Introductory Overview of Artificial Intelligence (AI) and Machine Learning (ML)

Alison Appling, U.S. Geological Survey

STAC workshop on AI/ML

February 24, 2025

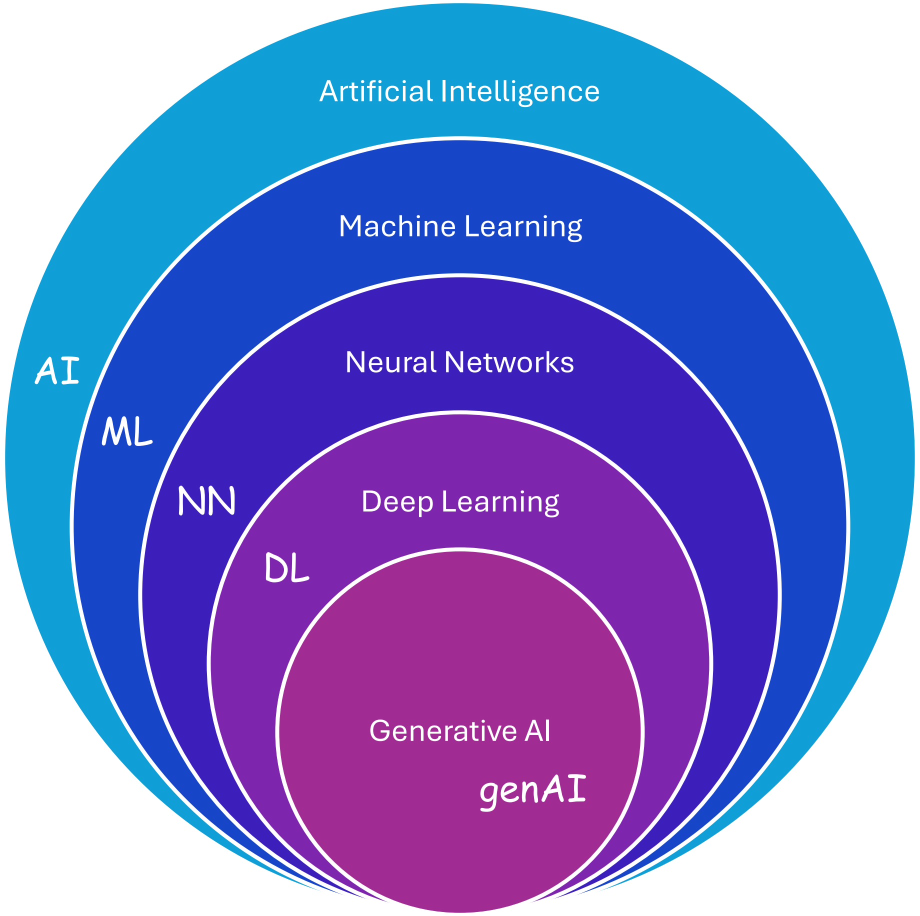
USGS
science for a changing world

Workshop on Leveraging Artificial Intelligence & Machine Learning to Advance Chesapeake Bay Research and Management

Chesapeake Science & Technical Advisory Committee (STAC)

stac

Artificial intelligence and some subsets



Act like a human

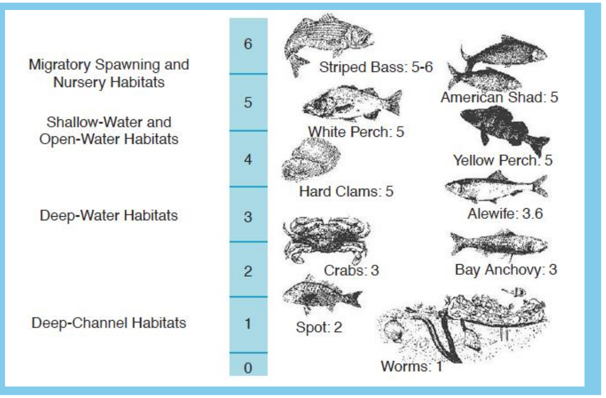
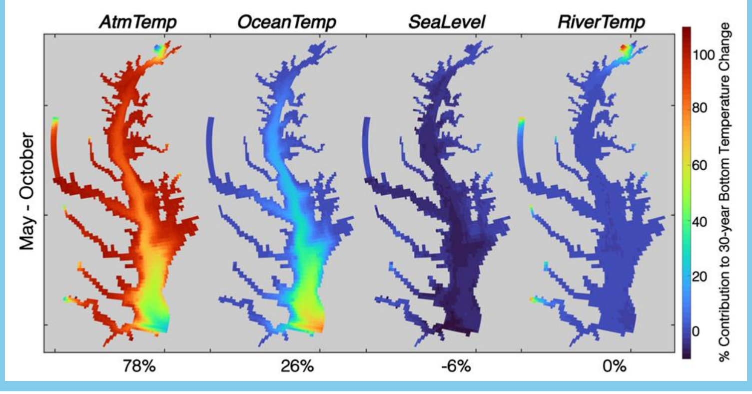
Learn to act like a human

Learn like a human

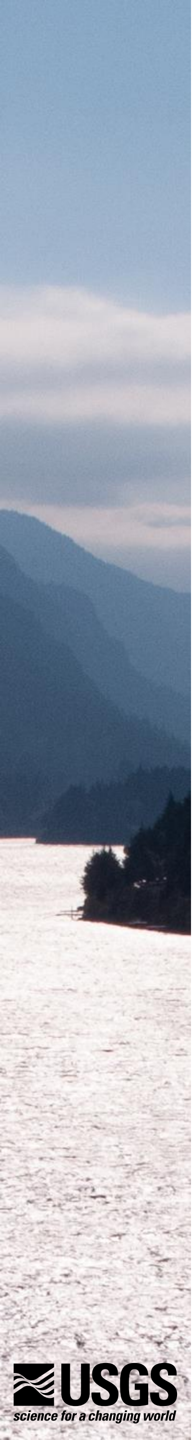
Learn like a smarter human

Learn from all the humans

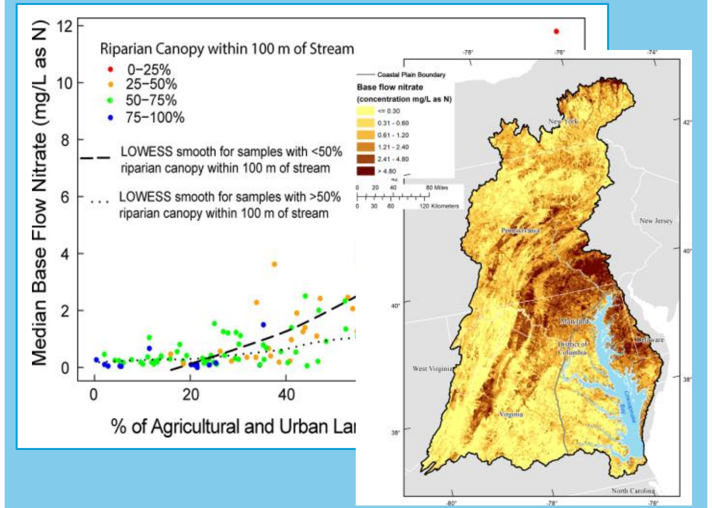
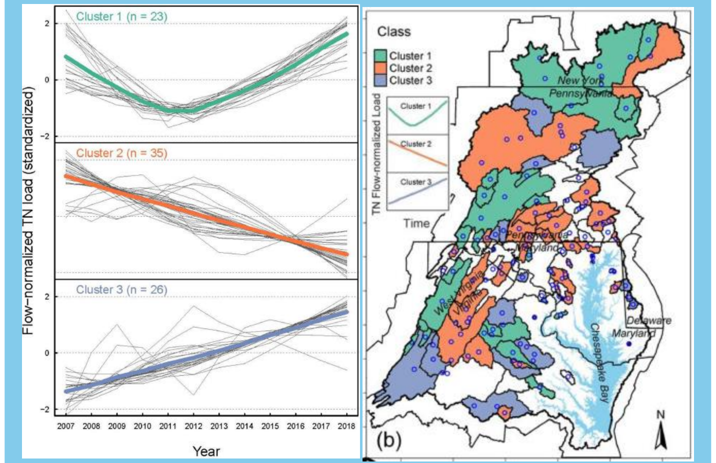
Machine learning paradigms

Categorical	Which location? Enough oxygen?	 
Continuous	How much?	

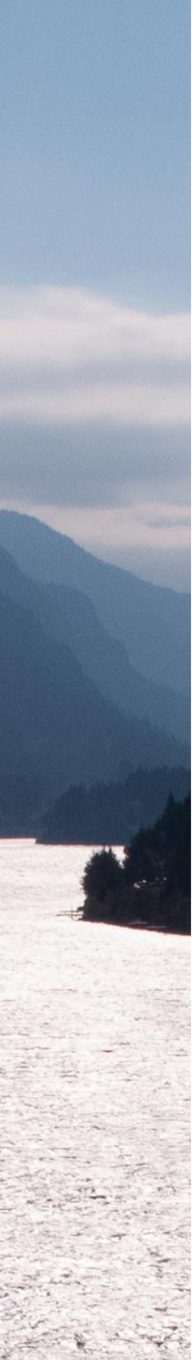
baydreaming CBP 2003
US-OCB



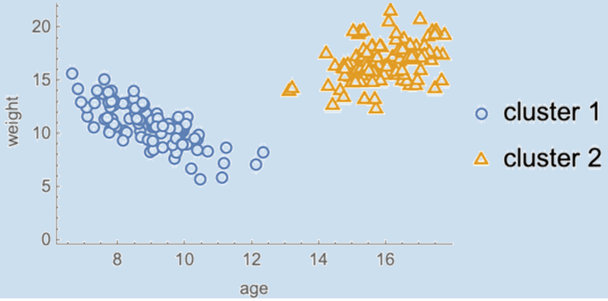
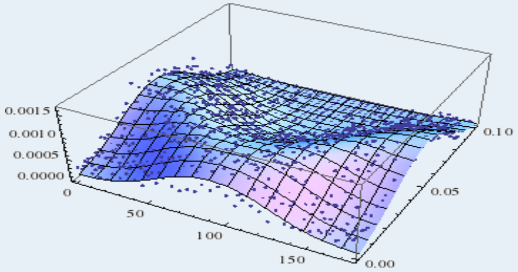
Machine learning paradigms

<table>
<tr><th></th><th>Supervised</th><th>Unsupervised</th></tr>
<tr><td>Categorical</td><td rowspan="2">Labeled training data
(true values known)

Wherry et al. 2000</td><td rowspan="2">Pile of observations
(need help organizing)

Zhang et al. 2022</td></tr>
<tr><td>Continuous</td></tr>
</table>

baydreaming wolfram

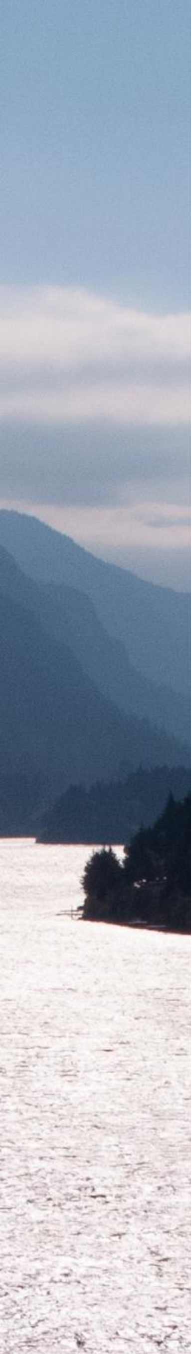


Machine learning paradigms

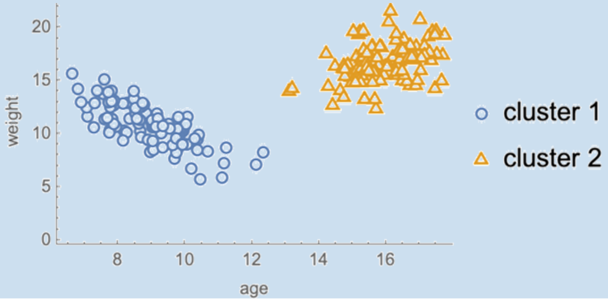
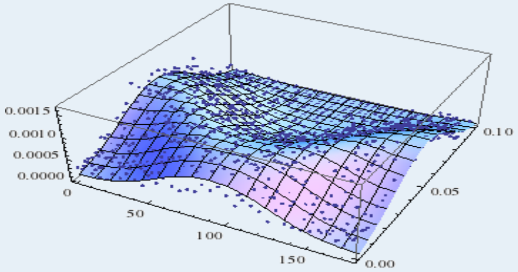
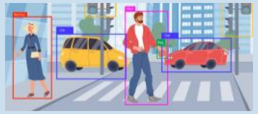
	Supervised	Unsupervised
Categorical	Classification  → Morel	Clustering 
Continuous	Regression 	Dimensionality Reduction 

wolfram wolfram
Dave Cote roboflow
DeepLobe Adobe Stock

Wolfran Intro to ML



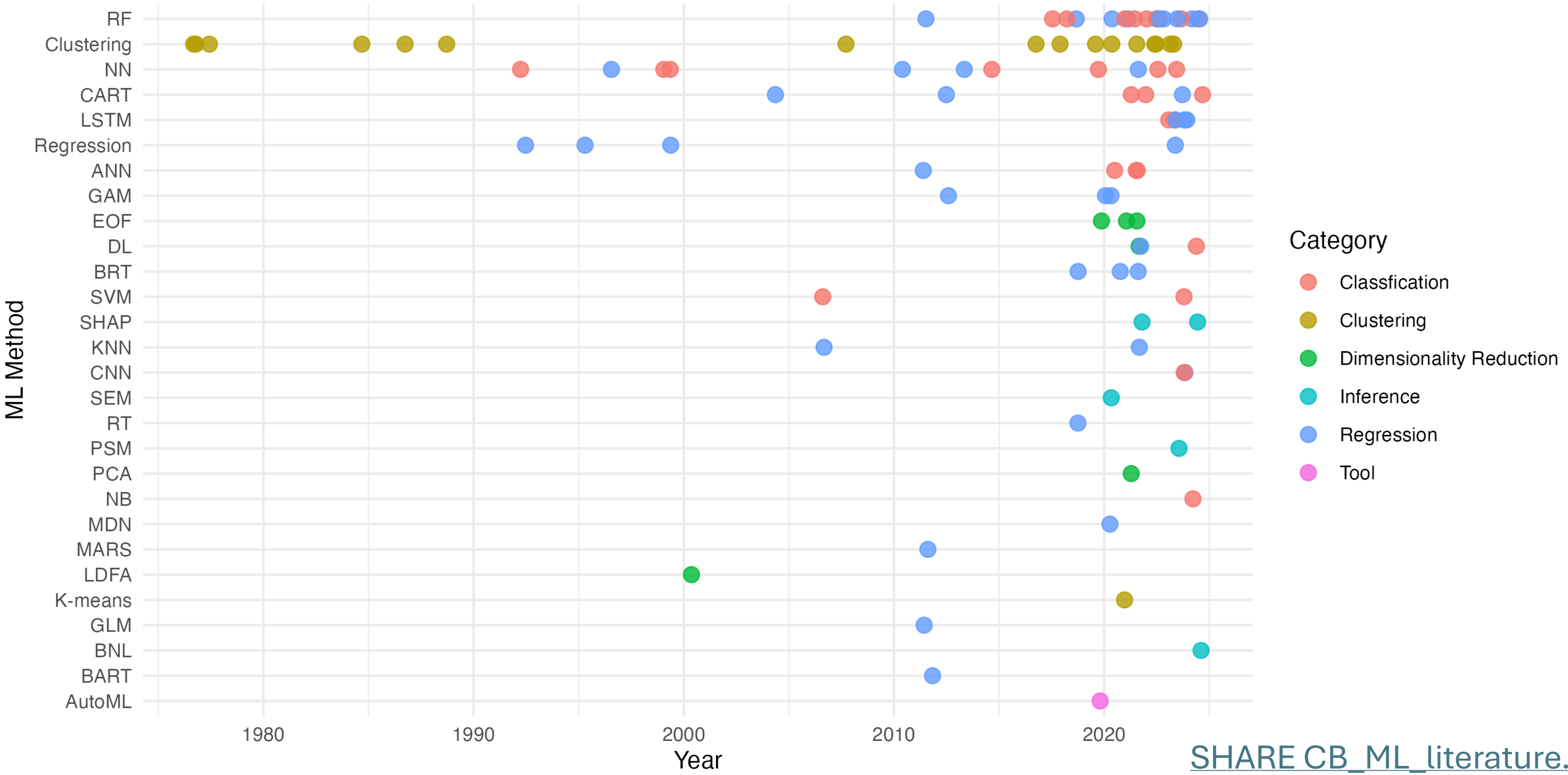
Machine learning paradigms

	Supervised	**Unsupervised**	***Reinforcement***
Categorical	Classification  → Morel	Clustering 	*and arguably also* *Semi-supervised* *Causal* *Online* *Active* *Transfer* *and other kinds of learning*
Continuous	Regression 	Dimensionality Reduction 	
Other	Object detection 	Anomaly detection Generative modeling 	Sequential actions …

