EAAE Masterclass: Machine Learning and Econometrics

17 April 2025

Dr. ir. C. (Koos) Gardebroek
Wageningen University, The Netherlands
koos.gardebroek@wur.nl

Outline

- What is Machine Learning (ML) and why study it?
- Purposes of Machine Learning
- Machine Learning in practice
- Hands on case: predicting soybean prices using several ML approaches.

What is Machine Learning?

Machine Learning (or statistical learning; *ML*): set of tools or algorithms for understanding data.

Novelty of machine learning is that it often follows a purely data-driven approach. Classical econometrics often more procedural:

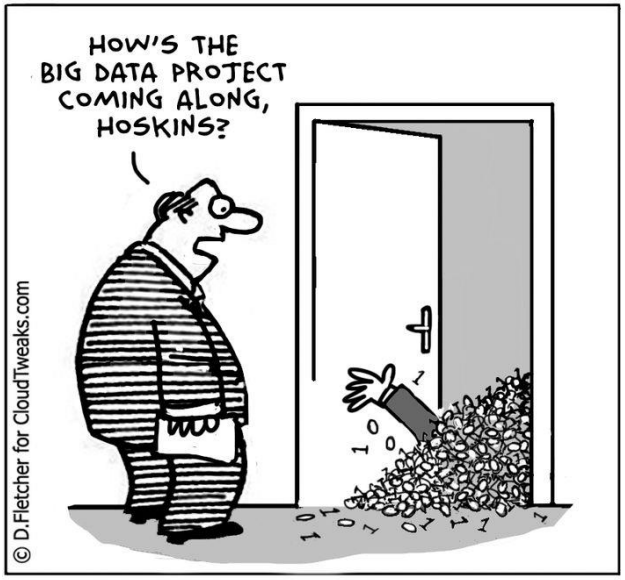
- Start from theory in selecting variables
- Choose a specific functional form a priori
- Choose a specific statistical technique (e.g. OLS) a priori

Contributions of novel ML literature are:

1. Many new tools (e.g. trees & forests, neural networks)
2. New ideas on how to analyse data.
3. Methods for analysing big data in automated way, even if there are more variables than observations.

Why study Machine Learning?

1. Offers new ways of analysing data.
2. In other research fields already used a lot, so interdisciplinary cooperation and communication may require knowledge.
3. More and more data becoming available, e.g. internet data, big data, climate data, text data. May require new tools.



Purposes of Machine Learning

Main purposes of ML are *prediction* and *classification*.

Inference is about understanding the relation between input and output variables. E.g. (James et al., 2023: 19):

- Which predictors are associated with the response?
- What is the relationship between the response and each predictor?
- Is the relationship linear or more complicated?

In regression analysis these questions are usually answered using hypothesis testing (e.g. t-test, F-test, LR-test). In ML this is (almost) absent, although other approaches are getting popular (e.g. Shapley values).

Often *trade-off* between *prediction accuracy* and *model interpretability* (and possibility of inference).

Purposes of Machine Learning

FIGURE 2.7. *A representation of the tradeoff between flexibility and interpretability, using different statistical learning methods. In general, as the flexibility of a method increases, its interpretability decreases.*

Source: James et al., 2023: 24.

So, why can't we do inference in ML?

- Standard errors often not given, or even impossible
- Certain ML methods select variables from large pool:
  - May randomly differ in different runs, many coefficients set to zero
  - Strong correlation would lead to large standard errors in regression due to multicollinearity, ML (e.g. LASSO, trees) just drops them.
  - Selecting some but not others → missing variable bias
- Regularization (setting key parameters, e.g. # of trees and depth) adds subjectivity.

In practice, superior prediction of ML is appreciated, at the cost of not being able to interpret and test the coefficients.

Not this LASSO…

So, why can't we do inference in ML?

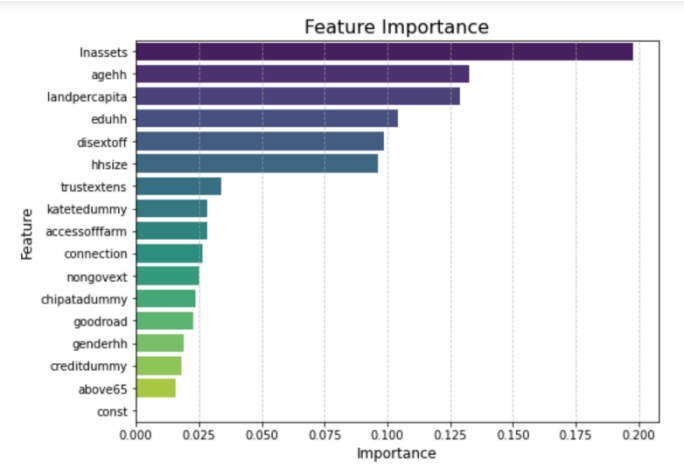
Efron (2020): Success of ML due to fact that prediction is easier than estimation or attribution (inference).

- Some ML methods combine individually weak learners (*ensembles*)
- Attribution demands much more of individual variables.

Efron (2020): so-called '*importance scores*' for predictors, e.g., in random forest, cannot be used to claim that certain variables are key due to *strong correlations* among the many variables

→ strong correlation does not matter for prediction

→ strong correlation makes attribution/inference very difficult



Machine Learning in practice

ML does not imply automatic implementation of the best model and training method. Still many choices:

1a. Choice of ML method
1b. Setting key parameters for method (e.g. # of trees; # of variables; shrinkage parameter)

2a. Encoding and transformation of variables (per unit; ln or linear; first-differences; normalised)
2b. Theory may be useful in many modeling decisions

