

# Artificial Intelligence for Earth System Science Summer School

NCAR  
UCAR



VAISALA



# Acknowledgements

## Organizing Committee

- Taysia Peterson
- David John Gagne
- Karthik Kashinath
- Rich Loft

## Hackathon Development Team

- Charlie Becker
- Gabrielle Gantos
- Keely Lawrence
- Gunther Wallach
- Ankur Mahesh
- Bill Petzke
- Aaron Bansemer
- Matt Hayman
- Siyuan Wang
- Alma Hodzic
- Andrew Gettelman
- Chih-Chieh (Jack) Chen

## Sponsors

- UCAR President's Council
- Vaisala: Eric Gritit
- Amazon Web Services: Zac Flamig
- NCAR Machine Learning Data Commons Reinvestment Project

## Livestream and Slido Team

- Paul Martinez
- Kelvin Tavaréz
- Mary Andreski
- Lisa Larson
- Gail Rutledge

# Speaker Acknowledgements

## Monday

9:10: David John Gagne

10:20: Dorit Hammerling

11:30: Ryan Lagerquist

## Tuesday

9:00: Karthik Kashinath

10:10: Chaopeng Shen

11:20: David Hall

## Wednesday

9:00: Sue Ellen Haupt

10:10: Jebb Stewart

11:20: Katie Dagon

