Machine Learning in Geotechnical Engineering

Krishna Kumar, UT Austin

Online course material

AI in Geotechnical Engineering | Search | SciML Course Repo ☆1 ⑂0

AI in Geotechnical Engineering
Home
Classification
Decision Tree
MLP
Activation Function ReLU
Automatic Differentiation
Gradient Descent
PINNs
PINNs oscillator
PINNs 1D consolidation
PINNs 1D consolidation [L-BFGS]

AI in Geotechnical Engineering

Krishna Kumar

A comprehensive course designed to introduce geotechnical engineers and scientists to the revolutionary field of Scientific Machine Learning (SciML). This course bridges traditional geotechnical engineering with cutting-edge AI techniques, focusing on solving real-world problems in soil mechanics, foundation engineering, and geotechnical modeling through Physics-Informed Neural Networks (PINNs) and advanced machine learning approaches.

The course emphasizes practical applications in geotechnical engineering, from soil classification and liquefaction prediction to solving complex consolidation problems and parameter identification. Participants will gain hands-on experience with industry-standard tools including Python, PyTorch, and modern scientific computing libraries.

Course Structure

Module 1: Foundations of Machine Learning in Geotechnical Engineering

00a-classification - Introduction to Classification Problems - Traditional geotechnical classification using Decision Trees, Random Forest, and XGBoost - Earthquake-induced liquefaction prediction with SHAP explainability - Real-world geotechnical dataset analysis and feature importance

00b-mlp-classification

- Multi-Layer Perceptrons for Geotechnical Applications
- From perceptrons to deep neural networks: theoretical foundations
- Universal Approximation Theorem and Sobolev spaces
- Activation functions, gradient descent, and automatic differentiation
- Advanced MLP implementation for liquefaction classification

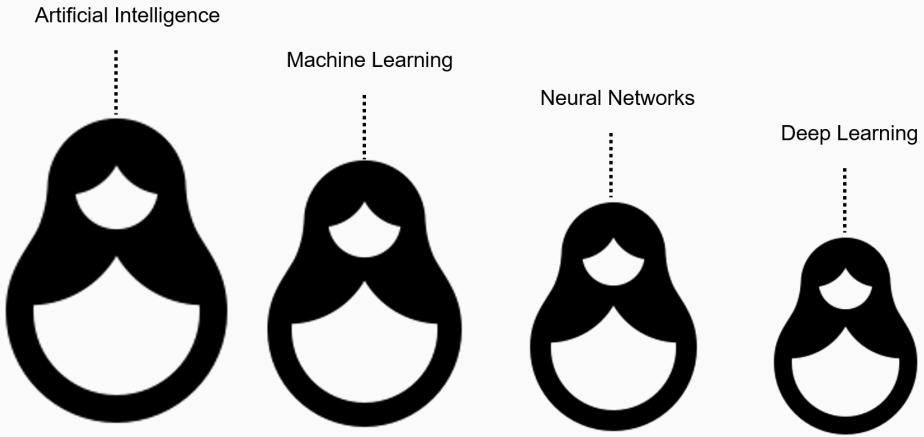
Table of contents

Course Structure
Module 1: Foundations of Machine Learning in Geotechnical Engineering
Module 2: Physics-Informed Neural Networks for Geotechnical Problems
Course Objectives
Target Audience
Key Features
Geotechnical Engineering Focus
Progressive Learning Approach
Strong Theoretical Foundation
Hands-On Implementation
Comprehensive Comparisons
Prerequisites
Essential Background
Helpful but Not Required
Learning Outcomes
Practical Skills
Theoretical Understanding
Professional Capabilities
Technical Requirements
Getting Started

https://kks32-courses.github.io/ai-geotech/

AI vs Machine Learning vs Deep Learning

- **Artificial intelligence**: build intelligent programs and machines that can creatively solve problems, which has always been considered a human prerogative.
- **Machine learning** is a subset of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. In ML, there are different algorithms (e.g. neural networks) that help to solve problems.
- **Deep learning** is a subset of machine learning, which uses the neural networks to analyze different factors with a structure that is similar to the human neural system.



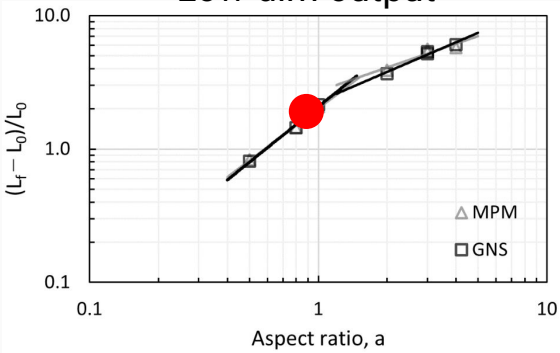
Why I dislike ML?

Input parameters

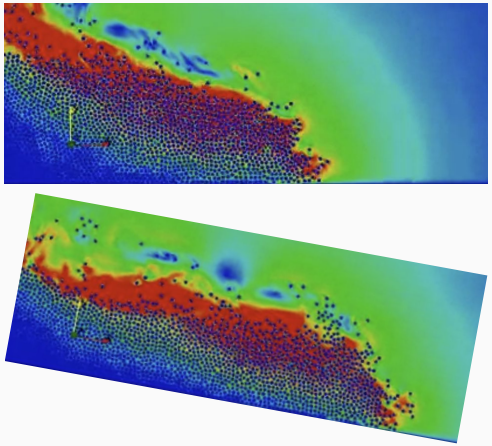
```
density = 1800 kg/m3
      n = 0.4
 Aspect ratio = 0.8
    Slope = 2.5°
```

Magical NN

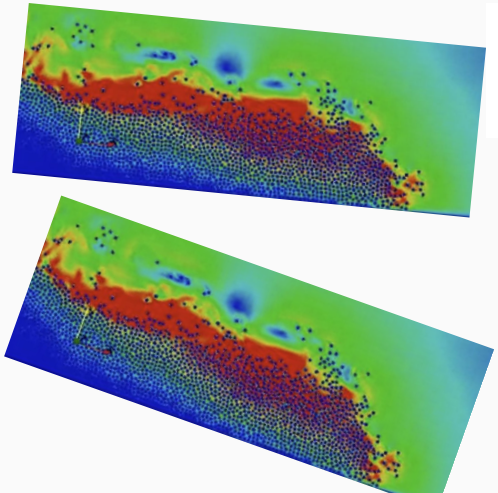
Low-dim output



Training



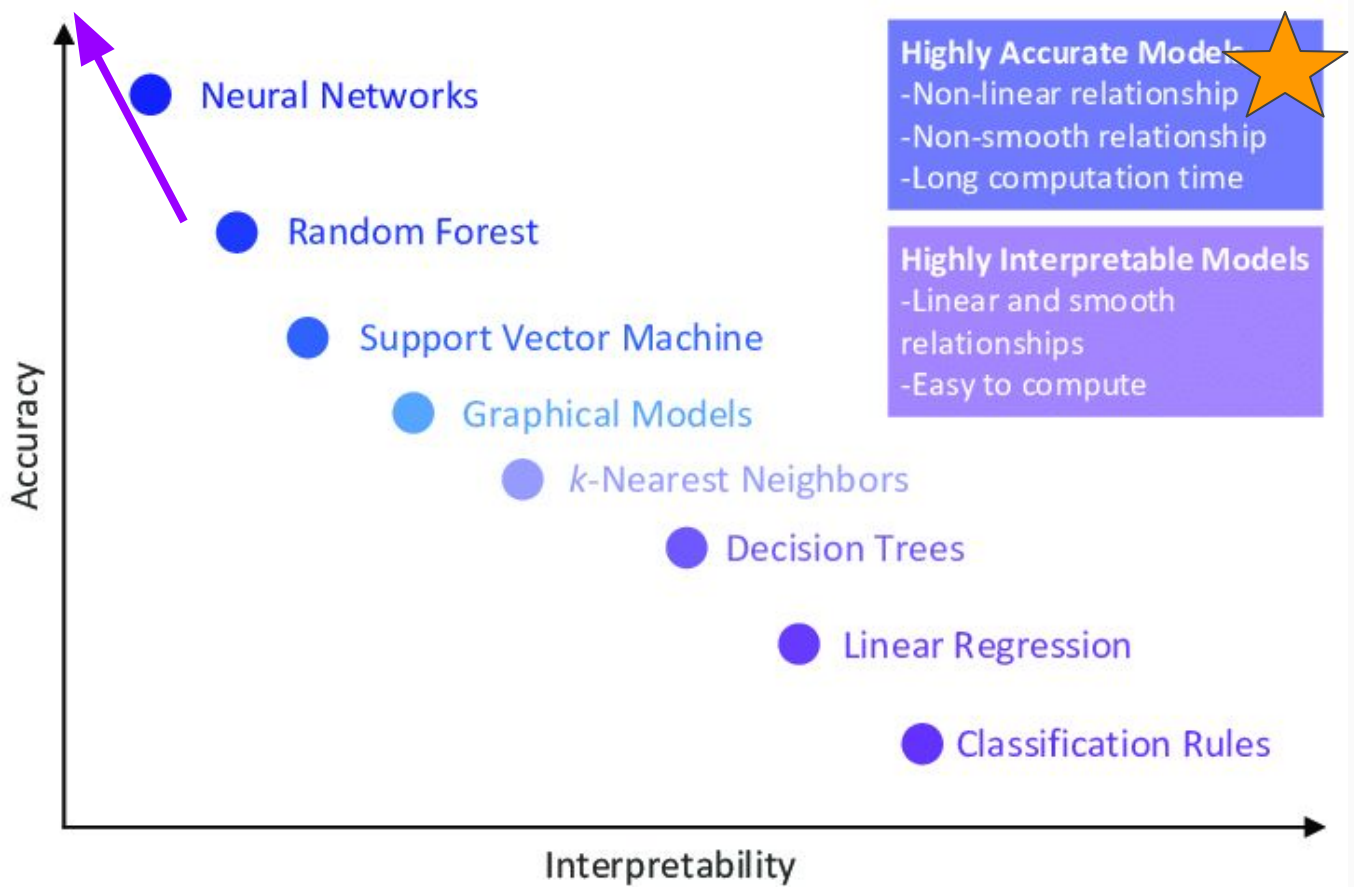
Testing



Interpolate

Extrapolate

Interpretability vs accuracy



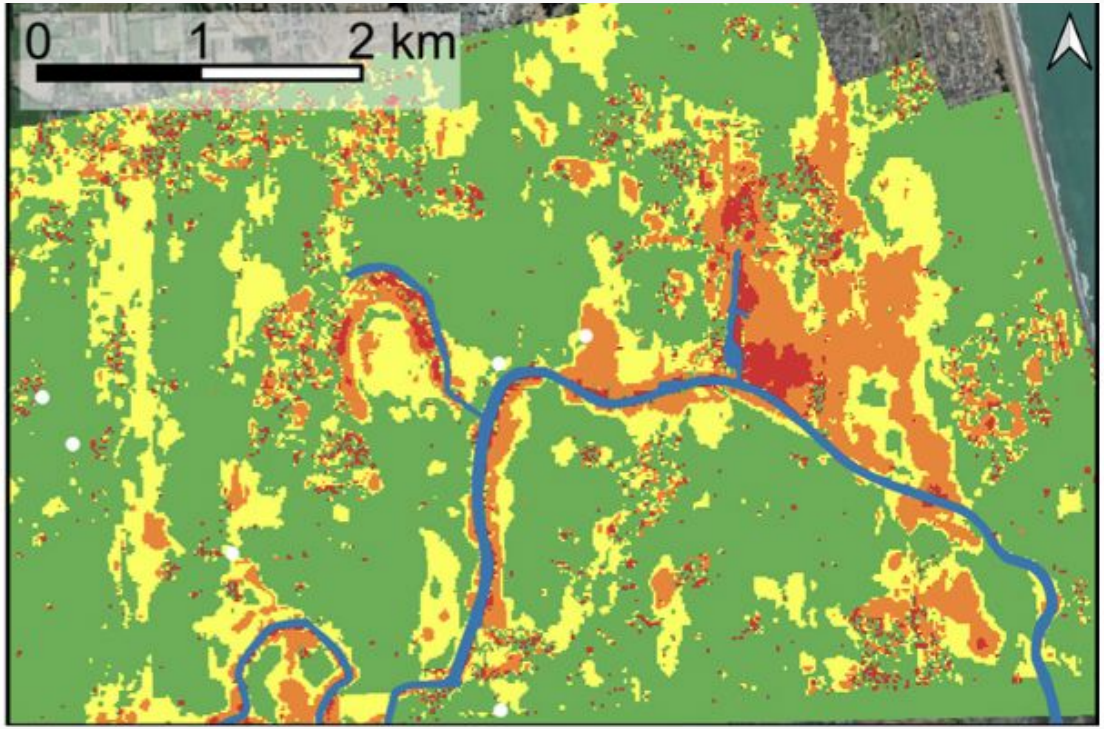
Lateral Spreading

Decision Trees
&
Random Forest

Predicting lateral spreading in NZ

Observed displacement (m)