wolfram wolfram
Dave Cote roboflow
DeepLobe Adobe Stock

Wolfram Intro to ML

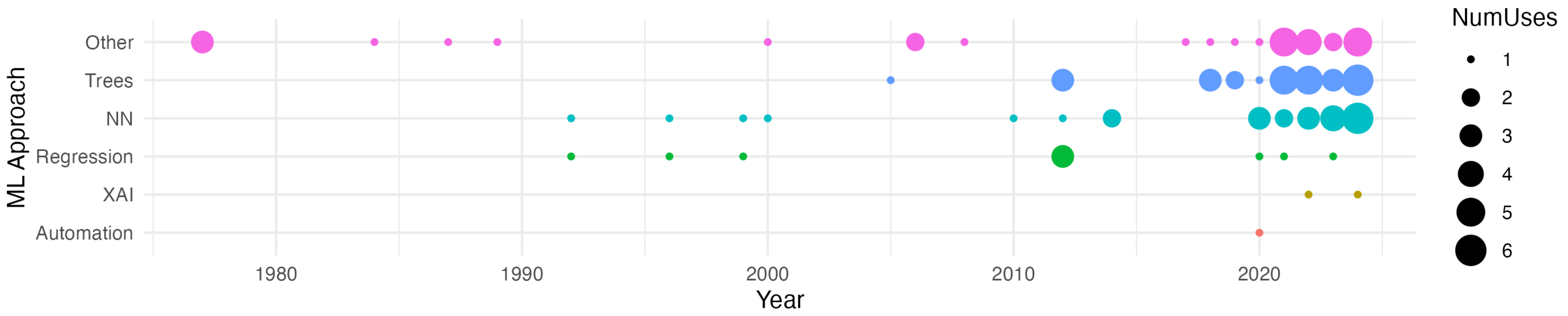
Chesapeake ML methods used



[SHARE CB_ML_literature.xlsx]

Preliminary information - Subject to revision. Not for citation or distribution.

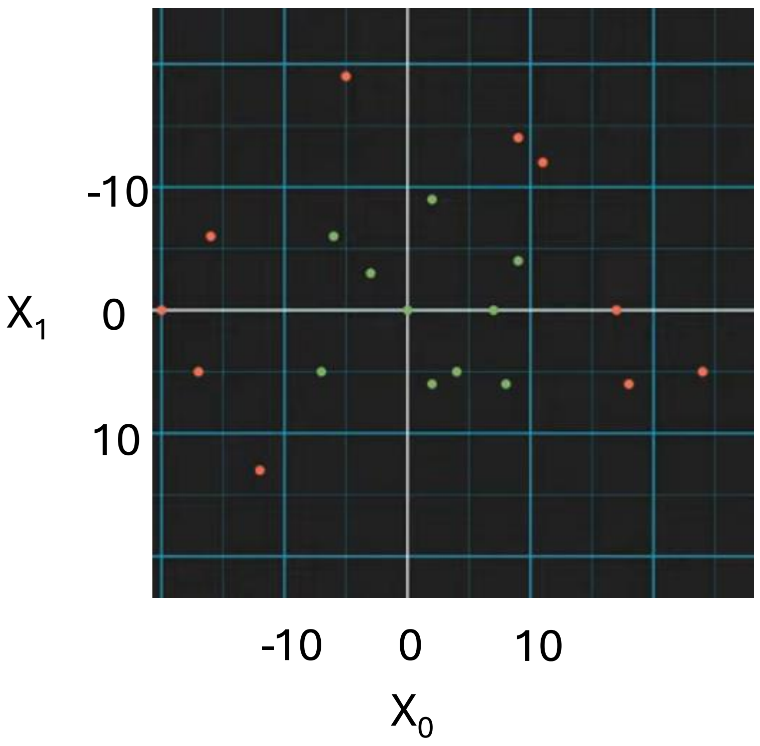
Chesapeake ML general approaches used



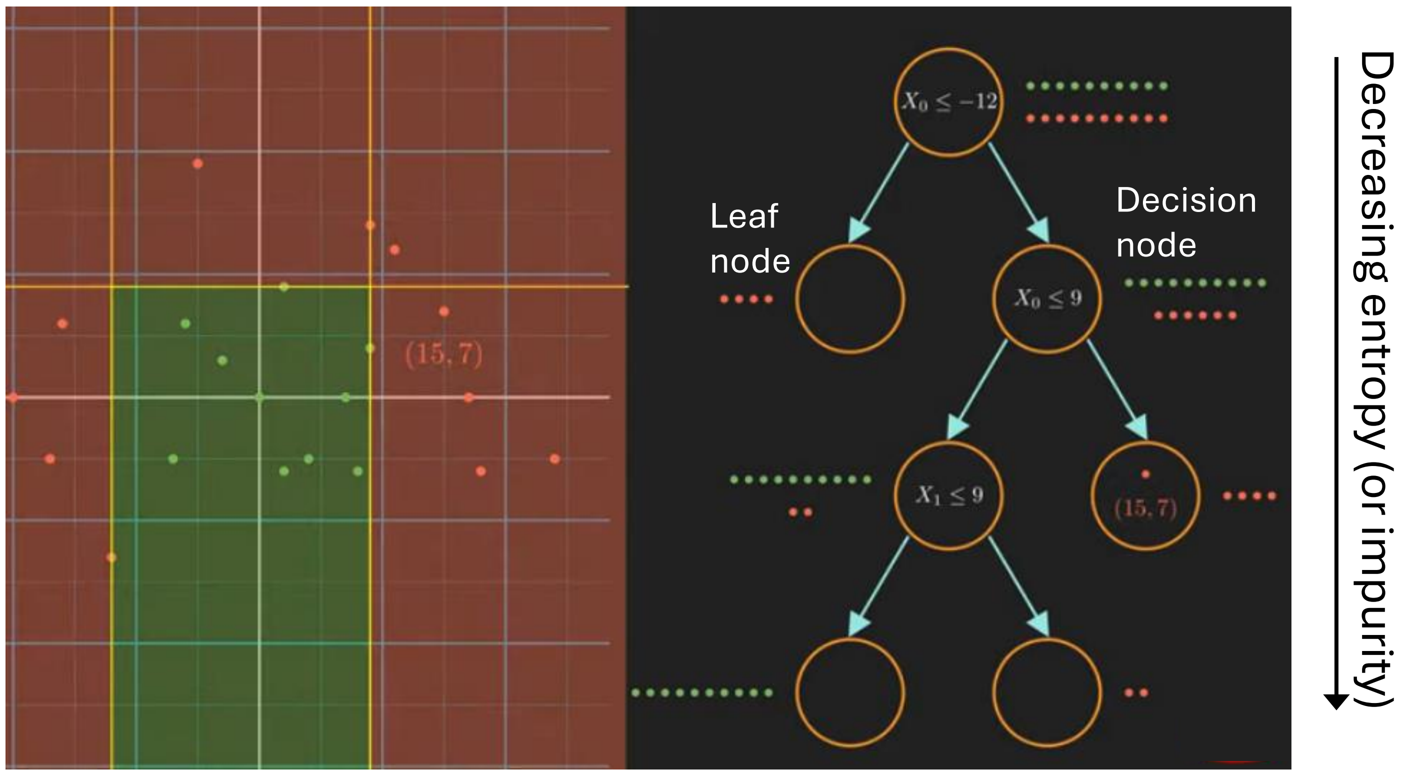
SHARE CB_ML_literature.xlsx

Decision trees for classification

Original data



Decision tree

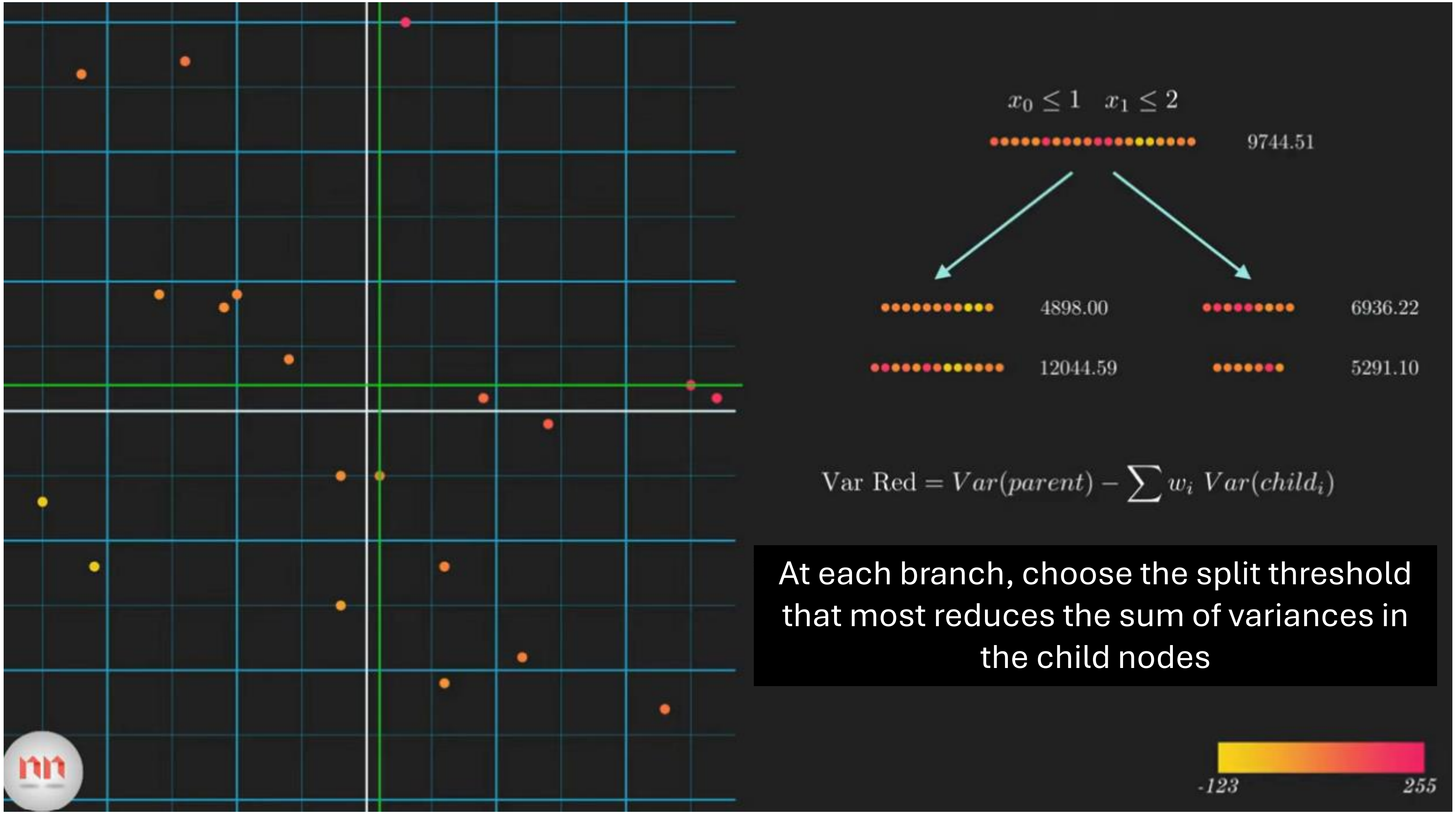


$$Entropy = \sum - p_i \log(p_i)$$

p_i = probability of class i

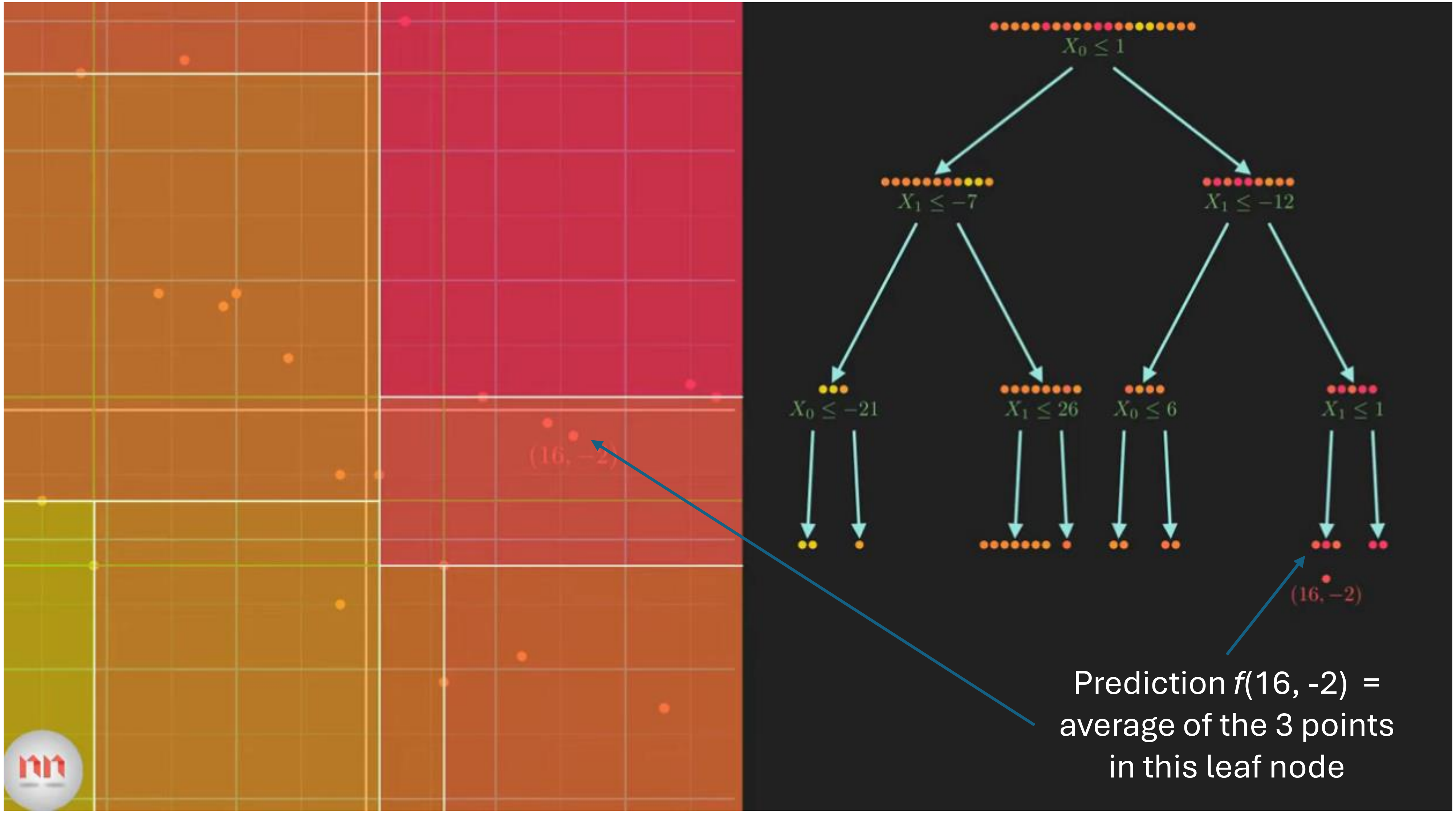
From [Tree Based Algorithms]