## Thursday

9:00: Amy McGovern

10:10: Imme Ebert-Uphoff

11:20: Mike Pritchard

## Friday

9:00: Mustafa Mustafa

10:10: Pierre Gentine

11:20: Claire Monteleoni



# Building a Strong Foundation: Defining ML Problems and Preprocessing

**David John Gagne**

*Machine Learning Scientist*

*National Center for Atmospheric Research*

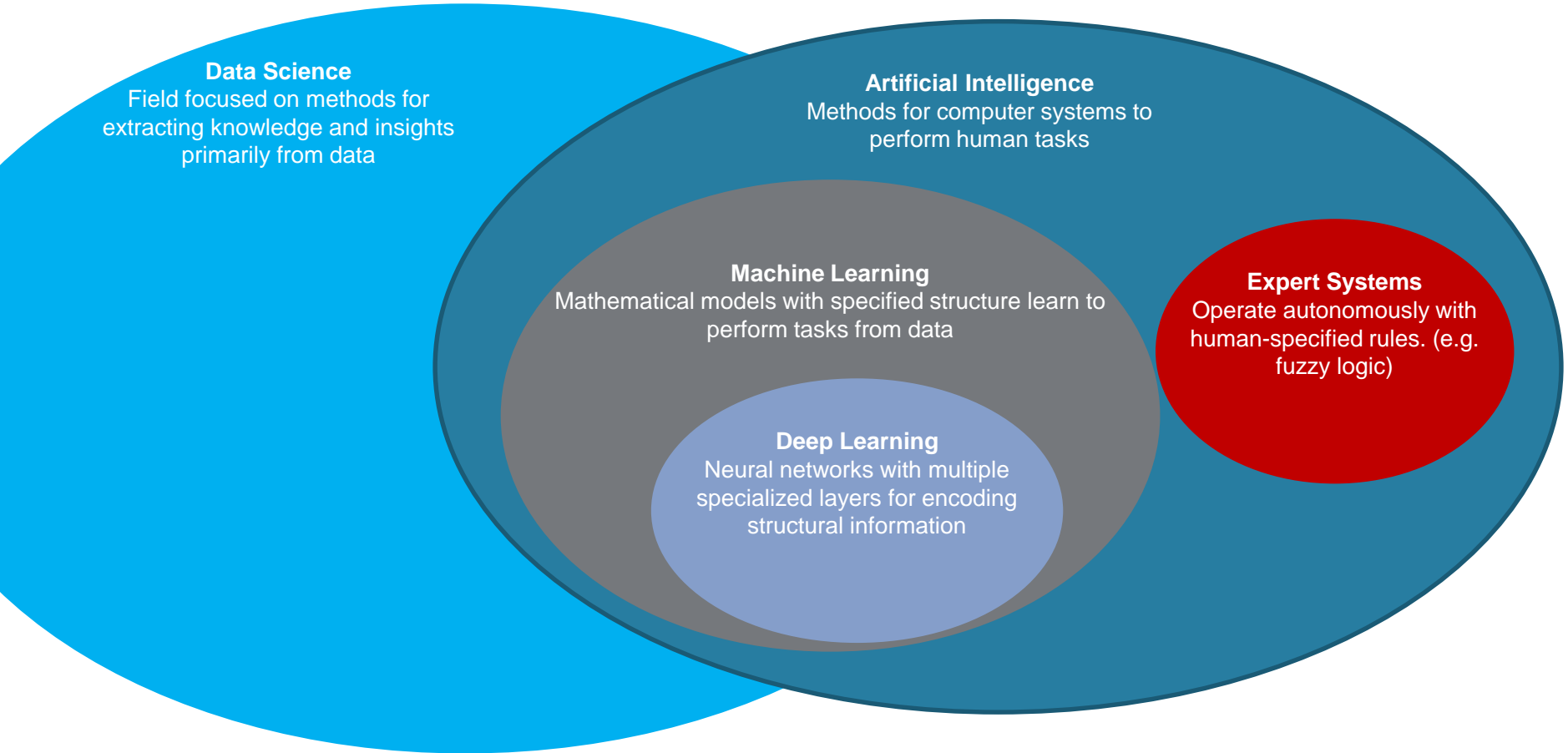
NCAR  
UCAR

June 22, 2020

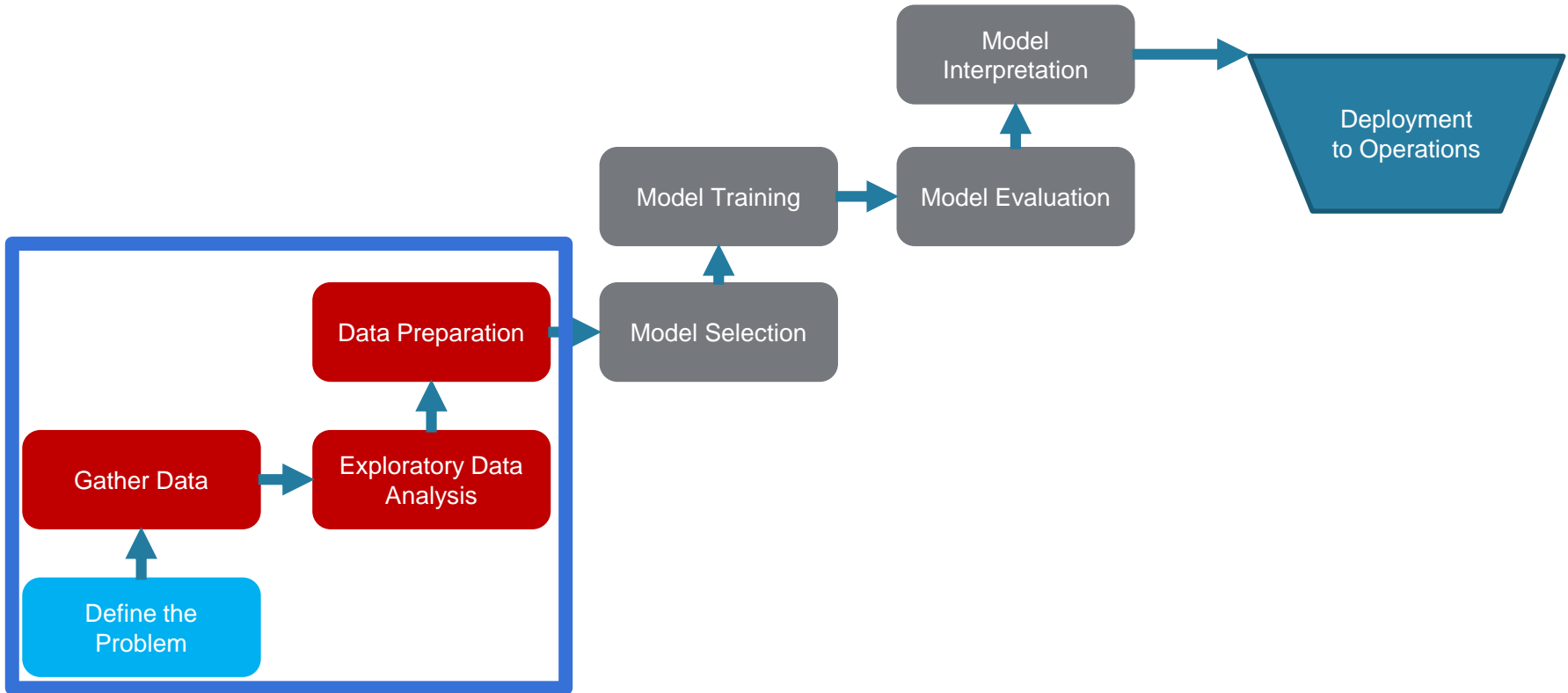
# Motivation

- Interest in AI and machine learning in the atmospheric sciences has exploded in the past three years
- Much of the attention has been focused on the algorithms
- However, choosing the right ML algorithm is not sufficient for creating a successful AI/ML system
- 80% of every machine learning project is spent on defining the ML problem and pre-processing the data
- This lecture will discuss the many important choices that must be made before training any ML models