- None (< 0.30)
- 0.30 - 0.50
- 0.50 - 1.00
- > 1.00



Durante and Rathje, 2021

Predicting lateral spreading in NZ

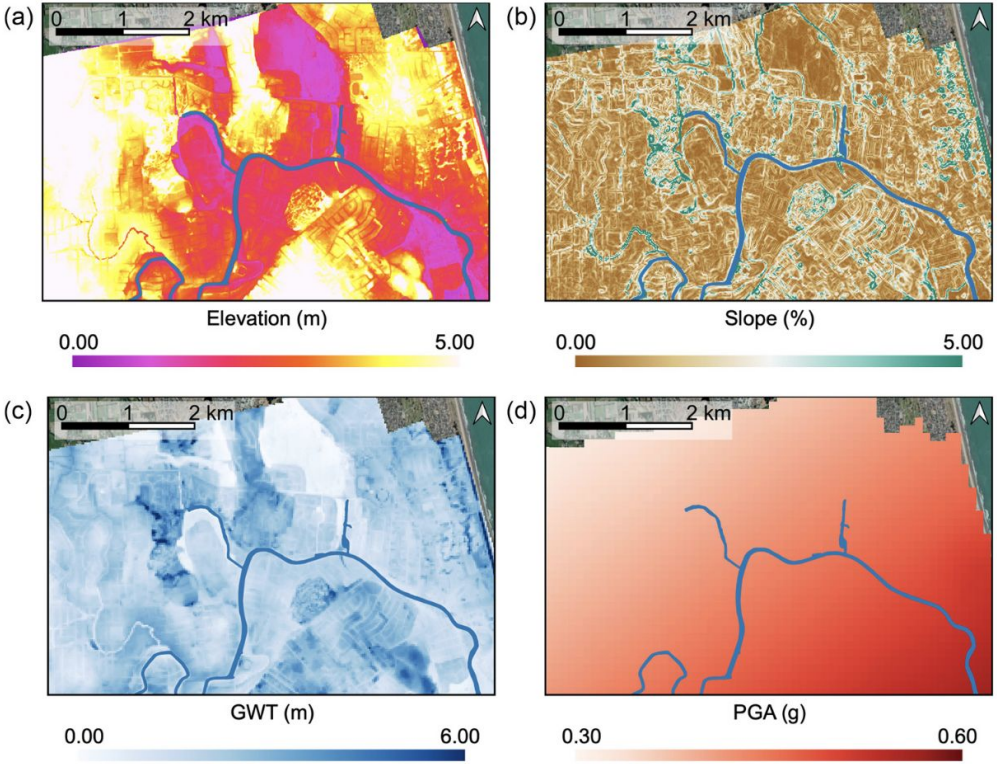
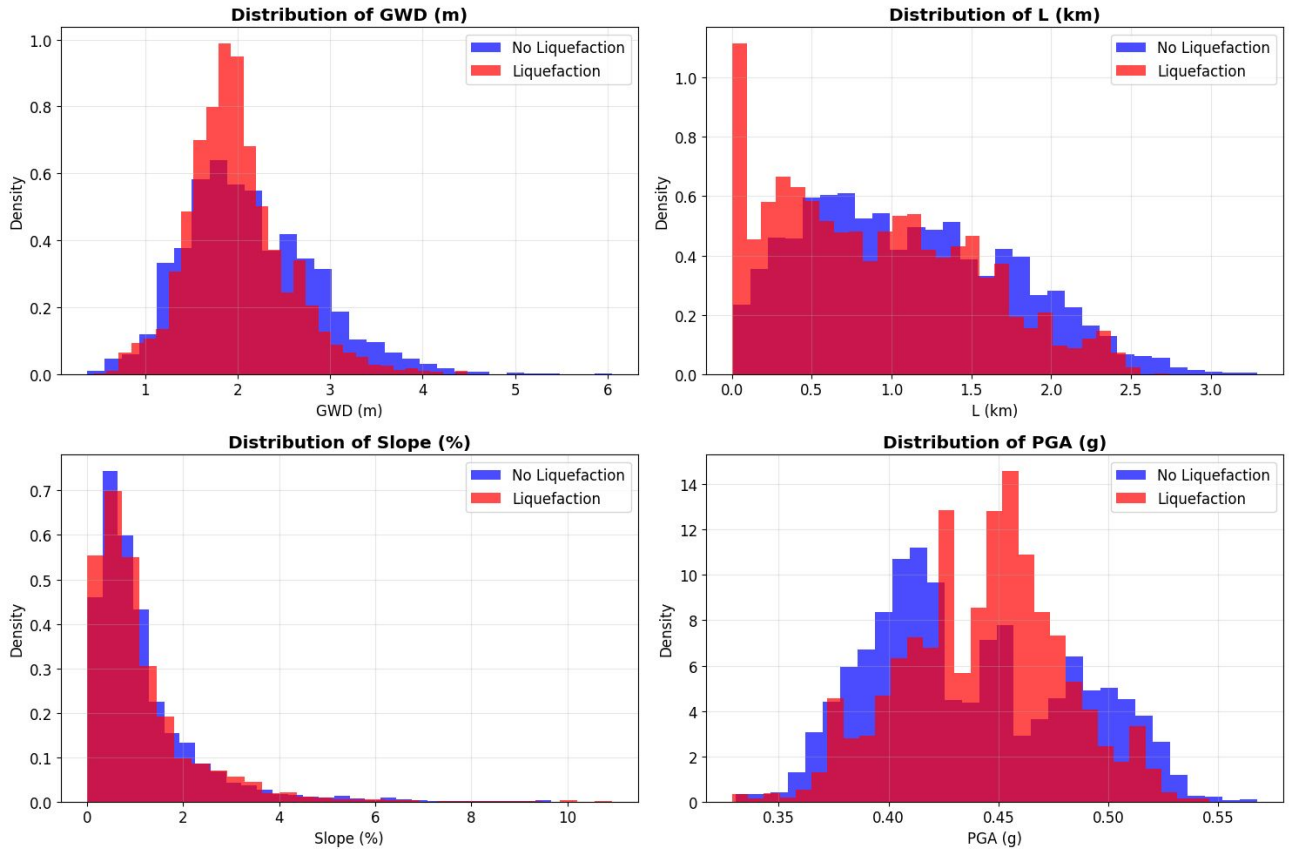


Figure 4. Spatial distribution of geometric and event-specific input features considered in ML models: (a) ground elevation, (b) ground slope, (c) GWT depth, and (d) PGA.

Durante and Rathje, 2021

Distribution of data



Random Forest model

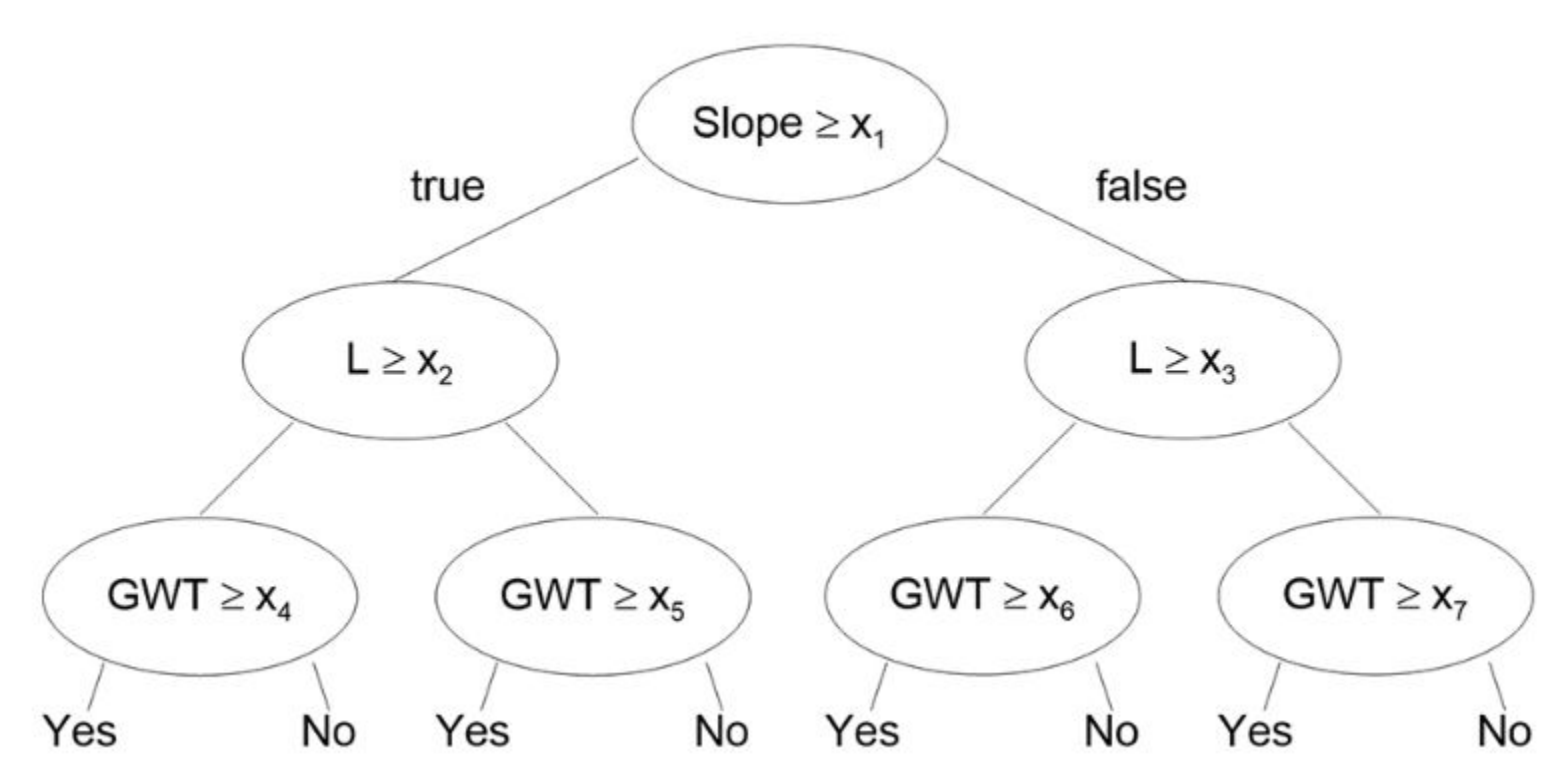
Table 3. Summary of features used in each RF model analyzed

		Feature					
		L (m)	GWT (m)	Slope (%)	PGA (g)	Elevation (m)	CPT data
No CPT data	Model 0	✓	✓	✓	○	○	○
	Model 1	✓	✓	✓	✓	○	○
	Model 2	✓	✓	✓	○	✓	○
	Model 3	✓	✓	✓	✓	✓	○
CPT data	Model 4	✓	✓	✓	✓	○	✓
	Model 5	✓	✓	✓	✓	✓	✓

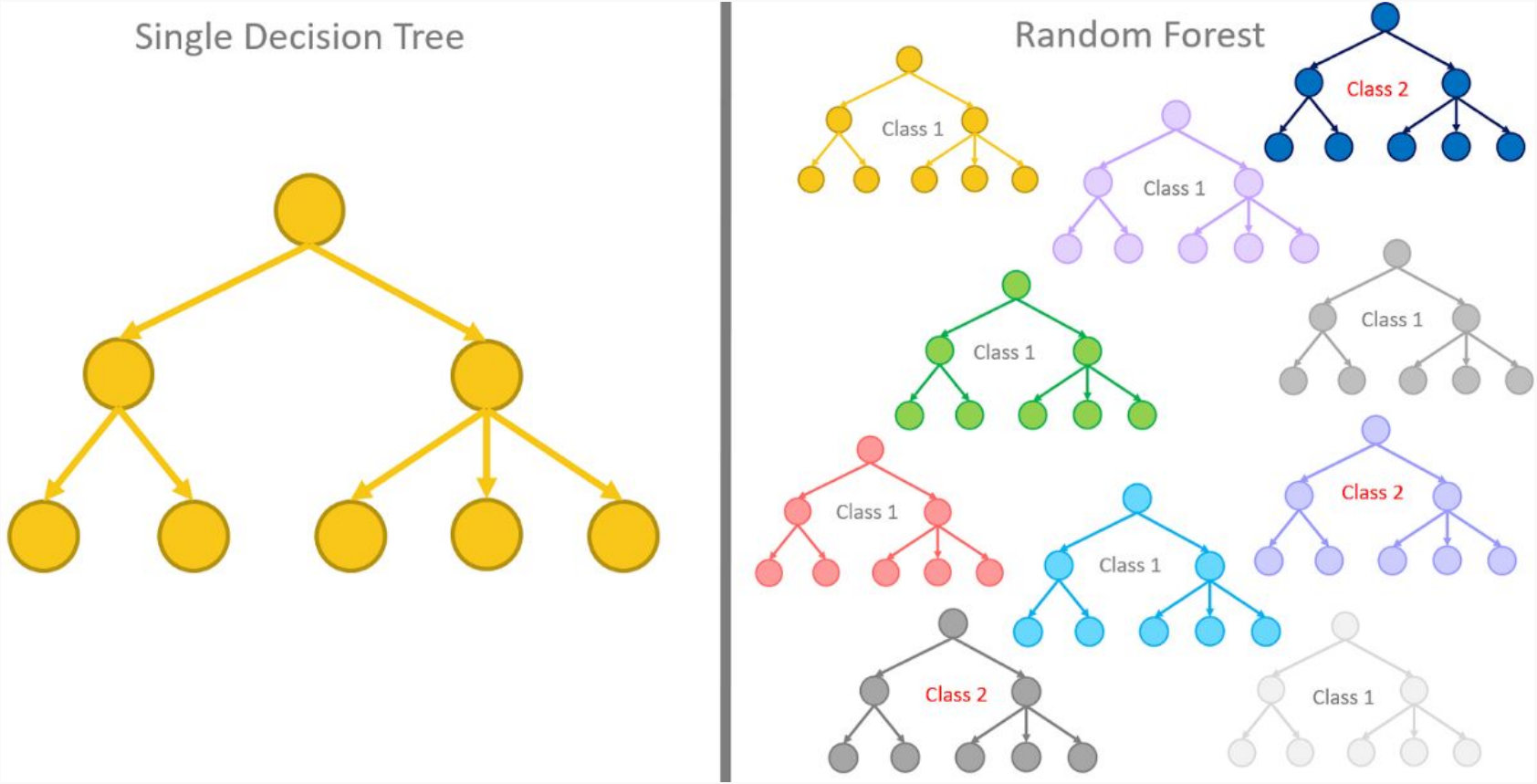
GWT: ground water table; PGA: peak ground acceleration; CPT: cone penetration tests.

Durante and Rathje, 2021

Classification - Decision Tree



Random Forest



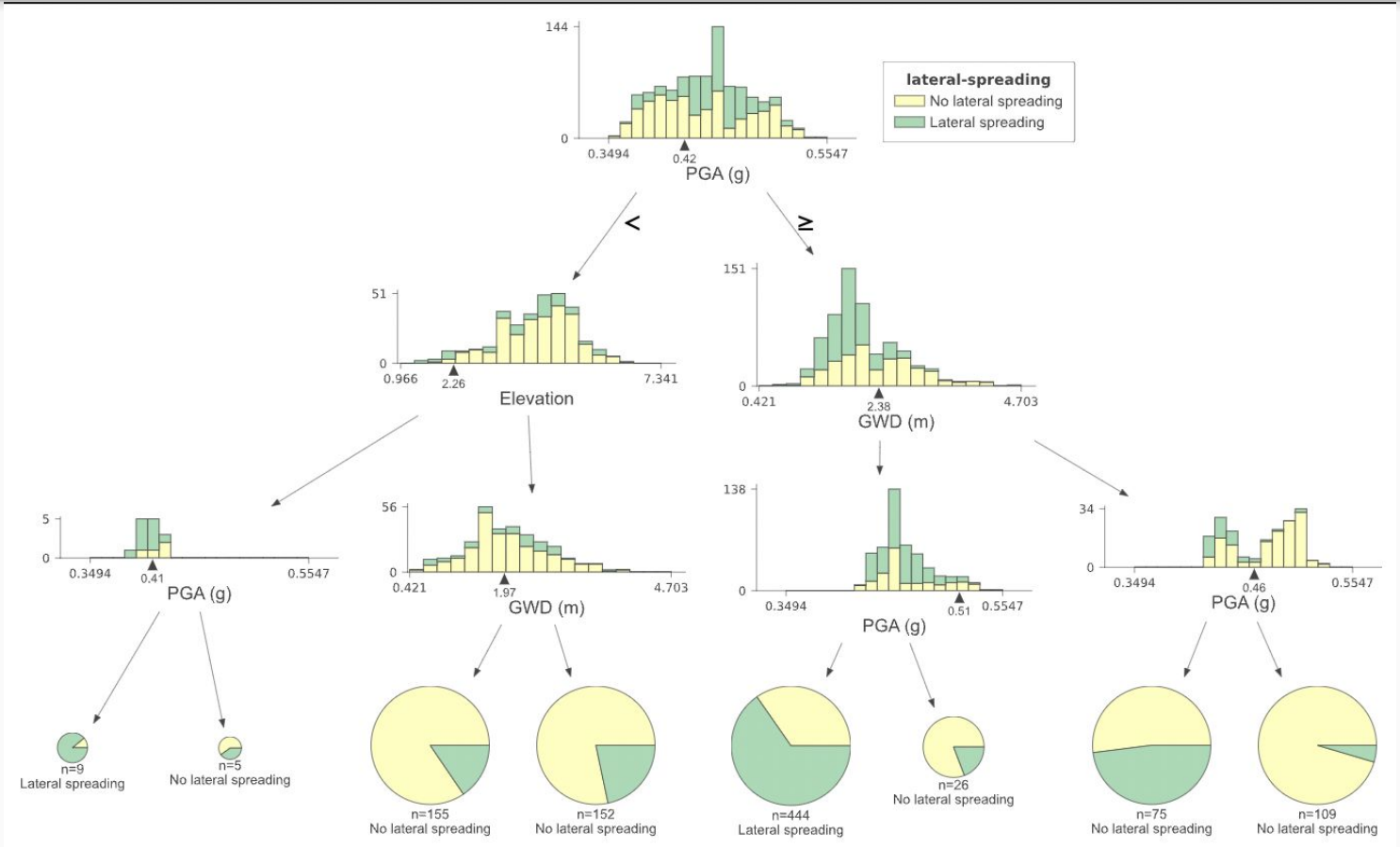
Random Forest - Performance

Table 4. Evaluation metrics for RF models for Yes/No and displacement classification problems