Decision trees for regression



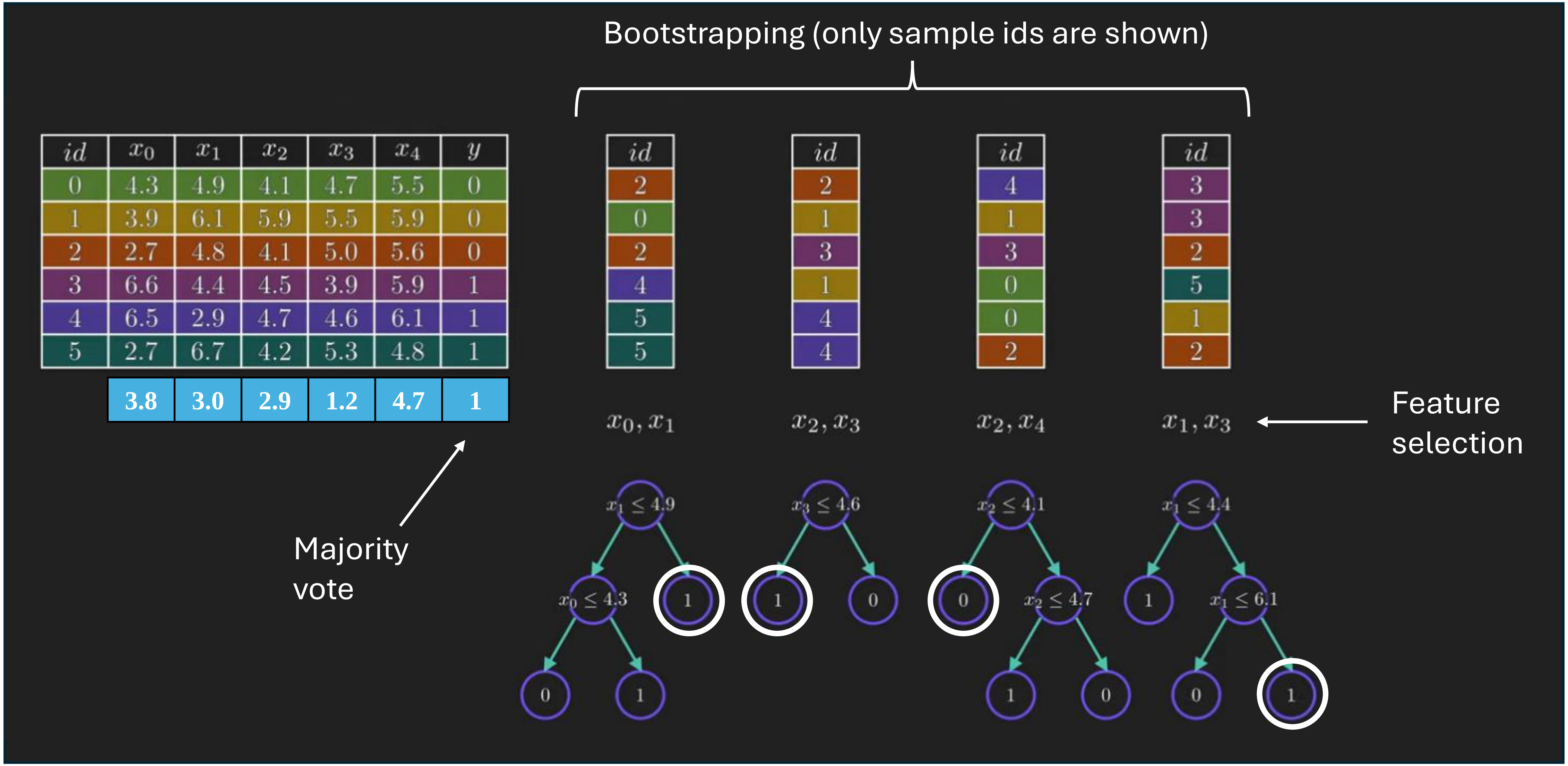
From [Tree Based Algorithms]

Decision trees for regression



From [Tree Based Algorithms]

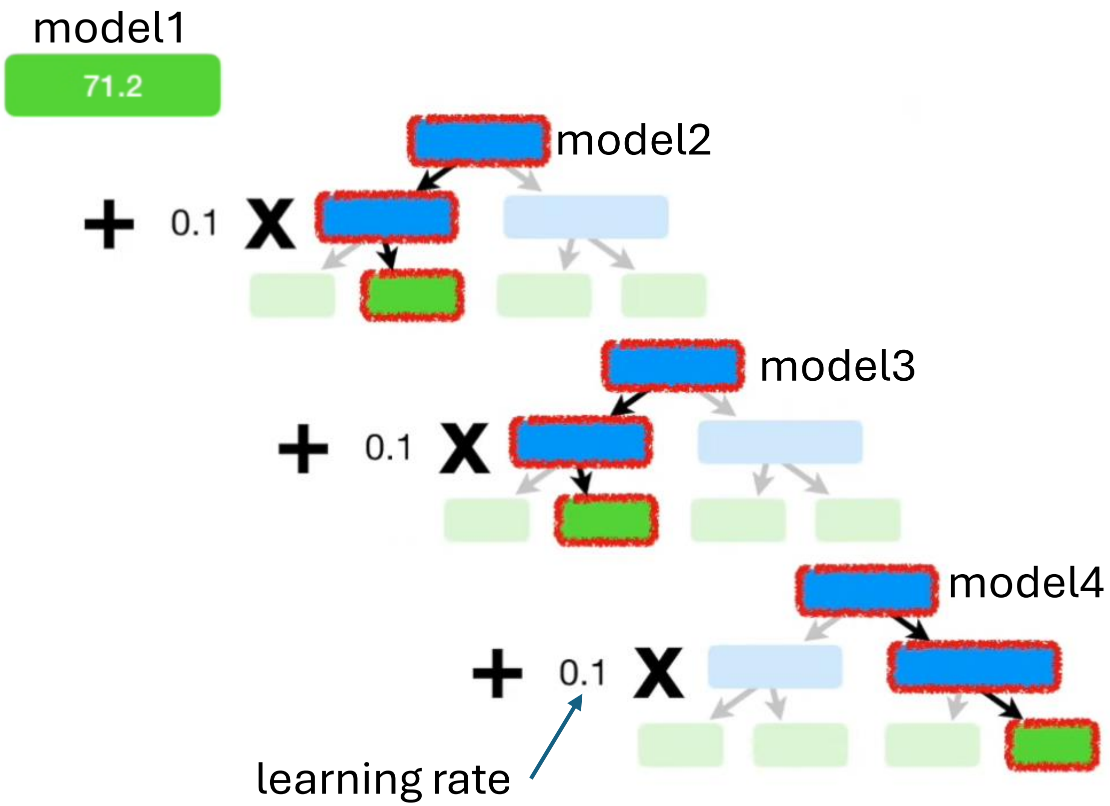
Random forest classification



From Tree Based Algorithms

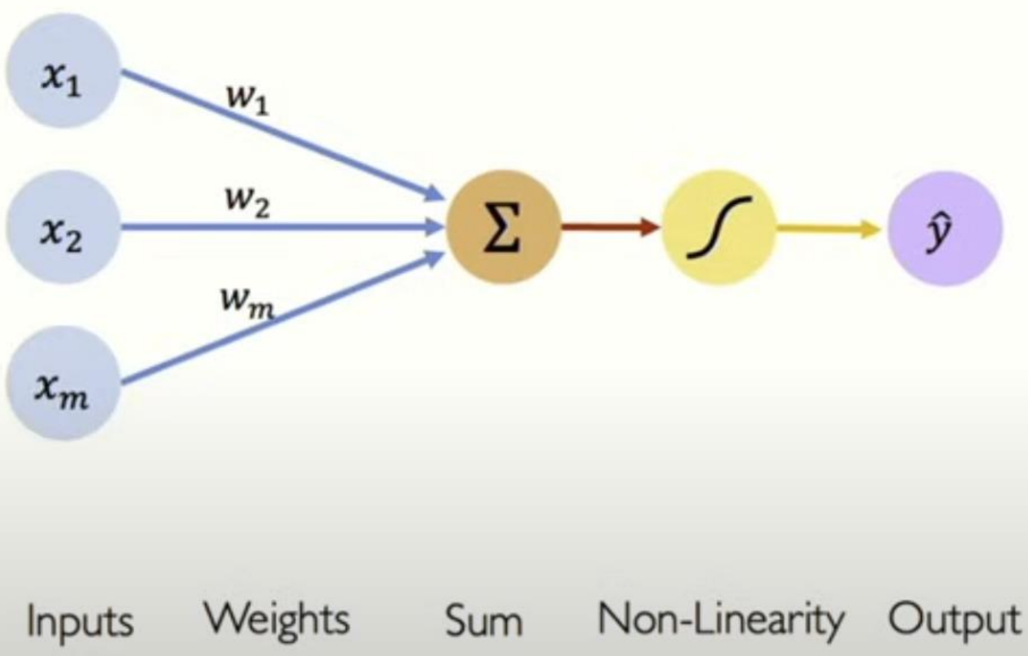
Gradient boosting

1. Initial prediction = average
2. Calculate pseudo-residuals
3. Built a weak (simple) learner of the residuals
4. Repeat
5. Prediction =
    model1 +
    model2 * learning rate +
    model3 * learning rate +
    model4 * learning rate + …



StatQuest - Gradient Boost Part 1

Neural networks – a neuron



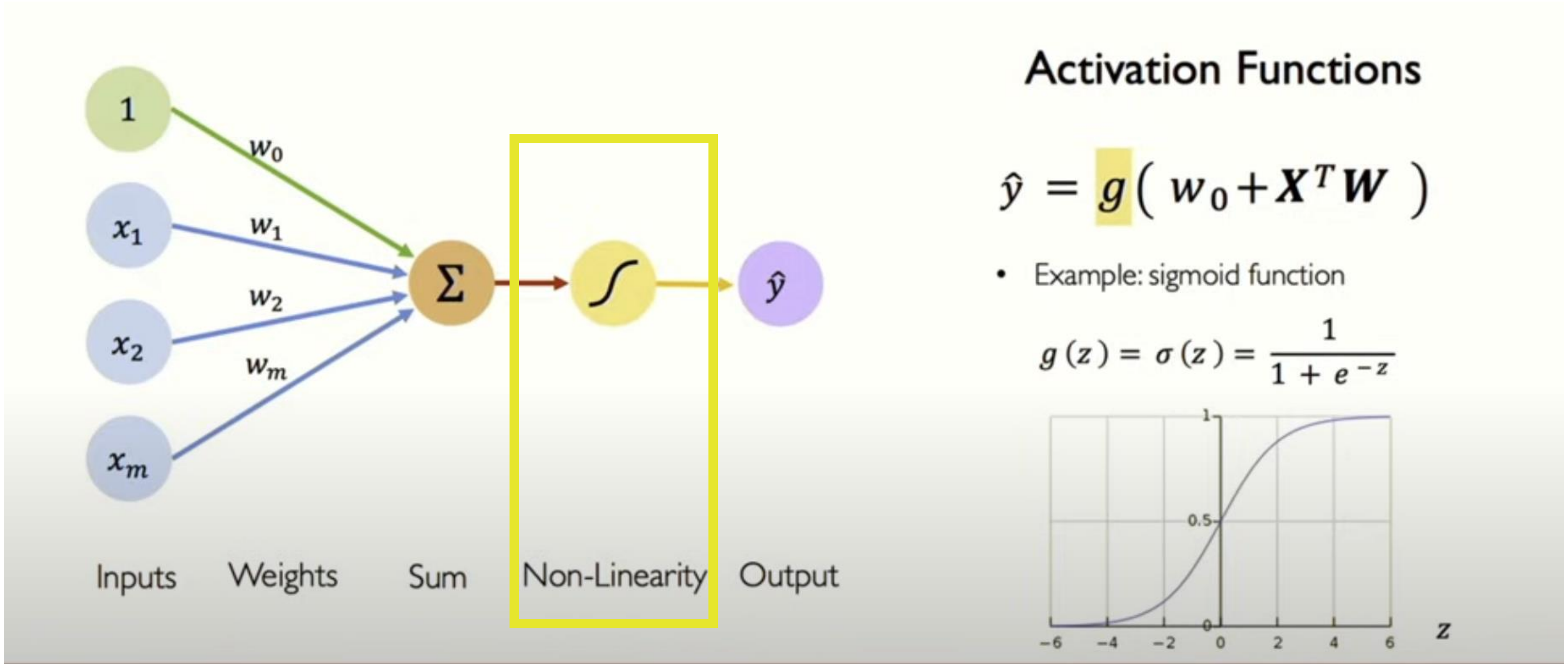
Output

Linear combination of inputs

$$\hat{y} = g\left(\sum_{i=1}^{m} x_i \, w_i\right)$$

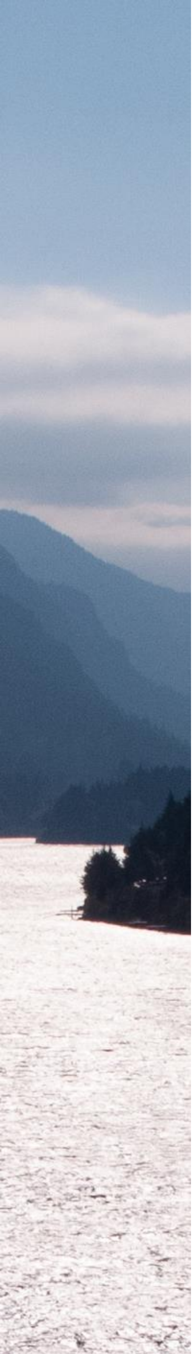
Non-linear activation function

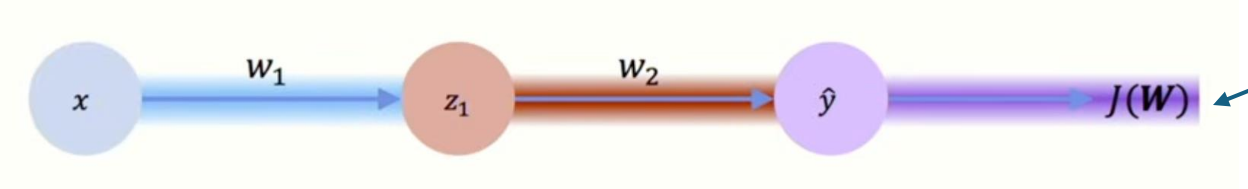
Neural networks – a neuron



MIT introduction to deep learning

Neural networks get feedback by backpropagation





A very, very simple neural network. Weights ***W*** are the model parameters. Loss ***J(W)*** is bigger when predictions are bad, e.g., RMSE

$$\frac{\partial J(\boldsymbol{W})}{\partial w_1} = \frac{\partial J(\boldsymbol{W})}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1} * \frac{\partial z_1}{\partial w_1}$$

