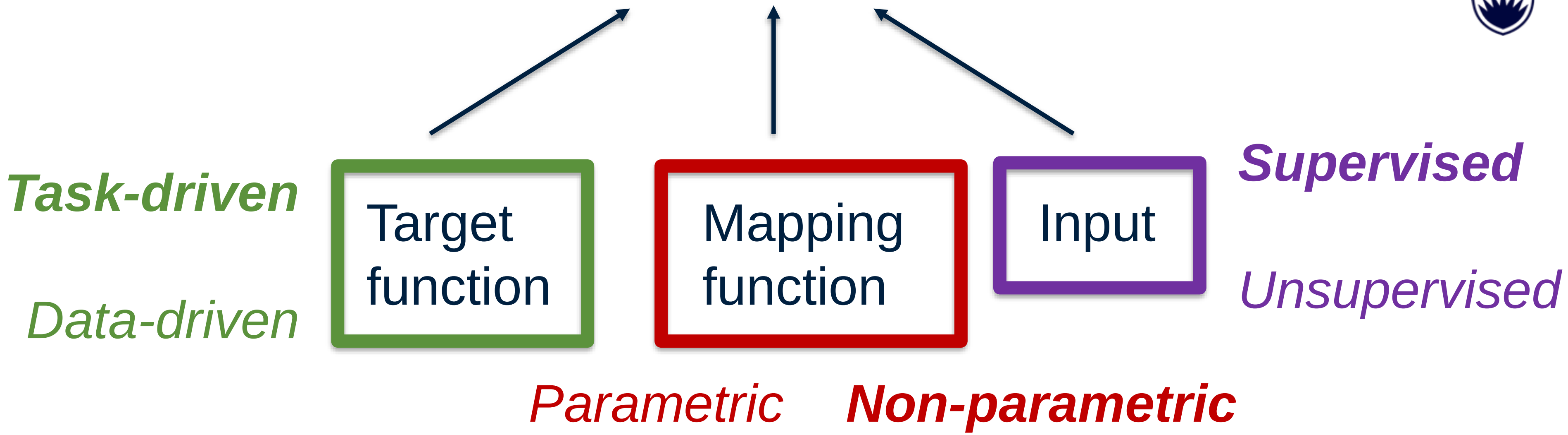
An introduction to machine learning models for gas sensor calibration

Prof. Naomi Zimmerman
University of British Columbia, Vancouver, Canada

US EPA Air Sensor QA Workshop: Gas Sensor Session
July 27, 2023

What is machine learning (ML)? Some definitions

$$\boldsymbol{y} = \boldsymbol{f}(\boldsymbol{x})$$



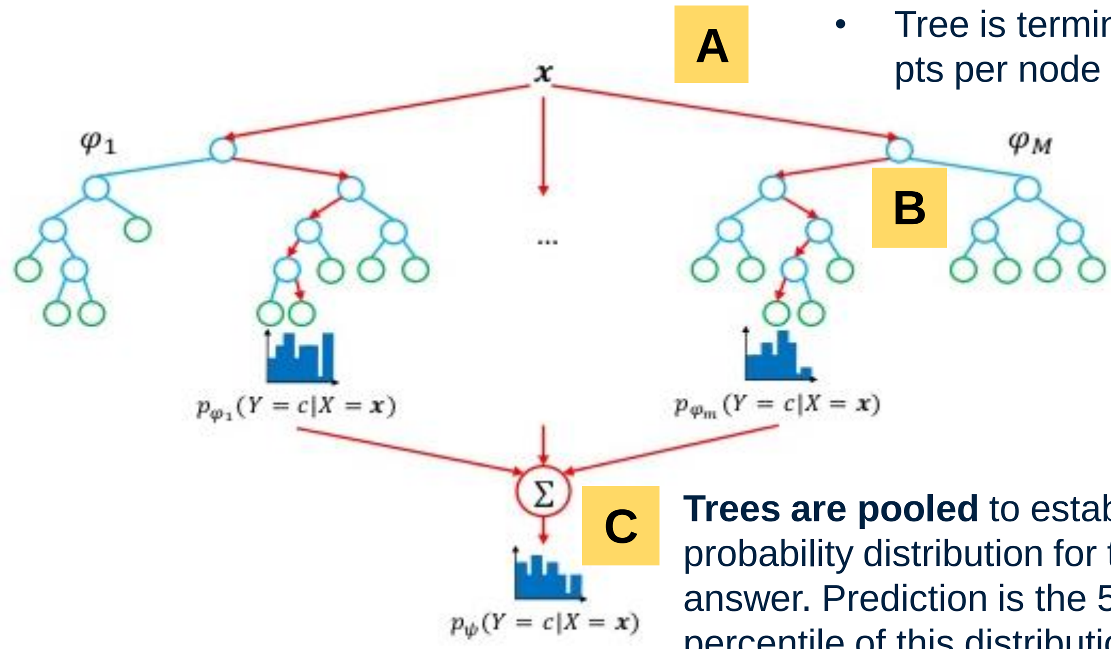
Why are these appealing tools for gas sensor calibration?