3. Choices on how to evaluate quality of the model, e.g., MSE, RMSE, MAE, MAPE

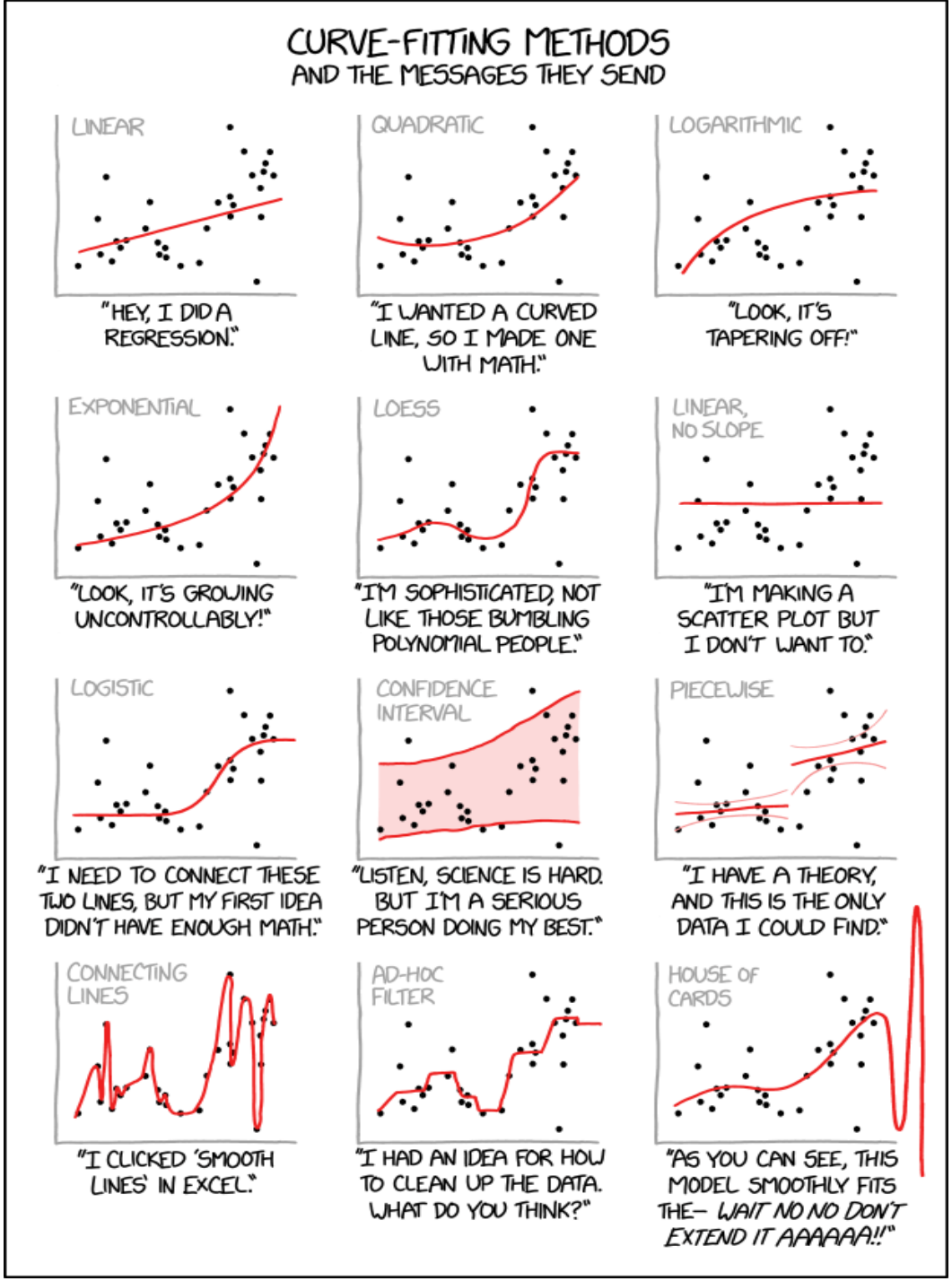
Machine Learning in practice

In econometrics models often estimated on all data available, and then used for inference or (out-of-sample) prediction.

Due to the prediction focus, ML typically has 3 phases: *training*, *validation*, and *testing*.

Available data is partly used to train (estimate) the model, validated on a (randomly) held out subsample of the training data, and finally used for testing (prediction in test sample).

A danger of more flexible ML approaches is *overfitting*: the model follows the data, including the errors, too closely → poor predictions in *test* stage. In other words, the improved training fit goes at the cost of testing (or prediction) fit.


CURVE-FITTING METHODS
AND THE MESSAGES THEY SEND
LINEAR
"HEY, I DID A REGRESSION."
QUADRATIC
"I WANTED A CURVED LINE, SO I MADE ONE WITH MATH."
LOGARITHMIC
"LOOK, IT'S TAPERING OFF!"
EXPONENTIAL
"LOOK, IT'S GROWING UNCONTROLLABLY!"
LOESS
"I'M SOPHISTICATED, NOT LIKE THOSE BUMBLING POLYNOMIAL PEOPLE."
LINEAR, NO SLOPE
"I'M MAKING A SCATTER PLOT BUT I DON'T WANT TO."
LOGISTIC
"I NEED TO CONNECT THESE TWO LINES, BUT MY FIRST IDEA DIDN'T HAVE ENOUGH MATH."
CONFIDENCE INTERVAL
"LISTEN, SCIENCE IS HARD. BUT I'M A SERIOUS PERSON DOING MY BEST."
PIECEWISE
"I HAVE A THEORY, AND THIS IS THE ONLY DATA I COULD FIND."
CONNECTING LINES
"I CLICKED 'SMOOTH LINES' IN EXCEL."
AD-HOC FILTER
"I HAD AN IDEA FOR HOW TO CLEAN UP THE DATA. WHAT DO YOU THINK?"
HOUSE OF CARDS
"AS YOU CAN SEE, THIS MODEL SMOOTHLY FITS THE– WAIT NO NO DON'T EXTEND IT AAAAAA!!"

Machine Learning in practice

Table 1

Performance of Different Algorithms in Predicting House Values

Method	*Prediction performance (R^2)*		*Relative improvement over ordinary least squares by quintile of house value*				
	Training sample	*Hold-out sample*	1st	2nd	3rd	4th	5th
Ordinary least squares	47.3%	41.7% [39.7%, 43.7%]	–	–	–	–	–
Regression tree tuned by depth	39.6%	34.5% [32.6%, 36.5%]	−11.5%	10.8%	6.4%	−14.6%	−31.8%
LASSO	46.0%	43.3% [41.5%, 45.2%]	1.3%	11.9%	13.1%	10.1%	−1.9%
Random forest	85.1%	45.5% [43.6%, 47.5%]	3.5%	23.6%	27.0%	17.8%	−0.5%
Ensemble	80.4%	45.9% [44.0%, 47.9%]	4.5%	16.0%	17.9%	14.2%	7.6%

Note: The dependent variable is the log-dollar house value of owner-occupied units in the 2011 American Housing Survey from 150 covariates including unit characteristics and quality measures. All algorithms are fitted on the same, randomly drawn training sample of 10,000 units and evaluated on the 41,808 remaining held-out units. The numbers in brackets in the hold-out sample column are 95 percent bootstrap confidence intervals for hold-out prediction performance, and represent measurement variation for a fixed prediction function. For this illustration, we do not use sampling weights. Details are provided in the online Appendix at http://e-jep.org.

Source: Mullainathan and Spiess (2017)

Machine Learning in practice

If no extensive test set available, *cross-validation* can be used: hold out a subset of training observations and use as validation set which fit proxies for test fit. *Validation set ≠ test set!*

Most popular is *K-fold cross-validation*: Randomly divide training sample in *k* folds or groups. Train the model *k* times, every time leaving out one of the folds and calculating measure of fit for the hold-out fold. Final measure is average over the *k* values, e.g., $MSE_{CV} = \frac{1}{k}\sum_{i=1}^{k} MSE_k$ → cross-validation error

Usually k set to 5 or 10.

Machine Learning in practice

Cross-validation useful for:

- Evaluating model performance (model assessment)
- Comparing performance of various models (model selection)
- Avoiding overfitting
- Getting better estimate of prediction error.

Time-series and *panel data* have a clear time ordering of data. So, randomly holding out obs. in cross-validation not ideal:

- Lagged values included in models cannot be taken.
- Obs. strongly correlate over time, so randomly held out obs. easy to predict → flawed validation fit measures.

Idea of *k*-fold cross-validation can be adapted for time-series and panel data, maintaining the time order of observations, by using *k* overlapping blocks of training and validation data.

