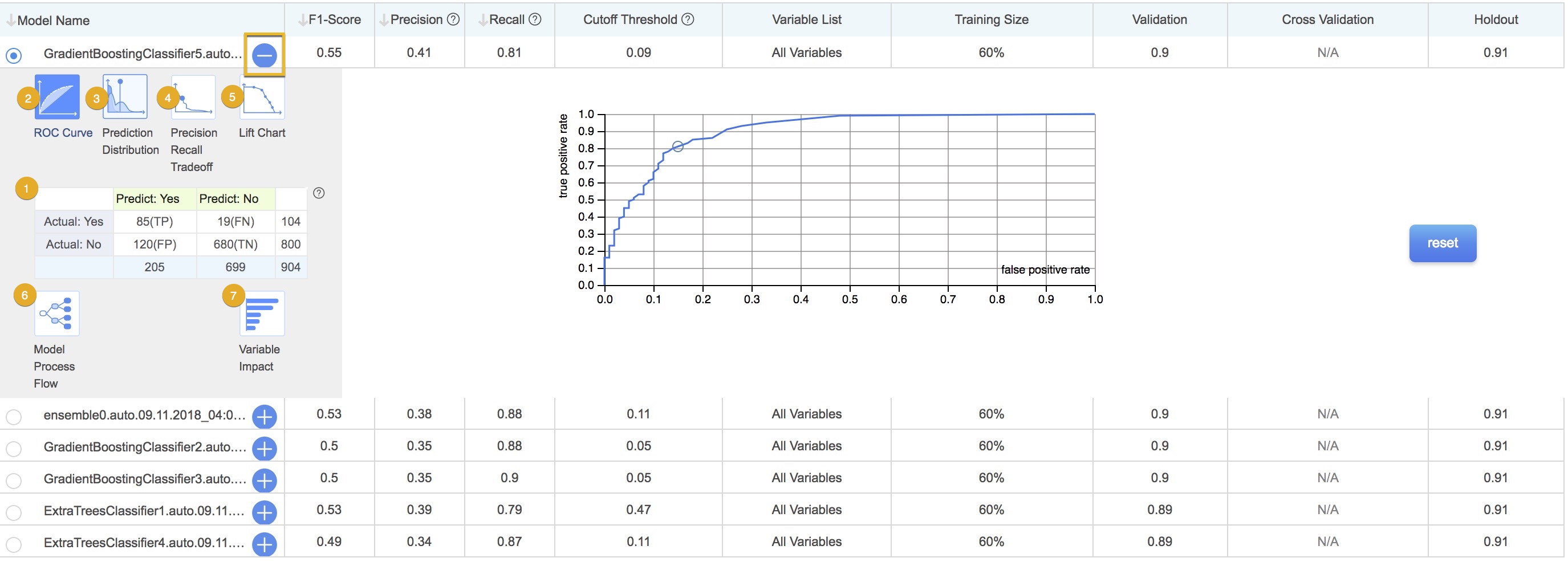
#### [C.2.2. Table of Models](#_table_of_models)

The table below displays all the models trained with the dataset. Each listed model has a:

* **Model Name**: The name of this model. This is usually "<algorithm\_name>.<model\_setting\_name>", where <model\_setting\_name> is automatically determined if you have selected [Automatic Modeling](#_automatic_modeling), or is set when configuring your [Advanced Modeling Setting](#_advanced_modeling_setting).
* **F1-Score**: This score is a measure of test set accuracy: F1 = 2 \* precision \* recall / (precision + recall).
* **Precision**: The proportion of all predictions (including true and false positives) that are true positives. Precision = TP/(TP+FP).
* **Recall**: The proportion of positives that the classification model correctly labels as positives. Recall =TP/(TP+FN).
* **Cutoff Threshold**: Many classifiers are able to produce a probability distribution over a set of classes (e.g. 1 or 0). Cut-off threshold is a probability value which can be used to determine whether an observation belongs to a particular class.
* **Variable List**: Shows which of the variables in the training dataset are included in the model.
* **Validation/Cross Validation**: The Validation/Cross Validation dataset is the data that is set aside to fine-tune an initial model to see if relationships between variables are correctly manipulated to create the best model possible.
* **Holdout**: Holdout is the proportion of the original dataset that has been set aside for the final testing of finished models. The data set aside for holdout is not used in the construction or tuning of the initial model, but is instead used to evaluate how well the model performs.