# The Data Science Taxonomy



# The Machine Learning Pipeline



# Should I choose machine learning?

## When ML works well

- Moderate to high coverage of possible space of inputs
- At least some plausible connection between inputs and outputs
- Non-ML approaches are too expensive or error-prone

## When ML works poorly

- Limited data coverage of possible inputs
- Little connection between inputs and outputs
- Current approaches to solving the problem are already effective



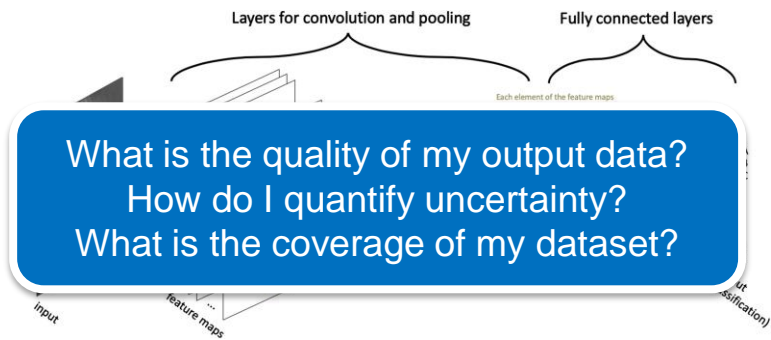
# Defining the Problem

- The most important part of any machine learning project is defining the problem properly
- Questions to ask:
  1. What are the ultimate goals of this project?
  2. What are the specific inputs and outputs needed to achieve the goals?
  3. What data are available for the inputs and outputs? What are the data limitations?
  4. What are the problem constraints (time, space, latency, physical)?
  5. How is the problem currently solved, and what are the limitations of those methods?