- Sensor operating principle: electrochemical or metal oxide most popular
- Both types can have varying *selectivity* – i.e., they might respond to more pollutants than just the target analyte
- Chemical reactions also influenced by environmental parameters like temperature and relative humidity
- Closed form models of these various effects difficult to capture – effects can be non-linear. Good candidate for machine learning models

Before you start building a machine learning model…

- **Sampling:** Aim to collect high quality data that represents the target sampling domain as closely as possible (in pollutant space and meteorological parameters such as T, RH)
- **Pre-process and clean your data:** do some QA/QC, remove outliers, determine how you will handle missing values etc. Many algorithms also recommend scaling your data on a 0-1 scale.
    - *Missing values*: for small gaps, you can use tools like a Kalman filter to impute missing values if you want to investigate time-trends. You might also omit whole periods if a threshold of data is missing (e.g., >25%)
- **External validation:** Set aside some data for model validation that is completely external to model building – this is a pure testing data set. Many folks use an 80:20 split

Popular models part 1: random forest



Bootstrap data into sub-samples

- # trees (M) user specified
- Tree is terminated either by min data pts per node or max tree depth

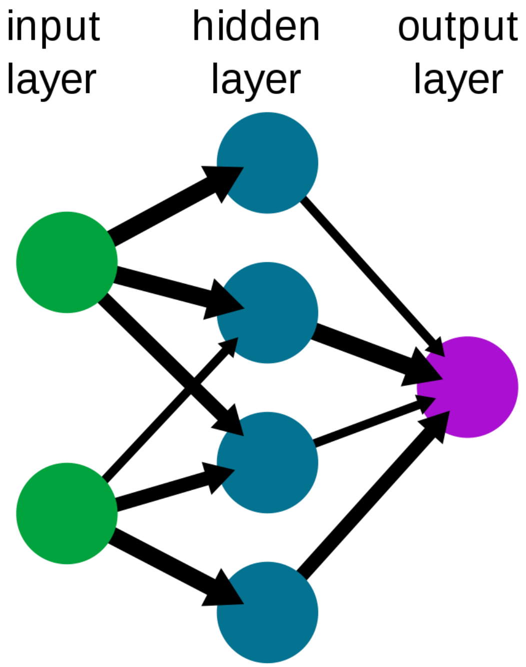
Split data by variable which best predicts the response

- Only a random fraction of variables are considered at each split
- The ideal number of variables to randomly consider can be tuned (m_{try})

Trees are pooled to establish a probability distribution for the answer. Prediction is the 50th percentile of this distribution

Image source: https://www.kdnuggets.com/2017/10/random-forests-explained.html

Popular models part 2: neural networks



- Slightly more complicated to explain
- Between each node in each layer, a *weight* and *bias* factor are fit to a user-specified **activation function**

$$f(x) = \left(\frac{e^x}{e^x + 1}\right)\ OR\ \ f(x) = \max(0, x)\ OR\ f(x) = \log(1 + \exp(x))$$