Validation with time series data

Figure 2-3. *Time series cross-validation in Python*
Source: Korstanje, J. (2023)

Drawback: Errors of shorter periods may be biased.

Validation with time series data

Figure 2-4. *Rolling time series cross-validation in Python*

Source: Korstanje, J. (2023)

Problem here could be that folds 1 and 2 are too far from test set.

Validation with panel data

Package PanelSplit (Frey and Simon, 2024) for splitting panel data:

	Train set					Test set				
	panel_data.loc[splits[0][0]]					panel_data.loc[splits[0][1]]				
Split 0	country_id	year	y	x1	x2	country_id	year	y	x1	x2
0	1	2001	1.62	-0.32	0.90					
4	2	2001	0.87	-0.17	-0.27					
8	3	2001	0.32	-1.10	-0.69					
1						1	2002	-0.61	-0.38	-0.68
5						2	2002	-2.30	-0.88	0.53
9						3	2002	-0.25	1.14	-0.85

	panel_data.loc[splits[1][0]]					panel_data.loc[splits[1][1]]				
Split 1	country_id	year	y	x1	x2	country_id	year	y	x1	x2
0	1	2001	1.62	-0.32	0.90					
1	1	2002	-0.61	-0.38	-0.68					
4	2	2001	0.87	-0.17	-0.27					
5	2	2002	-2.30	-0.88	0.53					
8	3	2001	0.32	-1.10	-0.69					
9	3	2002	-0.25	1.14	-0.85					
2						1	2003	-0.53	1.13	-0.12
6						2	2003	1.74	0.04	-0.69
10						3	2003	1.46	0.90	-0.67

	panel_data.loc[splits[2][0]]					panel_data.loc[splits[2][1]]				
Split 2	country_id	year	y	x1	x2	country_id	year	y	x1	x2
0	1	2001	1.62	-0.32	0.90					
1	1	2002	-0.61	-0.38	-0.68					
2	1	2003	-0.53	1.13	-0.12					
4	2	2001	0.87	-0.17	-0.27					
5	2	2002	-2.30	-0.88	0.53					
6	2	2003	1.74	0.04	-0.69					
8	3	2001	0.32	-1.10	-0.69					
9	3	2002	-0.25	1.14	-0.85					
10	3	2003	1.46	0.90	-0.67					
3						1	2004	-1.07	-1.10	-0.94
7						2	2004	-0.76	0.58	-0.40
11						3	2004	-2.06	0.50	-0.01

Applications of ML for agricultural economists

If prediction is core business of ML, what are potential uses?

- Prediction in markets and macro-variables
- Prediction of food security at micro-level (Zhou et al., 2022)
- Selection of relevant predictors in big data and new types of data, e.g. satellite and text data.
- Prediction in causal inference: IV regression, counterfactuals, propensity scores.
- Testing theories (are predictions in line with theory?)

Mullainathan and Spiess (2017): New methods may allow exploring new problems and answering new questions.

Advantages of ML

Coulombe et al. (2022) investigated *how* ML can be beneficial for macro-economic forecasting. Main conclusions are that ML is superior in picking up *non-linearities* in the data, and that k-fold cross-validation is the best approach to validate models.

Non-linearities arise in times of economic uncertainty.

Brignoli et al (2024a) compare classical time-series methods (ARIMA, VAR, VECM) with Neural Networks in forecasting various commodity prices. NNs perform well in *longer forecasting horizons* and in cases of *breaks* (volatile periods).

Advantages of ML

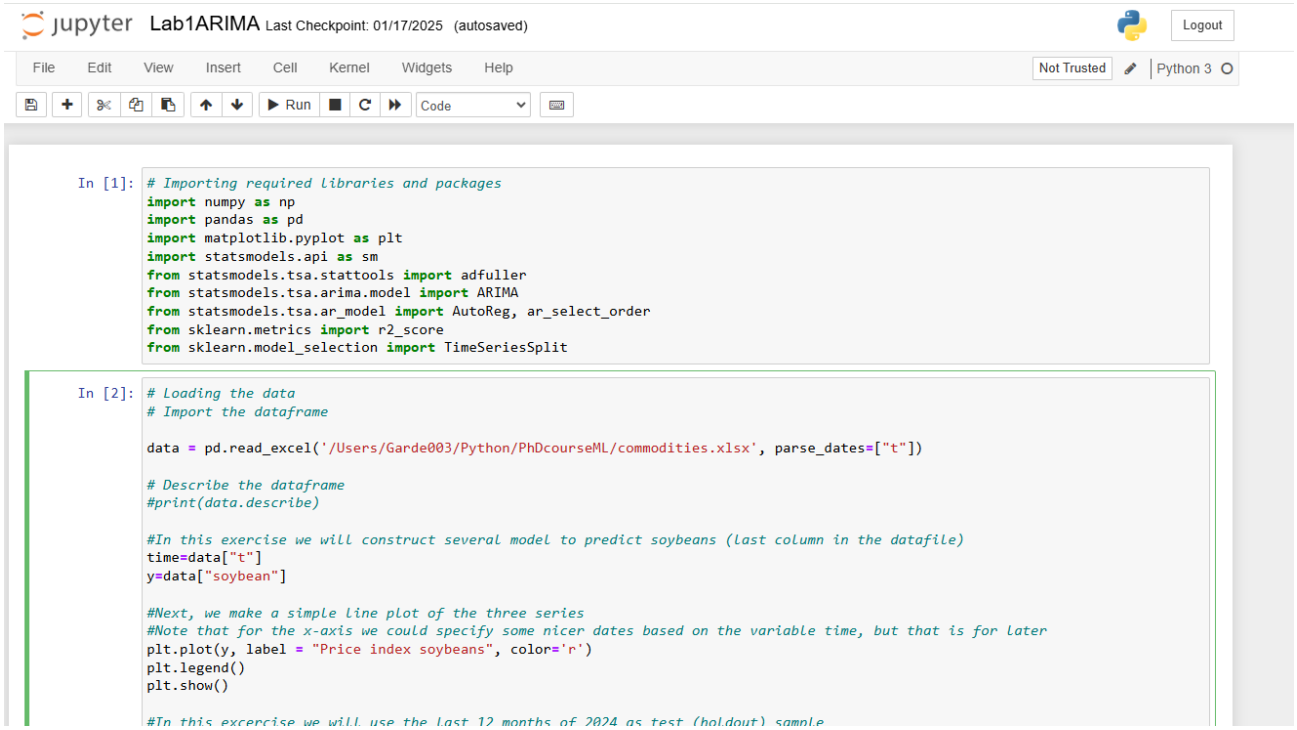
Brignoli et al. (2024b) investigate how ML perform in estimating treatment effects in a simulation experiment based on EU FADN panel data. Focus on selection of *functional form*, *non-linearities*, *controls*, *treatment effect heterogeneity*, and *latent confounding control*.

Authors conclude: ML methods outperform classical methods, particularly tree- and forest-based methods.

See Storm et al. (2020) for an elaborate overview of methods and applications.

