DEEP LEARNING MADE EASY WITH TERSE NETWORKS

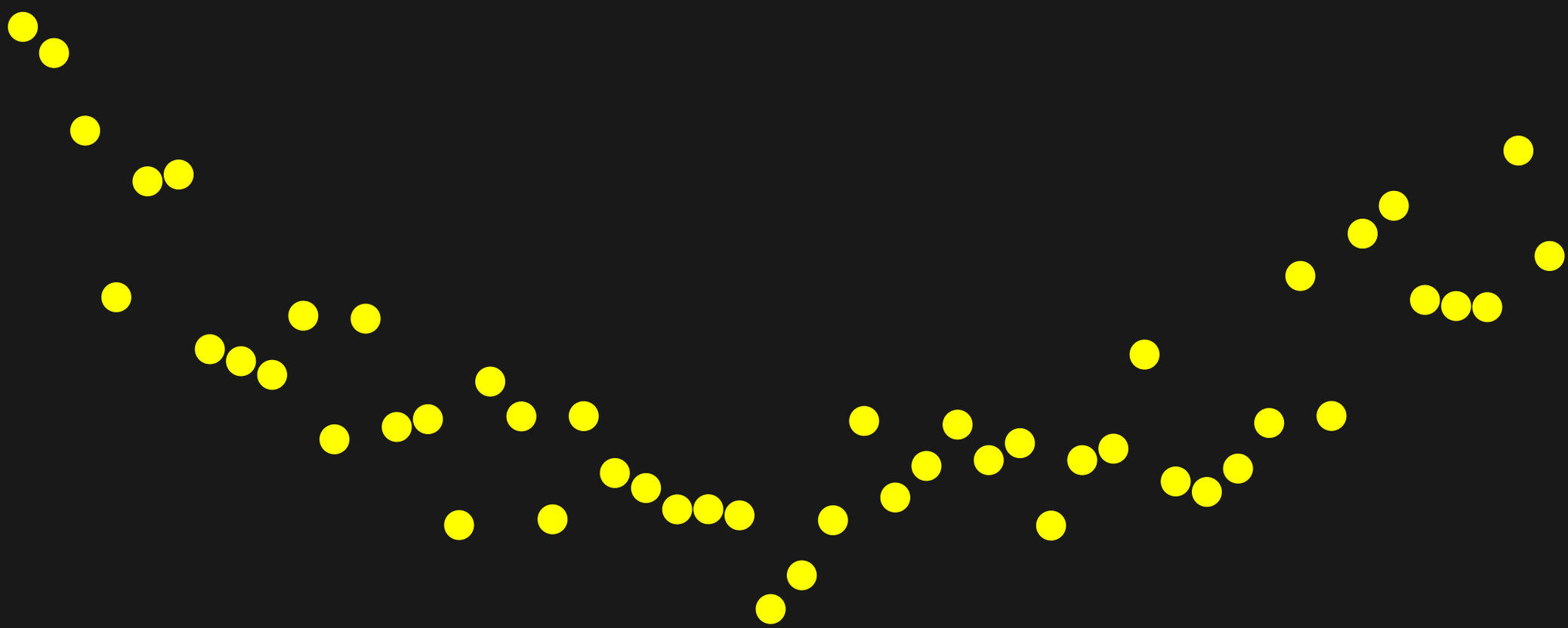
Sebastiaan Vermeulen

EIPC 2020

Econometrics: Can we recognize f?

Machine learning: Can we approximate f?