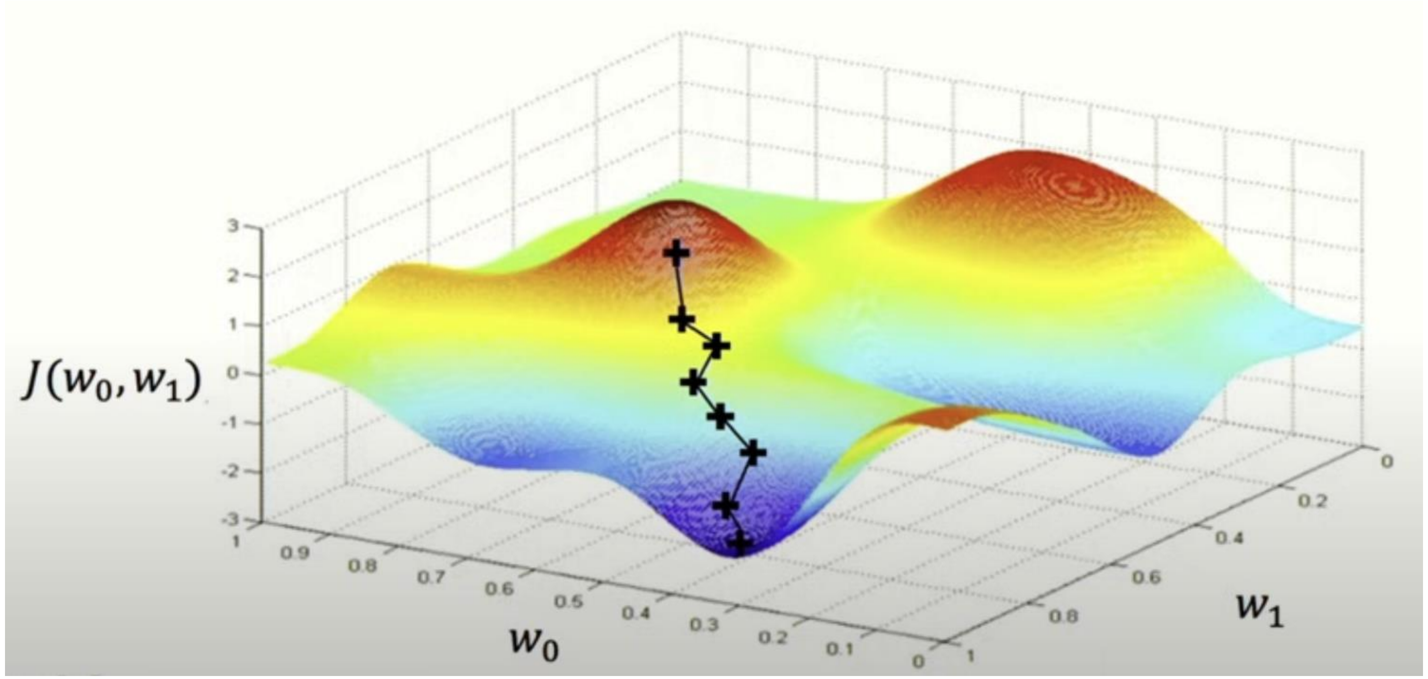
Gradient: change in the loss ($\partial J(\boldsymbol{W})$) per change in a weight (∂w_1)

Backpropagation: computation of the gradient for each weight, using the good old chain rule from calculus

$$\boldsymbol{W} \leftarrow \boldsymbol{W} - \eta \frac{\partial J(\boldsymbol{W})}{\partial \boldsymbol{W}}$$

Update: adjust each weight in the direction that reduces loss, at a rate that reflects its importance to the loss

Neural networks improve by gradient descent

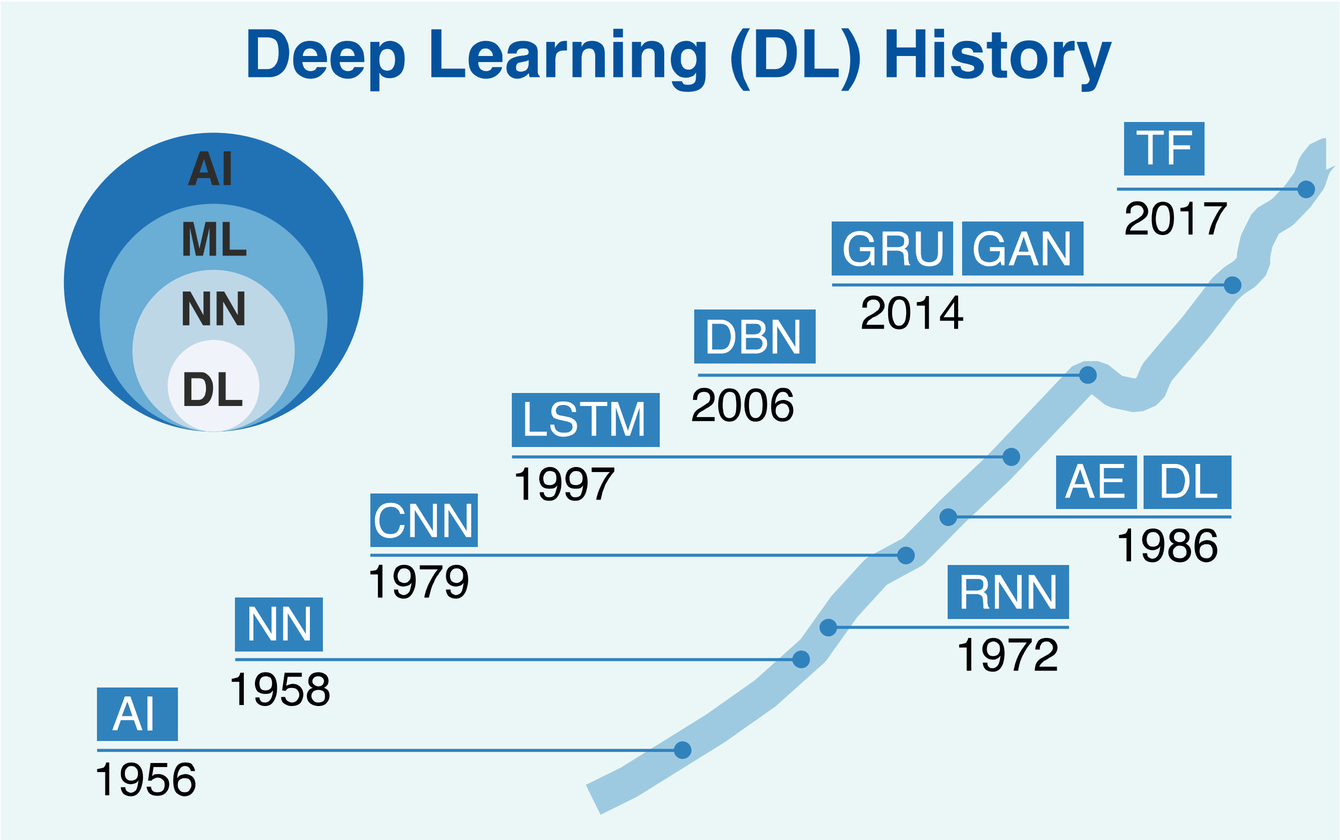


In each iteration:

- Adjust weights by some small amount in the direction of the gradient
- Run the model with new weights and recalculate loss and gradients

MIT introduction to deep learning

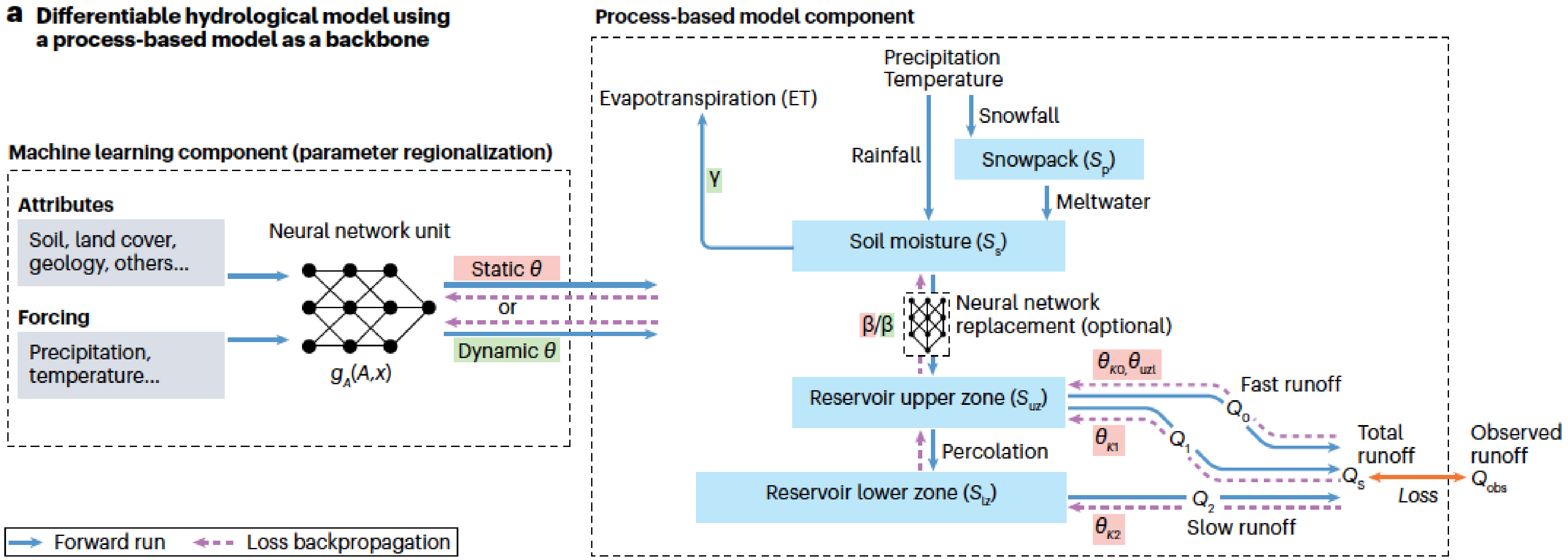
Neural networks are diverse and evolving



Zhi et al. 2024

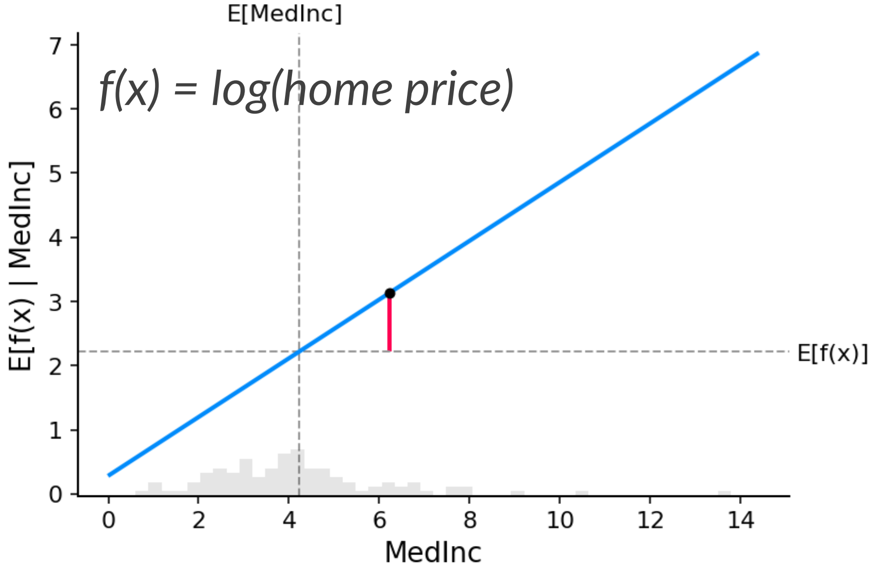
Neural networks can mix with conceptual models

- Differentiable models for water quantity/quality/ecology:
  - freely mix neural network and process-based components
  - training (calibration) is efficient because of back-propagation
  - model is more interpretable; learning can focus on select processes