#### [C.2.3. Additional Model Details](#_additional_model_details)

To view additional model details, click on the **+** next to the **Model Name** to expand the model selection.

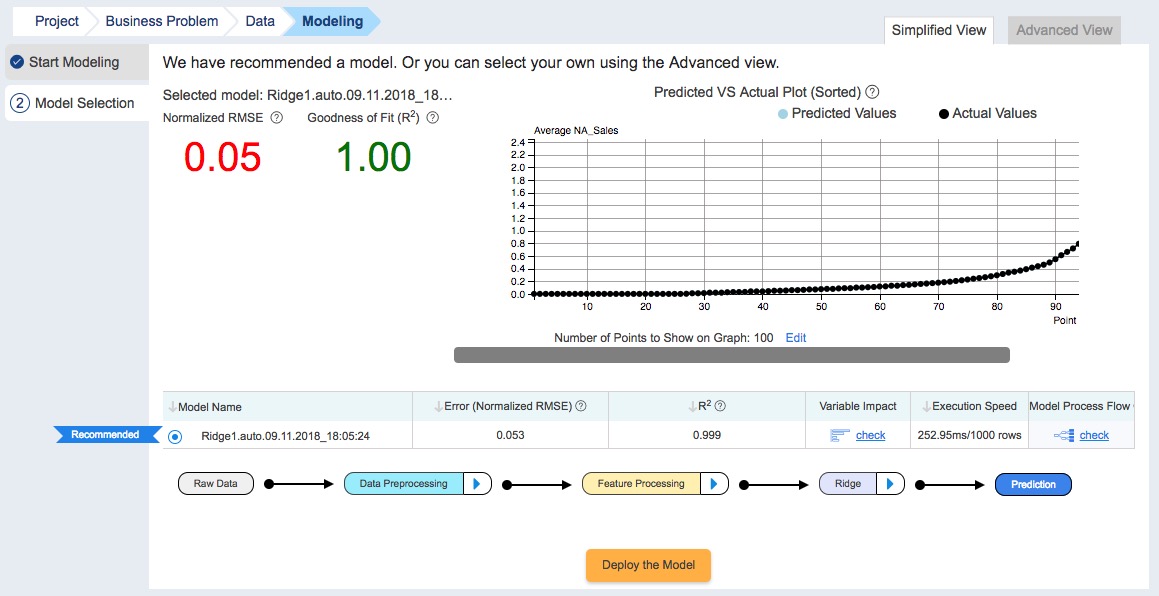


This displays a panel with the following charts:

1. **Error/Confusion Matrix**: This displays a matrix comparing predicted values with actual values, and shows the number of True Positives (TP), False Positives (FP), True Negatives (TN), and False Negatives (FN) predicted by this model.
2. **ROC Curves (adjustable)**: Charts the diagnostic ability of the binary classifier as its threshold varies. Click on the slider to adjust the ratio of these values.
3. **Prediction Distribution (adjustable)**: This plots Probability Threshold against Probability Density. Click on the slider to adjust the ratio of these values.
4. **Precision Recall Tradeoff (adjustable)**: This plots Recall (TP/(TP+FN)) against Precision (TP/(TP+FP)), showing us how accurate the model’s predictions are. Click on the slider to adjust the ratio of these values.
5. **Lift Chart**: The Lift Chart is used to determine the effectiveness of a model by showing the ratio between results obtained with and without using the predictive model.
6. **Model Process Flow**: Shows a step-by-step breakdown on how this model is built.
7. **Variable Impact**: Displays the impacts each variable in the model has on the predicted target variable. The higher this value is, the larger the effect a change in this variable has on the outcome predicted by the model.