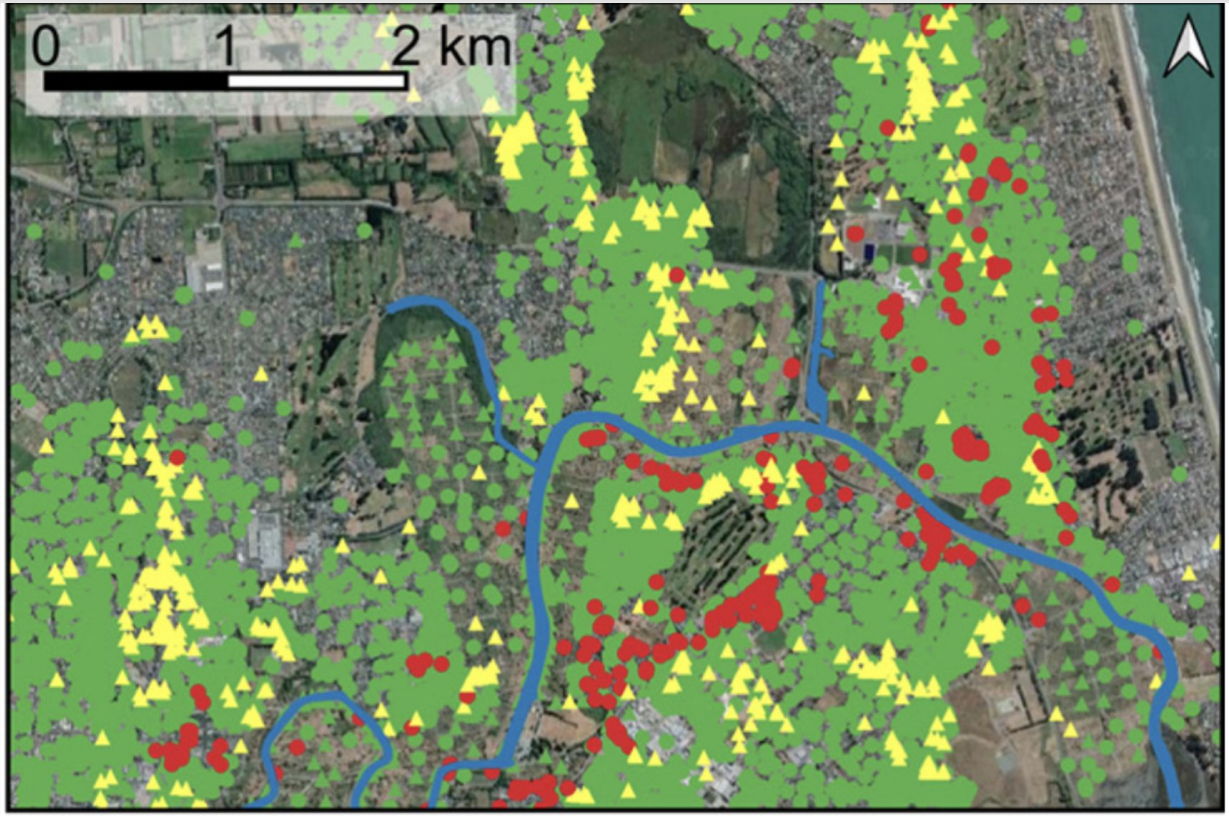
		No CPT data				CPT data	
		Model 0	Model 1	Model 2	Model 3	Model 4	Model 5
Y/N	Accuracy (Overall)	0.76	0.85	0.82	0.88	0.85	0.87
	Recall—Yes	0.58	0.75	0.74	0.82	0.76	0.78
	Recall—No	0.90	0.92	0.88	0.93	0.92	0.93
	Precision—Yes	0.80	0.88	0.82	0.89	0.87	0.89
	Precision—No	0.75	0.84	0.83	0.87	0.84	0.85
	ROC AUC	0.76	0.88	0.84	0.90	0.86	0.88
DISPL	Accuracy (Overall)	0.78	0.88	0.86	0.89	0.88	0.89
	Recall—Class 0	0.88	0.91	0.93	0.92	0.91	0.91
	Recall—Class 1	0.67	0.87	0.79	0.89	0.88	0.90
	Recall—Class 2	0.27	0.42	0.33	0.42	0.53	0.49
	Precision—Class 0	0.79	0.90	0.86	0.92	0.90	0.92
	Precision—Class 1	0.74	0.84	0.84	0.85	0.85	0.84
	Precision—Class 2	0.97	0.98	0.97	0.94	0.95	0.93
	ROC AUC	0.82	0.89	0.87	0.90	0.86	0.88

CPT: cone penetration tests; ROC: receiver operating characteristic; AUC: area under the ROC curve; Y/N: Yes/No Classification problem; DISP: Class Displacement Classification problem.

Visualizing the first tree in RF



Random forest prediction of lateral spreading



Durante and Rathje, 2021

▲ True Positive (TP) ● True Negative (TN) ● False Positive (FP) ▲ False Negative (FN)

SHAP Value: Contribution of each variable

A: $v(\{A\}) - v(\{\}) = 10 - 0 = 10$

B: $v(\{A,B\}) - v(\{A\}) = 60 - 10 = 50$

C: $v(\{A,B,C\}) - v(\{A,B\}) = 100 - 60 = 40$

$v(\{\}) = 0$

$v(\{A\}) = 10$

$v(\{B\}) = 20$

$v(\{C\}) = 30$

$v(\{A,B\}) = 60$

$v(\{B,C\}) = 70$

$v(\{A,C\}) = 90$

$v(\{A,B,C\}) = 100$

$\{\} \rightarrow \{A\} \rightarrow \{A,\ B\} \rightarrow \{A,\ B,\ C\} \parallel A = 10,\ B = 50,\ C = 40$

$\{\} \rightarrow \{A\} \rightarrow \{A,\ C\} \rightarrow \{A,\ B,\ C\} \parallel A = 10,\ B = 10,\ C = 80$

$\{\} \rightarrow \{B\} \rightarrow \{A,\ B\} \rightarrow \{A,\ B,\ C\} \parallel A = 40,\ B = 20,\ C = 40$

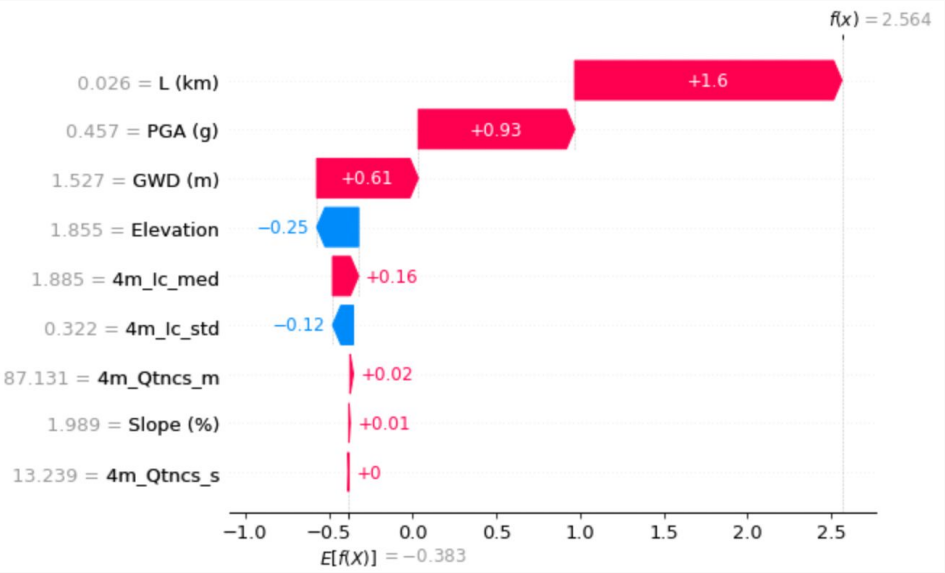
$\{\} \rightarrow \{B\} \rightarrow \{B,\ C\} \rightarrow \{A,\ B,\ C\} \parallel A = 30,\ B = 20,\ C = 50$

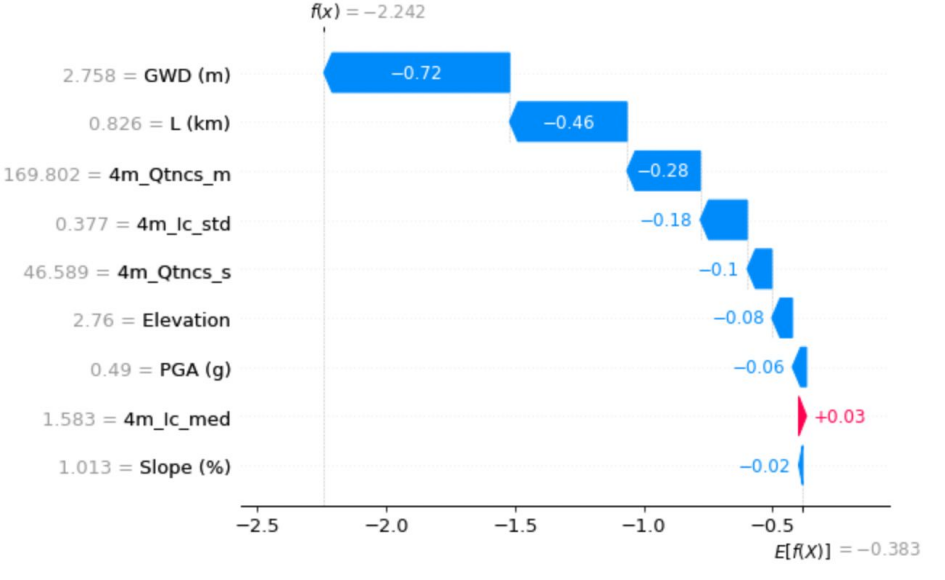
$\{\} \rightarrow \{C\} \rightarrow \{B,\ C\} \rightarrow \{A,\ B,\ C\} \parallel A = 30,\ B = 40,\ C = 30$

$\{\} \rightarrow \{C\} \rightarrow \{A,\ C\} \rightarrow \{A,\ B,\ C\} \parallel A = 60,\ B = 10,\ C = 30$

$$A_{avg} = 30,\ B_{avg} = 25,\ C_{avg} = 45$$

Random Forest post-hoc explanation

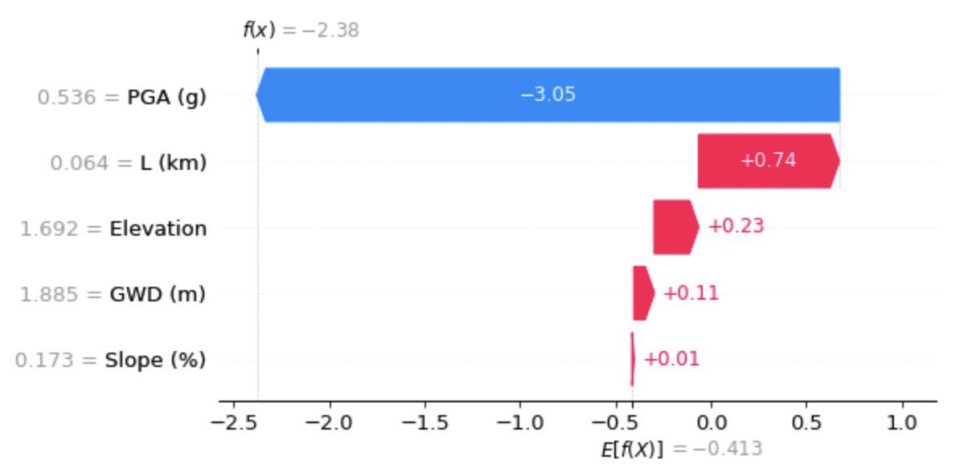




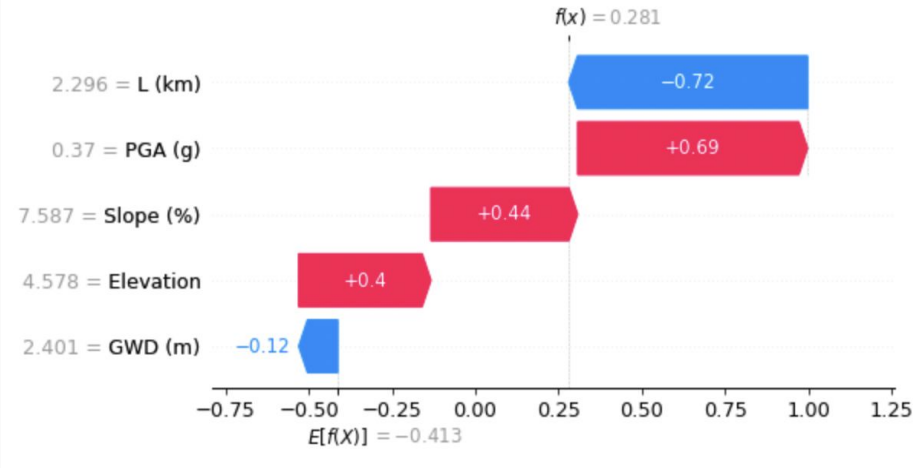
Lateral spreading - Prob (Lat spread) = 90%

