Regression Comparison: Julia's LinAlg vs Python's Scikit-learn

Manoj Yeddanapudy, Matthew Labay, Nolan Ales

Overview

- Purpose: to determine whether scikit-learn in Python or the linear algebra library in Julia is more accurate when given the same data set, specifically in the case of regression and interpolation. We can run the same, well-known regression or interpolation code on both systems, and analyze the differences in results between the two.

- Python's Scikit-learn library: https://github.com/scikit-learn/scikit-learn

- Julia's Linear Algebra library: https://github.com/JuliaLang/julia/tree/master/stdlib/LinearAlgebra

Methods

Imports (Julia): plots, statistics, LinAlg, benchmark, time

Imports (Python): Numpy, sklearn, MatPlotLib, timeit

Basic Methodology: We ran the same, well-known regression on both systems, and analyzed the difference in results between the two:

1) Generate noisy data using linspace and runge function like in class
2) Run regression on the same noisy data on both systems
3) Calculate MSE and R^2 from data generated by regression model (yfit)
4) Compare MSE and R^2
5) Compare runtime

Contribution

- Regression: widely used in a variety of machine learning applications to predict outputs based on specific input features.

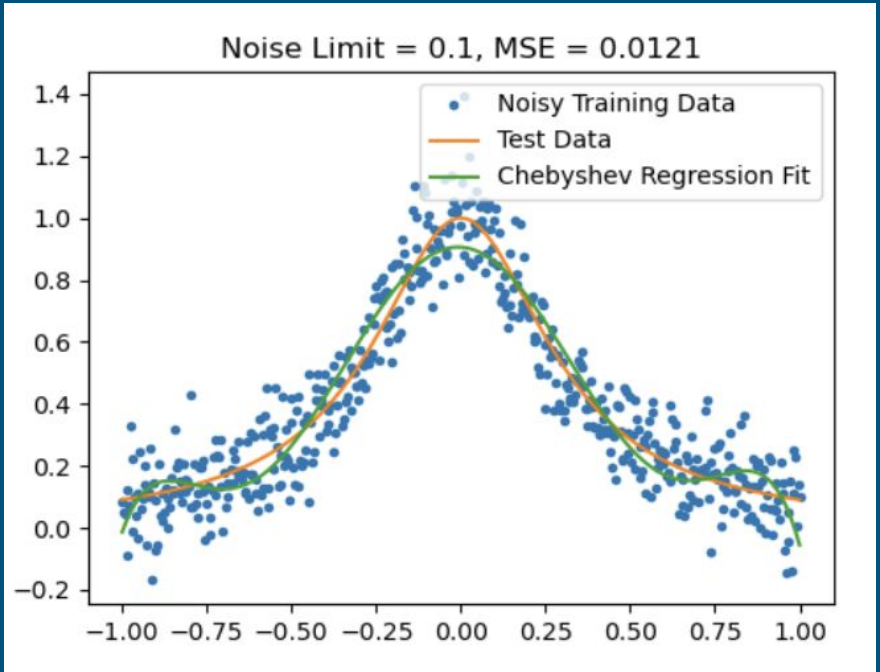
This project will provide valuable insights into the relative performance of scikit-learn and Julia for regression analysis, and help to determine the strengths and weaknesses of each library.

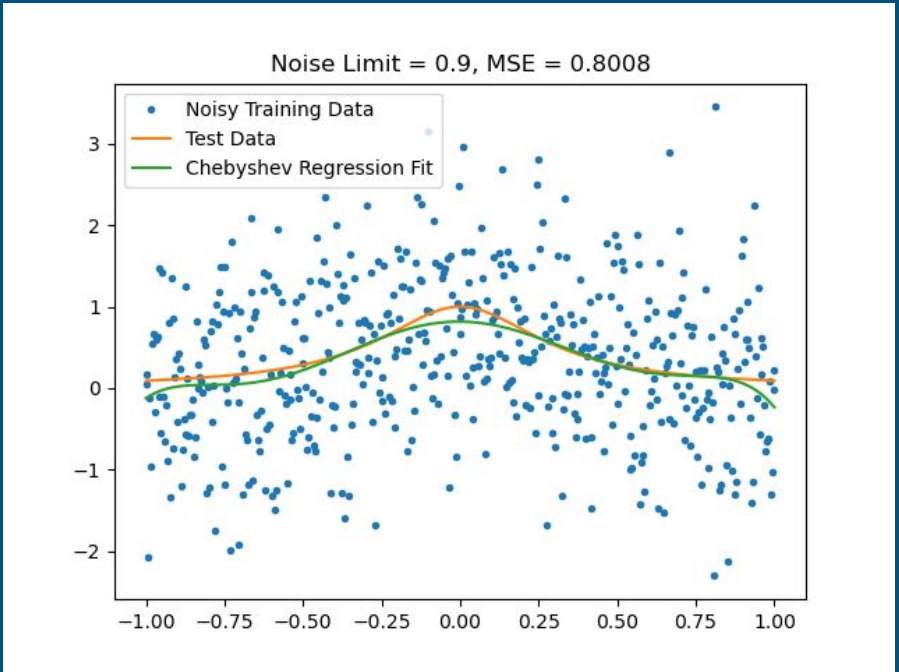
We aim to use insights from this project to better-understand how these differences might affect our outputs and efficiency, as all three of us use some sort of ML library for other classes/projects.