## [Appendix D: Model selection for Regression problems](#_model_selection_for_regression_problems)

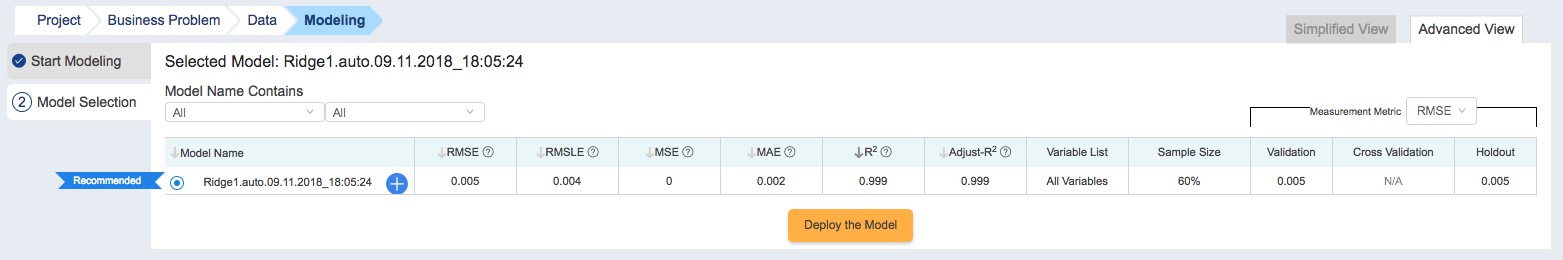
### [D.1. Simplified View](#_simplified_view_2)



When in the **Model Selection** tab for Regression problems, the following is displayed:

* **Selected Model**: Your currently selected machine learning model. This is selected from the list of trained machine learning models displayed below.
* **Normalized RMSE (Root Mean Square Error)**: This value will help you compare model performance: the smaller the Normalized RMSE, the better the model predicts data.
* **Quality of fit (R²)**: R² is a statistical measure of how close the regression line fits the given data. The higher the R² value is, the closer the model fits the data.
  + **Variable Impact**: Displays the impacts each variable in the model has on the predicted target variable. The higher this value is, the larger the effect a change in this variable has on the outcome predicted by the model.
* **Predicted v.s. Actual Plot (Sorted)**: Shows how close the values predicted by your models are to actual values. You can change the number of data points shown by clicking on **Edit** under the chart.
* **Table of trained models**: This section also shows a list of models that have been built with the training data. The listed models each have a:
  + **Model Name**: The name of this model. This is usually "<algorithm\_name>.<model\_setting\_name>", where <model\_setting\_name> is automatically determined if you have selected [Automatic Modeling](#_automatic_modeling), or is set when configuring your [Advanced Modeling Setting](#_advanced_modeling_setting).
  + **Error (Normalised RMSE)**: RMSE (root of mean square error) is square root of the standard error of the model, and is a measure of how close your model’s predictions are to the actual target variable values.
  + **Execution Speed**: The amount of time the model will take to process a given number of rows in a dataset.
  + **Variable Impact**: Click on **check** to see the impact each of the model’s variables has on the target variable. The higher this value is, the larger the effect a change in this variable has on the outcome predicted by the model.
  + **Model Process Flow**: Click on **check** to display a step-by-step breakdown on how this model is built.

### [D.2. Advanced View](#_advanced_view_2)



Users can use the **Advanced View** to view more detailed information about the models available for deployment.

#### [D.2.1. Top Section](#_top_section_2)

The top of the section displays the following:

* **Model Name Contains**: Allows you to filter the model name by algorithm and by Model Setting name.

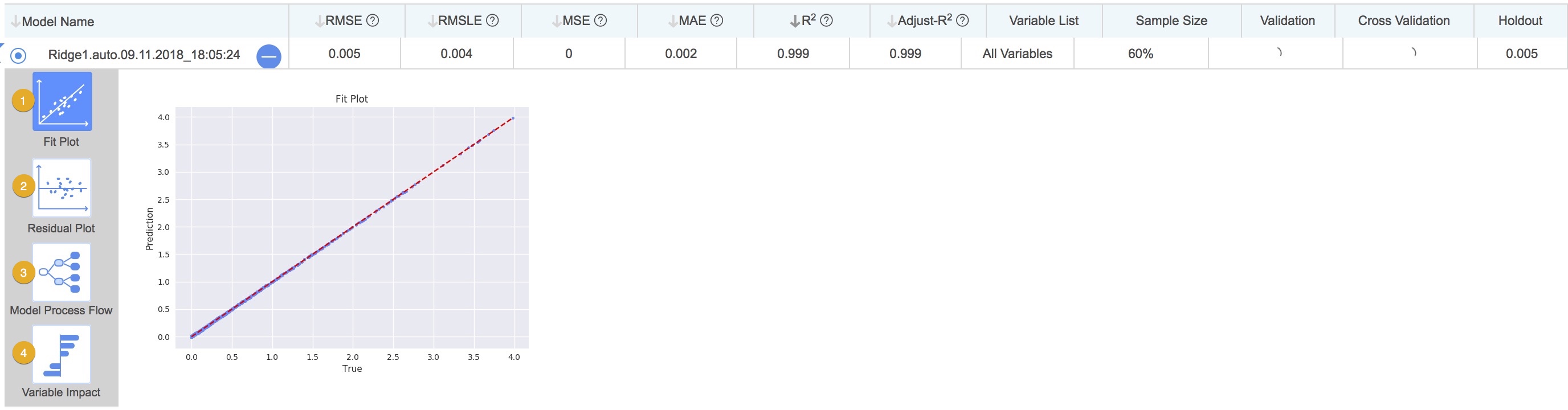
#### [D.2.2. Table of Models](#_table_of_models_2)