No lateral spreading - Prob (Lat spread) = 7%

Incorrect learning

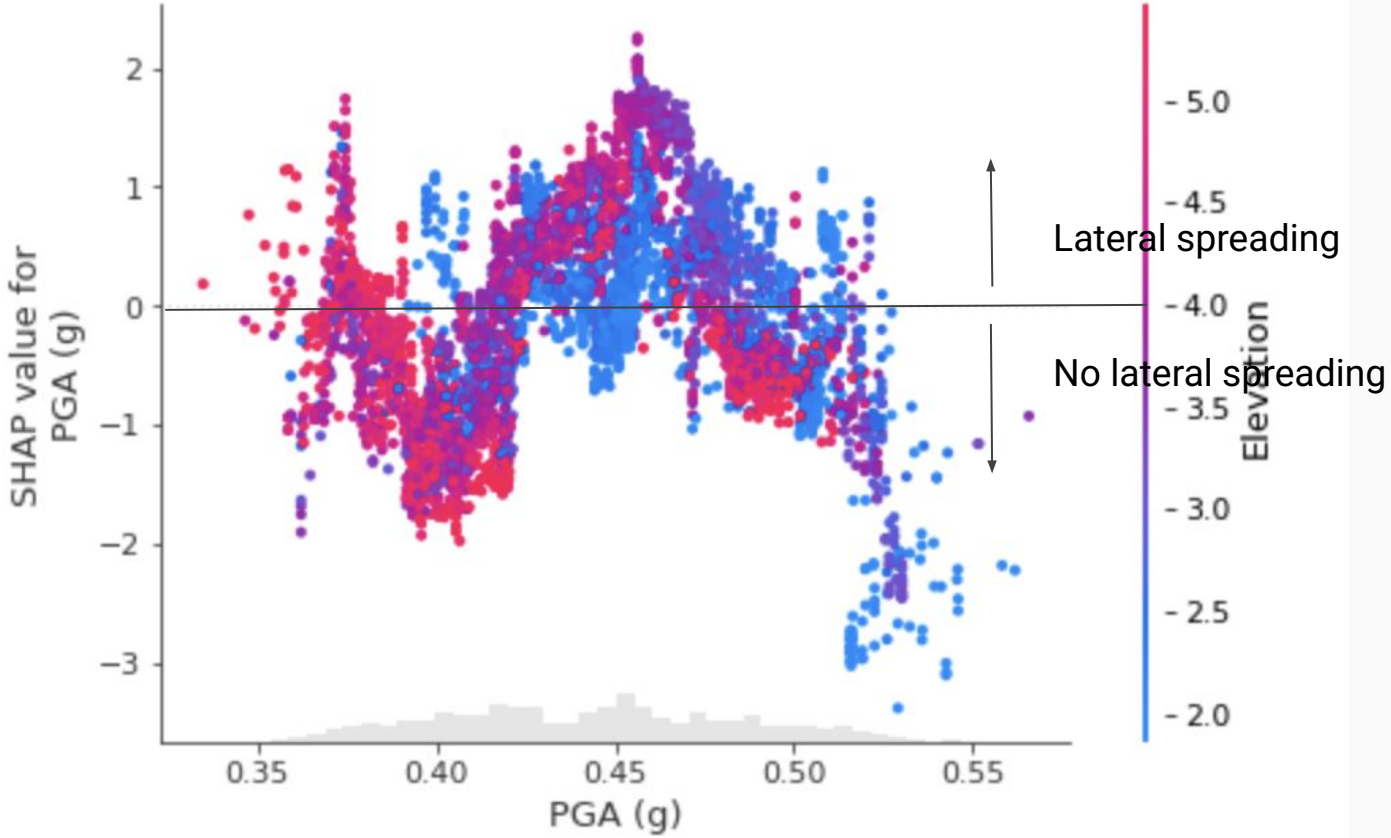


Incorrectly predicting no lateral spreading
Prob (Lat spread) = 8%

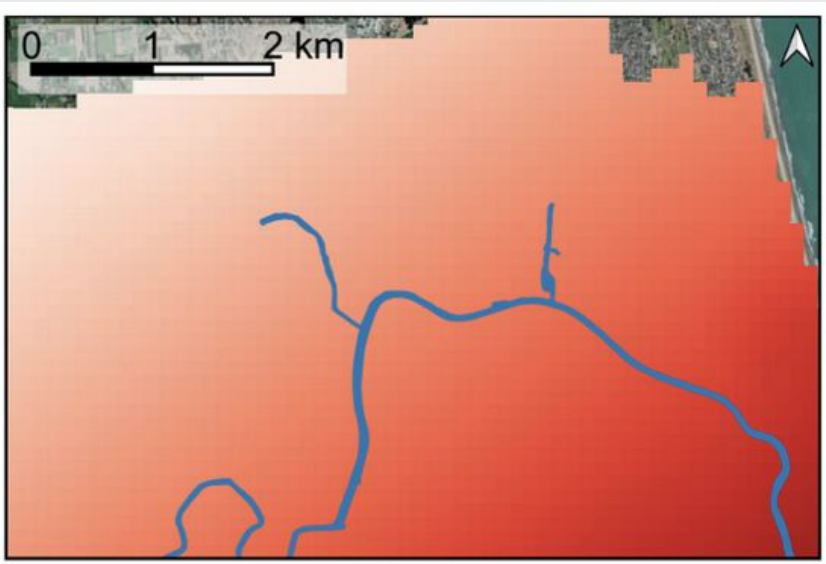


Incorrectly predicting lateral spreading
Prob (Lat spread) = 53%

Why PGA has a bad influence?

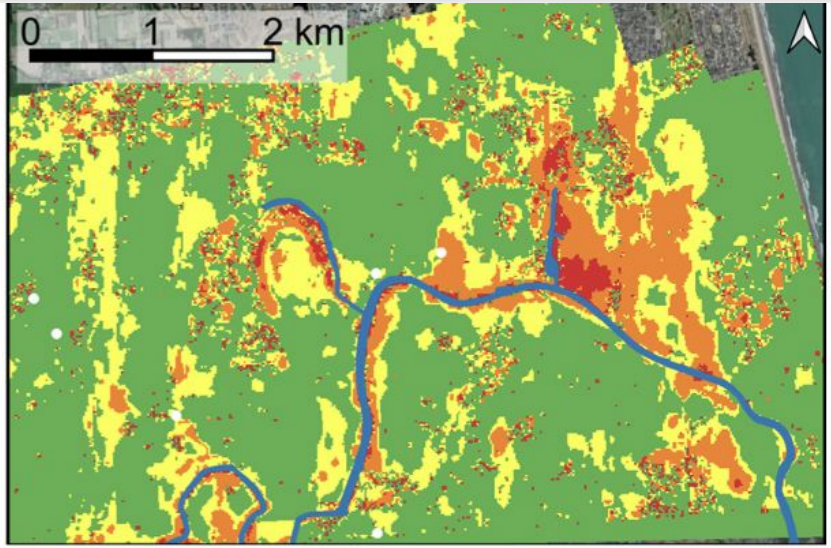


Bad learning of PGA relation



PGA (g)

0.30 0.60



Observed displacement (m)

- None (< 0.30)
- 0.30 - 0.50
- 0.50 - 1.00
- > 1.00

Multi-Layer Perceptron

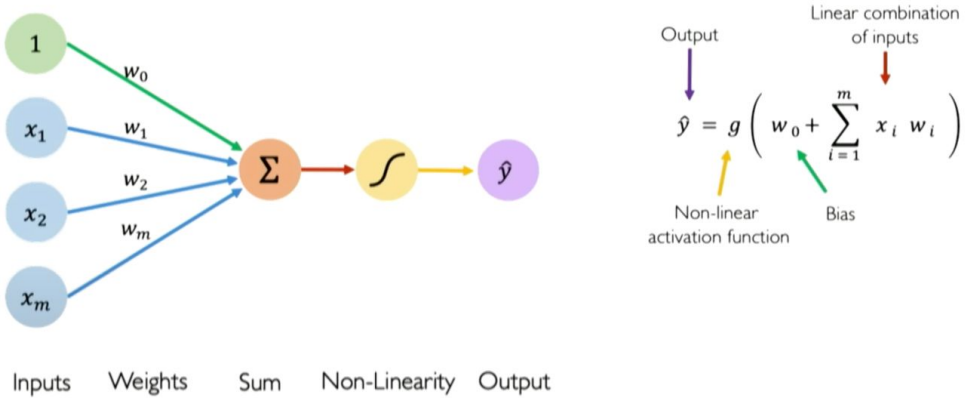
Now let's try with MLP

A perceptron computes:

$$z = \mathbf{w}^T\mathbf{x} + b = \sum_{i=1}^{n} w_i x_i + b$$

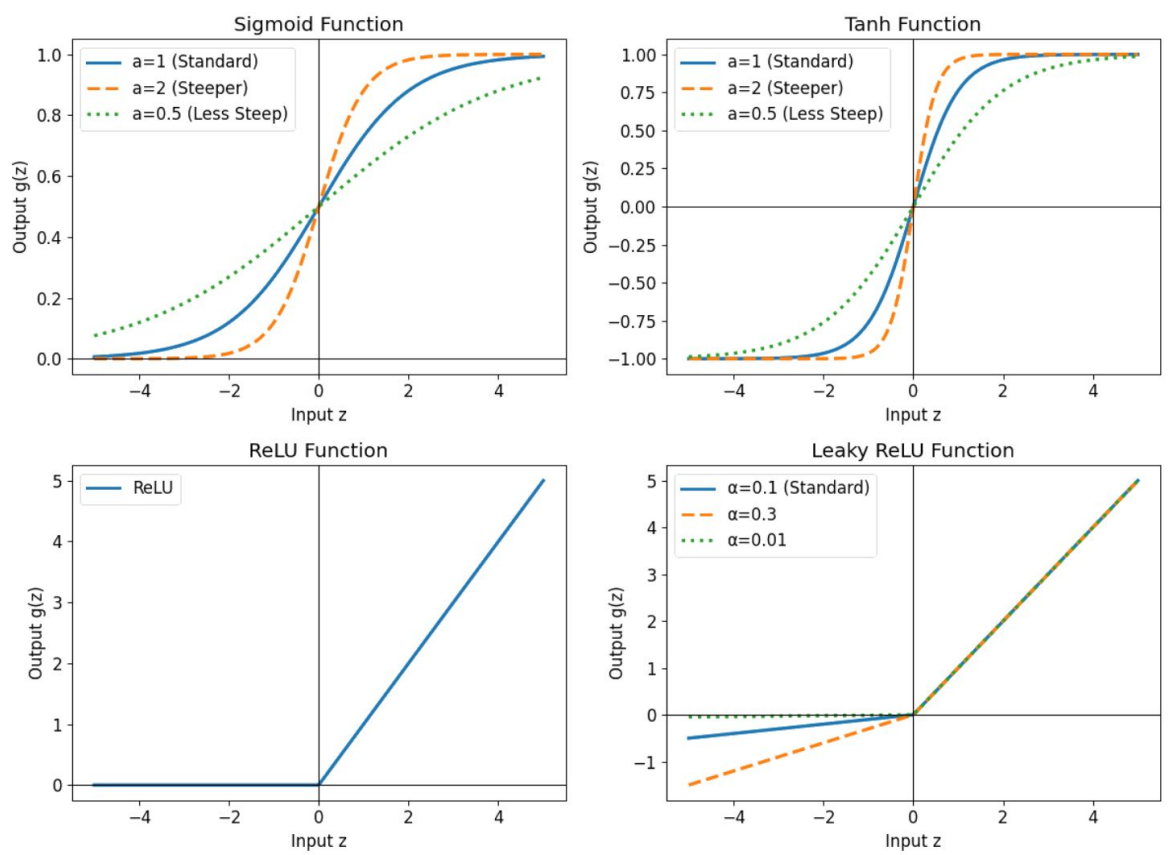
$$\hat{y} = g(z)$$

where g is the activation function.

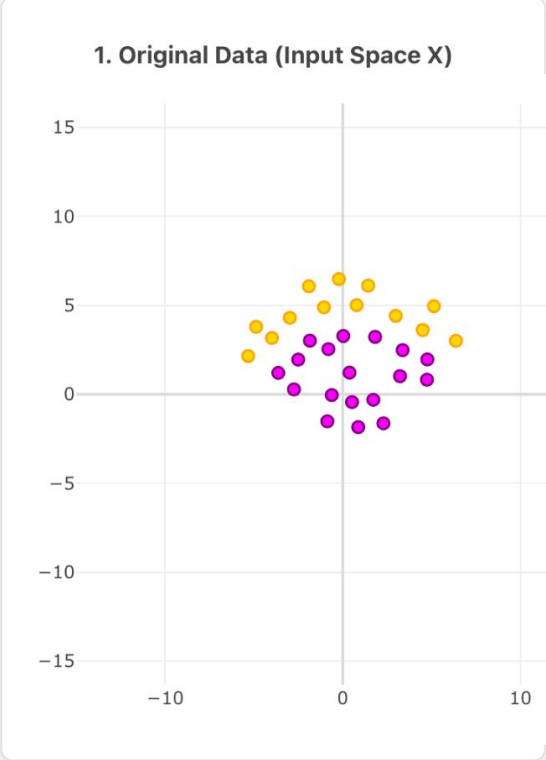


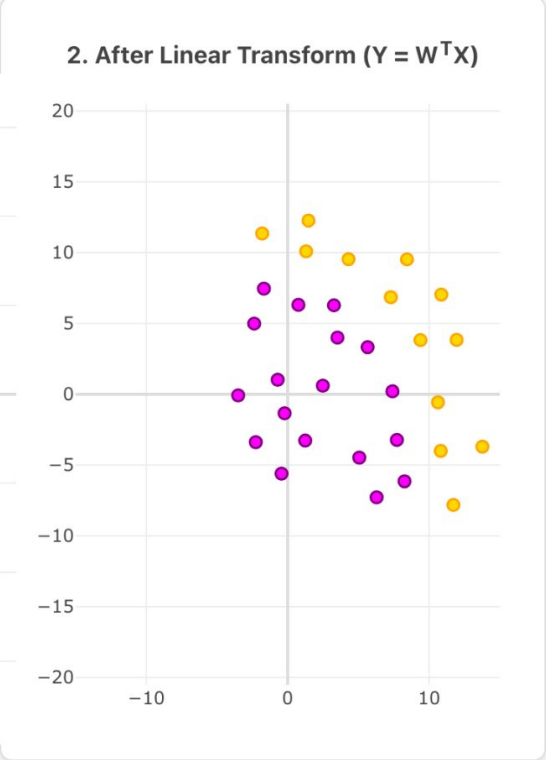
Perceptron architecture

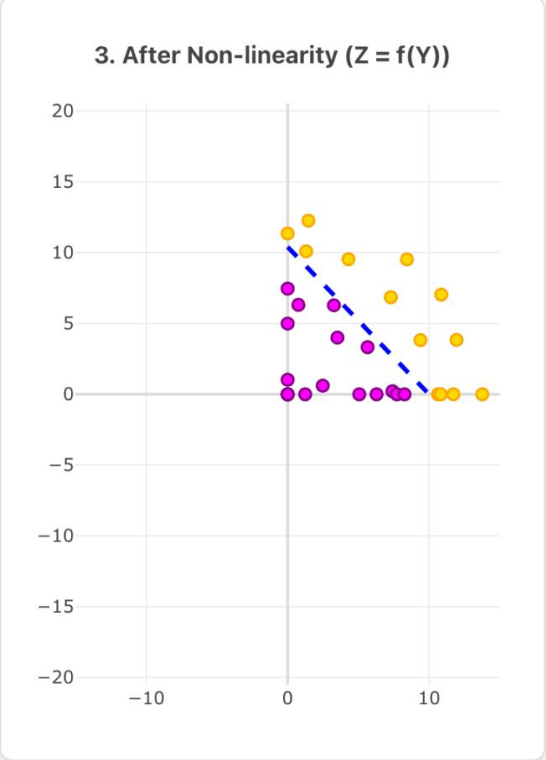
Why do we need non-linearity?



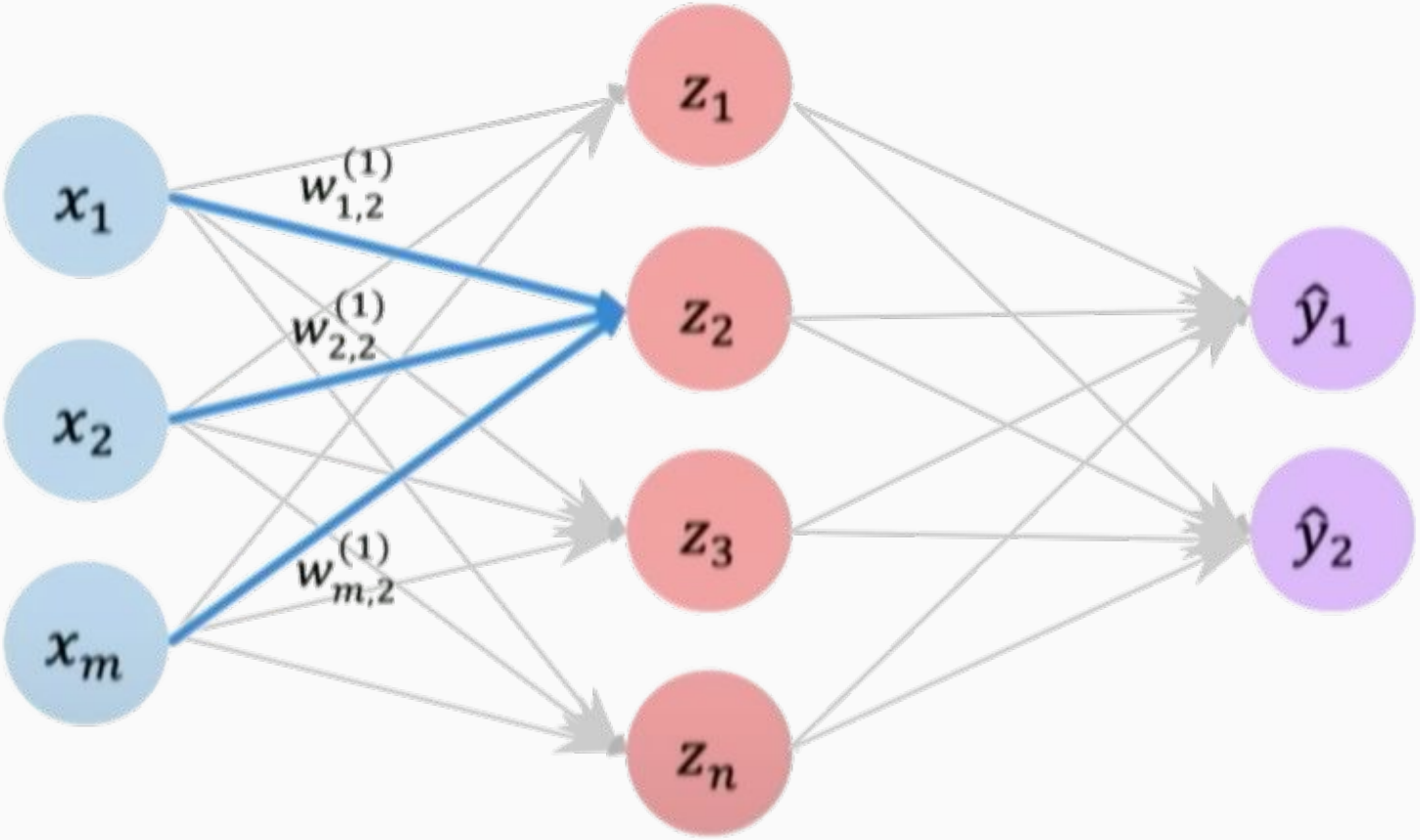
Why do we need non-linearity?



