A2 Report

Scikit-learn

Table of Contents

Our Implementation of 2 Issues

Feature: Adding Poisson splitting criterion in RandomTreeRegressor (Issue 19304):

Description

This feature needs 3 tasks to be completed:

- Adds input validation to the parameter “y” where it ensures none of the values inside array “y” are negative i.e., less than zero for splitting criterion “poisson”.
- Expand test cases for RandomForestRegressor class to check “poisson” splitting criterion and input validation.
- Change the docstring of RandomForestRegressor class to add “poisson” as one of the splitting criterion.

Changes

To add input validation and docstring, the code changes are done in the _forest.py file which is under directory sklearn/ensemble. Inside the docstring of RandomForestRegressor class there is a description section about the “criterion” parameter where we added “poisson” to indicate it supports the poisson splitting. For input validation of parameter “y”, we made changes in the fit() method of BaseForest class. The changes we made involved creating a temporary variable storing “y” converted into an array so that we can loop through a 2-dimensional “y” and check its values. We then check for criterion to be “poisson” and if so each value of “y” is checked using the python in-built function all() and ensuring none of the values are non-positive. If a non-positive value is found then a ValueError is raised indicating that the input to the fit() method was incorrect. Overall this was a minor change, thus it didn’t really affect the overall architecture.

- File changed: sklearn/ensemble/_forest.py (Lines 306 - 310 & Line 1318 & Line 1322)

Testing

The test cases that ensure the implementation of input validation is successful are as follows, the cases can be run by using the command “pytest 19304-testsuite.py” in your terminal opened in the a2 directory after successfully installing scikit-learn. If you are unable to do so, please move the .py file into the scikit-learn directory and run the command from that directory:

- Case 1: “y” has all the values greater than or equal to zero.
  This test case should not raise a ValueError as all values of “y” are positive and it should return a predicted value which is checked using assertTrue.

- Case 2: “y” has all values equal to zero.
  This test case should raise a ValueError as all values of “y” are zeros and it should return an array of zeros which is checked using assertTrue.

- Case 3: “y” has a negative value at the start of the array.
  This test case should raise a ValueError as one of the values of “y” is non-positive and the test checks whether an error was raised.

- Case 4: “y” has a negative value at the middle of the array.
  This test case should raise a ValueError as one of the values of “y” is non-positive and the test checks whether an error was raised.

- Case 5: “y” has a negative value at the end of the array.
  This test case should raise a ValueError as one of the values of “y” is non-positive and the test checks whether an error was raised.

Bug: Potential Division by 0 in Gaussian Progression (Issue 18318):

Description

This bug was caused by a division operation where there was no check if the denominator in the division is 0. Of course this is a bug, since division by 0 is not possible and results in an error. The task needed to fix this bug would be to add a check before the division occurs, and ensure the denominator is not 0. If it is 0, we must avoid the division.

Changes

To add the fix to the divide by zero error, the code changes occur inside _gpr.py file which is under directory sklearn/gaussian_process. The changes we made is to check if the variable "self._y_train_std" is equal to zero, if it is then we don't divide by self._y_train_std and instead we assign y to be "y - self._y_train_mean". This is due to the fact that the standard deviation is 0, so we only care about the mean values. Otherwise if "self._y_train_std" is NOT equal to zero we keep the original value of y which was "y = y - self._y_train_mean / self._y_train_std". Overall this was a minor change, thus it didn't really affect the overall architecture.

- Files Changed: sklearn/gaussian_process/_gpr.py (Lines 202 - 207)

Testing

The test cases that ensure the implementation of the divide by zero is successful are as follows, the cases can be run by using the command "pytest 18318-testsuite.py" in your terminal opened in the a2 directory after successfully installing scikit-learn. If you are unable to do so, please move the .py file into the scikit-learn directory and run the command from that directory:

- Case 1: "y" has multiple non-zero values.
  This test case should return a fit value which contains non-zero values which is checked using assertTrue.