Hands-on case: Predicting Soybean prices

- Data: 22 commodity prices Jan. 2003-Dec. 2024 (264 obs.)
- Goal: create models to predict soybean prices
- 12 months in 2024 are test sample; other training sample
- Various ML techniques explained and compared, also to standard time-series techniques.
- Analysis done in Python (Jupyter notebooks)



Hands-on case: Predicting Soybean prices

Classical time-series econometrics

- Select relevant variables, e.g., based on theory or interest
- Test for (non-)stationarity, e.g., ADF , PP or KPSS test
- Decide on # of lags, trends and constants
- Test for cointegration: long-run equilibria among series
- Use model for prediction (or IRF analysis for VAR/VECM)

First, some standard tests were done in Python:

- All 22 series non-stationary; first-differences all stationary
- Bivariate Engle-Granger cointegration tests: soybeans cointegrated with 6 other commodities: palm oil, canola oil, maize, wheat, potassium, soy oil.

So, how does Machine Learning deal with all this?

Hands-on case: Predicting Soybean prices

Variable Selection Techniques (shrinkage)

- Ridge Regression
- Least Absolute Shrinkage and Selection Operator (LASSO)
  - Regular/Adaptive/Group LASSO
- Elastic Net (combo of Ridge Reg. and LASSO)
- Smoothly Clipped Absolute Deviation (SCAD)

Used to select a subset of variables from a large set. Often combined with other ML methods.

How: Add a *penalty term* to a regression objective, shrink various coefficients towards zero (RR) or exactly zero (LASSO).

LASSO: $\min_{\beta} RSS = \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p}\left|\beta_j\right|$

Hands-on case: Predicting Soybean prices

Regular LASSO with TS cross-validation, grid search on optimal penalty term, and up to 2-period lagged diffs. (65 variables) [1]:

```
# Finally, we will create one-month and two-month lagged values of all prices and include these in a CVLASSO

# Creating extra columns with lagged values for variables
# Specify the columns to lag (all except the date column)
columns_to_lag = [col for col in X_traindiff.columns]

# Specify the number of lags
lags = [1, 2]  # Example: lagged by 1 and 2 periods

# Create lagged variables
for col in columns_to_lag:
    for lag in lags:
        X_traindiff[f"{col}_lag_{lag}"] = X_traindiff[col].shift(lag)

# Drop rows with missing values
X_traindiff = X_traindiff.dropna().reset_index(drop=True)

#Note we also need to drop the first 2 obs. in y_traindiff since otherwise dimension does not match with X_traindiff
y_traindiff=y_traindiff.iloc[2:]

# Use TimeSeriesSplit for cross-validation
tscv = TimeSeriesSplit(n_splits=5)

# Fit LassoCV with time series splits
lasso_cv = LassoCV(cv=tscv, random_state=42, max_iter=10000)
lasso_cv.fit(X_traindiff, y_traindiff)

# Get the optimal alpha (lambda)
optimal_lambda = lasso_cv.alpha_
print(f"Optimal Lambda (alpha): {optimal_lambda}")

#vif_data2["Feature"] = X_traindiff.columns

# Extract non-zero coefficients
nonzero_mask = lasso_cv.coef_ != 0
final_coefficients = pd.DataFrame({
    "Feature": X_traindiff.columns,
    "Coefficient": lasso_cv.coef_
}).loc[nonzero_mask]

print("\nFinal Non-Zero Coefficients:")
print(final_coefficients)
```

```
Optimal Lambda (alpha): 0.000558350182106155

Final Non-Zero Coefficients:
           Feature  Coefficient
0          crudoil     0.000616
5         coconoil     0.022167
9            maize     0.073036
13           sugar     0.009096
15             dap    -0.029706
16          potass    -0.004206
17          copper     0.039579
19          soyoil     0.302928
20         soymeal     0.356785
37  sunfloil_lag_1     0.033423
38  sunfloil_lag_2     0.019282
40     maize_lag_2     0.012473
47     sugar_lag_1    -0.029217
48     sugar_lag_2    -0.007088
51       dap_lag_1    -0.014190
52       dap_lag_2     0.001931
54    potass_lag_2     0.020663
55    copper_lag_1    -0.003254
62   soymeal_lag_2     0.056252
```

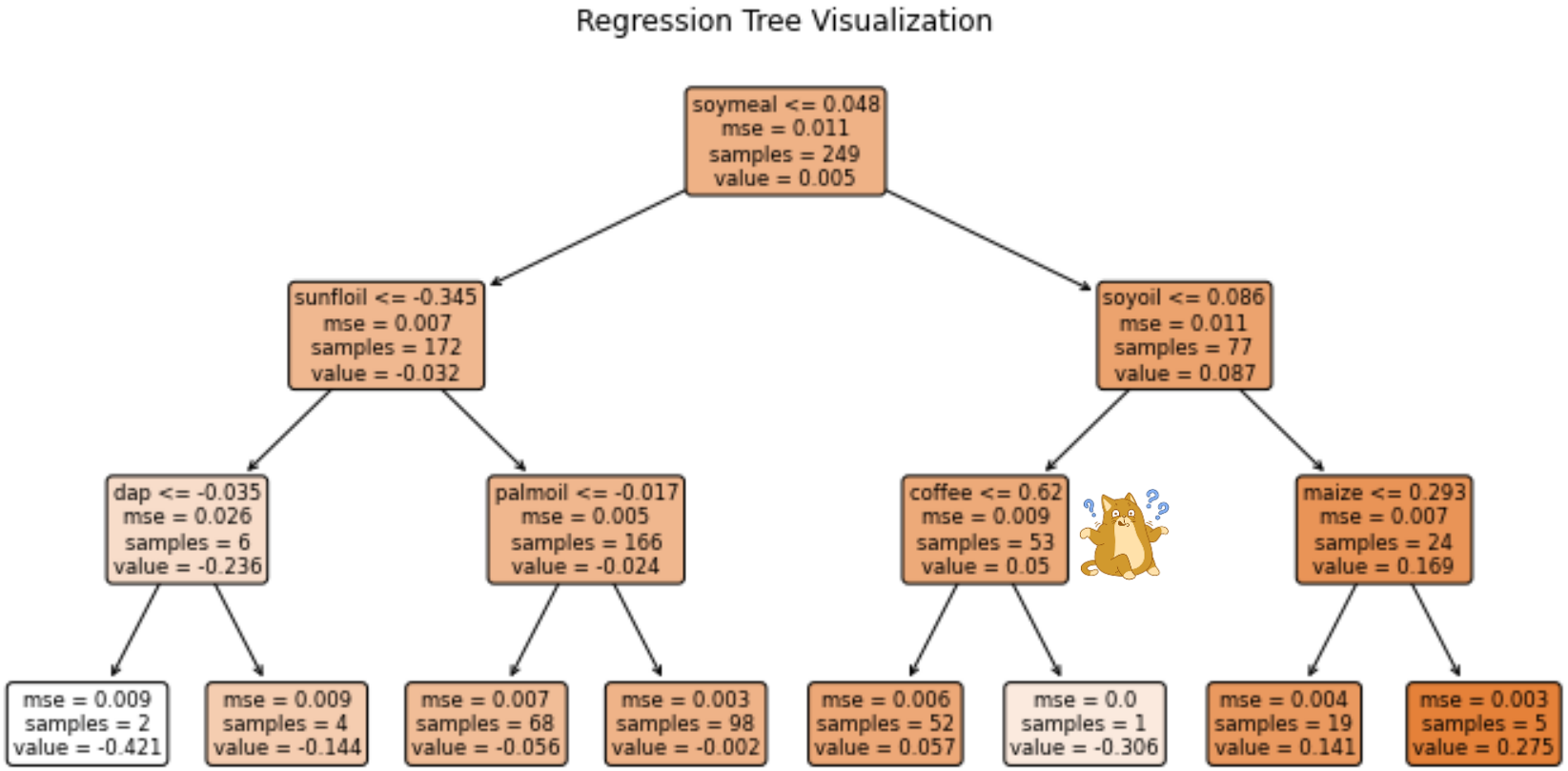
[1] Credits to ChatGPT for helping this noob with Python coding…