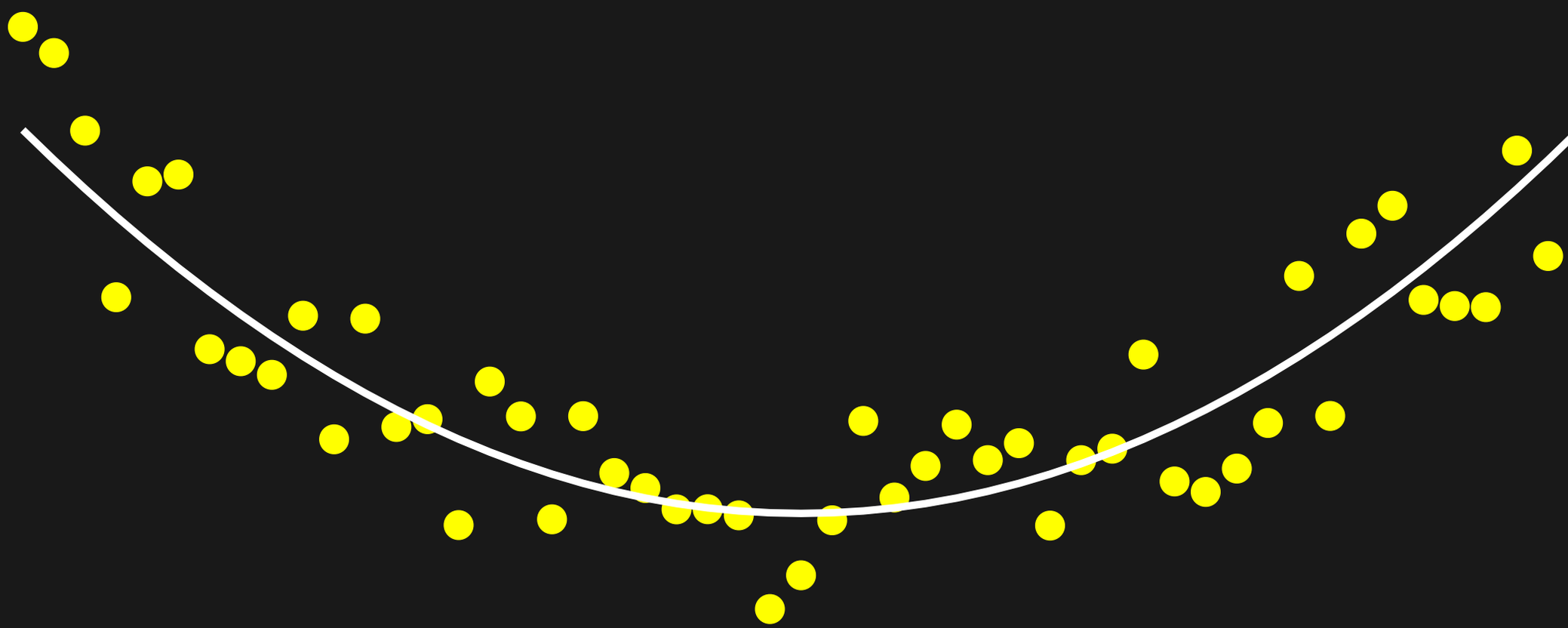
$$y = f(x) + \epsilon$$



Econometrics: Can we recognize f?

Machine learning: Can we approximate f?