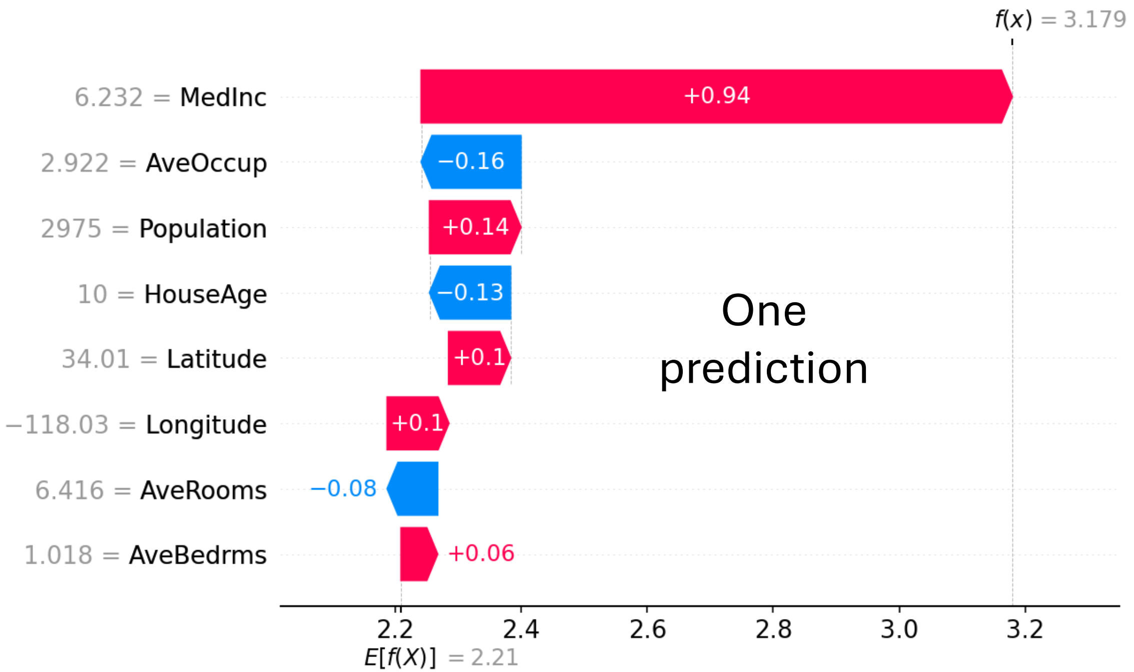


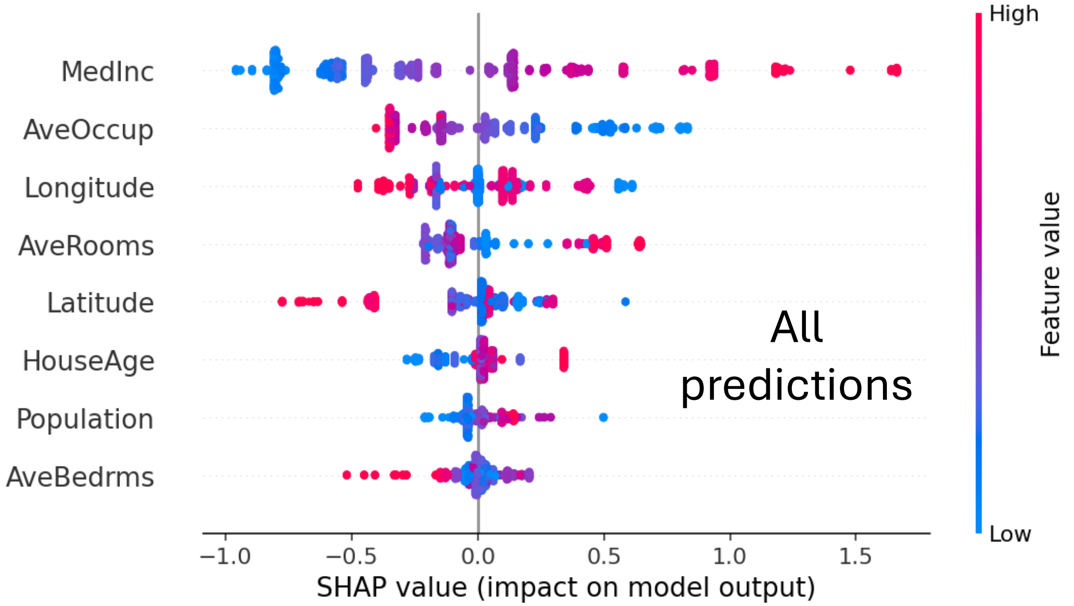
Shen et al. 2023 (no paywall on arXiv)

SHapley Additive exPlanations (SHAP)



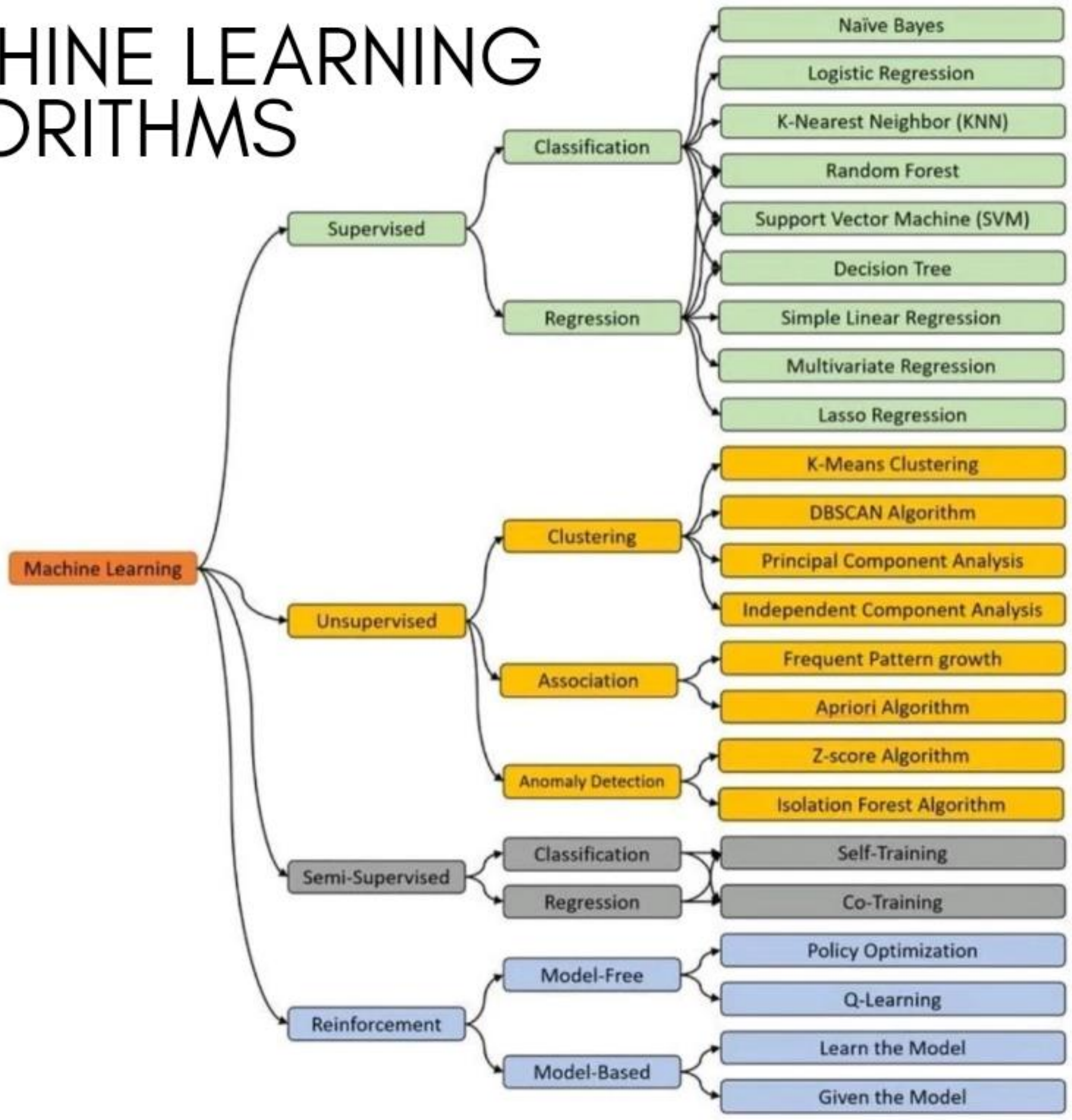
- Provides attribution – quantifies the effect of each input variable value on the prediction
- Additive – SHAP values sum to the difference between the baseline and current model prediction



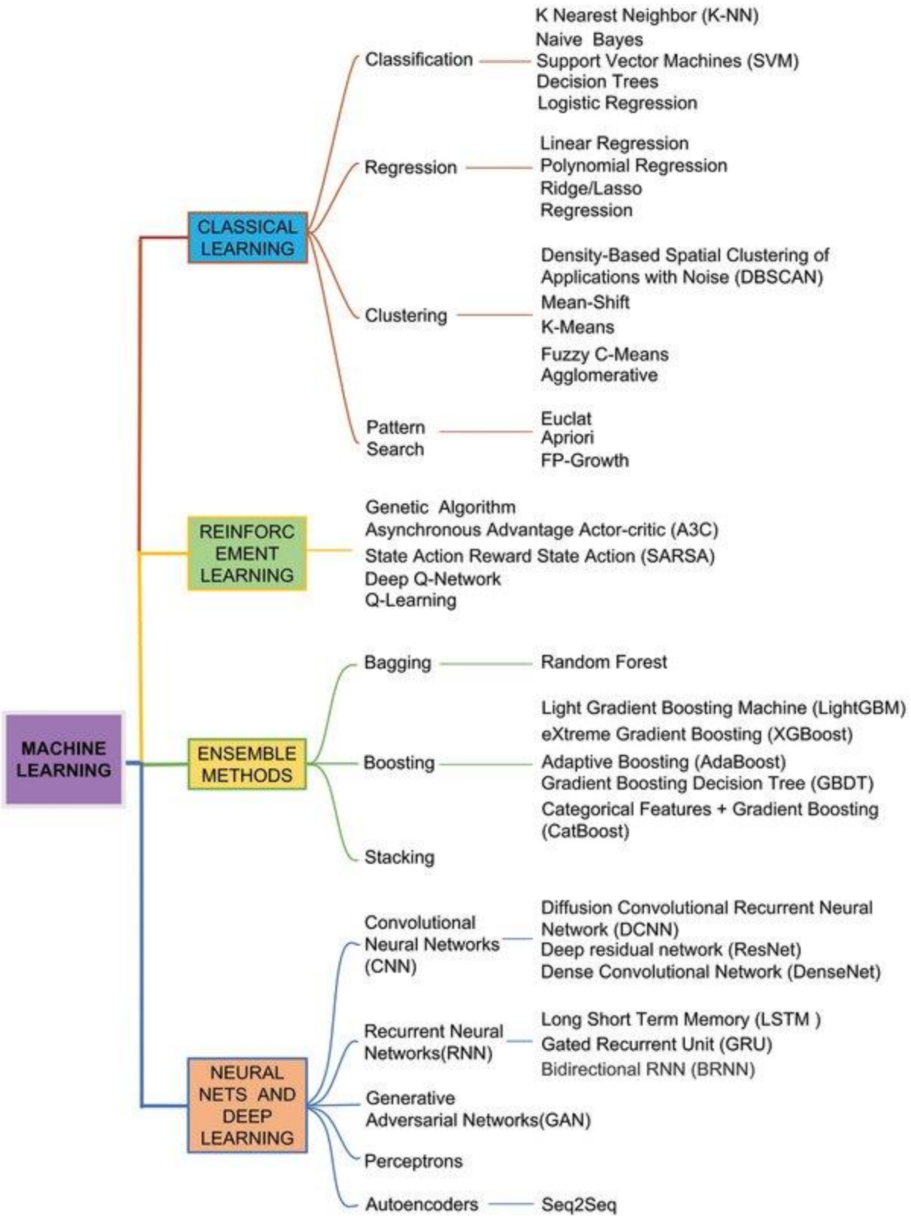


shap.readthedocs.io

MACHINE LEARNING ALGORITHMS

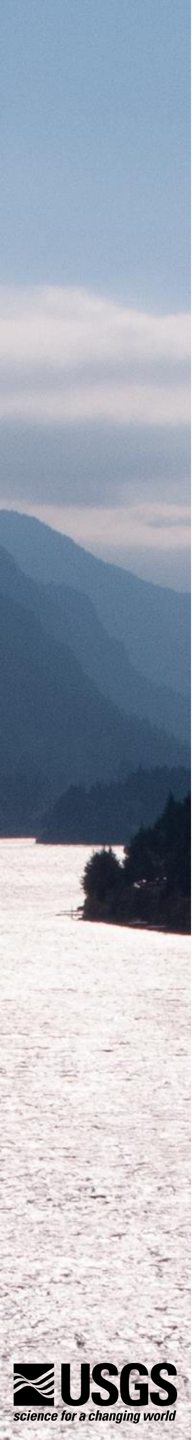


[Abdullah 2024]

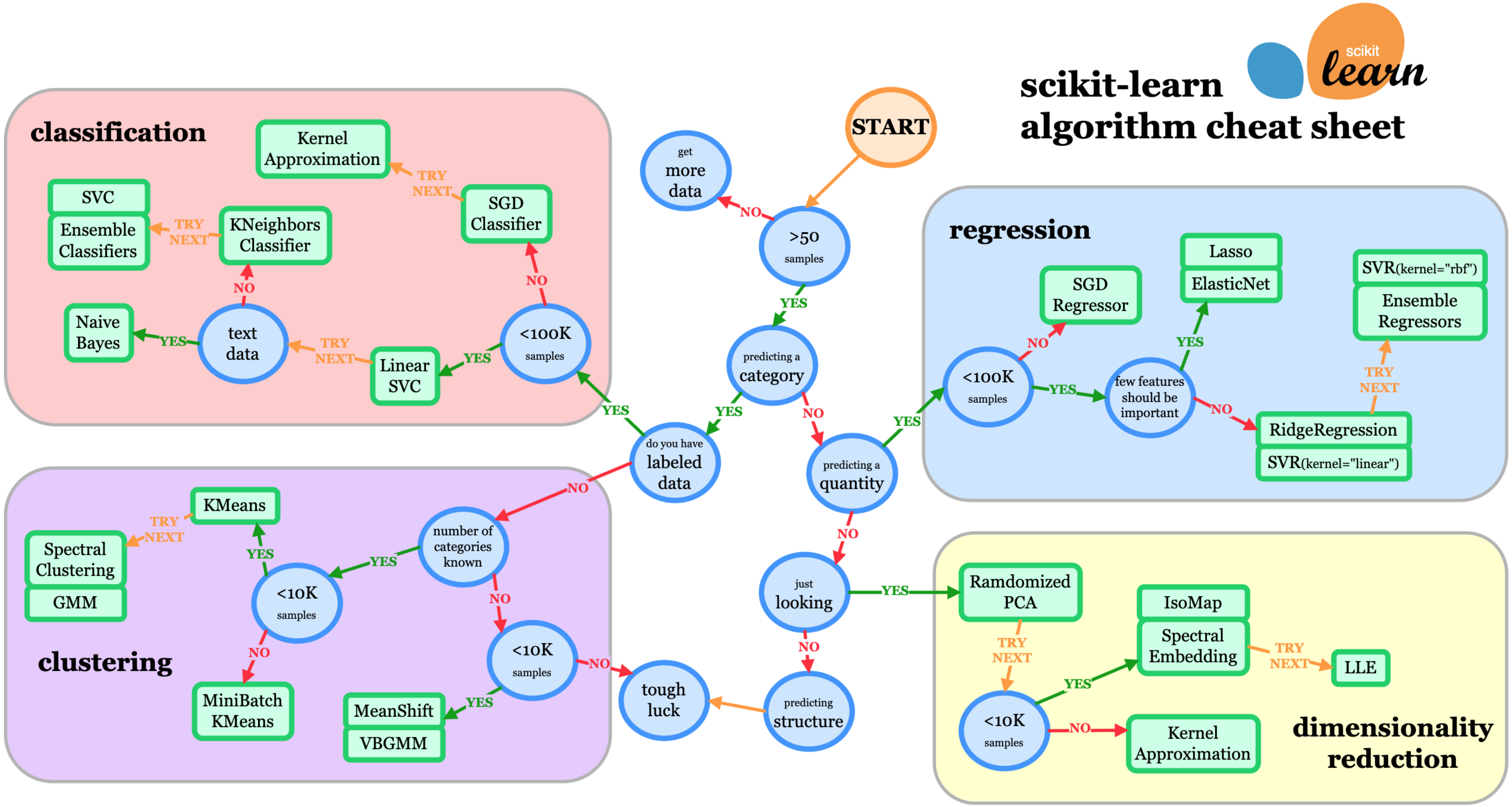


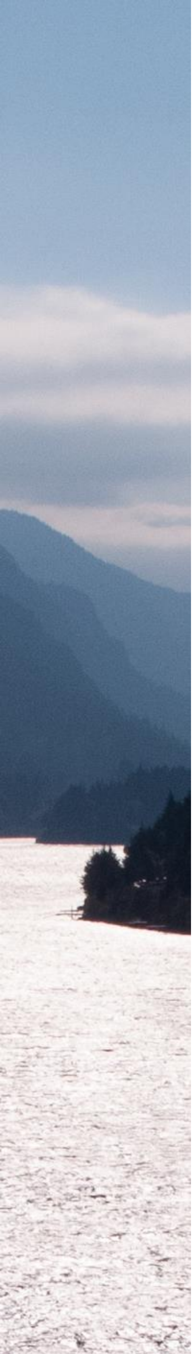
[Gao et al. 2021]