Hands-on case: Predicting Soybean prices

Regression trees and Ensembles of trees

- Simple regression trees
- Bootstrap aggregation (Bagging) of trees
- Random Forests
- Boosting
- Bayesian Additive Regression Trees (BART)

Hands-on case: Predicting Soybean prices



Predicted differences in 2024:

```
Mean Squared Error: 0.0142
[-0.00233248 -0.00233248 -0.00233248 -0.00233248  0.05694006 -0.00233248
 -0.00233248 -0.00233248 -0.00233248 -0.00233248 -0.00233248 -0.00233248]
```

Hands-on case: Predicting Soybean prices

Simple tree is inferior (weak learner)

- Many similar predictions.
- If data changes, tree will change.

Ensembles of trees combine trees using various 'tricks':

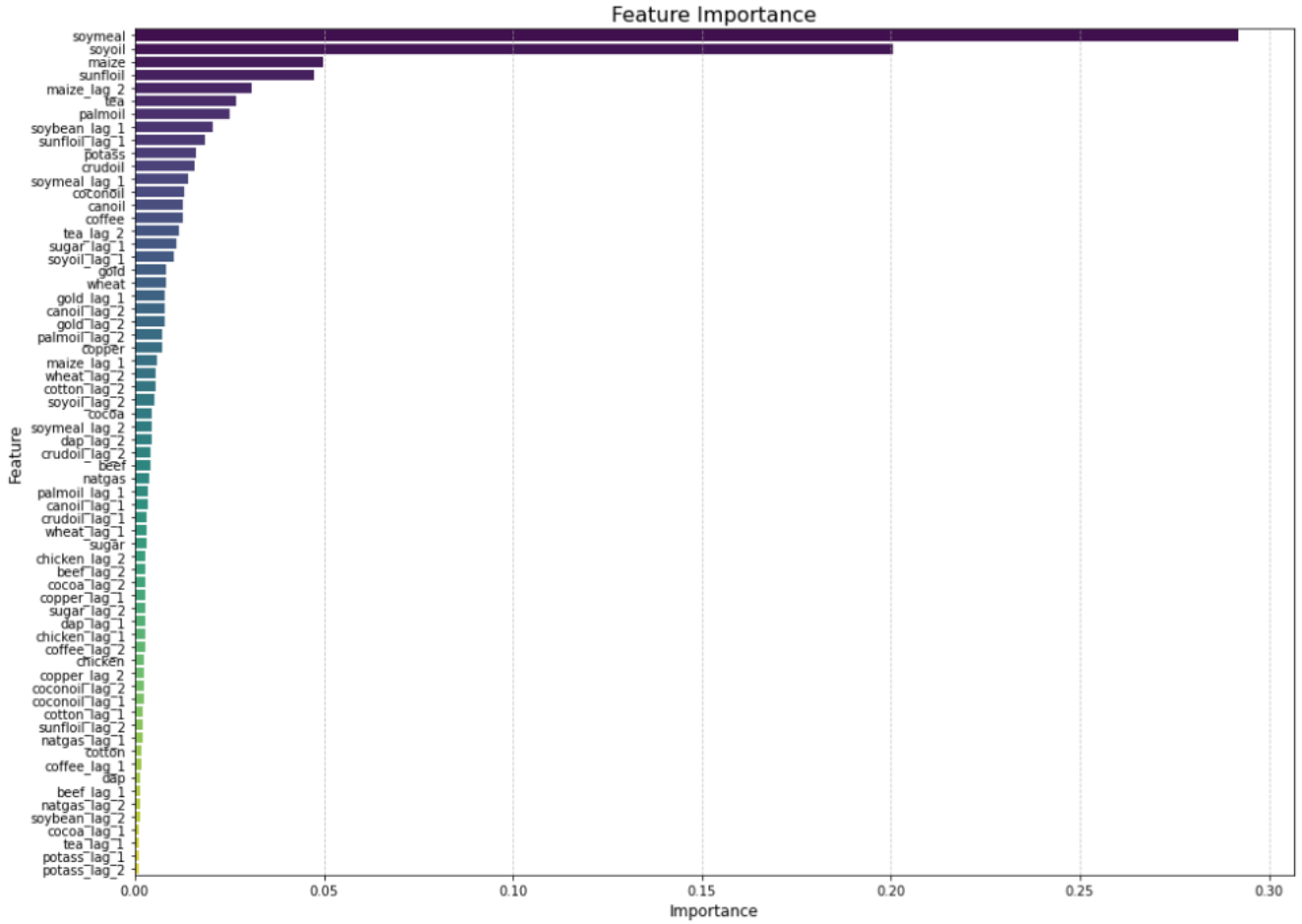
- Taking random subsets of training data (*Bagging*)
- Randomly select variables for different trees (*Random For.*)
- Updating earlier trees/Learning (*Boosting* and *BART*)

Gardebroek et al. (2024) used a mixed-effects Random Forest model to forecast acreages of biomaterials (hemp and flax) in several European countries.

