Extensible NcML for AI/ML Ready Data on the THREDDS Data Server

Leo Matak[1,2], Tara Drwenski[1], Hailey Johnson[1], Thomas Martin[1];
[1]University Corporation for Atmospheric Research (UCAR), NSF Unidata Program Center, Boulder, USA;
[2]Department of Civil and Environmental Engineering, University of Houston, Houston, Texas, USA

Summary

- ➢ The THREDDS Data Server (TDS) hosts a vast array of data
- ➢ For certain Machine Learning applications, data preprocessing is desirable
- ➢ In this project I added a mechanism for custom, server-side data processing (in Java) which allows an implementation of any preprocessing routines

How to do it?

- ➢ Implement the service provider by following the interface shown below
- ➢ Write the NcML to define the desired data transformation
- ➢ Results? Data integrity while enabling virtual preprocessing directly on the server

Introduction

- The THREDDS Data Server (TDS) is a web server developed by NSF Unidata, a program under the University Corporation for Atmospheric Research (UCAR).
- The TDS provides access to scientific datasets using standard data access protocols.

Generic server side data processing

- **Data preprocessing** can enhance AI performance
- This summer internship project aimed to **develop extensible NcML for data preprocessing using the service provider mechanism of Java.**
- **Server administrators to set up any desired data transformations.**
- Admins can then **virtually transform** and preprocess their data **without altering the original datasets**.

GOAL

How it works?

- We use Java's Service Provider pattern, where a concrete implementation of a service interface can be loaded at runtime without any hardcoding or modifying of the existing code.
- Admins can now make a custom implementation of the following interface:

```
public interface EnhancementProvider {

  boolean appliesTo(Enhance enhance, AttributeContainer
attributes, DataType dt);

  Enhancement create(VariableDS var);
}
```

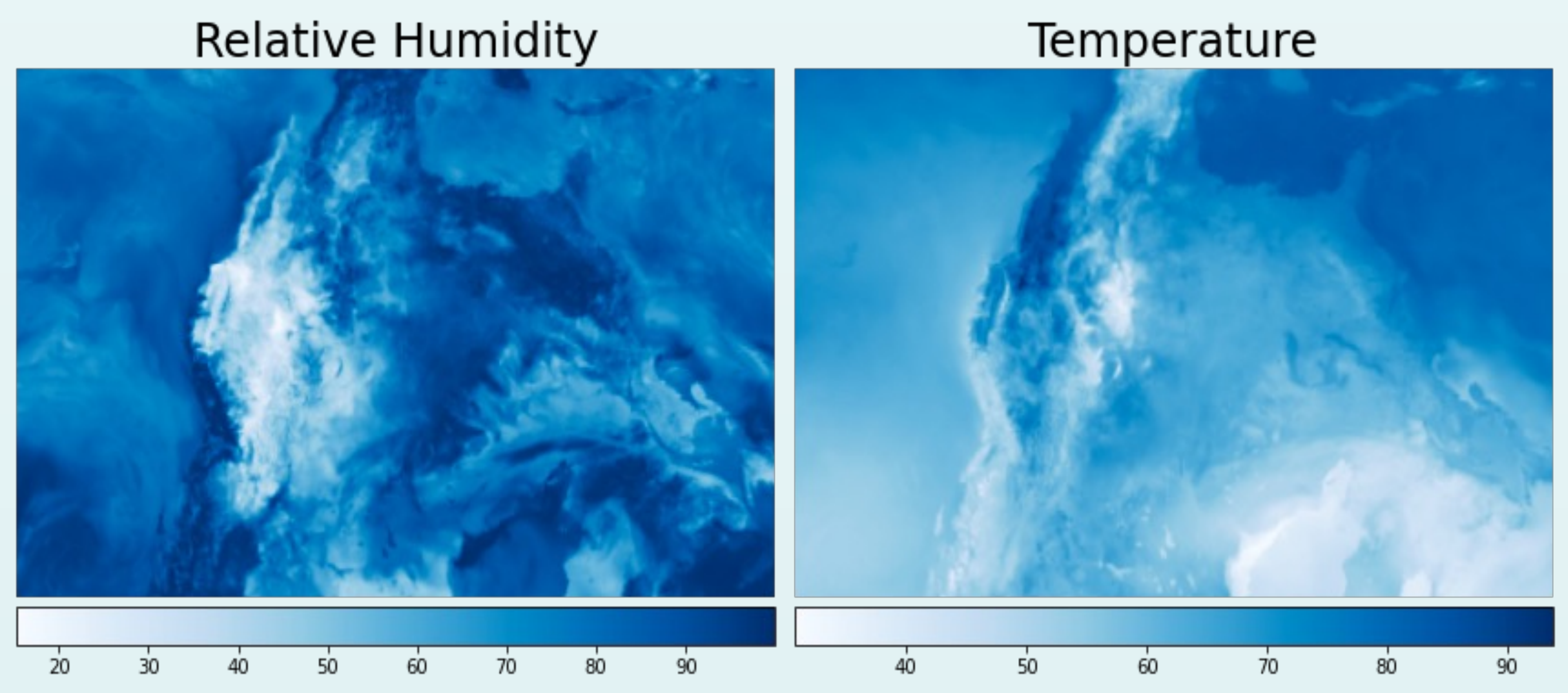
- The implemented data transformation is defined using the NcML (which stands for NetCDF Markup Language) for any dataset and variable
- Here shown is an example of using the **Classifier** class on temperature raw data:

```
<variable name="Temperature_height_above_ground">
    <attribute name="classify" value="0 65 0; 65 85 1; 85 inf 2"/>
</variable>
```