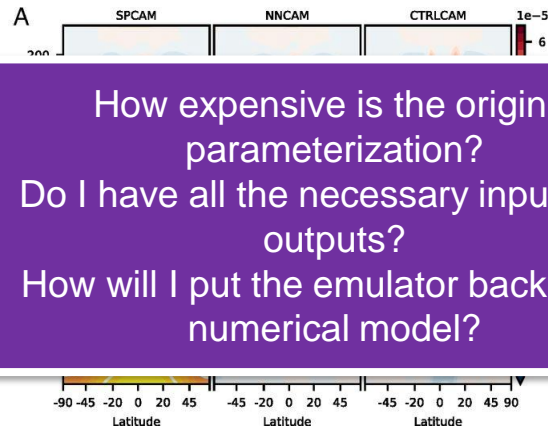
# Machine Learning Problem Examples



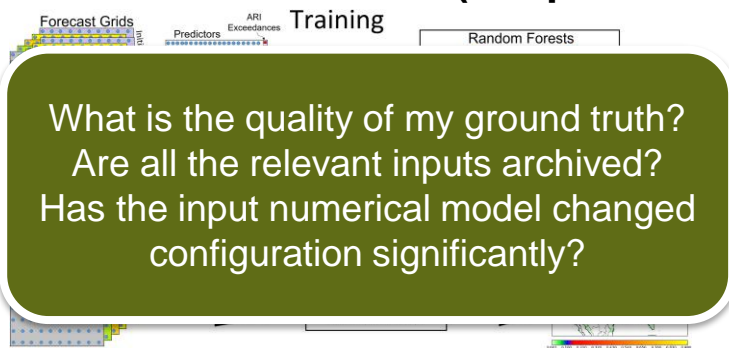
## Object Segmentation (Kurth et al. 2018)



## Observation Diagnosis (Wimmers et al. 2019)



## Parameterization Emulation (Rasp et al. 2018)



## Model Post-Processing (Herman and Schumacher 2018)

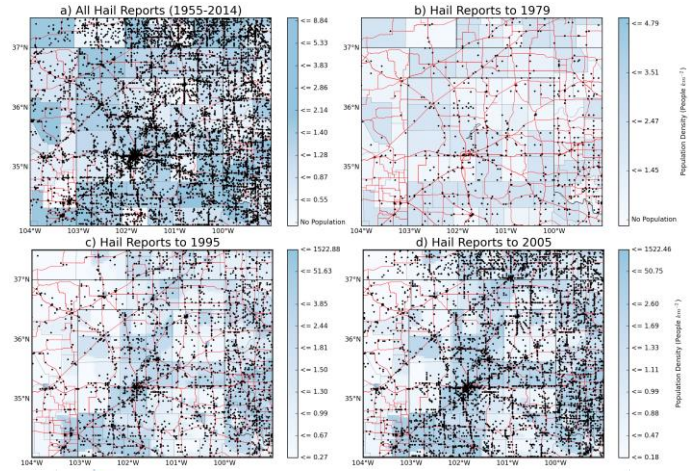
# Data Gathering

## Choose your data gathering adventure

	Use Existing Data	Gather Your Own Data	Generate Synthetic Data
Benefits	Long archive Freely available Retrieve necessary subsets Can compare different versions	Gather exactly what you need Control experiment design	Control properties of data Repeatable
Perils	File formats Lack of metadata/ provenance Inappropriate variables or pre-processing for problem Biased sampling	Expensive Quality of data gathering No access to past Your responsibility to avoid bad data sampling and processing practices	May be computationally expensive Not from real world Setting up infrastructure is time-consuming

# Bias in Data

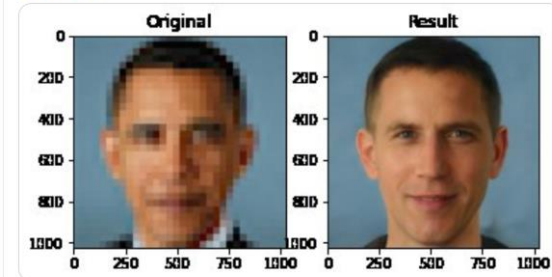
- Observational data are not collected randomly
- Data may be biased by the collection process, especially with report datasets
- ML models trained on biased data and biased assumptions will propagate that bias into their predictions
- Diverse datasets should be gathered for both training and validation