The table below displays all the models trained with the dataset. Each listed model has a:

* **Model Name**: The name of this model. This is usually "<algorithm\_name>.<model\_setting\_name>", where <model\_setting\_name> is automatically determined if you have selected [Automatic Modeling](#_automatic_modeling), or is set when configuring your [Advanced Modeling Setting](#_advanced_modeling_setting).
* **RMSE (Root Mean Square Error)**: RMSE is the square root of MSE. Usually used to compare the likelihood of predictive errors across different models
* **RMLSE (Root Mean Logarithmic Square Error)**: This is the square root of the logarithmic value of MSE.
* **MSE (Mean Square Error)**: MSE measures the prediction error of the model. It is the sum of the difference between values predicted by the model and actual values, divided by the total number of data points.
* **MAE (Mean Absolute Error)**: This measures the prediction error of the model. It is the sum of absolute value of the difference between values predicted by the model and actual values.
* **R²**: R² is a statistical measure of how close the regression line fits the given data. The higher the R² value is, the closer the model fits the data.
* **R² Adjusted**: An Adjusted R² is a modified version of R² which has been adjusted for the number of predictors in the model. It is always lower than R².
* **Variable List**: Shows which of the variables in the training dataset are included in the model.
* **Validation/Cross Validation**: The Validation/Cross Validation dataset is the set of data that is set aside from the original dataset to fine-tune an initial model to see if relationships between variables are correctly manipulated to create the best model possible.
* **Holdout**: The holdout is the proportion of the original dataset that has been set aside for the final testing of finished models. The data set aside for holdout is not used in the construction or tuning of the initial model, but is instead used to evaluate how well the model performs.

#### [D.2.3. Additional Model Details](#_additional_model_details_2)

To view additional model details, click on the **+** next to the **Model Name** to expand the model selection.



This displays a panel with the following charts:

1. **Fit plot**: The fit plot charts the predictive accuracy of a model by plotting actual target variable values in the dataset (x-axis) against predicted target variable values (y-axis). A straight line (y = x) is drawn on the graph, showing all the points on the graph where a predicted target variable value is equal to its corresponding actual target variable value. The more points on the graph that are close to or on the line, the better the predictive performance of the model.
2. **Residual plot**: A "residual" is the difference between the actual target variable value and the predicted target variable value for a given data point. The residual plot charts actual target variable values (y-axis) against their corresponding residuals (x-axis) to show possible issues with applying the model to the given dataset.  
   Click on **Diagnose** to get R2-Learn to make recommendations on how to improve your model by comparing the shape of your current model’s residual plot to known residual plot shapes. In the dialog box that opens, select from the following a residual plot shape that most closely resembles your model’s residual plot:
   * Randomly distributed residual plot.
   * Y-axis unbalanced residual plot.
   * X-axis unbalanced residual plot.
   * Outliers residual plot.
   * Nonlinear residual plot.
   * Heteroscedasticity residual plot.
   * Large Y-axis data points residual plot.
3. **Model Process Flow**: Shows a step-by-step breakdown on how this model is built.
4. **Variable Impact**: Displays the impacts each variable in the model has on the predicted target variable. The higher this value is, the larger the effect a change in this variable has on the outcome predicted by the model.