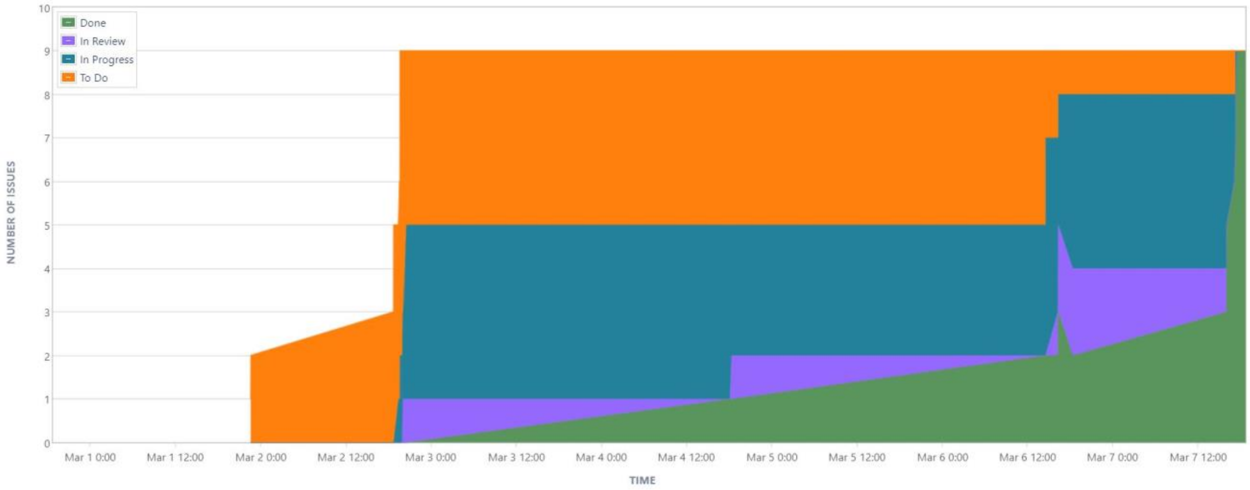
- Case 2: "y" has a single non-zero value.
  This test case should return a fit value which contains a single zero value which is checked using assertTrue.

- Case 3: "y" has multiple zero values.
  This test case should return a fit value which contains multiple zero values which is checked using assertTrue.

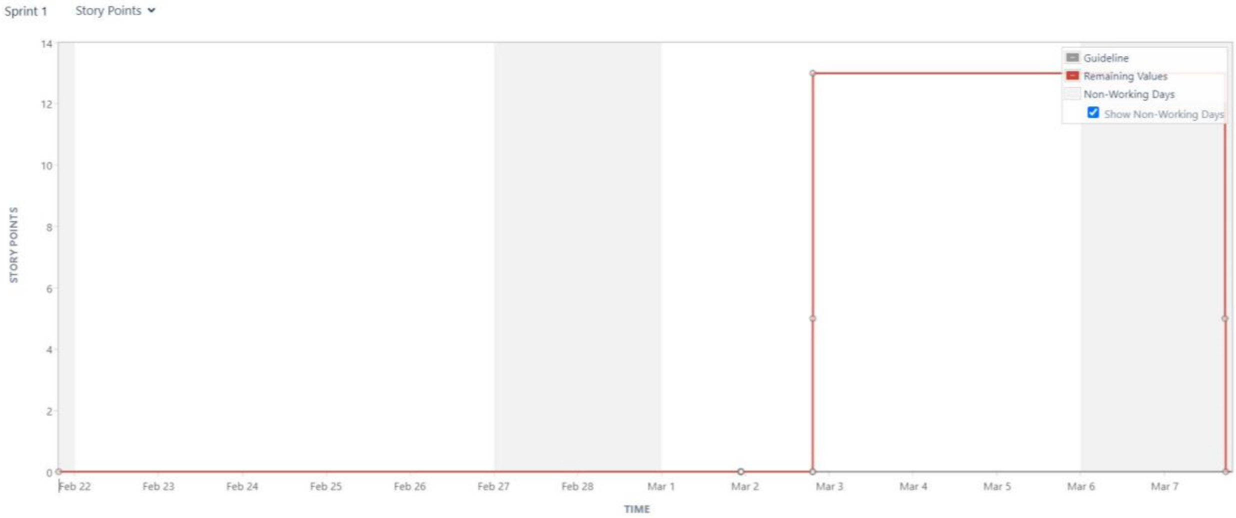
- Case 4: "y" has a single zero value.
  This test case should return a fit value which contains a single zero value which is checked using assertTrue.

Development Process

JIRA



Cumulative Flow Diagram



Burndown Chart