Allen and Tippett 2015



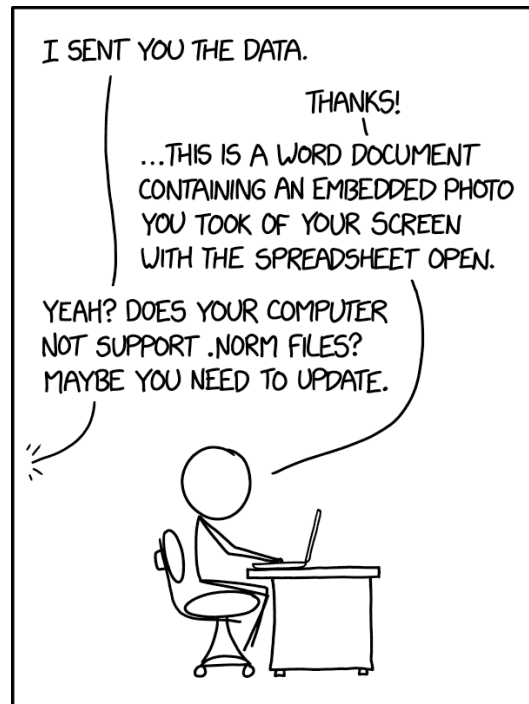
Replying to @tg\_bomze



6:14 AM · Jun 20, 2020 · Twitter for Android

# Data File Formats

- Pick data file formats based on the following criteria
  - Structure of the data
  - Size of the dataset
  - Needs of the users
- Your raw data will likely not be in an ideal format for machine learning
  - Legacy file format
  - Chunked or strided with a less than ideal memory access pattern
  - Missing or inconsistent variable names, formatting, etc.
- Key decision point: text vs binary
  - Text: easier to inspect, more portable, but is bulky and prone to manual mis-formatting
  - Binary: can store large datasets in a consistent format compactly, but requires special libraries to read



SINCE EVERYONE SENDS STUFF THIS WAY ANYWAY, WE SHOULD JUST FORMALIZE IT AS A STANDARD.

From xkcd

# Tabular and Structured Data Formats

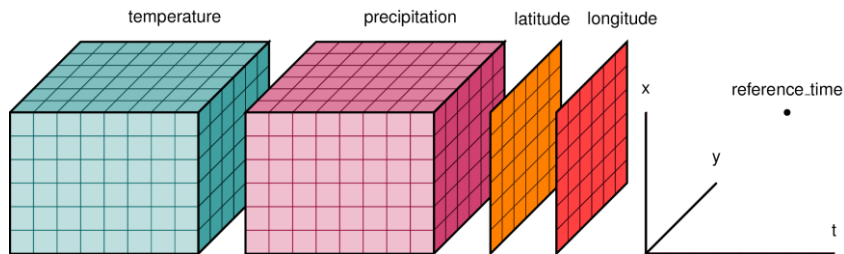
- CSV: plaintext files containing data separated by columns. Very portable and readable but is row-oriented and doesn't scale well for multi-GB or TB datasets
- Apache Parquet: binary open source columnar data format that offers compression and support for different data formats
- XML, JSON, and YAML: hierarchical text-based data formats with decreasing amounts of extra syntax around data. Useful for config files and storing non-tabular data

# Geospatial Data Formats

- GRIB: World Meteorological Organization standard format for gridded data. Can be highly compressed, but metadata is stored outside files, which is a problem for custom variables
- netCDF4/HDF5: hierarchical, self-describing, binary data format that supports compression of individual variables. Works well on supercomputers but performs poorly in the cloud
- Zarr: new binary hierarchical data format that breaks dataset into a large number of small binary files. Better suited for cloud data storage