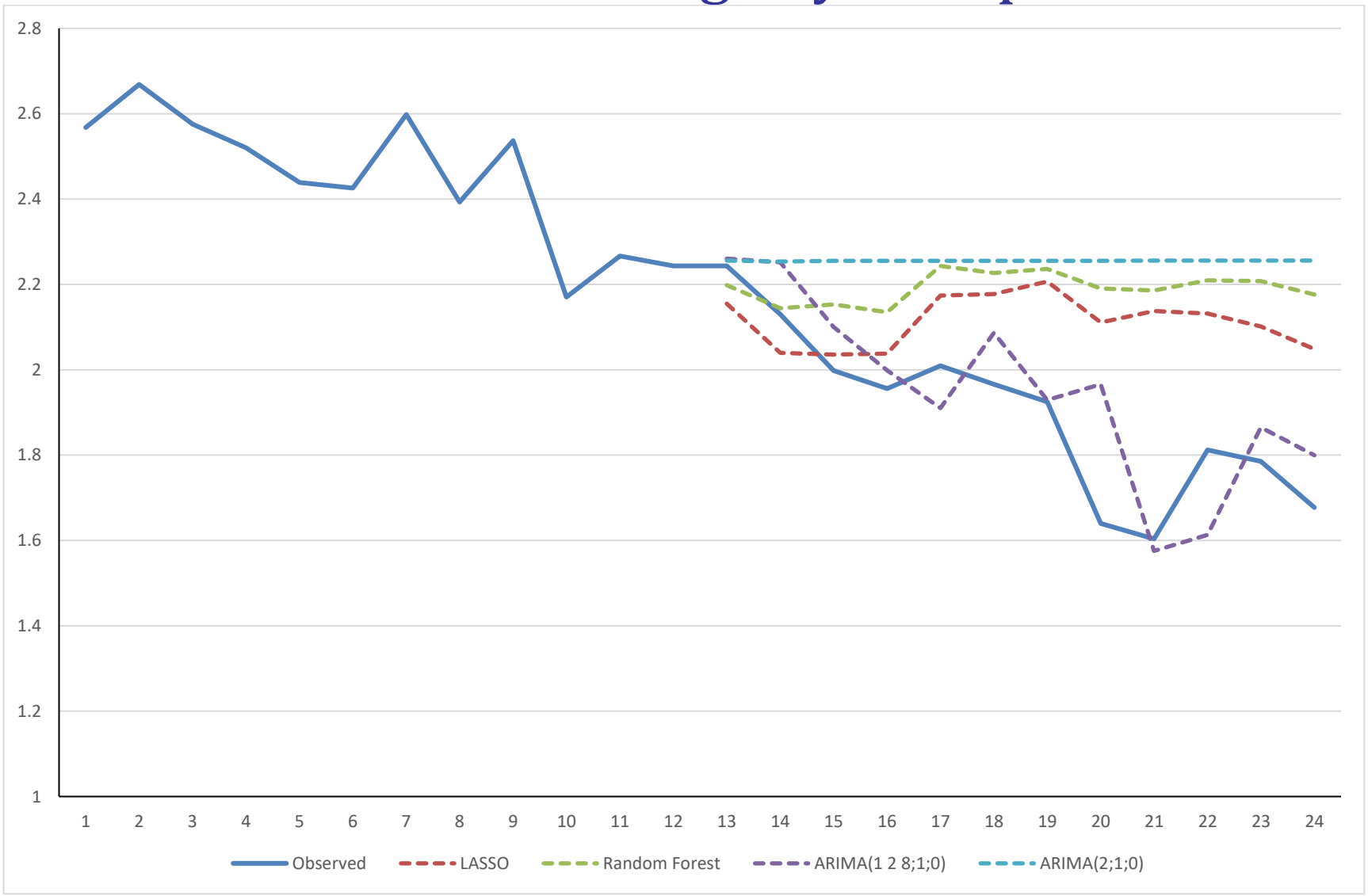
Hands-on case: Predicting Soybean prices

Prediction for Random Forest using the 65 variables and max. tree depth of 5:

```
Name: soybean, dtype: float64
[-0.04544962 -0.05394099  0.00887809 -0.01820988  0.10832755 -0.01647553
  0.00973835 -0.04592832 -0.0047426   0.02349475 -0.00143631 -0.03169992]
```



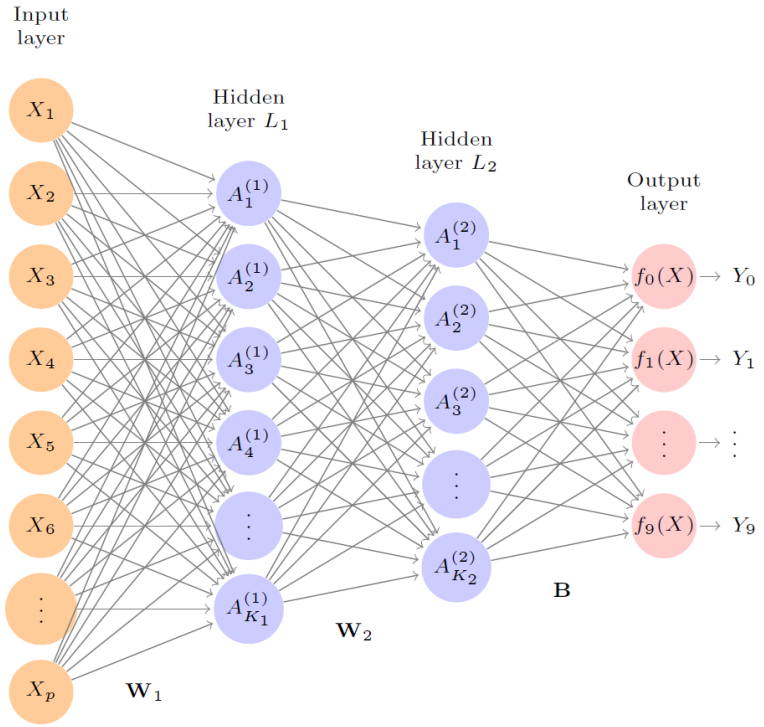
Hands-on case: Predicting Soybean prices



Hands-on case: Predicting Soybean prices

Neural Networks (Deep Learning)

- Simple feedforward Neural Network



p =784 variables
K_1 = 256
K_2 = 128
Output units: 10
Weights: 235146 (!)

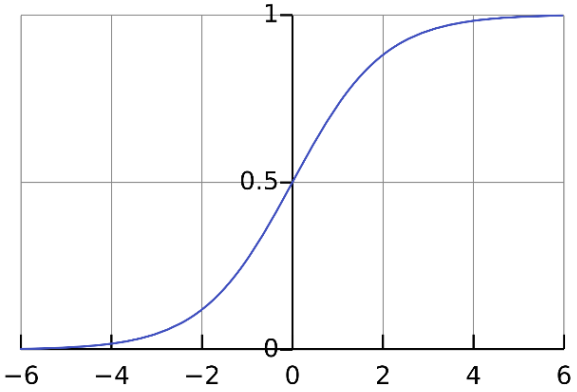
Source: James et al., 2023: 400.

Hands-on case: Predicting Soybean prices

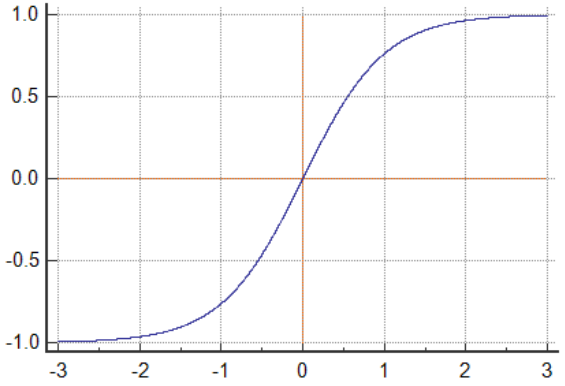
Inside the neurons are so-called *activation functions* with *weights*, transforming the data. Various options (e.g. linear, piecewise linear, softmax), but very popular are:

1. Sigmoid or logistic function: $h(X) = \frac{e^X}{1+e^X} = \frac{1}{1+e^{-X}}$

2. Hyperbolic tangent function: $h = \tanh(X) = \frac{e^X - e^{-X}}{e^X + e^{-X}}$



Logistic function



Hyperbolic tangent

Hands-on case: Predicting Soybean prices

A single layered Neural Network reminds of:

- Non-linear regression if output is continuous
- (Non-linear) binary choice if output is binary
- (Non-linear) multinomial choice if multiple integer outputs

However, the above models often have a *fixed functional form* (e.g. Logit), whereas NN is very *flexible*.