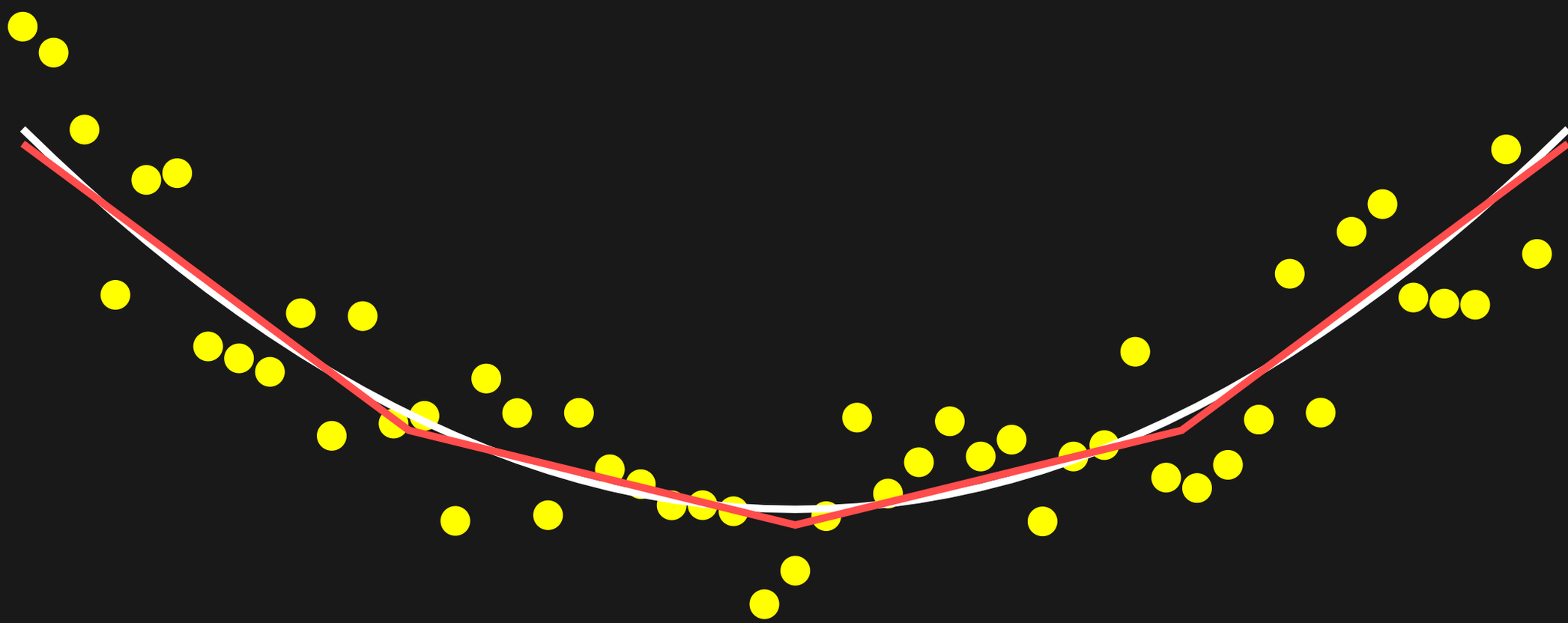
$$y = f(x) + \epsilon$$



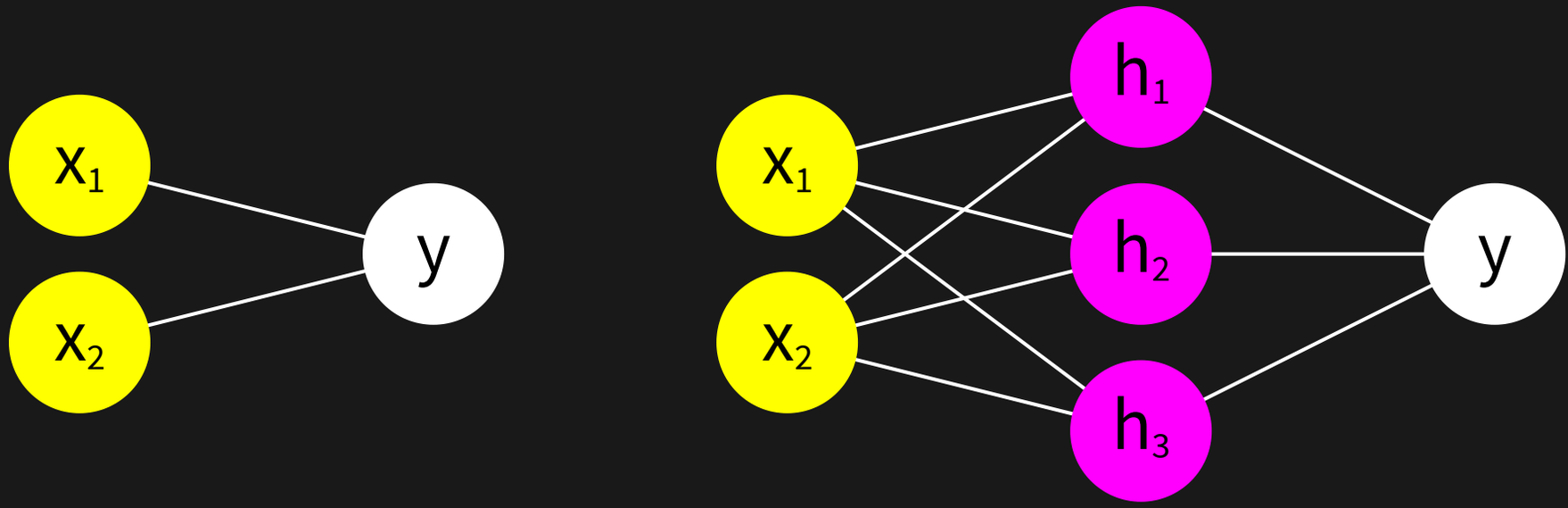
Econometrics: Can we recognize f?

Machine learning: Can we approximate f?