- Weights and biases to fit x are estimated using backpropagation from training data, essentially by minimizing the residuals between the true data and the output from the neural network after initializing them with guesses

$$x = input\ \times weight + bias$$

- Initial guess for weights is normally randomly selected from normal distribution, initial guess for bias normally 0

https://commons.wikimedia.org/w/index.php?curid=5084582

Hyper-parameters and tuning your models

- A hyper-parameter is an external value that affects how the model “learns” or is built
- There are different approaches for scanning range of possible hyperparameters
- Choosing optimal hyperparameter could be based on a target function like root mean square error minimization, etc.
- Some hyperparameters include:

Random Forest	Neural Networks
# trees in your “forest”	# of hidden layers
m_{try}	# nodes per hidden layer
Tree complexity	Learning rate
Sampling scheme	Batch size
Splitting rule	Activation function

Tools to assess importance of predictors

- There are a few approaches to this, but I will share the one I think is most interpretable: **permutation importance**

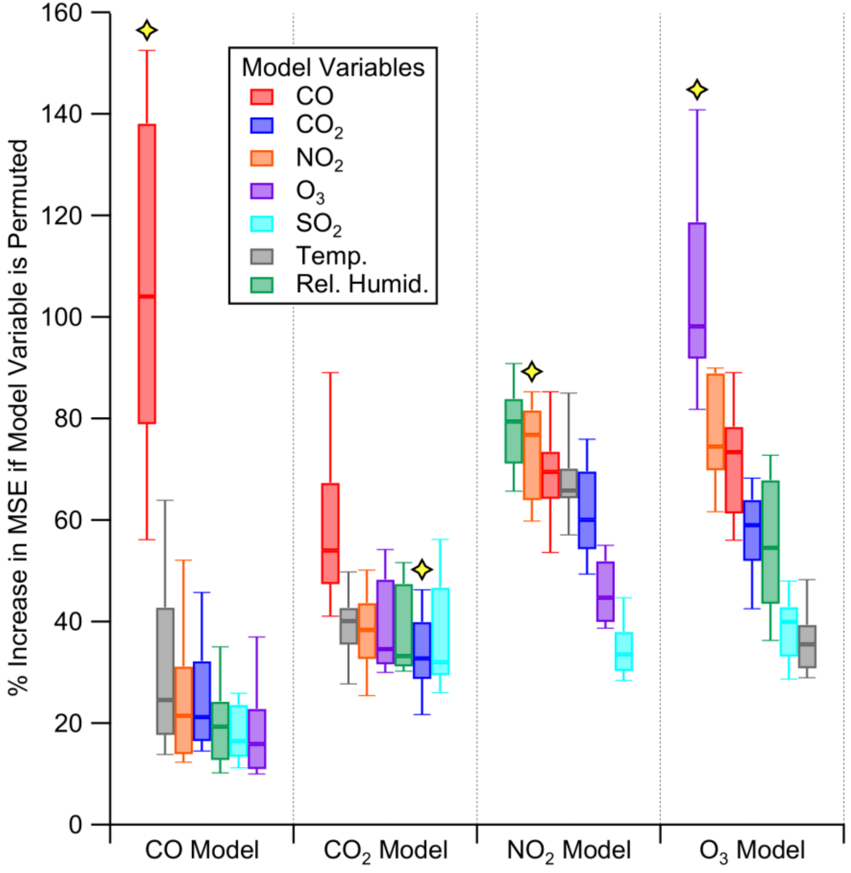
- After your models are built, you randomly shuffle (or *permutate*) your input variables one at a time

- If a variable is important, this process will result in less accurate predictions, since the resulting data no longer corresponds to anything observed in the real world

- The more important a variable, the more error we will see, since we are breaking a strong relationship that our model learned during training