USGS science for a changing world



Choosing an ML method





Automatic ML selection (AutoML)

H2O AutoML

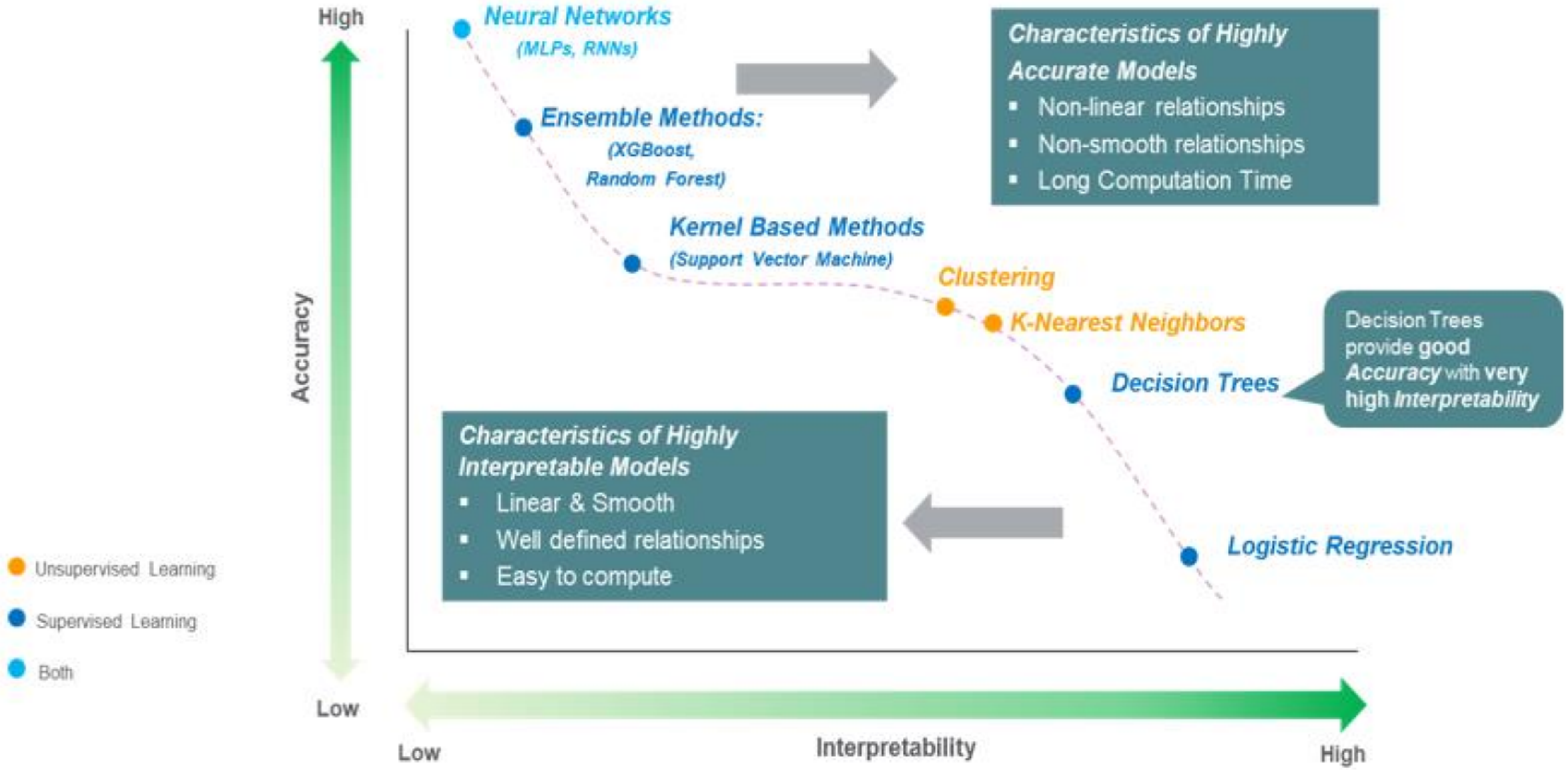
Amazon SageMaker

Many others

- No-code or low-code fitting & selection of ML algorithms
- Try out many different algorithms
- Tune model hyperparameters (i.e., configurations)
- Can combine into ensembles
- Apply model explanation tools (XAI)

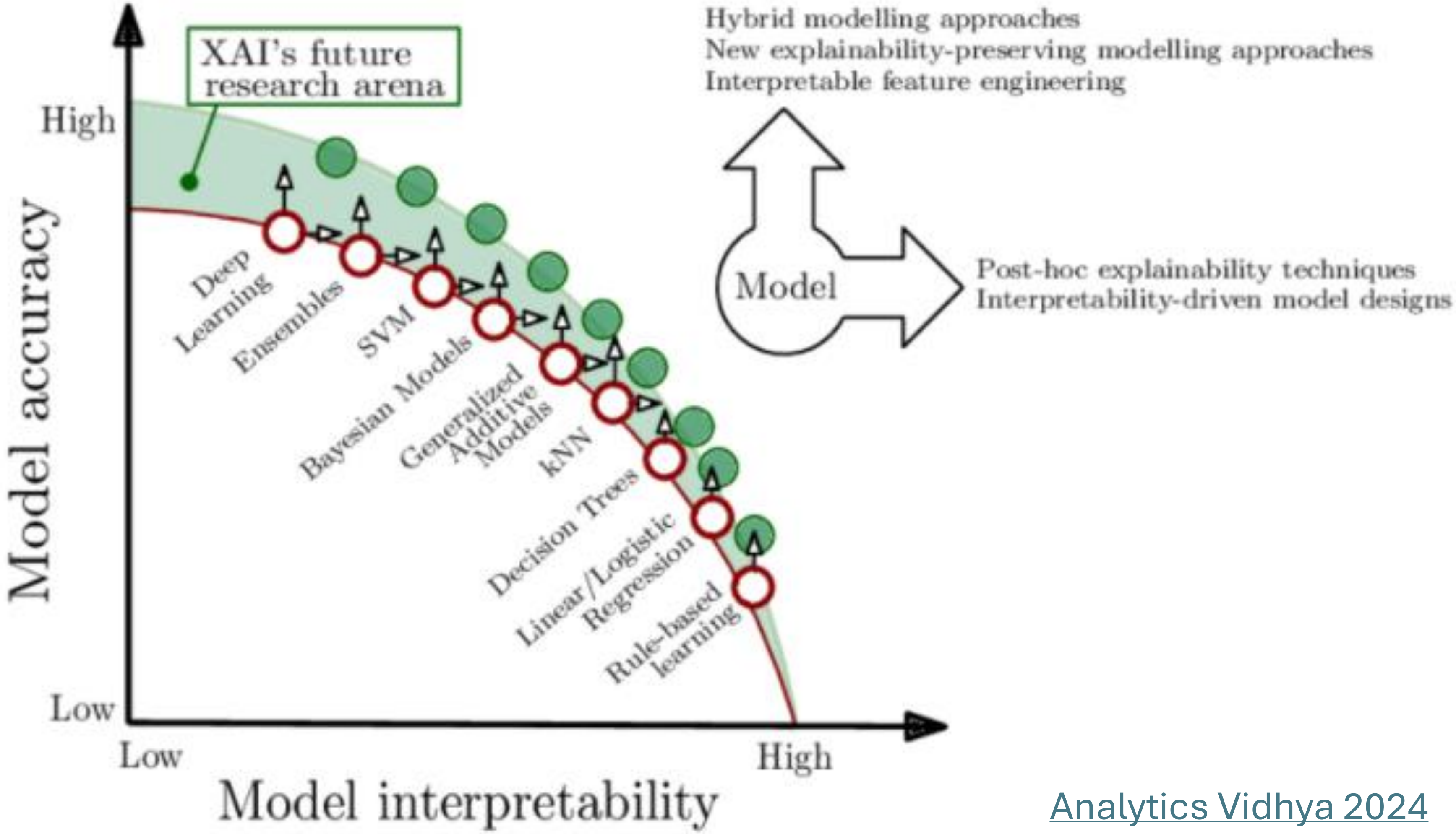
e.g., H2O AutoML; Amazon SageMaker Canvas;
an overview at superannotate automl-guide

Accuracy vs interpretability



[Rane 2018]

Accuracy *and* interpretability?



[Analytics Vidhya 2024]()

GenAI use case: Hydro/ecological predictions

Ecological Informatics
Volume 80, May 2024, 102545

Foundation models in shaping the future of ecology

Albert Morera [a b]

2 A Foundation Model for the Earth System

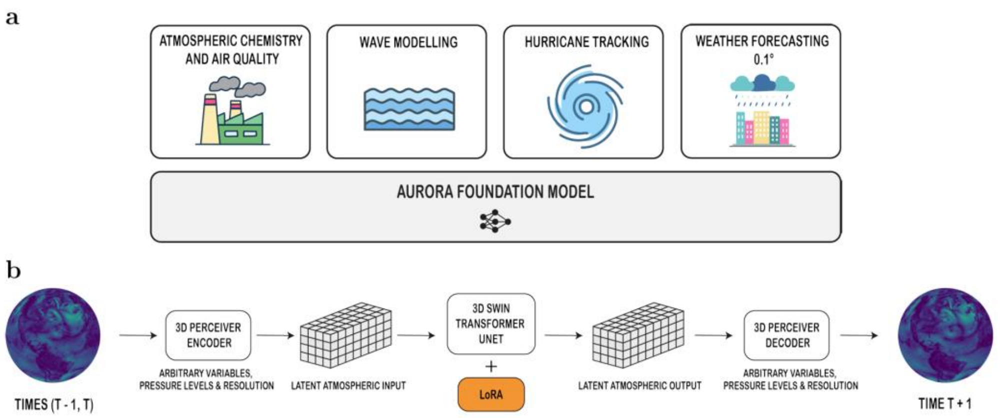
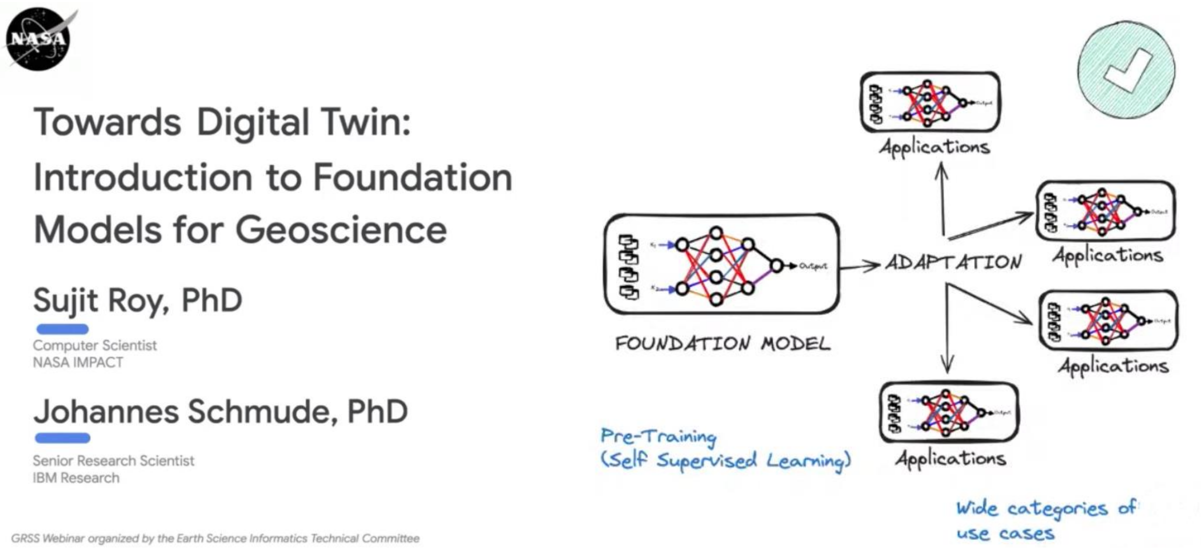
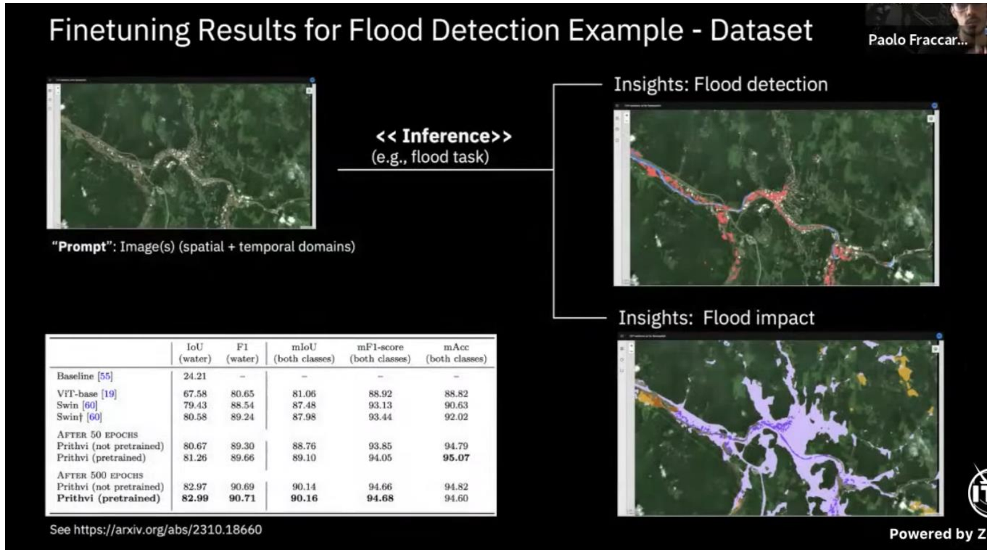


Fig. 1: Aurora is a 1.3 billion parameter foundation model for the Earth system. a: Aurora is pretrained on multiple heterogeneous datasets with different resolutions, variables, and pressure levels. The model is then fine-tuned





	IoU (water)	F1 (water)	mIoU (both classes)	mF1-score (both classes)	mAcc (both classes)
Baseline [55]	24.21	–	–	–	–
ViT-base [19]	67.58	80.65	81.06	88.92	88.82
Swin [60]	79.43	88.54	87.48	93.13	90.63
Swin† [60]	80.58	89.24	87.98	93.44	92.02
After 50 epochs					
Prithvi (not pretrained)	80.67	89.30	88.76	93.85	94.79
Prithvi (pretrained)	81.26	89.66	89.10	94.05	**95.07**
After 500 epochs					
Prithvi (not pretrained)	82.97	90.69	90.14	94.66	94.82
Prithvi (pretrained)	**82.99**	**90.71**	**90.16**	**94.68**	94.60

GenAI use case: Code development

Provide python code to implement a LSTM with three target variables

Certainly! Below is an example of how to implement an LSTM model in Python using Keras with TensorFlow as the backend. This example is designed for a dataset with three target variables.

