Predicting Temperature of Nuclear Waste Canisters

Stefan Mignerey
Giona Ghezzi

Introduction

In this project, we predict the transient temperatures around a nuclear waste canister. We used an XGboost model after testing a neural network and a K-nearest neighbors model. It achieved a score of 7.1 on the public leaderboard on **Kaggle.com**.

Data preprocessing

Clipping outliers: Pressures below -2'000 and temperatures above 150°C were cut.
Faulty/deleted sensors imputation: KNN (k = 5) and linear interpolation on partially missing data of a sensor. This acted as data augmentation to regularize.

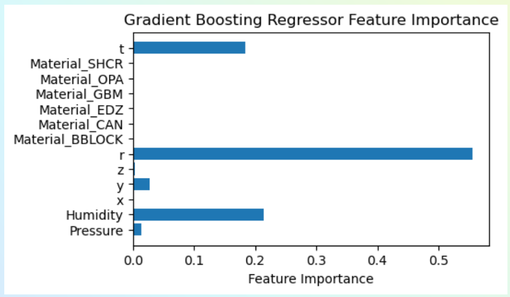
Feature representation

To create a single matrix, we first created the dictionaries (one array for each timestamp) and then we stacked them vertically, the data was then shuffled and normalized (z-score) for better performance on the neural network. We used all features provided.

	Pressure	Humidity	X	Y	Z	r	Materials ...	t
Sensors								

Features importance

By implementing an XGboost algorithm we could easily see which features were important. It appeared that the material played no role in the regression and probably affected badly the neural network that didn't have L1 regularization.



Model

After unsatisfactory results on a neural network, with a [14, 30, 25, 32, 1] architecture and ReLU activations preceded by layer normalization, we chose to use an XGBoost model. With a randomized search we came to the following hyper-parameters:

- Portion of the data for base learners: 80%
- Number of boosting stages: 500
- Max depth of each regressor: 6
- Learning_rate: 0.1

The learning stopped when the test loss did not improved over the last iteration.

Model explanation

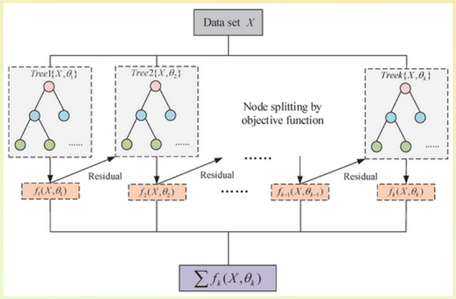


Figure 1: Diagram of XGboost

XGboost works by sequentially adding trees. It initiate a weak learner and makes a prediction, then gives the error as input to a new weak learner to learn the error, by summing the 2 results, we get closer to an accurate prediction. We can iterate with new weak learners and minimize the error.

This model is highly customizable, quite robust to over-fitting, selects important features automatically and is more robust to imbalanced data compared to other models like neural networks.

Results

All the results were obtained on a randomized split of the dataset: 87.5 % for training and 12.5 % for testing, with an 8-fold cross-validation for the neural network. The metric used is the mean squared error.

Model	Training	Validation	Kaggle
KNN (k=7)	None	None	81.5
Neural Net	0.078	0.080	70.1
XGboost	0.227	0.408	7.6

Discussion

Globally the project was very interesting, creating a machine learning model from 'scratch' really taught us what we would face in our later career. Unfortunately the neural net was not able to provide satisfactory scores on Kaggle and fulfill our expectations, even if it was able to perform well on the training and test sets. The Kaggle score was our main objective and had no correlation with our computed loss, which was frustrating and made us rely on Kaggle more than on the loss.
Nevertheless, we explored multiple theories like biased data caused by the imputation, bad clipping or overfitting, but couldn't impove the score. We concluded the unbalanced data could be the issue and explaining why the XGboost model, more robust to it, performed 10 times better on the same data.

Future

We should start with multiple models from the start, automatical tune the hyperparameters, and continue with the best one, automating the pipeline would have allowed us to identify the best model sooner.

More about our models on:

GitHub

Figure 1: Guo, Rui et al. (2020). Degradation State Recognition of Piston Pump Based on ICEEMDAN and XGBoost. Applied Sciences. 10. 6593. 10.3390/app10186593.