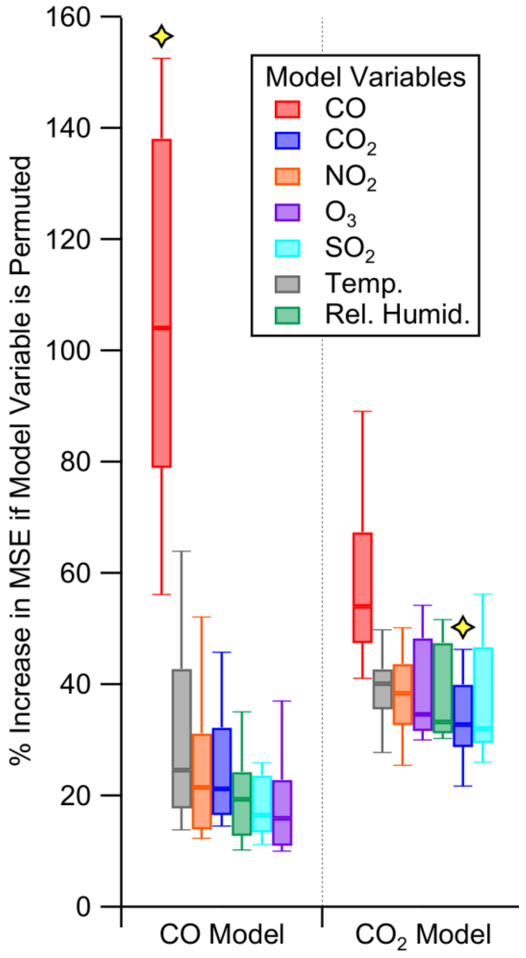
Example of permutation importance

- Consider the case where we have built four separate random forest calibration models for four gases: CO, CO_2, NO_2 and O_3
- Each model has 7 input variables: sensor-reported signals for CO, CO_2, NO_2, O_3, SO_2, T, RH
- In the CO calibration model, there is the most error if we permute the CO sensor signal – this suggests a relatively simple model: the CO sensor is an important input in calibrated CO measurements
- In the NO_2 model, RH is actually the most important variable, but not by a huge margin



Interpreting model performance

- There are standard accuracy and precision metrics for assessing model performance – the US EPA has released performance targets for O_3 and are a good place to start for any gas sensor model
- More holistically, if you are building a ML model, consider that they are functions of their environment – they will likely not transfer well if the deployment conditions are different from the calibration conditions
- Let's return to the permutation error plot for CO_2 as an example – this calibration was done near vehicle emissions – thus it leverages the more accurate vehicle emission sensor signals to correct CO_2 – this model would likely not work very well indoors!



Best practices: highlights

ML models can be powerful tools to help with gas sensor calibration – they are especially appealing due to the operating principles of many popular gas sensors, just keep in mind:

- **Evaluate your model:** hyperparameters, interpreting variable importance etc. can all tell you something about the underlying nature of the calibration model

- **Space is the place:** keep in mind that your calibration models will likely not work well in new spaces/places – especially for pollutants with more complex models (e.g., the CO vs. CO_2 example). *You might risk under-estimating a pollutant or mis-representing risk if applied to the wrong context. Use common sense – do the numbers make sense?*

- **Routinely reassess:** A calibration model isn't static – sensors degrade or conditions change. Consider tracking this and rebuilding when performance dips below targets

Resources

Websites

- **US EPA Air Sensor Performance Targets and Testing Protocols:** **https://www.epa.gov/air-sensor-toolbox/air-sensor-performance-targets-and-testing-protocols**

- **StatQuest: https://statquest.org/**
  (their YouTube videos are some of the easiest to understand that I have encountered)

Textbooks:

- **Hands-On Machine Learning with R,** *Bradley Boehmke & Brandon Greenwell*

- **R for Data Science,** *Hadley Wickham & Garrett Grolemund*

Thank you!

Questions for later? Email me anytime at

nzimmerman@mech.ubc.ca

This research was undertaken, in part, thanks to funding from the Canada Research Chairs program.