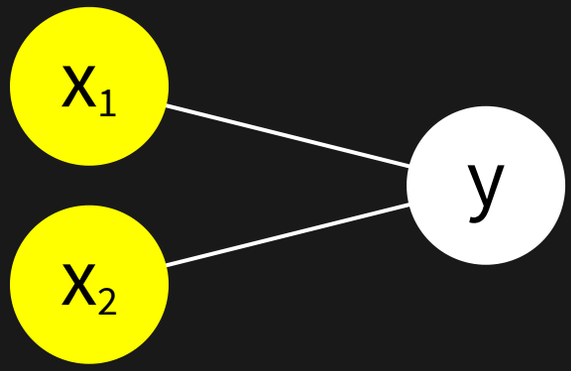
$$y = f(x) + \epsilon$$

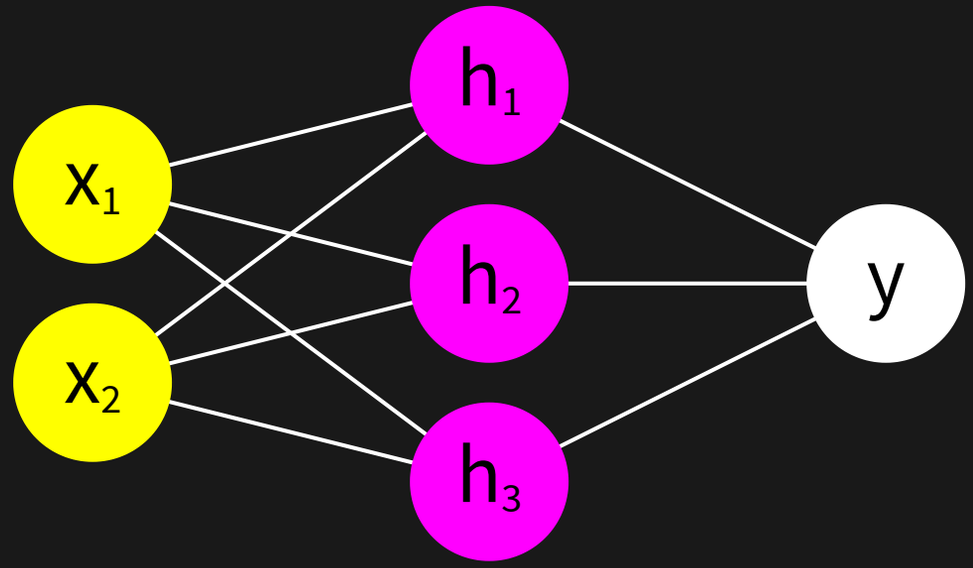


Neural Networks



Neural Networks

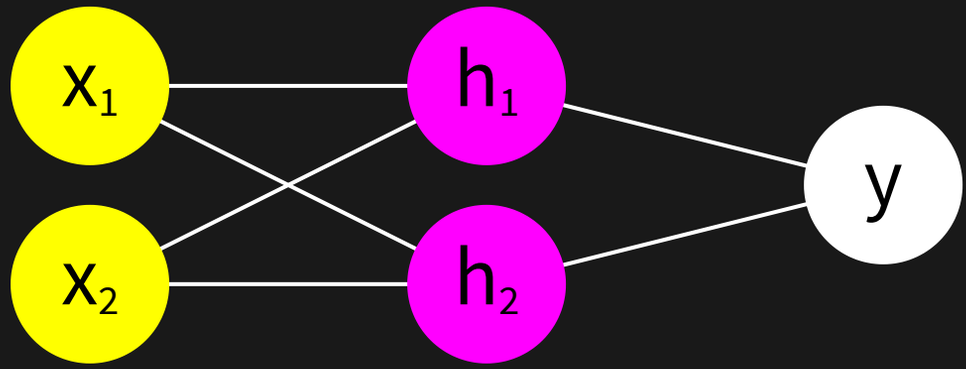




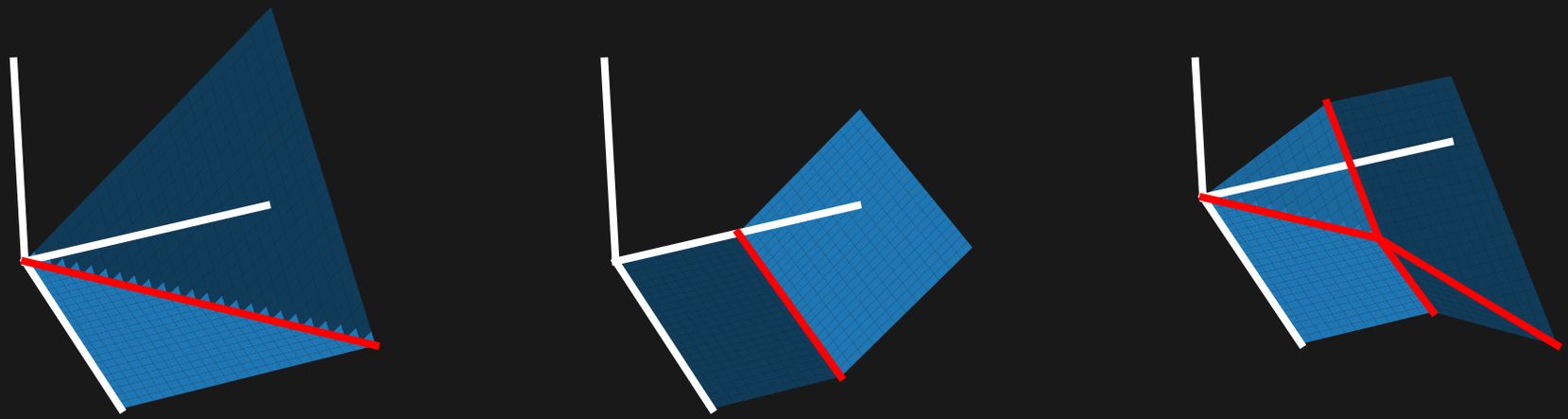
$$h_1(x_t) = \begin{bmatrix} u(a_{11} + x_t' b_{11}) \\ \dots \\ u(a_{n_1 1} + x_t' b_{n_1 1}) \end{bmatrix} \quad h_k(x_t) = \begin{bmatrix} u\big(a_{1k} + h_{k-1}(x_t)' b_{1k}\big) \\ \dots \\ u\big(a_{n_k k} + h_{k-1}(, x_t)' b_{n_k k}\big) \end{bmatrix} \quad y_t = h_k(x_t)'\beta$$

Activation functions

Partitioning the 'input space'



$$(x_2 - x_1)^+ - (x_2 - 0.5)^+$$



Redundancy in the network formula

- Every node is scale-invariant
- Every layer is permutation-invariant