# Data Preparation: Transformations

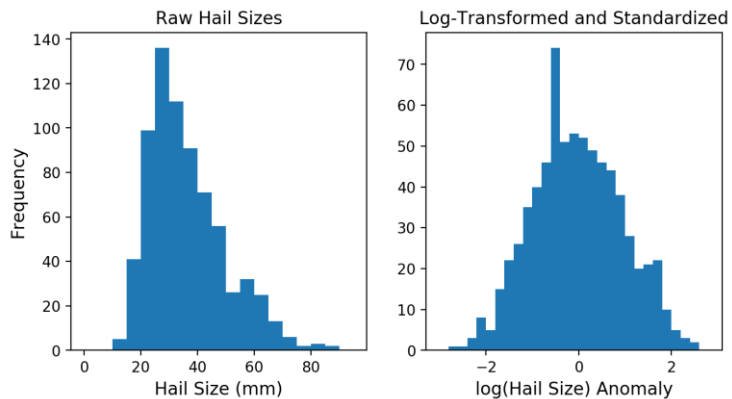
## Reshaping and Sampling



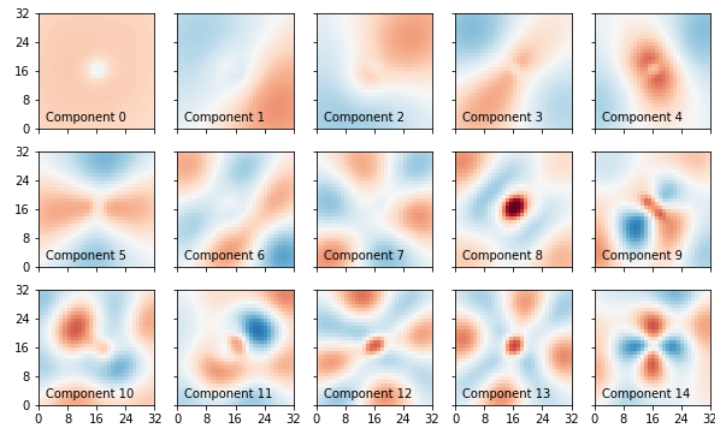
From xarray.pydata.org

Time	Lat	Lon	Temp	Precip
0	35	-124	28	0
1	32	-94	15	24
2	45	-53	-2	5
...	...	...	...	...

## Data Scaling and Standardization

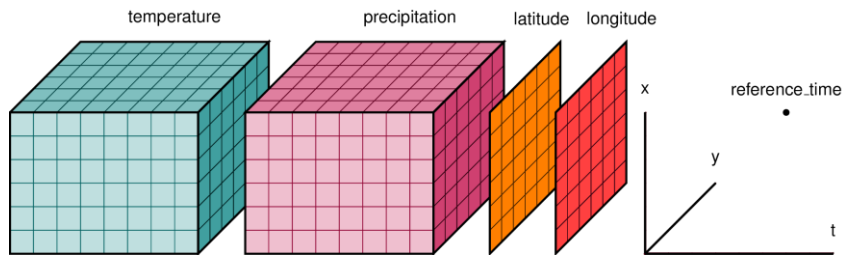


## Dimensionality Reduction





# Reshaping

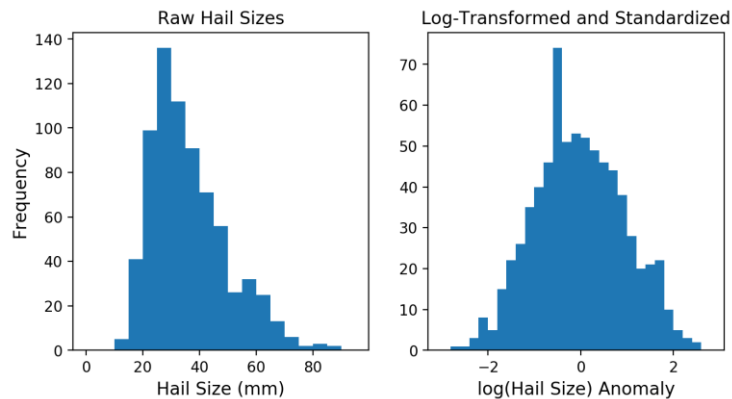


Time	Lat	Lon	Temp	Precip
0	35	-124	28	0
1	32	-94	15	24
2	45	-53	-2	5
...	...	...	...	...

- Gridded ESS data products are arranged in time-> variable->grid cell order for traditional data analysis purposes
- ML models generally require a vector or tensor of multiple variables at one location and time
- If all variables in same file, `xarray.Dataset.to_dataframe` function will automatically convert gridded data to tabular form
- Sampling 2D patches requires custom code