Prerequisites

Make sure you have the necessary libraries installed:

bash Copy code

```
pip install numpy pandas tensorf
```

Example Code

python

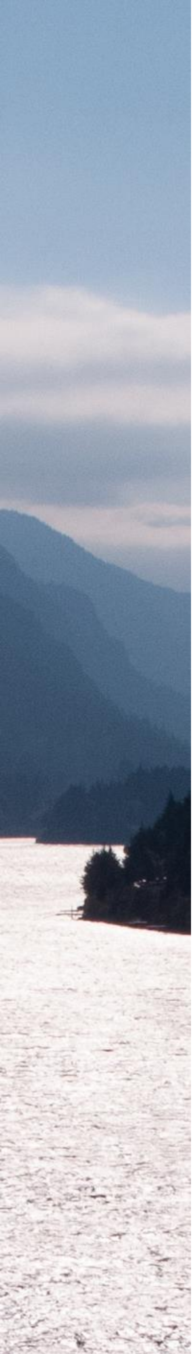
```
import numpy as np
import pandas as pd
from sklearn.preprocessing impor
from tensorflow.keras.models imp
from tensorflow.keras.layers imp
from sklearn.model_selection imp

# Sample data generation
# Assume we have a DataFrame `da
data = pd.DataFrame({
    'feature1': np.random.rand(1
    'feature2': np.random.rand(1
    'target1': np.random.rand(10
    'target2': np.random.rand(1000),
    'target3': np.random.rand(1000)
})

# Features and targets
X = data[['feature1', 'feature2']].values
y = data[['target1', 'target2', 'target3']].values
```

```
# Scale the data
scaler_X = MinMaxScaler()
scaler_y = MinMaxScaler()

X_scaled = scaler_X.fit_transform(X)
y_scaled = scaler_y.fit_transform(y)

# Reshape input to be [samples, time steps, features]
y, modify as needed
e[0], 1, X_scaled.shape[1]))

ing sets
est_split(X_scaled, y_scaled, test

t_shape=(X_train.shape[1], X_train

 target variables

_squared_error')

ch_size=32, validation_data=(X_tes

print(f'Test Loss: {loss}')

# Make predictions
predictions = model.predict(X_test)

# Inverse transform the predictions to get original values
predictions_inverse = scaler_y.inverse_transform(predictions)
```

Explanation

1. **Data Preparation**: The code generates sample data with two features and three target variables. In practice, you would load your dataset here.
2. **Scaling**: The input features and target variables are scaled using `MinMaxScaler`.
3. **Reshape Input**: The input data is reshaped to fit the LSTM model's requirements.
4. **Model Building**: A simple LSTM model is created with one LSTM layer followed by a dropout layer and a dense layer for the output.
5. **Training**: The model is compiled and trained using the training set.
6. **Evaluation**: The model is evaluated on the test set, and predictions are made.

Feel free to modify the number of epochs, batch size, and model architecture based on your specific requirements and dataset characteristics.



GenAI use case: Literature exploration

- Browse papers by "too long; didn't read" (TLDR) summary

SEMANTIC SCHOLAR | chesapeake bay machine learning

About 23,600 results for "chesapeake bay machine learning"

Fields of Study | Date Range | Has PDF | Author | Journals & Conferences

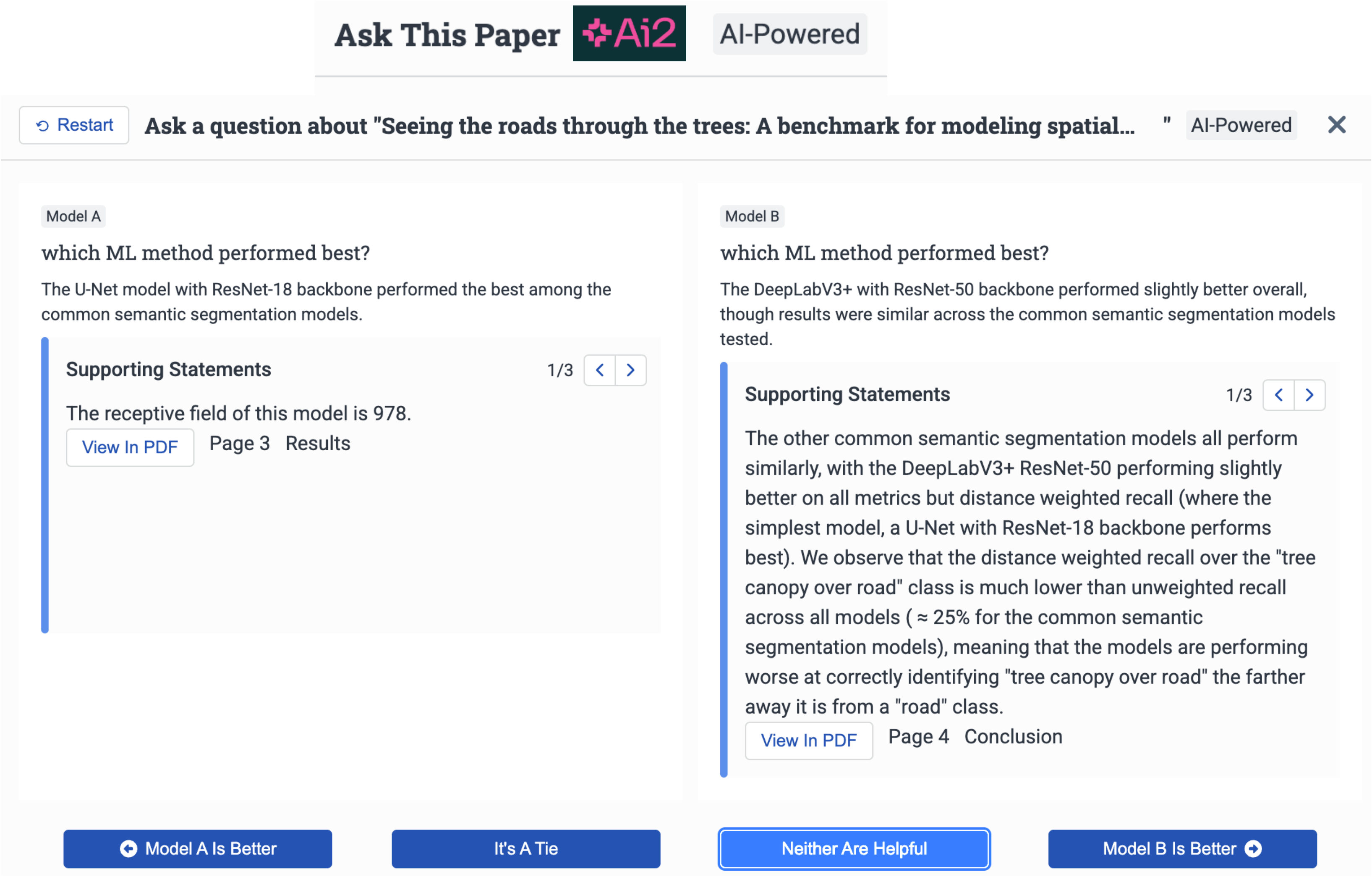
Machine Learning-Based Wave Model With High Spatial Resolution in Chesapeake Bay
Jian Shen · Zhengui Wang · Jiabi Du · Yinglong J. Zhang · Qubin Qin
Environmental Science, Computer Science · Earth and Space Science · 1 March 2024
TLDR A **machine learning** model using long short-term memory to simulate large-scale, high-resolution waves and shows promise as an alternative for wave forecasting and coupling with sediment and ecological models. Expand
PDF · Wiley · Save · Cite

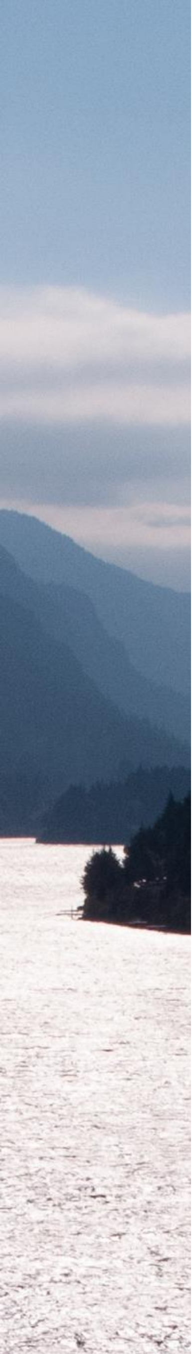
Use of interpretable machine learning to identify the factors influencing the nonlinear linkage between land use and river water quality in the Chesapeake Bay watershed
Zhenyu Zhang · Jinliang Huang · S. Duan · Yaling Huang · Juntao Cai · Jing Bian
Environmental Science, Computer Science · Ecological Indicators · 1 July 2022
32 · Publisher · Save · Cite

Explainable machine learning improves interpretability in the predictive modeling of biological stream conditions in the Chesapeake Bay Watershed, USA.
K. Maloney · C. Buchanan · +5 authors · Matthias Schmid · Environmental Science, Biology · Journal of Environmental Management · 1 September 2022
14 · PDF · PubMed · Save · Cite

GenAI use case: Literature exploration

- Browse papers by "too long; didn't read" (TLDR) summary
- Ask for specific information



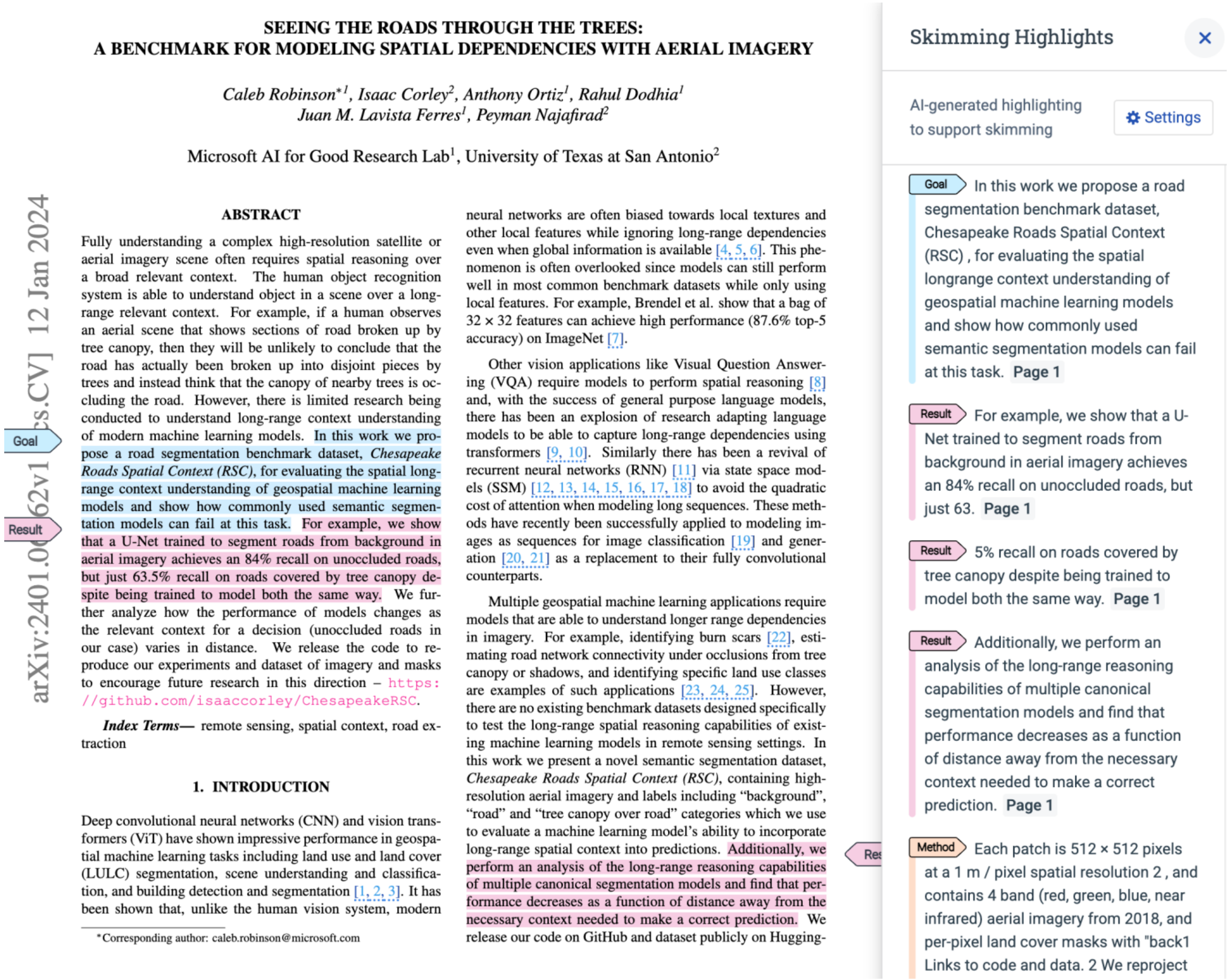


USGS
science for a changing world

GenAI use case: Literature exploration

SEMANTIC READER

- Browse papers by "too long; didn't read" (TLDR) summary
- Ask for specific information
- Skim for key points more quickly
- Future: summarize multiple papers together


SEEING THE ROADS THROUGH THE TREES:
A BENCHMARK FOR MODELING SPATIAL DEPENDENCIES WITH AERIAL IMAGERY

Caleb Robinson[*1], *Isaac Corley*[2], *Anthony Ortiz*[1], *Rahul Dodhia*[1]
Juan M. Lavista Ferres[1], *Peyman Najafirad*[2]

Microsoft AI for Good Research Lab[1], University of Texas at San Antonio[2]

arXiv:2401.06762v1 [cs.CV] 12 Jan 2024

ABSTRACT

Fully understanding a complex high-resolution satellite or aerial imagery scene often requires spatial reasoning over a broad relevant context. The human object recognition system is able to understand object in a scene over a long-range relevant context. For example, if a human observes an aerial scene that shows sections of road broken up by tree canopy, then they will be unlikely to conclude that the road has actually been broken up into disjoint pieces by trees and instead think that the canopy of nearby trees is occluding the road. However, there is limited research being conducted to understand long-range context understanding of modern machine learning models. In this work we propose a road segmentation benchmark dataset, *Chesapeake Roads Spatial Context (RSC)*, for evaluating the spatial long-range context understanding of geospatial machine learning models and show how commonly used semantic segmentation models can fail at this task. For example, we show that a U-Net trained to segment roads from background in aerial imagery achieves an 84% recall on unoccluded roads, but just 63.5% recall on roads covered by tree canopy despite being trained to model both the same way. We further analyze how the performance of models changes as the relevant context for a decision (unoccluded roads in our case) varies in distance. We release the code to reproduce our experiments and dataset of imagery and masks to encourage future research in this direction – https://github.com/isaaccorley/ChesapeakeRSC.