Stakeholder

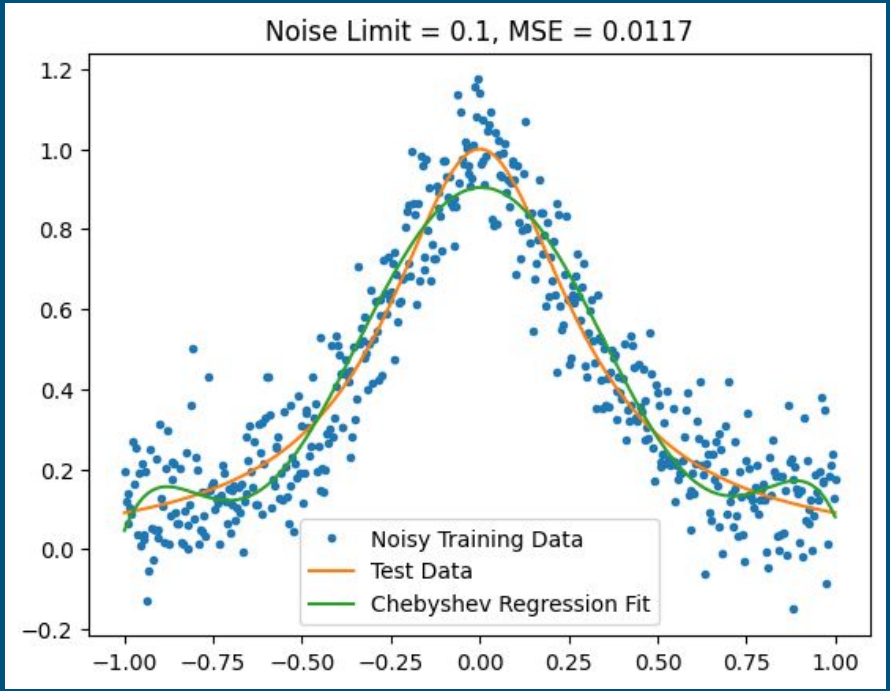
- The stakeholders for this problem are anyone using linear algebra computations:
  - Data Analysts
  - Data Scientists
  - Researchers

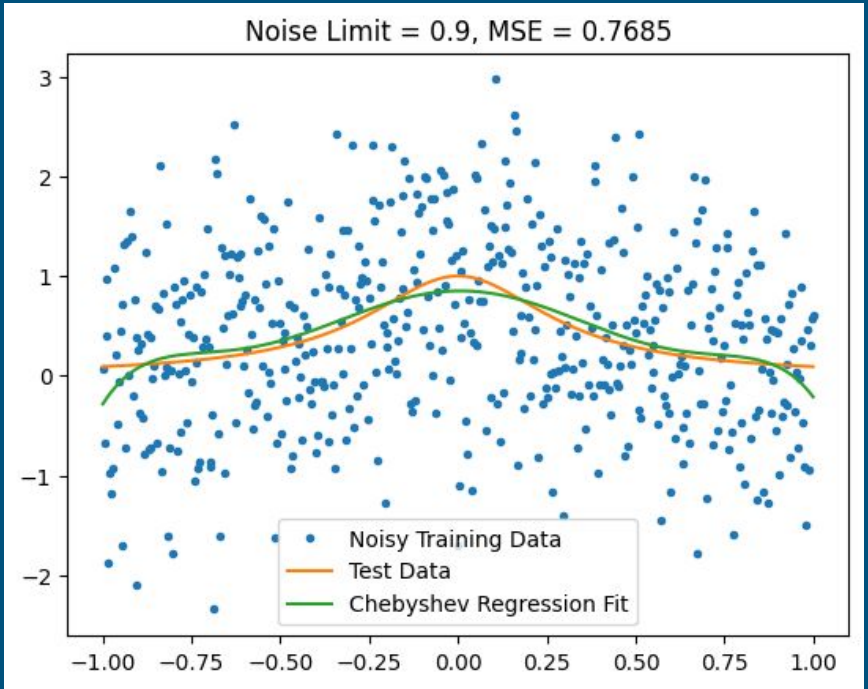
Python Graph





Julia Graph


Noise Limit = 0.1, MSE = 0.0117
Noisy Training Data
Test Data
Chebyshev Regression Fit


Noise Limit = 0.9, MSE = 0.7685
Noisy Training Data
Test Data
Chebyshev Regression Fit

Analysis + Impacts

Both packages return fairly similar results as a baseline but differ when given different noise limits.

Accuracy Discrepancies:

- Noise limit: 0.1
- Python MSE: 0.0121
- Julia MSE: 0.0117

Runtime Discrepancies:

- Python: 0.76403220900010 64 seconds
- Julia: 0.118733 seconds (206.14 k allocations: 16.594 MiB, 74.65% compilation time)

Origin of Discrepancies

The cause of these differences likely originates from the fact that Julia is known for its high performance, especially when it comes to linear algebra operations, which makes linear regression models run faster in Julia compared to Scikit-learn. Other variables that likely contribute to these discrepancies include the fit method and the fact that SciKit learn is deep learning oriented, meaning there is more going on in the back end than a purely linear algebra focused library. It comes down to the two libraries being geared towards different applications.

Questions?

Feel free to reach out to us with any questions!

We are available on Zulip, or by email:

- Manoj Yeddanapudy (maye7573@colorado.edu)
- Matthew Labay (mala5209@colorado.edu)
- Nolan Ales (noal5803@colorado.edu)

Thank You!