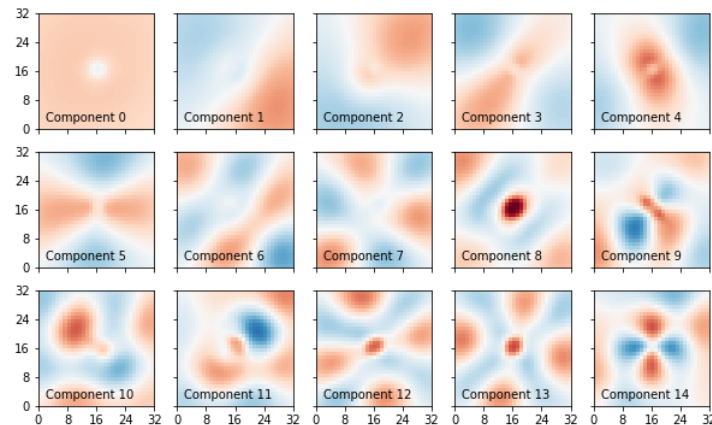
# Data Transform and Scaling

- Different input variables often have different ranges and distributions
- In regression and neural network models, weights are initialized in the same range, so inputs and outputs with larger ranges may have more influence
- Rescaling data ensures that all variables get similar consideration initially
  - Subtracting the mean and dividing by the standard deviation
  - Rescaling values from 0 to 1 with minimum and maximum
- Log or Box-Cox transform can make exponentially distributed values more Gaussian
- Scaling statistics should be calculated only on training data to prevent information leakage

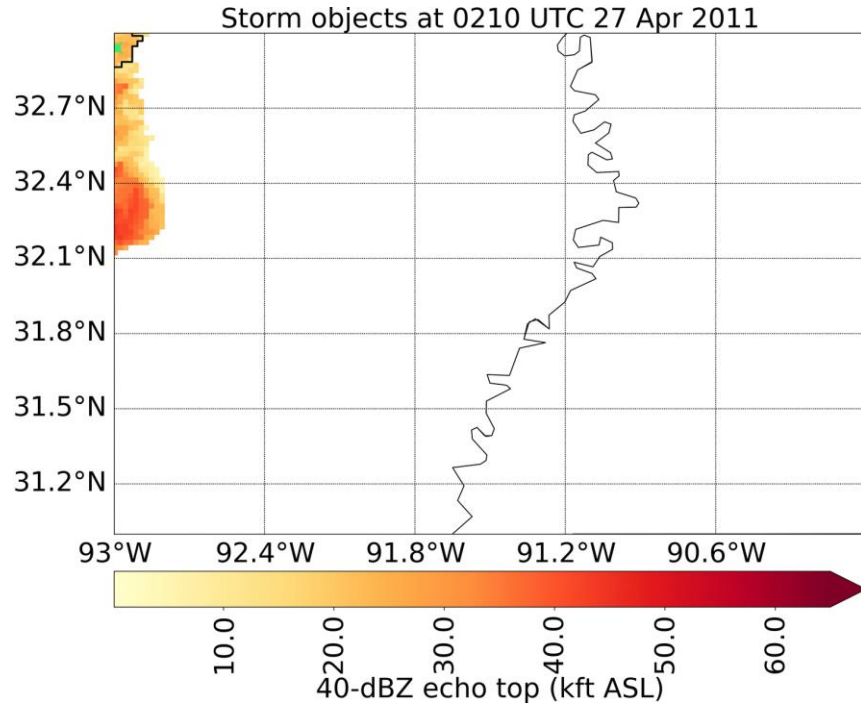


# Dimensionality Reduction

- Why reduce dimensionality?
  - Data contains more input variables than examples
  - Data has high frequency modes that are less relevant to the problem
  - Want to visualize in 2D space
- Dimensionality Reduction Methods
  - Principal Component Analysis
  - Fourier or Wavelet Transforms
  - t-distributed Stochastic Neighbor Embedding
  - Autoencoders



# Object Identification and Tracking



From Ryan Lagerquist

- Some Earth system phenomena can be identified and tracked as discrete events
- Heuristic object identification systems use fixed thresholds and computer vision techniques
- Tracking can be accomplished by matching objects in time with centroid or overlap calculations
- Lots of edge cases and parameter tradeoffs to make