$$h_1(x_t) = \begin{bmatrix} u(a_{11} + x_t' b_{11}) \\ \dots \\ u(a_{n_1 1} + x_t' b_{n_1 1}) \end{bmatrix}$$

$$h_k(x_t) = \begin{bmatrix} u\big(a_{1k} + h_{k-1}(x_t)' b_{1k}\big) \\ \dots \\ u\big(a_{n_k k} + h_{k-1}(, x_t)' b_{n_k k}\big) \end{bmatrix}$$

$$y_t = h_k(x_t)' \beta$$

TERSENET: DEVOID OF SUPERFLUITY

TERSENET: DEVOID OF SUPERFLUITY

Fix the partitioning

TERSENET: DEVOID OF SUPERFLUITY

Fix the partitioning

Ensure every region has identifiable regression

TERSENET: DEVOID OF SUPERFLUITY

Fix the partitioning

Ensure every region has identifiable regression

TEsselated Regression Simplices Encoding network

TERSENET: DEVOID OF SUPERFLUITY

Fix the partitioning

Ensure every region has identifiable regression

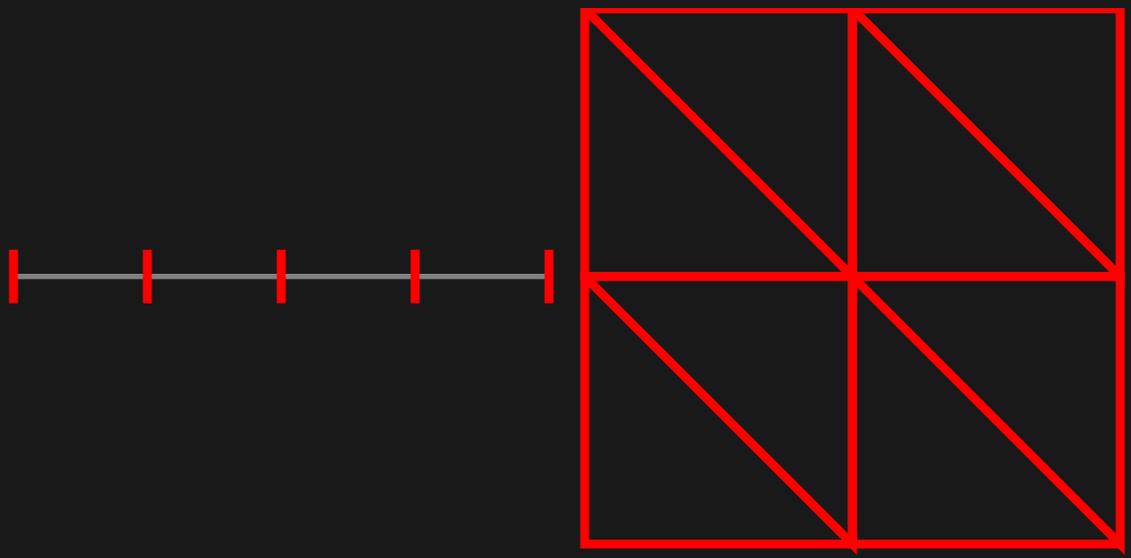
TEsselated Regression Simplices Encoding network