System nature of NN also reminds of VAR/VECM or simultaneous equations models. However, the latter are linear in parameters, whereas NN is highly *non-linear*.

Problem of simple feed-forward NNs: not suited for dealing with *past values/sequences*. We could manually create many lagged values, but that would lead to huge number of weights.

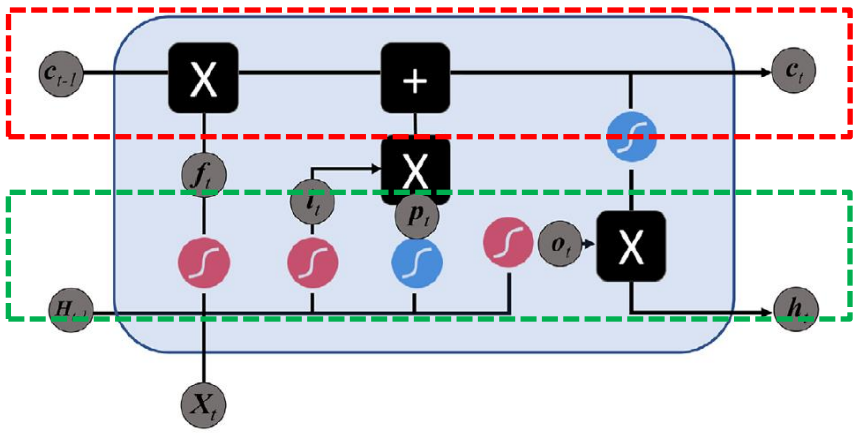
Hands-on case: Predicting Soybean prices

Recurrent Neural Networks (RNNs) consider sequential nature of inputs X_i (e.g. time order). Drawback: no long memory: long-term information decays quickly (vanishing gradient problem).

Long-short-term memory (LSTM) and *Gated Recurrent Unit (GRU)* add several functions to the hidden state, that allow for separating short-term and long-term learning.

This is done by combining different activation functions via different trajectories in the hidden states.

Hands-on case: Predicting Soybean prices



Red circle: logistic/sigmoid function; Blue circle: tanh function.

The Cell state captures the *long-term memory*. The incoming value c_{t-1} can be multiplied or something may be added.

But no weights or biases can modify it, so not affected by vanishing (or exploding) gradient problem.

Processing of the hidden state updates *short-term memory*.

Hands-on case: Predicting Soybean prices

Comparison of simple VAR(2) model forecasts for 2024 with:

- NN1: Simple Feed Forward NN without a hidden layer, 2 output layers with linear activation functions, and using 2-period values as inputs (mimics simple VAR2): only 10 parameters to train.

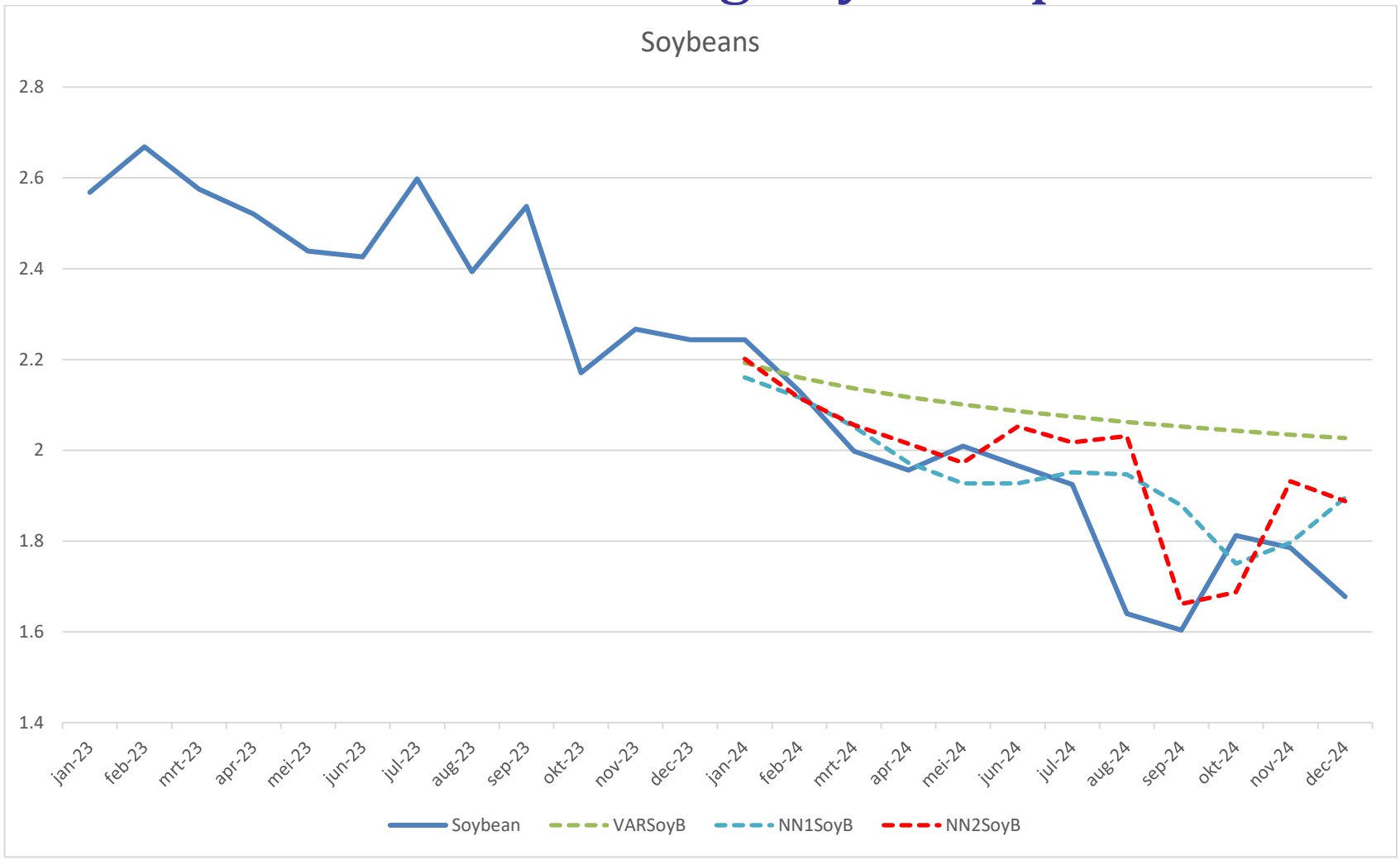
Part of *Tensorflow/Keras* code*:*

```
model1 = Sequential([
    Dense(2, activation='linear', input_shape=(4,)),  # Hidden layer
])
```