Index Terms— remote sensing, spatial context, road extraction

1. INTRODUCTION

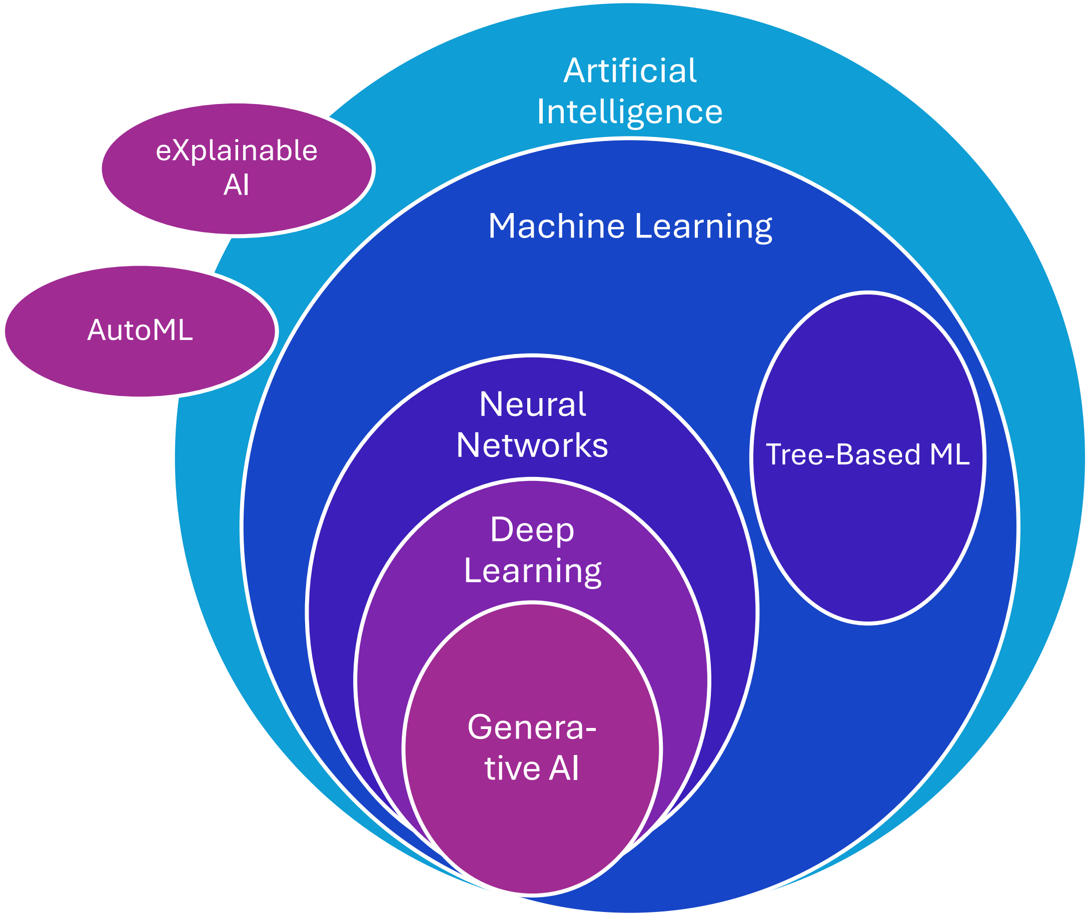
Deep convolutional neural networks (CNN) and vision transformers (ViT) have shown impressive performance in geospatial machine learning tasks including land use and land cover (LULC) segmentation, scene understanding and classification, and building detection and segmentation [1, 2, 3]. It has been shown that, unlike the human vision system, modern neural networks are often biased towards local textures and other local features while ignoring long-range dependencies even when global information is available [4, 5, 6]. This phenomenon is often overlooked since models can still perform well in most common benchmark datasets while only using local features. For example, Brendel et al. show that a bag of 32 × 32 features can achieve high performance (87.6% top-5 accuracy) on ImageNet [7].

Other vision applications like Visual Question Answering (VQA) require models to perform spatial reasoning [8] and, with the success of general purpose language models, there has been an explosion of research adapting language models to be able to capture long-range dependencies using transformers [9, 10]. Similarly there has been a revival of recurrent neural networks (RNN) [11] via state space models (SSM) [12, 13, 14, 15, 16, 17, 18] to avoid the quadratic cost of attention when modeling long sequences. These methods have recently been successfully applied to modeling images as sequences for image classification [19] and generation [20, 21] as a replacement to their fully convolutional counterparts.

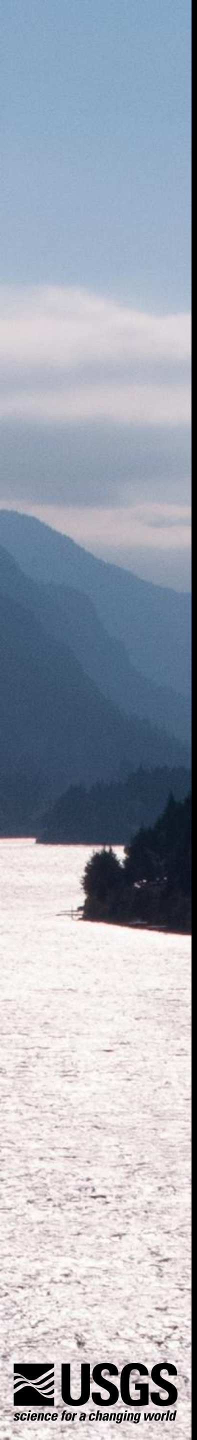
Multiple geospatial machine learning applications require models that are able to understand longer range dependencies in imagery. For example, identifying burn scars [22], estimating road network connectivity under occlusions from tree canopy or shadows, and identifying specific land use classes are examples of such applications [23, 24, 25]. However, there are no existing benchmark datasets designed specifically to test the long-range spatial reasoning capabilities of existing machine learning models in remote sensing settings. In this work we present a novel semantic segmentation dataset, *Chesapeake Roads Spatial Context (RSC)*, containing high-resolution aerial imagery and labels including "background", "road" and "tree canopy over road" categories which we use to evaluate a machine learning model's ability to incorporate long-range spatial context into predictions. Additionally, we perform an analysis of the long-range reasoning capabilities of multiple canonical segmentation models and find that performance decreases as a function of distance away from the necessary context needed to make a correct prediction. We release our code on GitHub and dataset publicly on Hugging-

*Corresponding author: caleb.robinson@microsoft.com

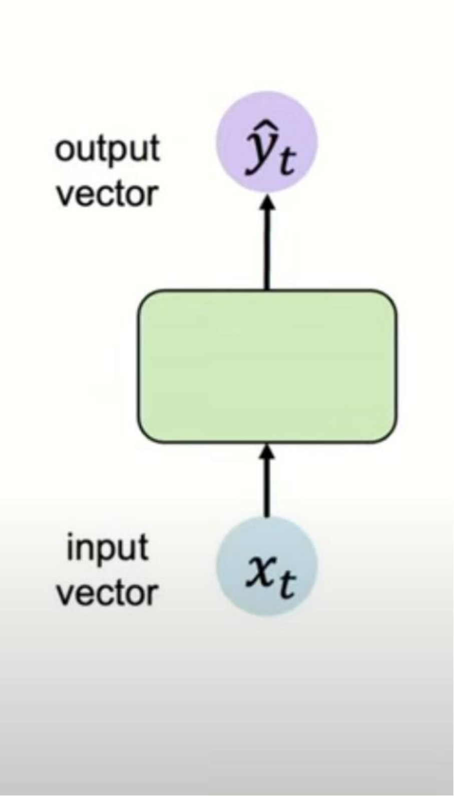
Conclusions



- Chesapeake AI is mostly ML (useful for brevity?)
- Many great algorithms & tools exist and are emerging
- Neural networks are endlessly flexible…but tree ensembles sometimes still outperform
- Generative AI is a whole new game with new uses to consider

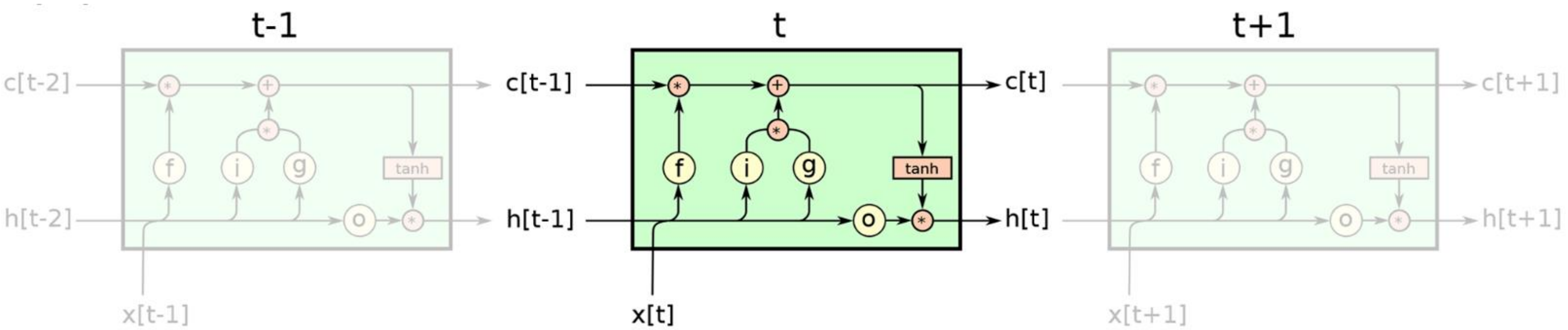

USGS
science for a changing world

Recurrent neural networks (e.g., for timeseries)



MIT introduction to deep learning

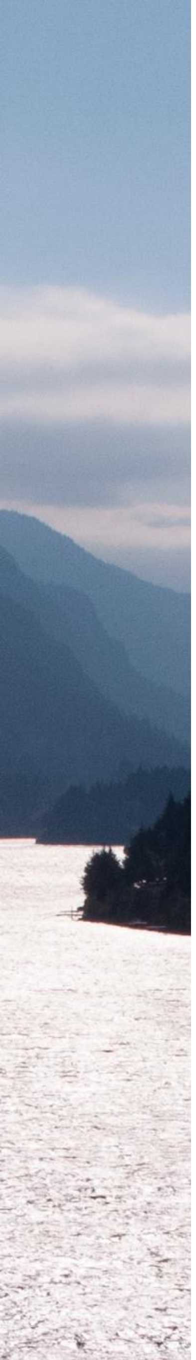
Long short-term memory (LSTM) networks (even better for timeseries)



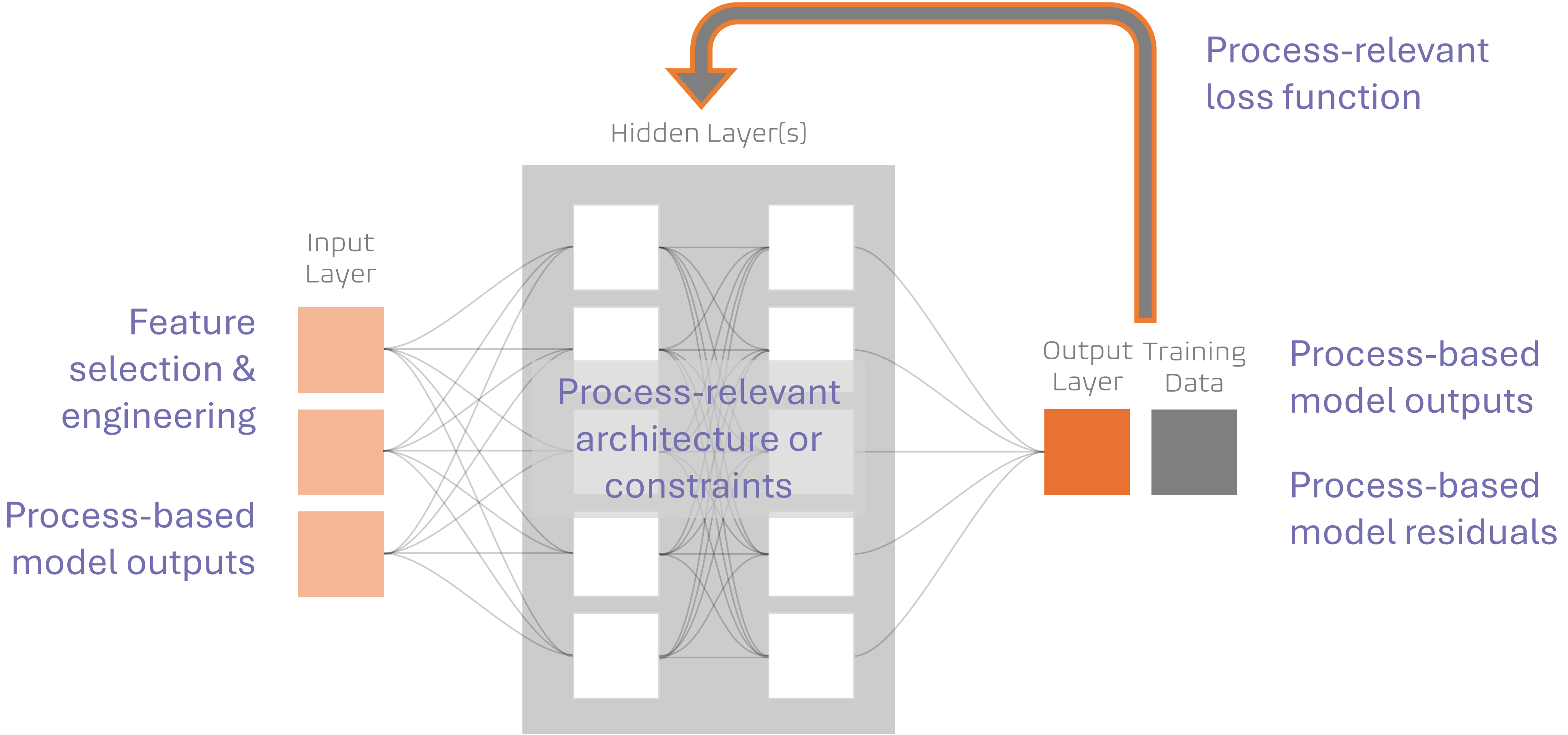
input gate: $$\boldsymbol{i}[t] = \sigma(\underline{\mathbf{W}_{\mathrm{i}}}\boldsymbol{x}[t] + \underline{\mathbf{U}_{\mathrm{i}}}\boldsymbol{h}[t-1] + \underline{\boldsymbol{b}_{\mathrm{i}}}),$$

learnable parameters

MIT introduction to deep learning and Kratzert and others, 2019



Process-guided deep learning (PGDL)



Concepts expanded from [Willard et al. 2022]
Figure by Ellen Bechtel, modified from [Appling et al. 2022]