TERSENET: DEVOID OF SUPERFLUITY

Fix the partitioning

Ensure every region has identifiable regression

TEsselated Regression Simplices Encoding network

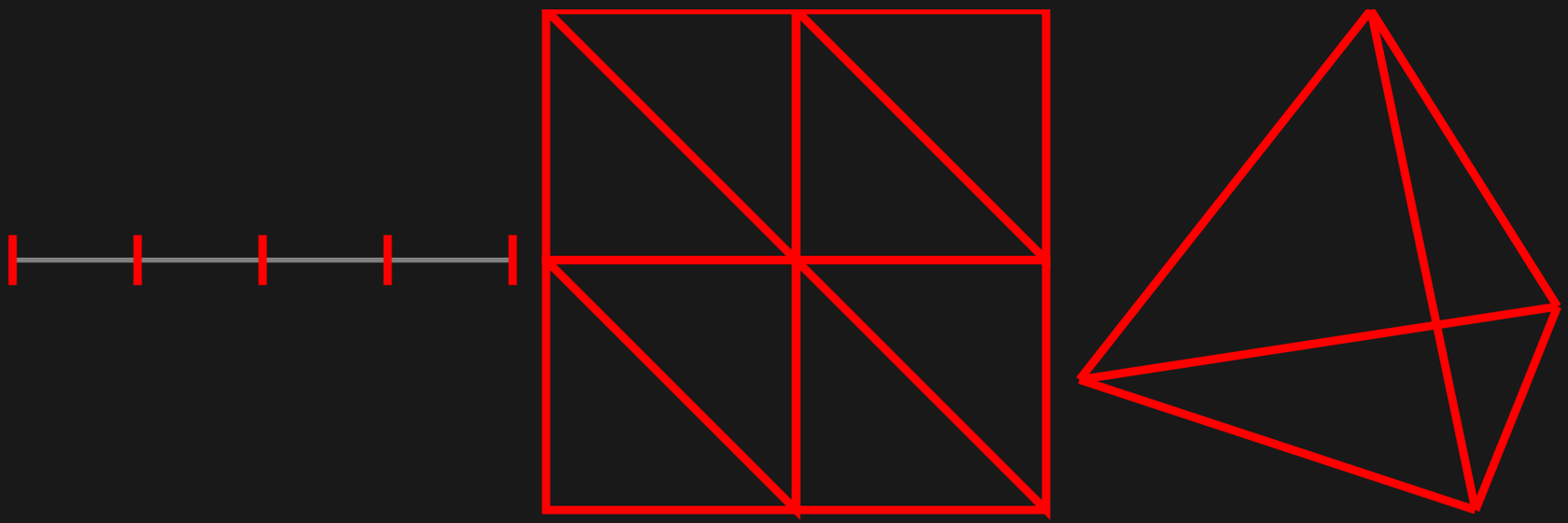


TERSENET: DEVOID OF SUPERFLUITY

Fix the partitioning

Ensure every region has identifiable regression

TEsselated Regression Simplices Encoding network

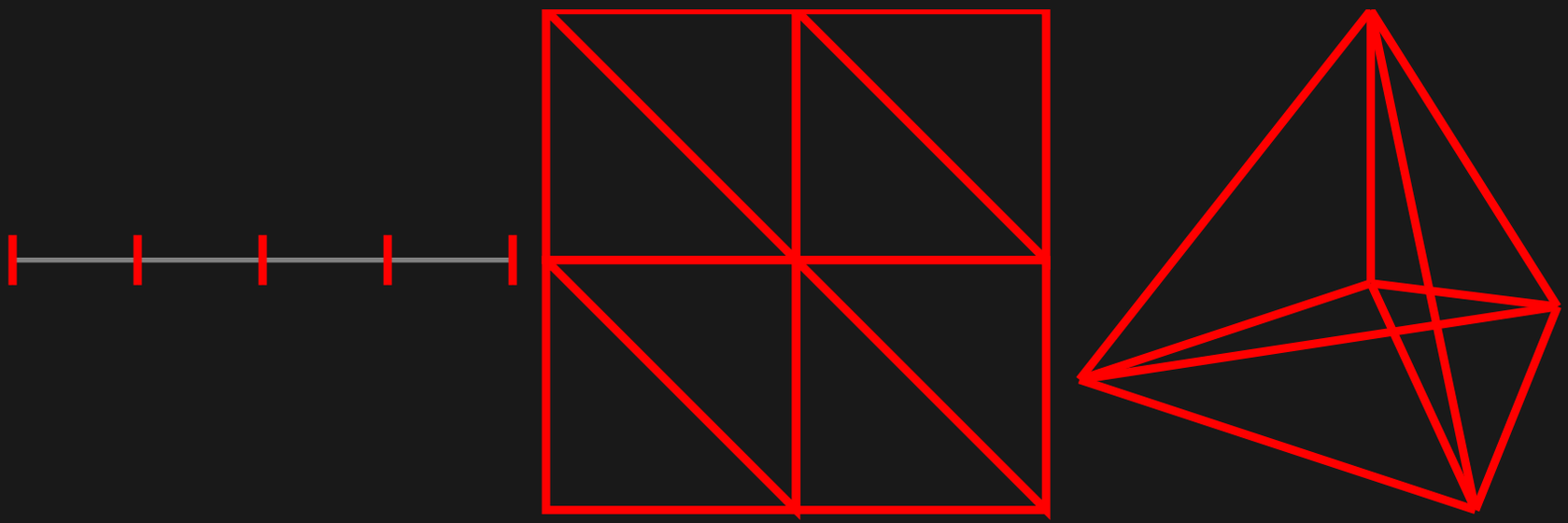


TERSENET: DEVOID OF SUPERFLUITY

Fix the partitioning

Ensure every region has identifiable regression

TEsselated Regression Simplices Encoding network



TERSENET IS LIKE A LINEAR MODEL:

$$h_{1,i+dj}(x_t) = (j - n(x_{t,i})^+)^+$$

$$h_{2,k}(x_t) = \left(1 - \sum_{i=1}^{d} (j_i - n(x_{t,i})^+)^+\right)^+, \quad j_i = w(k,i)$$

$$\hat{y}_t = h_2(x_t)' \beta$$

TERSENET IS LIKE A LINEAR MODEL:

$$h_{1,i+dj}(x_t) = (j - n(x_{t,i})^+)^+$$

$$h_{2,k}(x_t) = \left(1 - \sum_{i=1}^{d} (j_i - n(x_{t,i})^+)^+\right)^+, \quad j_i = w(k,i)$$

$$\hat{y}_t = h_2(x_t)' \, \beta$$

TERSENET IS LIKE A LINEAR MODEL:

$$h_{1,i+dj}(x_t) = (j - n(x_{t,i})^+)^+$$

$$h_{2,k}(x_t) = \left(1 - \sum_{i=1}^{d} (j_i - n(x_{t,i})^+)^+\right)^+, \quad j_i = w(k,i)$$

$$\hat{y}_t = h_2(x_t)' \beta$$

(Regularized) least squares with OLS or LARS

PERFORMANCE (THEORETICAL)

- TERSEnet is parameter efficient for C^2 functions

$$\|f - g\|_\infty = O(W^{-2/d})$$

- TERSEnet parameters have unique global (easy) optimum

PERFORMANCE (PRACTICAL)

TERSEnet is

PERFORMANCE (PRACTICAL)

TERSEnet is

- interpretable

PERFORMANCE (PRACTICAL)

TERSEnet is

- interpretable
- robust

PERFORMANCE (PRACTICAL)

TERSEnet is

- interpretable
- robust
- fast

PERFORMANCE (PRACTICAL)

TERSEnet is

- interpretable
- robust
- fast

(Currently) only for vanilla deep regression

(Currently) only for vanilla deep regression

Insights have wider applications