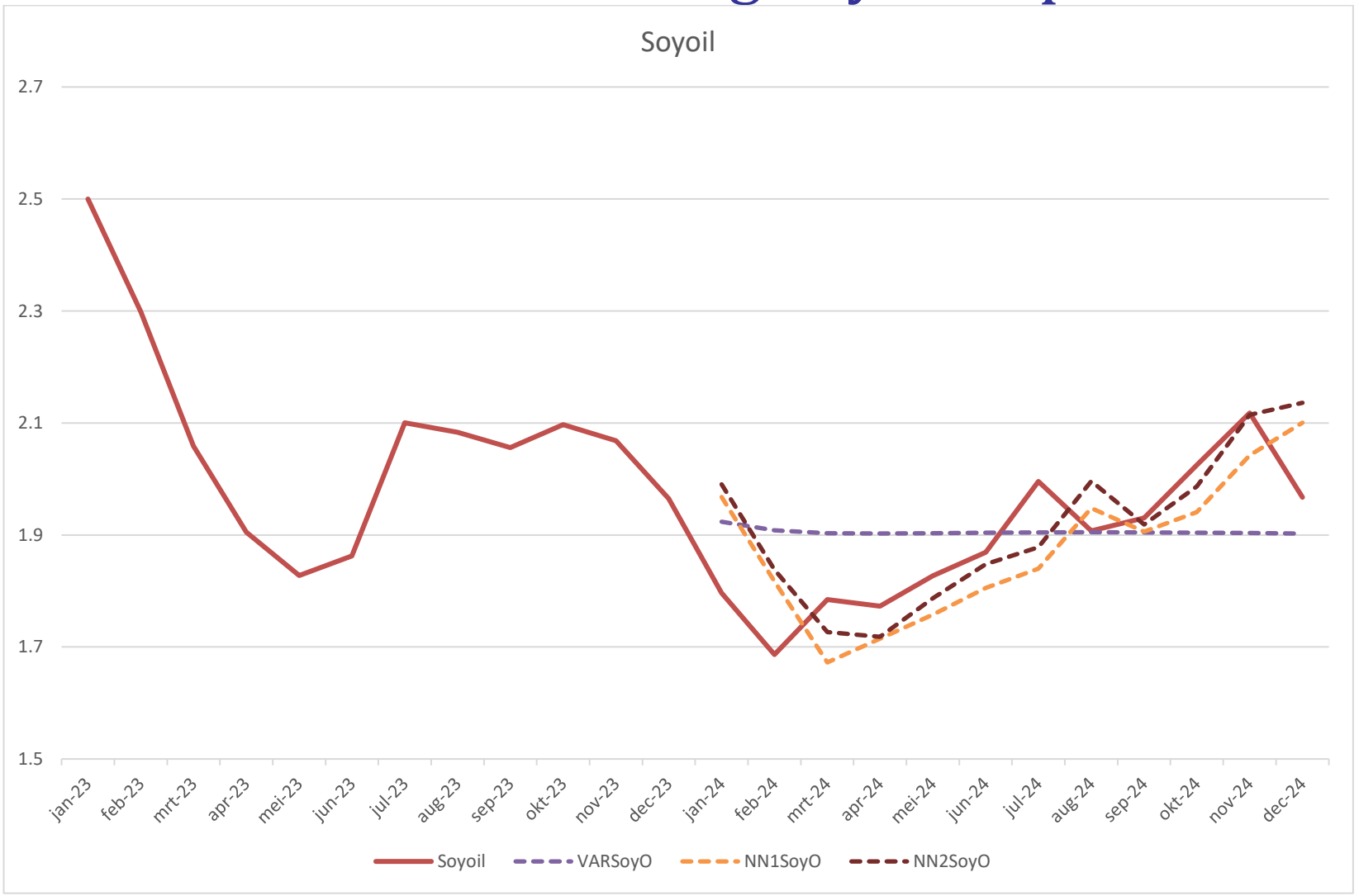
- NN2: Feed-forward NN with 2 hidden layers, total 117 parameters to train.

```
# Create a simple feedforward neural network
model2 = Sequential([
    Dense(10, activation='relu', input_shape=(4,)),  # First hidden layer
    Dense(5, activation='tanh'),  # Second hidden layer
    Dense(2, activation='linear')  # Output layer for regression
])
```

Hands-on case: Predicting Soybean prices



Hands-on case: Predicting Soybean prices



Ok, enough for today

Hope you got a basic understanding of what ML can and can't do.

Several items not discussed:

- Support Vector Machines
- Interpretable Machine Learning (Shapley Values and SHAP)
- Opening up the black boxes
- Many other new routines

Time-series forecasting as an example, but we could also focus on cross-sectional or panel data.

Appetite for more?

5-day PhD course ***Machine Learning and Econometrics***

dr. ir. Koos Gardebroek, Wageningen University, The Netherlands

What is Machine Learning how does it differ from and complement traditional econometrics? Pros and cons of ML methods with typical agricultural cases. Theory in morning sessions, hands-on Python coding in afternoons.

Topics

Introduction to Machine Learning and Python
Variable selection methods
Random trees and forests, bagging and boosting
Deep Learning and Neural Networks
Vector Support Machines and Interpretable Machine Learning

Your own institute? Or spring 2026 Wageningen
Interested? Contact me at: koos.gardebroek@wur.nl

Thank you for your attention!

Any questions?

References

Athey, S., & Imbens, G. W. (2019). Machine learning methods that economists should know about. *Annual Review of Economics*, 11(1), 685-725.

Brignoli, P. L., de Mey, Y., & Gardebroek, C. (2024). Everything under control: comparing machine learning and classical econometric impact assessment methods using FADN data. *European Review of Agricultural Economics*, Online first.

Brignoli, P. L., Varacca, A., Gardebroek, C., & Sckokai, P. (2024). Machine learning to predict grains futures prices. *Agricultural Economics*, 55(3), 479-497.

Gardebroek, C., Urdu, D., Guo, X., Kornelis, M., Harbers, C. J., & van Ruiten, C. A. J. (2024). Transformative Forecasting Methodologies for the Bioeconomy. Web publication/site, Wageningen University & Research.

Goulet Coulombe, P., Leroux, M., Stevanovic, D., & Surprenant, S. (2022). How is machine learning useful for macroeconomic forecasting? *Journal of Applied Econometrics*, 37(5), 920-964.

Efron, B. (2020). Prediction, estimation, and attribution. *International Statistical Review*, 88, S28-S59.

Frey, E., & Seimon, B. (2024). panelsplit (Version 0.4.2) [Computer software]. https://doi.org/10.5281/zenodo.10777259

James, G., Witten, D., Hastie, T., Tibshirani, R., & Taylor, J. (2023). *An introduction to statistical learning: With applications in python*. Springer Nature.

Korstanje, J. (2021). *Advanced forecasting with Python* (pp. 243-251). United States: Apress.

Mullainathan, S., & Spiess, J. (2017). Machine learning: an applied econometric approach. *Journal of Economic Perspectives*, 31(2), 87-106.

Storm, H., Baylis, K., & Heckelei, T. (2020). Machine learning in agricultural and applied economics. *European Review of Agricultural Economics*, 47(3), 849-892.

Zhou, Y., Lentz, E., Michelson, H., Kim, C., & Baylis, K. (2022). Machine learning for food security: Principles for transparency and usability. *Applied Economic Perspectives and Policy*, 44(2), 893-910.