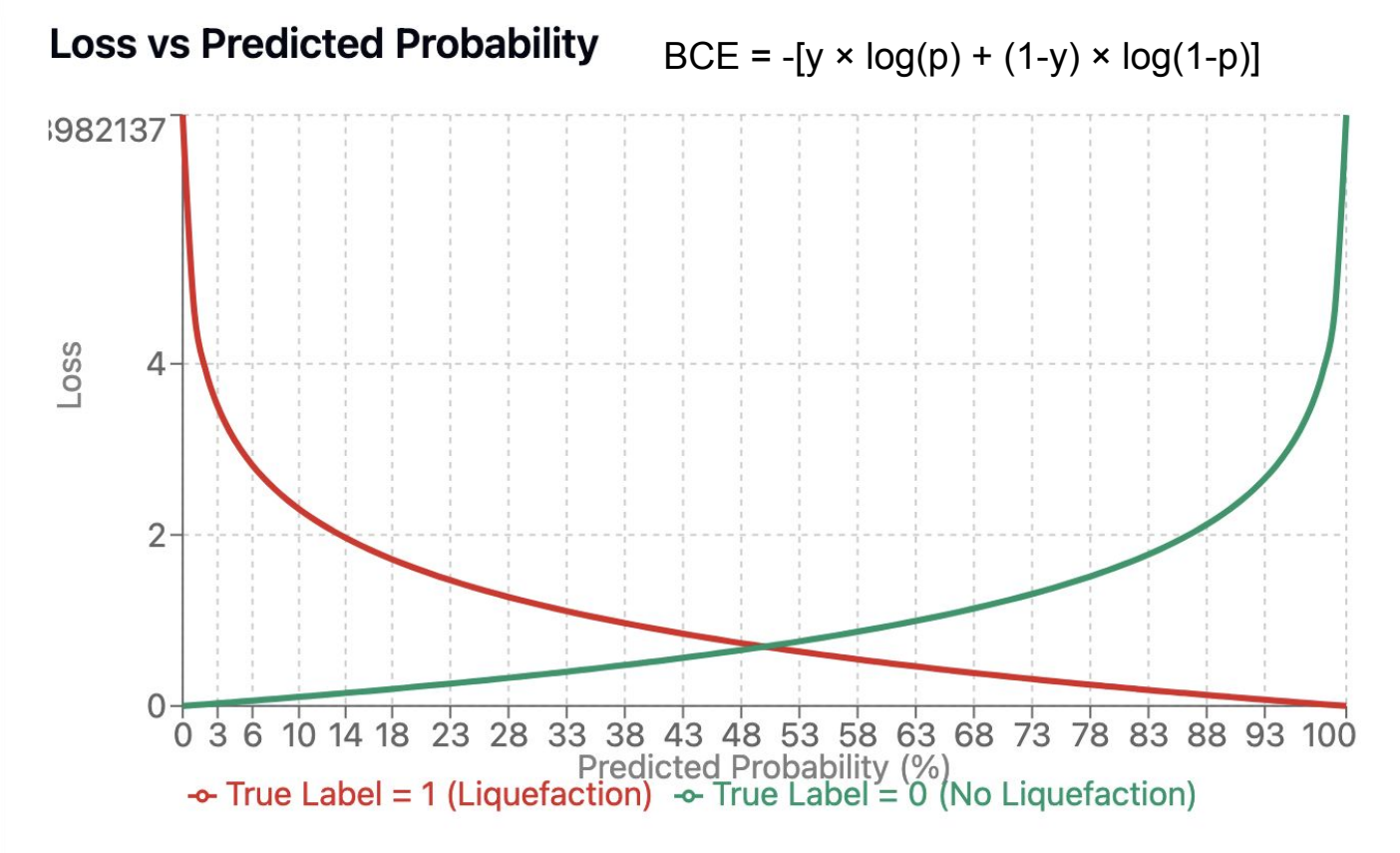




Multi-Layer Perceptron

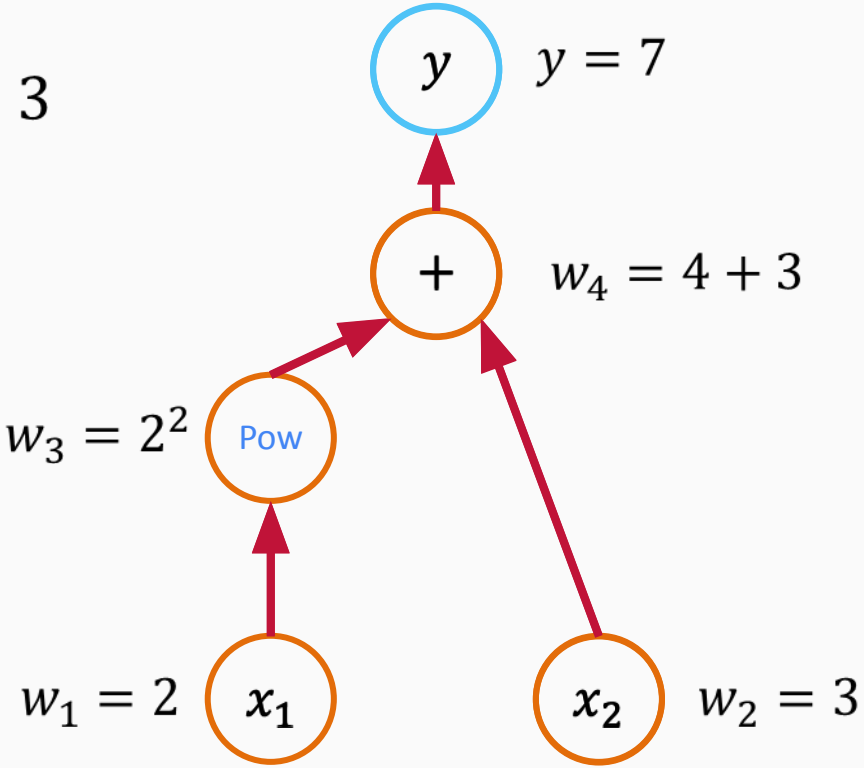


Define a misfit (loss)



Automatic Differentiation: Forward Pass

$x_1 = 2, x_2 = 3$

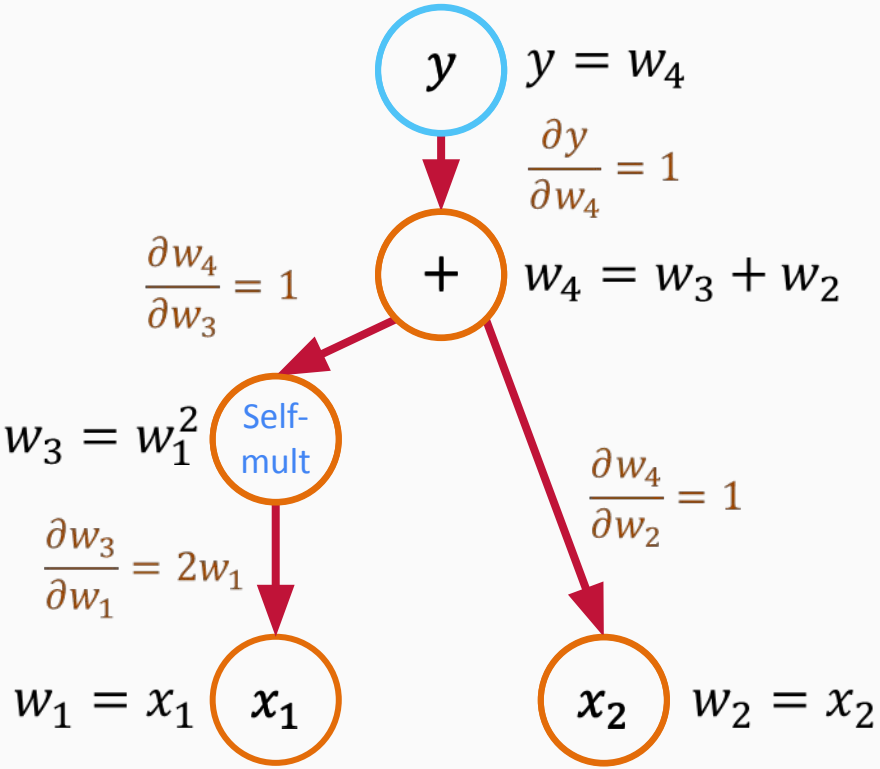


$$y = x_1^2 + x_2$$

$$y = 2^2 + 3$$

$$y = 7$$

Automatic Differentiation: Reverse Mode



$$y = x_1^2 + x_2$$

$$\nabla y = \left(\frac{\partial y}{\partial x_1}, \frac{\partial y}{\partial x_2}\right)$$

$$\boxed{\nabla y = (2x_1, 1)}$$

We can find the gradient with respect to the output y

Using $x_1 = 2, x_2 = 3$:

$$\nabla y = (4, 1)$$

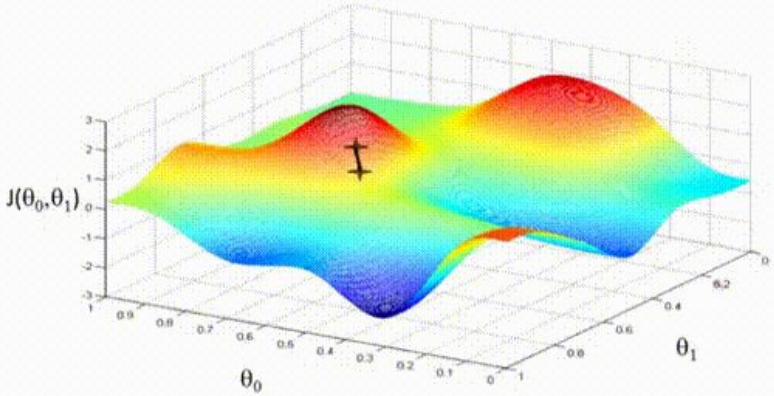
Minimizing the loss: Stochastic Gradient Descent

Basic Algorithm:

1. Initialize weights randomly
2. Loop until convergence:
3. Compute gradient $\frac{\partial \mathcal{L}}{\partial \theta}$
4. Update weights: $\theta \leftarrow \theta - \eta \frac{\partial \mathcal{L}}{\partial \theta}$
5. Return weights

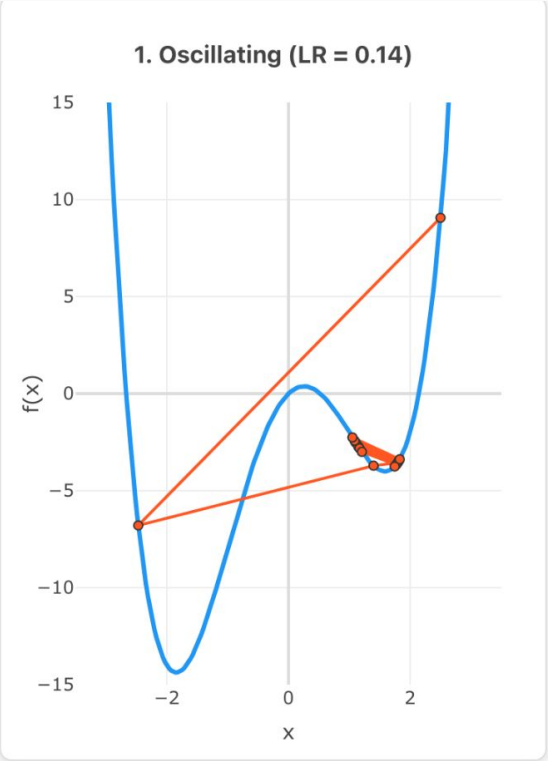
$$\theta_{new} = \theta_{old} - \eta \nabla \mathcal{L}(\theta)$$

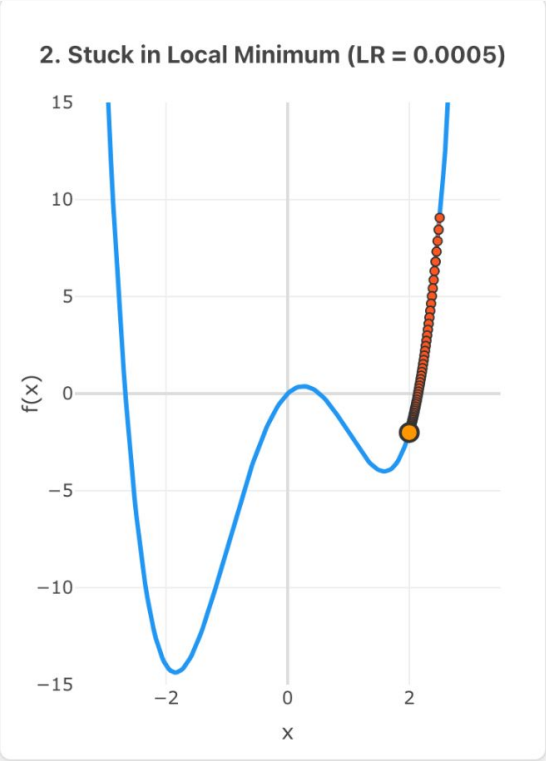
where η is the learning rate.