# Feature Selection

- How to find the right balance between minimizing the number of inputs and maximizing performance?
- Global feature selection methods, like sequential forward selection
- Model-based feature selection, like LASSO
- Conditional feature selection, like decision trees

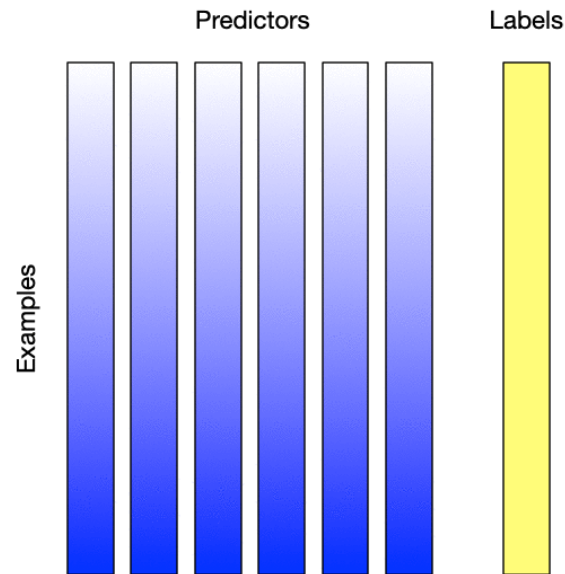
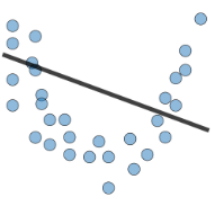
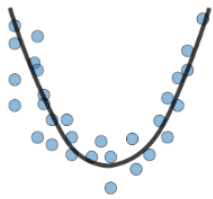
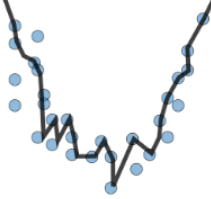
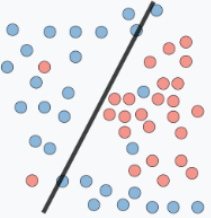
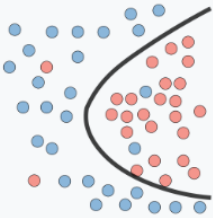
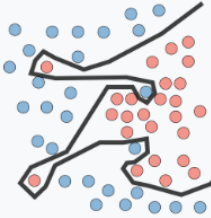





Illustration of sequential forward selection. At each step, the effect of each unselected predictor (blue bar) is tested and the best one (red bar) is chosen. This is repeated until performance no longer improves or all predictors have been chosen.

From McGovern et al. 2020

# Training/Validation/Test Sets

- Goal: produce a ML model that will generalize, or perform well operationally.
- How do we estimate generalization ability?
- Training Set
  - Used to optimize a model's weights or structure for one set of hyperparameters
  - More complex models will almost always improve on training set scores
- Validation Set
  - Used to assess the performance of one or more models
  - Can be used to choose hyperparameters
  - Should be independent of training data unless cross-validation is used
- Test Set
  - Data unseen during training and validation
  - Should be used for final assessment and not model selection
- How to split the data
  - If data points are independent, random splits are fine
  - Splitting process should account for spatial and temporal dependencies

	Underfitting	Just right	Overfitting
Symptoms	<ul style="list-style-type: none"><li>• High training error</li><li>• Training error close to test error</li><li>• High bias</li></ul>	<ul style="list-style-type: none"><li>• Training error slightly lower than test error</li></ul>	<ul style="list-style-type: none"><li>• Very low training error</li><li>• Training error much lower than test error</li><li>• High variance</li></ul>
Regression illustration			
Classification illustration			
Deep learning illustration			
Possible remedies	<ul style="list-style-type: none"><li>• Complexify model</li><li>• Add more features</li><li>• Train longer</li></ul>		<ul style="list-style-type: none"><li>• Perform regularization</li><li>• Get more data</li></ul>

# Summary

- Defining your ML problem well at the beginning will save you a lot of time later in the process
- Data gathering procedures can strongly impact the resulting ML model structure and predictions
- Pre-processing choices are key for efficiently transforming your data into a format suitable for machine learning