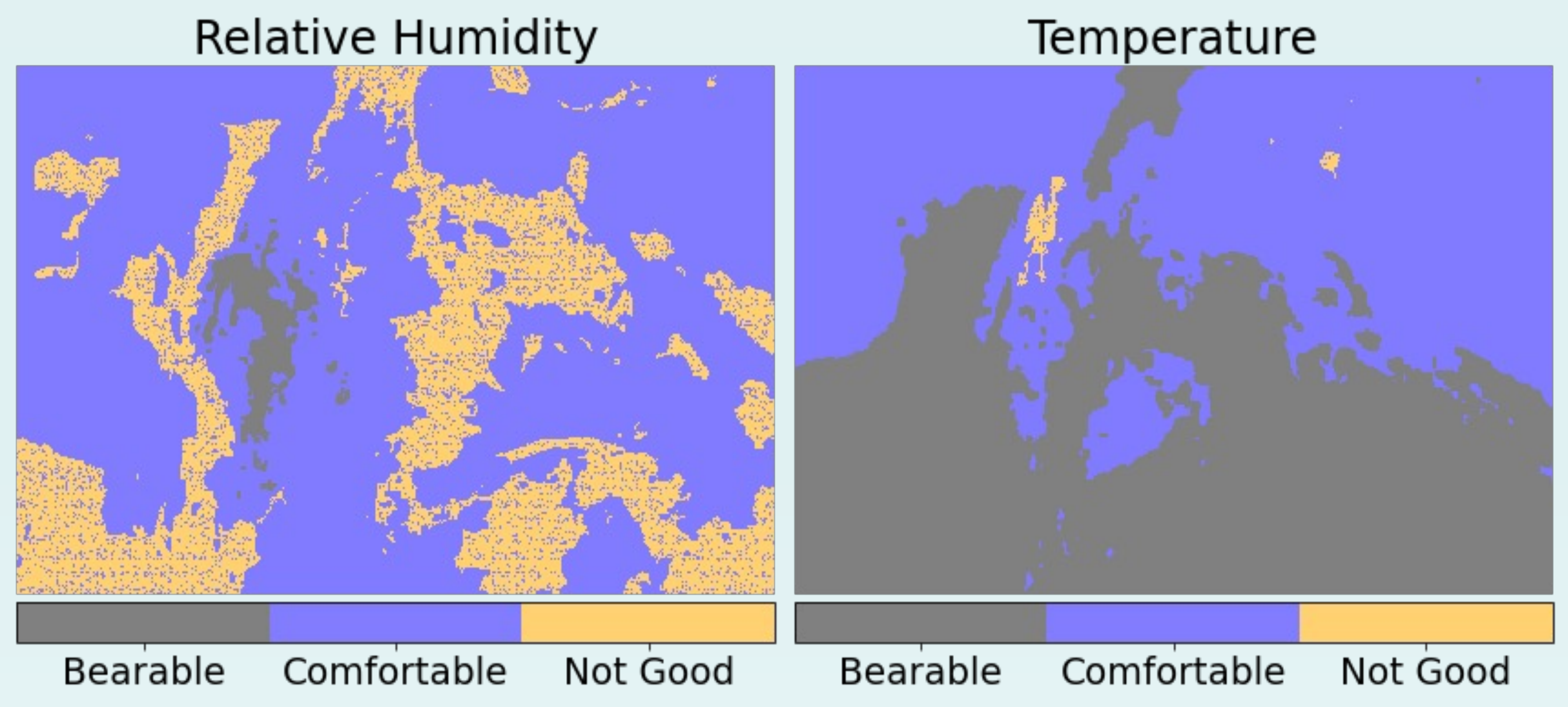
- The above code will perform the following classification:

Temperature [F]	[0,65)	[65,85)	[85,inf)
Assigned Class	0 (bearable)	1 (Comfort.)	2 (Not good)

Raw Data



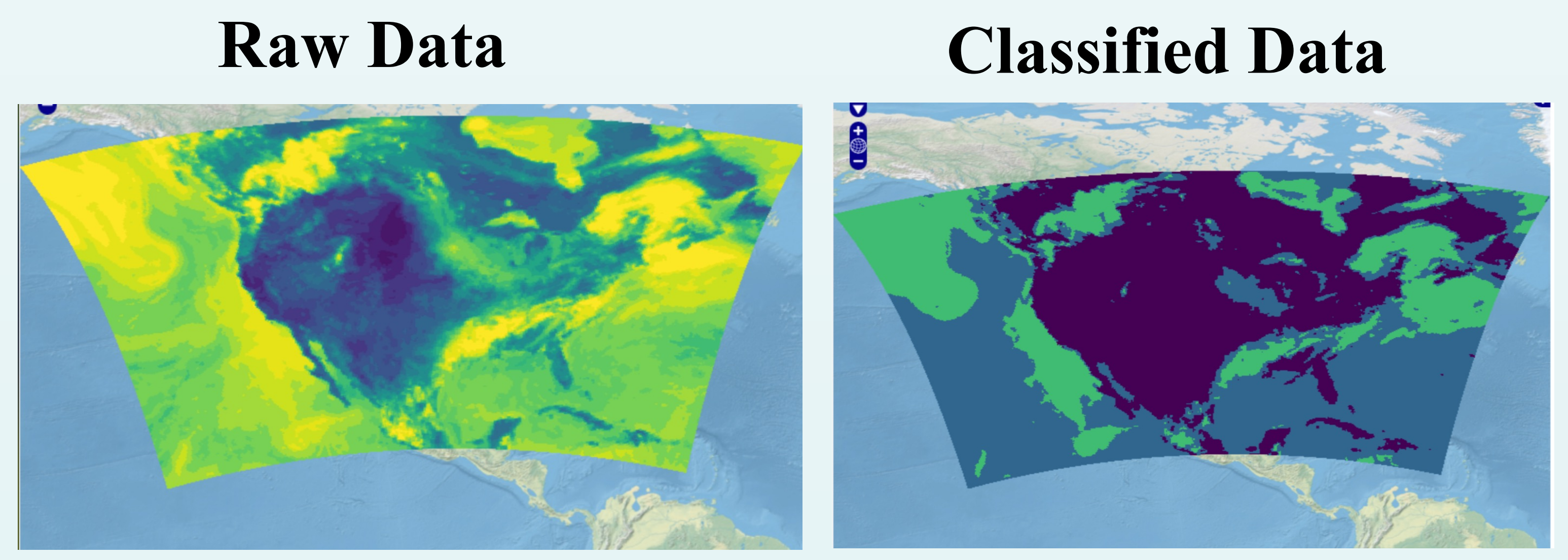
Classified Data



Data processing directly on TDS

- Data can be virtually modified directly on the server side
- Server-side computation preserves data integrity and encourages reproducible workflows
- Here shown is another example from TDS[1] where relative humidity variable is being classified as specified with the NcML

```
<variable name="Relative_humidity_height_above_ground">
   <attribute name="classify" value="0 45 0; 45 75 1; 75 100 2"/>
</variable>
```



Conclusion

- Users can now easily preprocess data using our extensible NcML service.
- Developed solution offers generic server-side data processing.
- TDS Administrators can configure any necessary data transformations.
- Users of TDS, latest snapshot, can benefit from pre-established preprocessing routines set up by admins.
- AI/ML data preprocessing, for instance the transformation in the scikit-learn package can be seamlessly integrated, offering versatile data transformation capabilities.

This project was part of my NSF Unidata 2024 Summer Internship

References

[1] https://thredds-test.unidata.ucar.edu/thredds

This material is based upon work supported by NSF Unidata under award #2403649 from the U. S. National Science Foundation. Any opinions, findings and conclusions or recommendations expressed in this material do not necessarily reflect the views of the NSF.