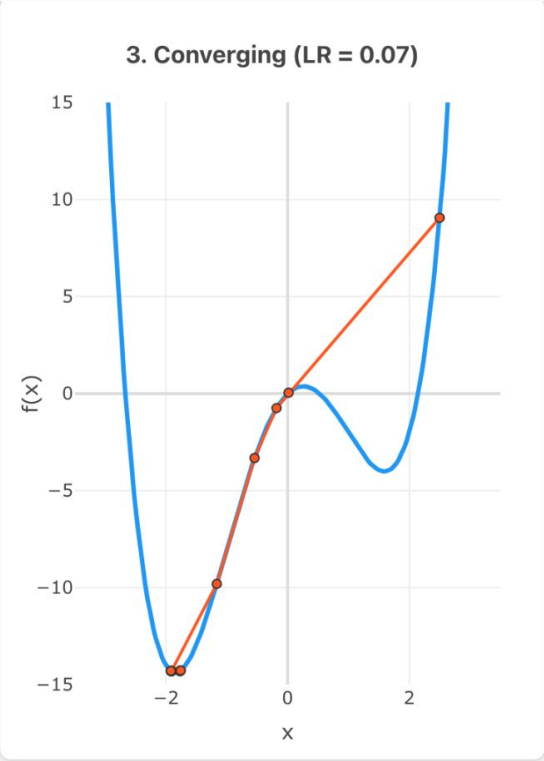


Andrew Ng

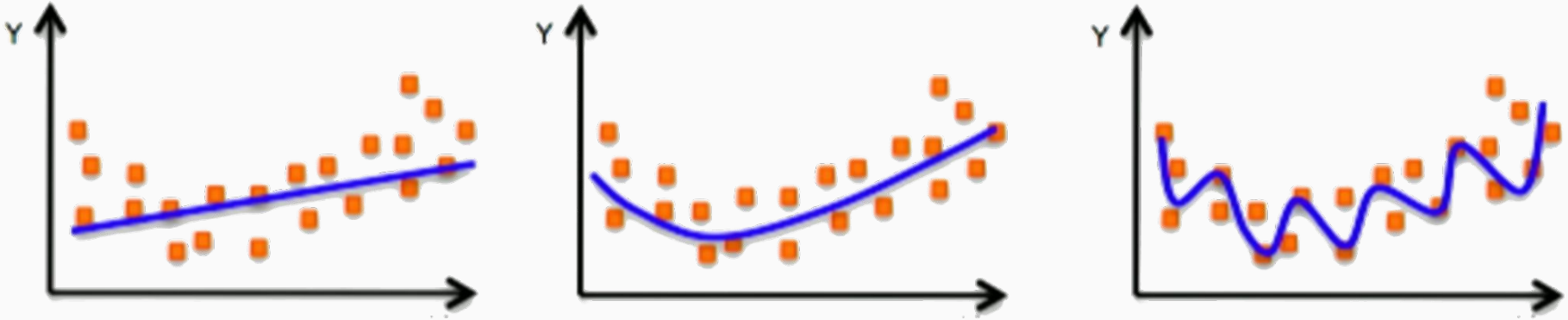
Gradient Descent and Learning Rate







Identifying overfitting

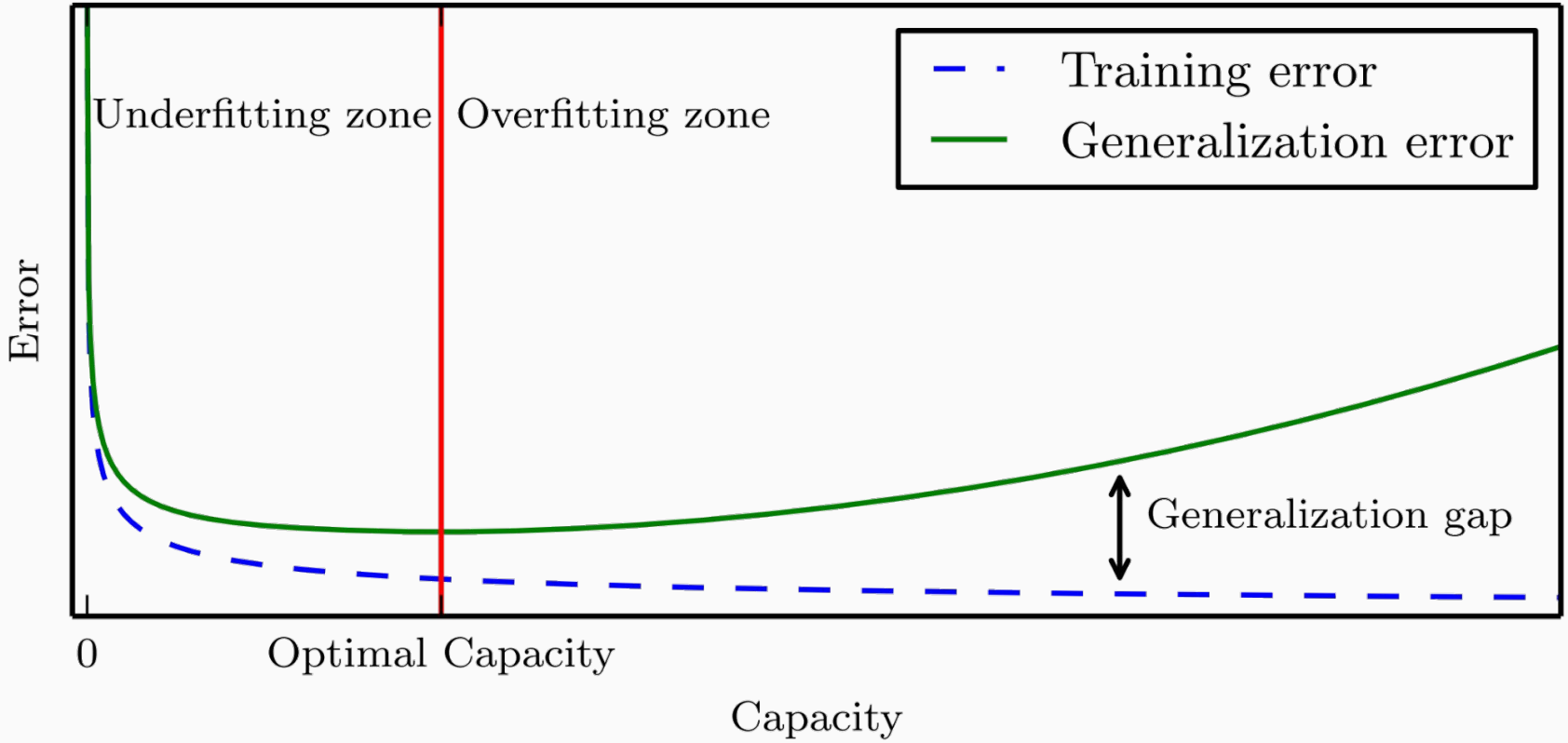


Underfitting
Model does not have capacity to fully learn the data

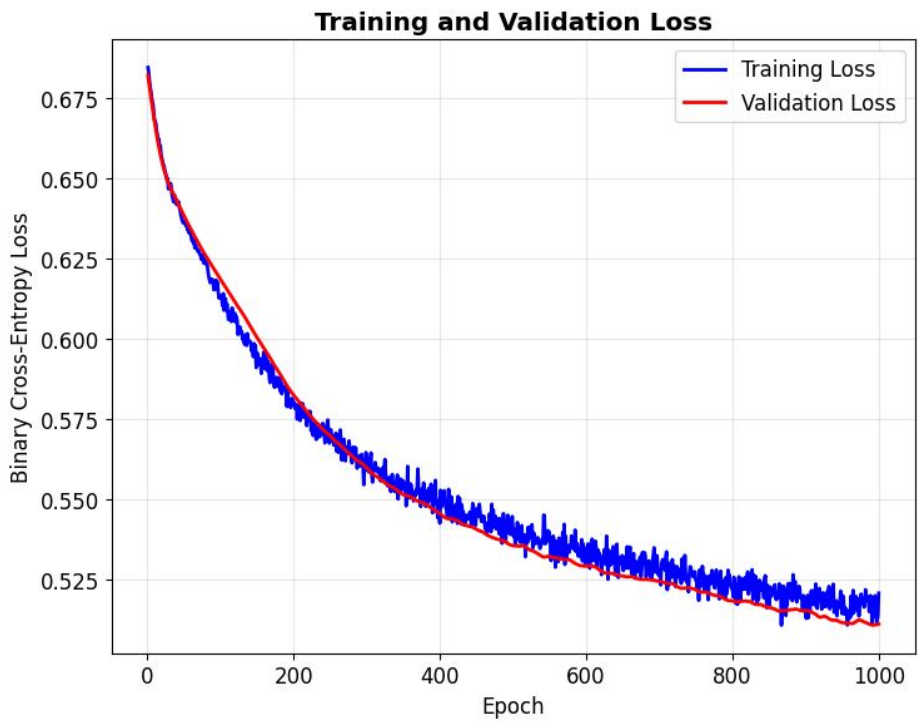
Ideal fit

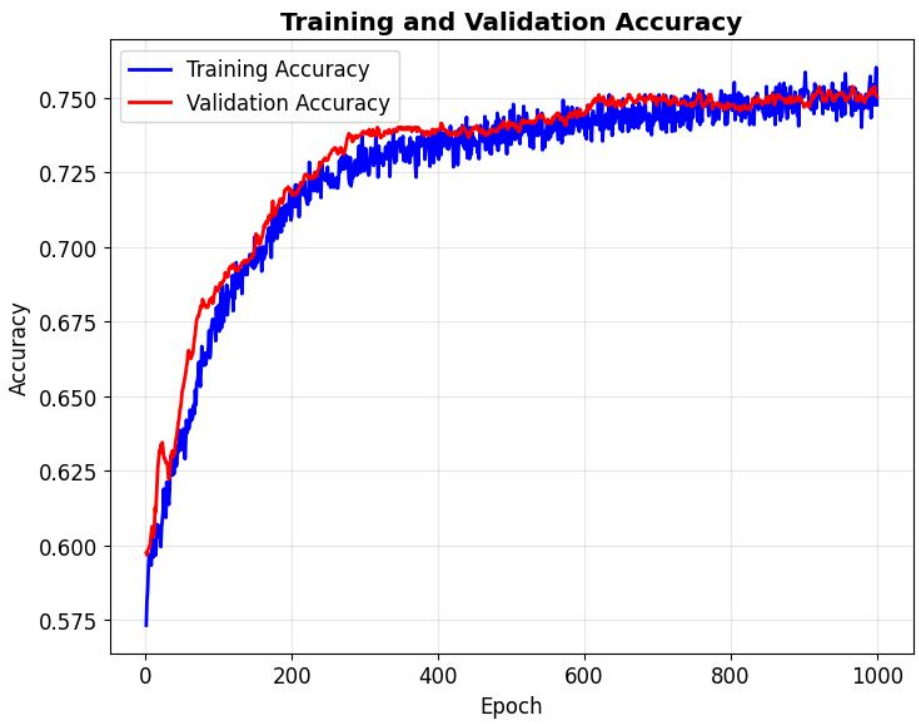
Overfitting
Too complex, extra parameters, does not generalize well

How to identify overfitting?

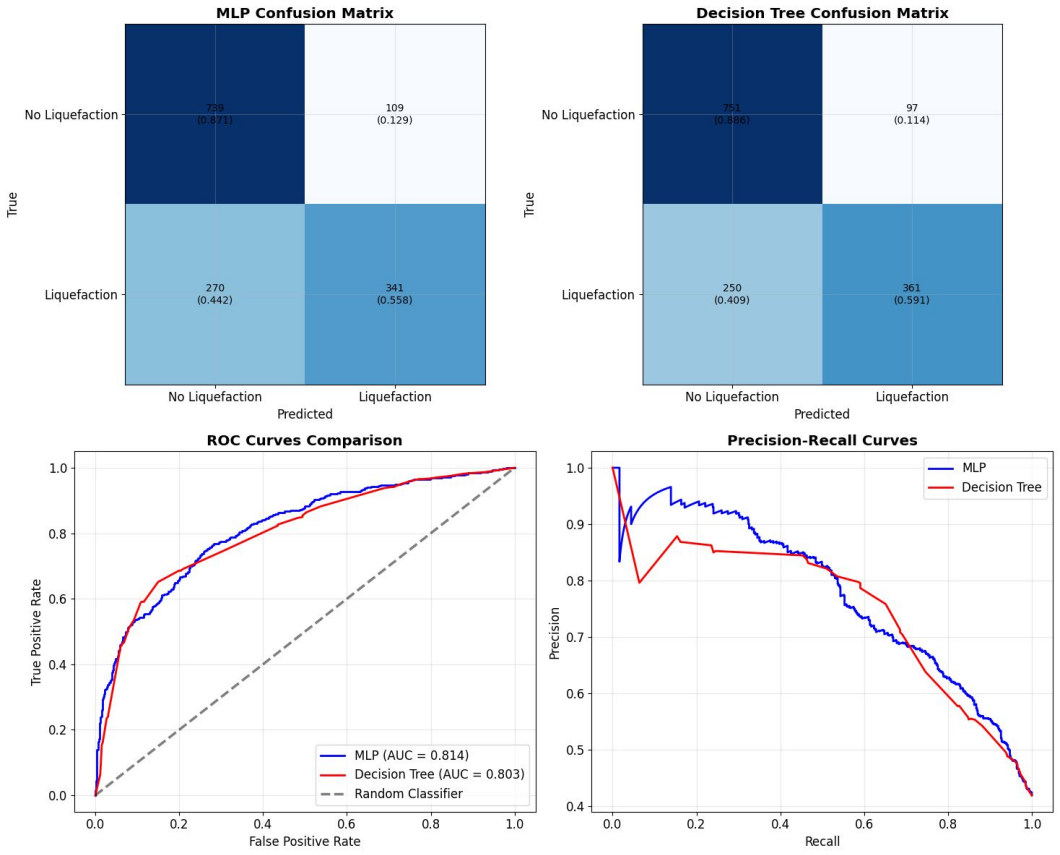


Training vs validation

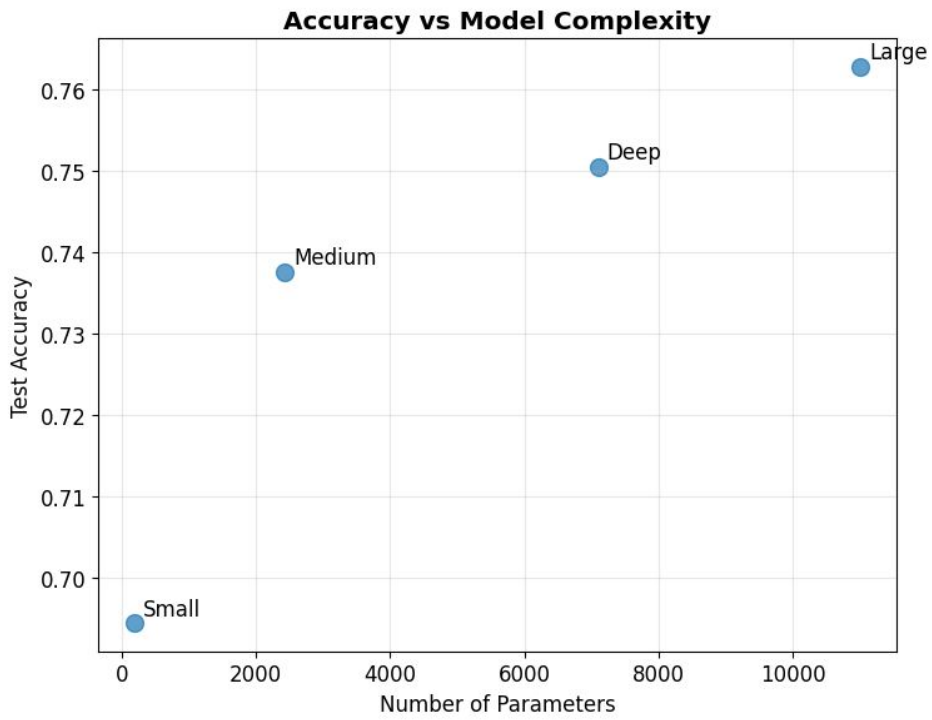


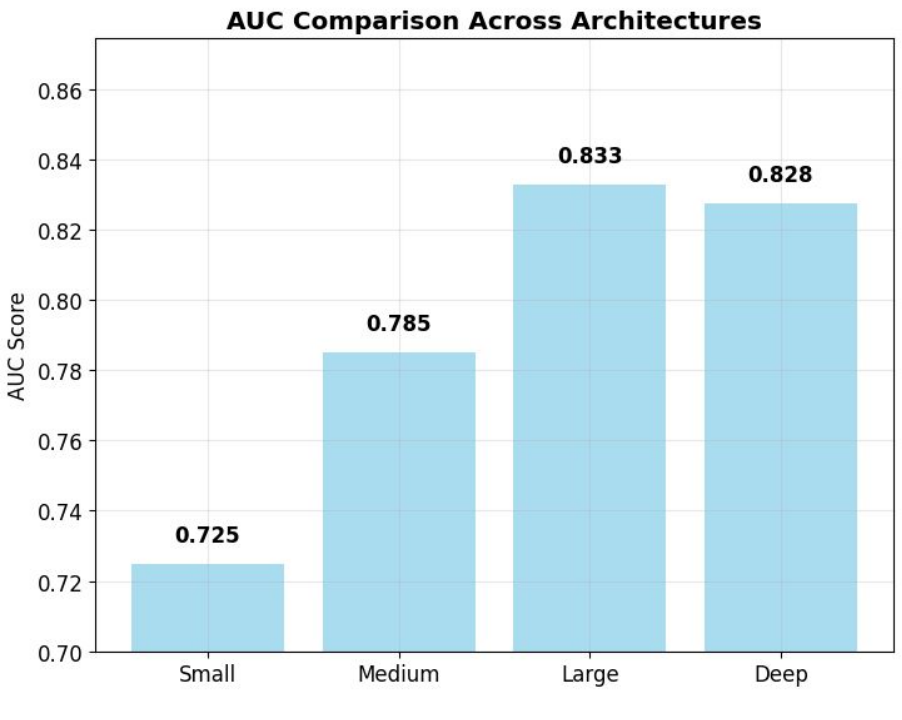


MLP results


MLP Confusion Matrix
No Liquefaction
Liquefaction
True
739
(0.871)
109
(0.129)
270
(0.442)
341
(0.558)
No Liquefaction
Liquefaction
Predicted
Decision Tree Confusion Matrix
No Liquefaction
Liquefaction
True
751
(0.886)
97
(0.114)
250
(0.409)
361
(0.591)
No Liquefaction
Liquefaction
Predicted
ROC Curves Comparison
True Positive Rate
False Positive Rate
MLP (AUC = 0.814)
Decision Tree (AUC = 0.803)
Random Classifier
Precision-Recall Curves
Precision
Recall
MLP
Decision Tree

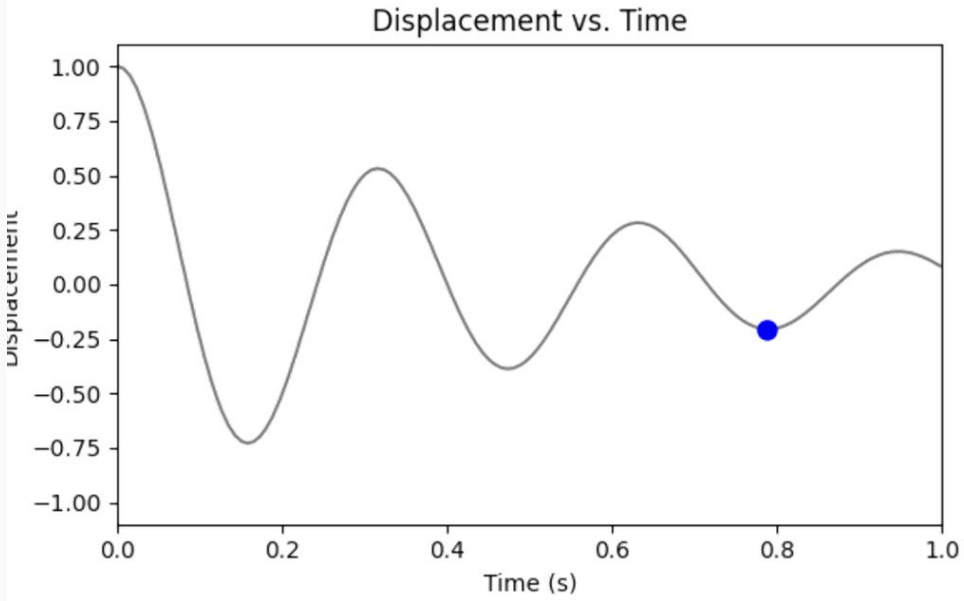
Deeper/Larger Networks are better!

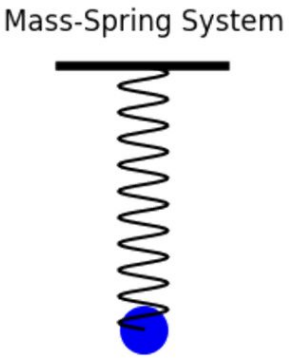




Physics Informed Neural Network

Damped Harmonic Oscillator





A mass m attached to a spring (constant k) with damping (coefficient c). The displacement $u(t)$ from equilibrium satisfies:

$$m\frac{d^2u}{dt^2} + c\frac{du}{dt} + ku = 0$$

Stage 1: The Data-Only Approach

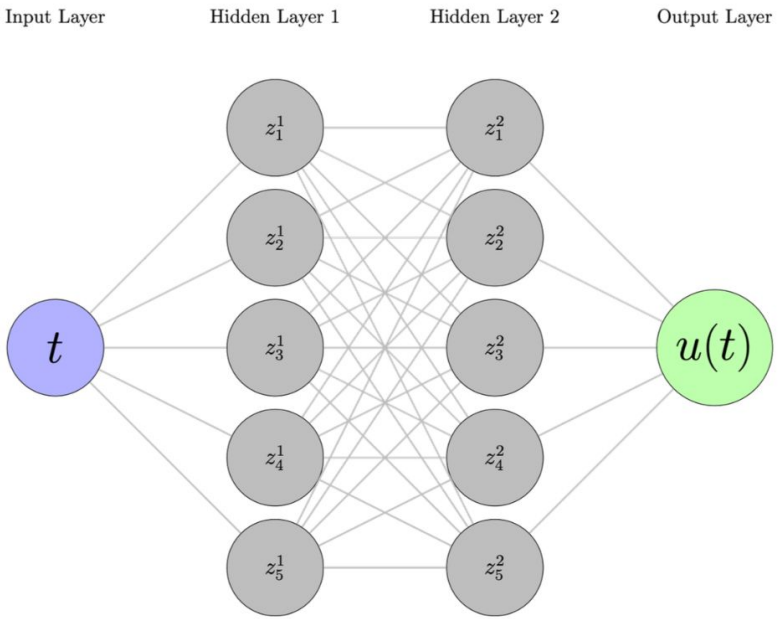
Idea: Train a standard neural network to fit the sparse data.

Loss Function: Mean Squared Error

$$\mathcal{L}_{\text{data}}(\theta) = \frac{1}{N}\sum_{i=1}^{N}|\hat{u}_\theta(t_i) - u_i|^2$$

Architecture:

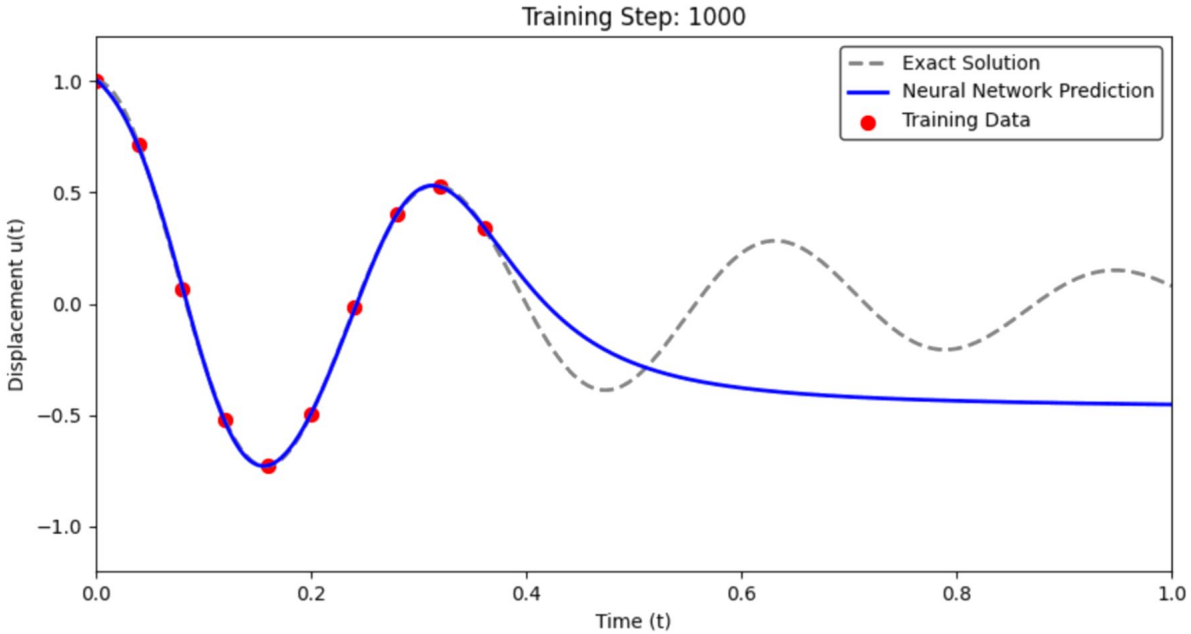
- Input: Time t
- Hidden Layers: Tanh activations
- Output: Displacement $\hat{u}_\theta(t)$



Standard NN for function fitting.

The Failure of the Data-Only Approach

Result: The network fits the training points but fails catastrophically between them.



The network overfits to the sparse data and does not respect the underlying physics.

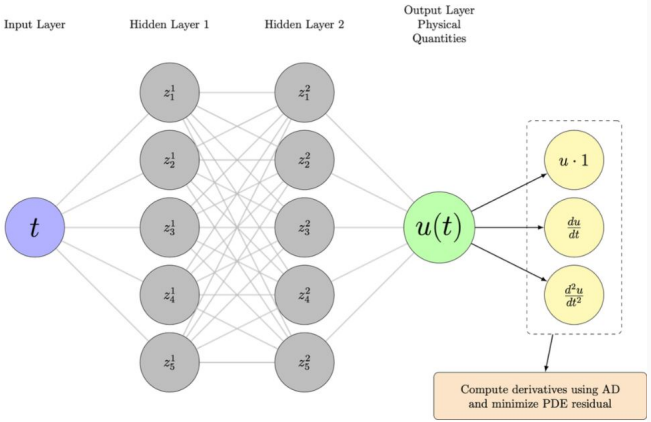
Stage 2: Enter Physics-Informed Neural Networks

The Key Insight: Don't just fit data. Enforce the differential equation itself!

Physics Residual: We define a residual based on the ODE:

$$\mathcal{R}_\theta(t) = m\frac{d^2\hat{u}_\theta}{dt^2} + c\frac{d\hat{u}_\theta}{dt} + k\hat{u}_\theta$$

If the solution is correct, $\mathcal{R}_\theta(t)$ should be zero.



PINN architecture with physics loss.

The Complete PINN Loss Function

The total loss is a combination of data fit and physics enforcement.

Total Loss:

$$\mathcal{L}_{\text{total}}(\theta) = \mathcal{L}_{\text{data}}(\theta) + \lambda \mathcal{L}_{\text{physics}}(\theta)$$

Data Loss

Ensures the solution passes through the measurements.

$$\mathcal{L}_{\text{data}}(\theta) = \frac{1}{N_{\text{data}}} \sum_{i=1}^{N_{\text{data}}} |\hat{u}_\theta(t_i) - u_i|^2$$

Physics Loss

Ensures the solution obeys the ODE at random "collocation" points.

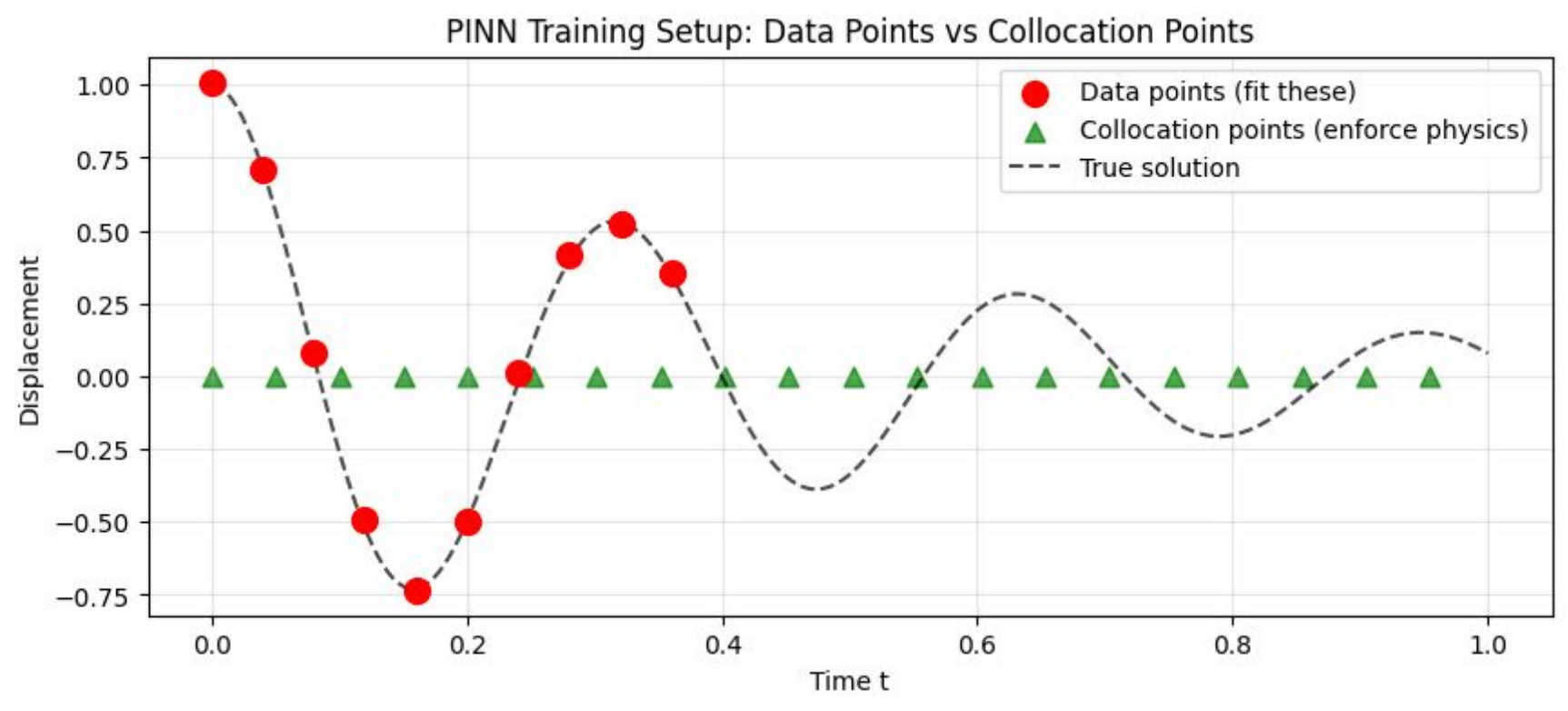
$$\mathcal{L}_{\text{physics}}(\theta) = \frac{1}{N_{\text{colloc}}} \sum_{j=1}^{N_{\text{colloc}}} |\mathcal{R}_\theta(t_j)|^2$$

Secret Weapon: Automatic Differentiation

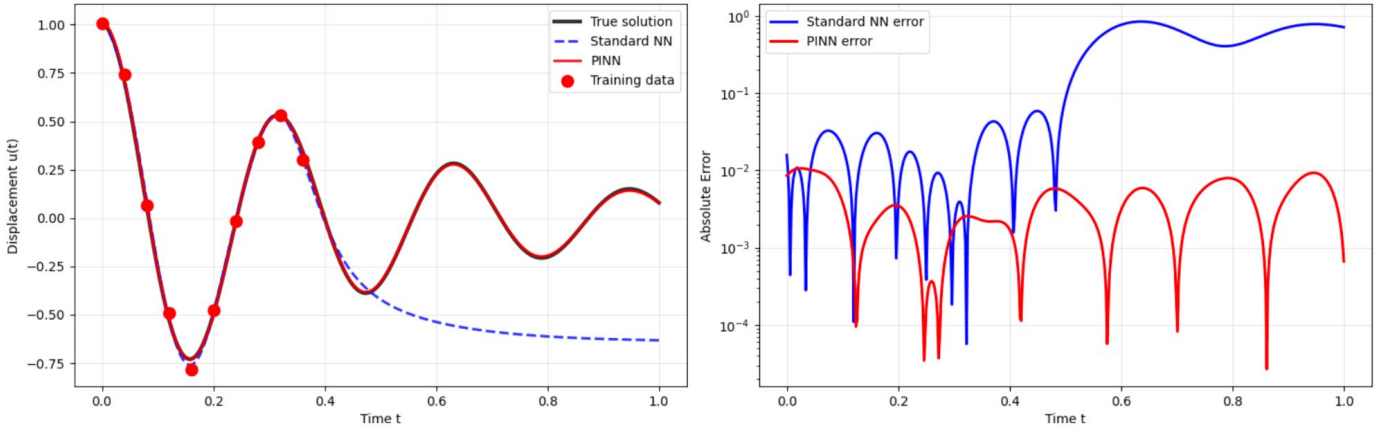
Critical Question: How do we compute $\frac{d\hat{u}_\theta}{dt}$ and $\frac{d^2\hat{u}_\theta}{dt^2}$?

```python
# First derivative: du/dt
du_dt = torch.autograd.grad(
    outputs=u,
    inputs=t_colloc,
    grad_outputs=torch.ones_like(u),
    create_graph=True,  # Allow higher-order derivatives
    retain_graph=True
)[0]

# Second derivative: d²u/dt²
d2u_dt2 = torch.autograd.grad(
    outputs=du_dt,
    inputs=t_colloc,
    grad_outputs=torch.ones_like(du_dt),
    create_graph=True,
    retain_graph=True
)[0]

# Physics residual: R = m * d²u/dt² + c * du/dt + k * u
residual = m * d2u_dt2 + c * du_dt + k * u
```

PINNs: Collocation points



PINNs vs Data-Driven NN



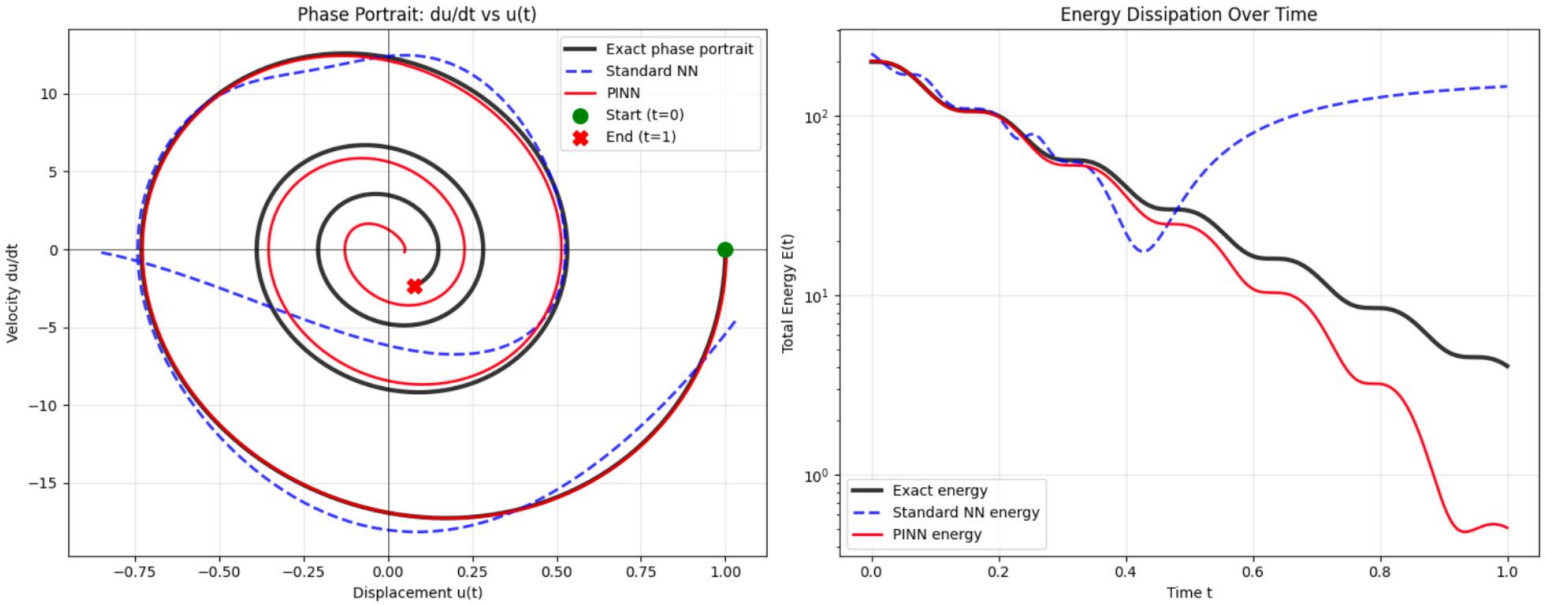
Placeholder for the comparison plot from the notebook.

Observation:

- **Standard NN:** Fits data points, but fails to generalize.
- **PINN:** Fits data points AND follows the physics, resulting in a globally accurate solution.

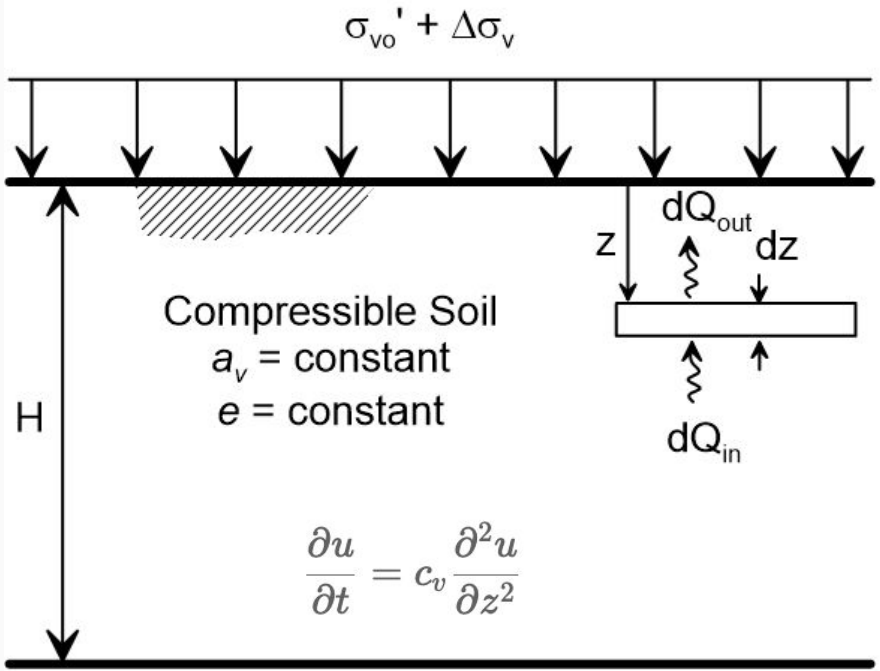
PINNs vs Data-Driven NN

The ultimate test: Does the PINN learn physically consistent derivatives?

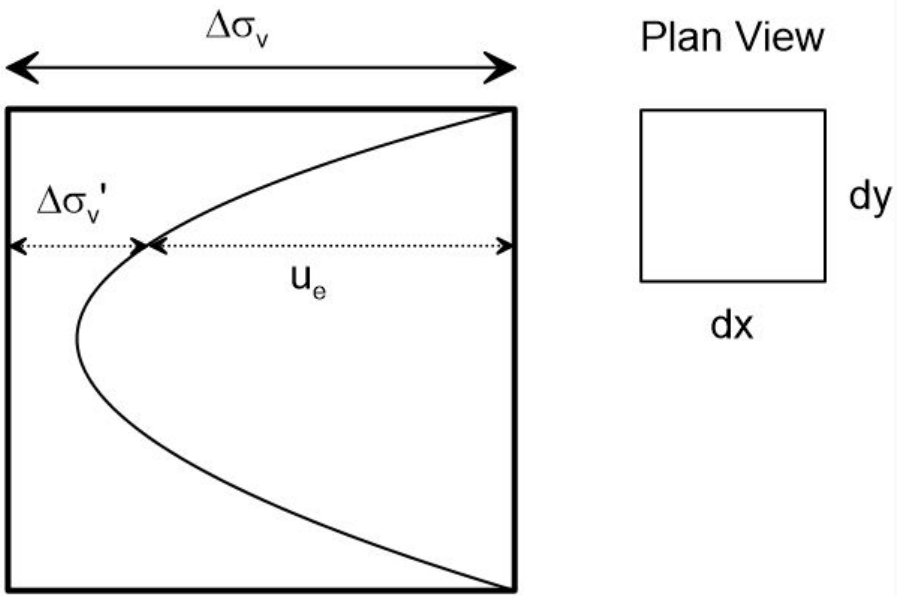


Phase portrait plots.

PINNs for 1D Consolidation



$$\frac{\partial u}{\partial t} = c_v \frac{\partial^2 u}{\partial z^2}$$



$$u(z,t) = \frac{4u_0}{\pi} \sum_{n=0}^{\infty} \frac{\sin\left(\frac{(2n+1)\pi z}{2H}\right)}{2n+1} \exp\left(-\frac{(2n+1)^2 \pi^2 c_v t}{4H^2}\right)$$

https://uclageo.com/SoilMechanicsNotes/Section3.3.php

PINNs: 1D Consolidation

$$\frac{\partial u}{\partial t} = c_v \frac{\partial^2 u}{\partial z^2}$$

$$\mathcal{L}_{\text{total}}(\theta) = \mathcal{L}_{\text{data}}(\theta) + \lambda_{\text{phys}}\mathcal{L}_{\text{physics}}(\theta) + \lambda_{\text{bc}}\mathcal{L}_{\text{BC}}(\theta) + \lambda_{\text{ic}}\mathcal{L}_{\text{IC}}(\theta)$$

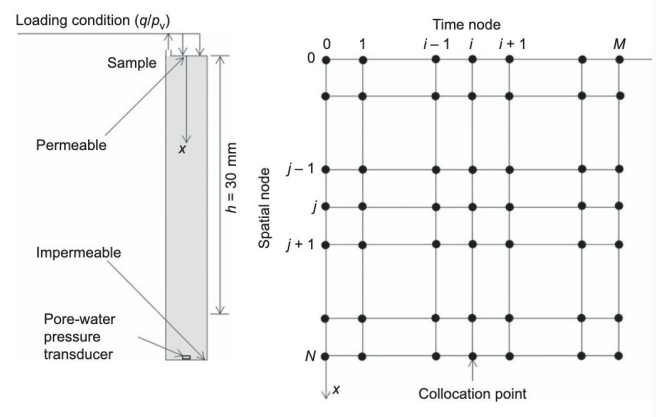
Data Loss: $\mathcal{L}_{\text{data}}(\theta) = \frac{1}{N_{\text{data}}} \sum_{i=1}^{N_{\text{data}}} |\hat{u}_\theta(z_i, t_i) - u_i|^2$

Physics Loss: $\mathcal{L}_{\text{physics}}(\theta) = \frac{1}{N_{\text{colloc}}} \sum_{j=1}^{N_{\text{colloc}}} |\mathcal{R}_\theta(z_j, t_j)|^2$

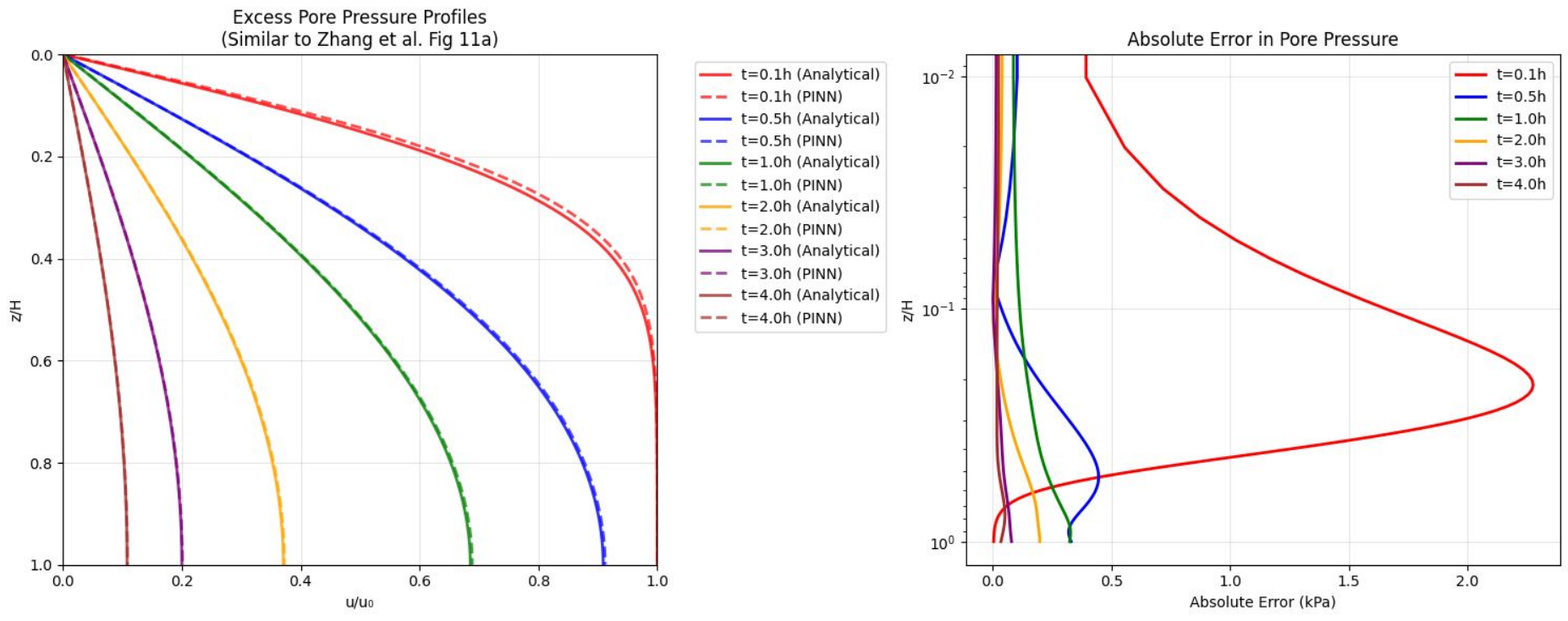
Boundary Conditions:

- **Top drainage**: $\mathcal{L}_{\text{BC}}(\theta) = \frac{1}{N_{\text{BC}}} \sum_k |\hat{u}_\theta(0, t_k)|^2$
- **Bottom no-flow**: $\mathcal{L}_{\text{BC}}(\theta) = \frac{1}{N_{\text{BC}}} \sum_k \left|\frac{\partial \hat{u}_\theta}{\partial z}(H, t_k)\right|^2$

Initial Condition: $\mathcal{L}_{\text{IC}}(\theta) = \frac{1}{N_{\text{IC}}} \sum_l |\hat{u}_\theta(z_l, 0) - u_0|^2$



PINNs 1D Consolidation: Result



Chishiki-AI

Chishiki-AI

About Fellowships Courses News

https://www.chishiki-ai.org/

Chishiki-AI

AI-powered Civil Engineering Community

NSF has awarded a $7 million grant to Chishiki.ai, an innovative project spearheaded by the University of Texas at Austin in collaboration with Cornell University. Chishiki.ai is a groundbreaking initiative designed to revolutionize the field of Civil and Environmental Engineering (CEE) through the integration of artificial intelligence (AI).