Machine Learning Models

Gabriel Vasconcelos

June 15, 2018

Outline

- Framework
- Complete Subset Regression
- Shrinkage and Regularization
- Bayesian VAR
- Regression Trees
- Tree Based Algorithms
- Artificial Neural Networks

Framework

- Most of the models preseted here will follow the framework below:

$$y_{t+h} = f(x_t) + \varepsilon_{t+h}$$

- where:
  - y_t is the dependent (response) variable,
  - x_t are the independent variables (controls, characteristics),
  - ε_{t+h} is an error term,
  - $f(\cdot)$ is a mapping function that is a linar function for most models and a nonlinear function for Trees and Neural Networks.

Framework

- Most models use the **direct forecasting** framework. In this case we estimate one model for each forecasting horizon h.
- The recursive forecasting is used only in the VAR framework, where a single model is estimated of y_{t+1} on $y_t, \ldots, y_{t-j}$ and the forecasts are iterated until the desired horizon.
- The y_t in the VAR framework is not the response cariable alone, but all the variables in the VAR system.

Complete Subset Regression

- The complete subset regression is a combination of small linear models.
- Suppose we have K independent variables in x_t, the complete subset regression is estimated following this steps:
  1. Set $k < K$ as the number of variables in each linear regression,
  2. Estimate all possible combinations of models with k variables,
  3. Compute the forecast for each model and take their average as the final forecast.

Complete Subset Regression

- There are some minor ajustments which can be made in the CSR:
  1. One may chose to use fixed controls in each regression, for example, an autorregressive component and seasonal dummies.
  2. The number of variables K may be to big for the model to be computationaly feasible. In this case one can use a pre-testing to select a smaller number of variables.
     - Use the LASSO to select the variables,
     - Use t-statistics to select the variables based on individual regressions of each variable on y_{t+h} and the fixed controls.

Complete Subset Regression

- The CSR is simple but it has presented some good results,
- It is very robust to overfitting,
- It provides good forecasts when the data is noisy and the relation between the variables is linear.

Shrinkage and Regularization

- Models that are estimated by minimizing a Loss function L plus a penalty function P, which penalizes a set o parameters θ.

$$\min_{\theta} L(\theta; y_{t+h}, x_t) + P(\theta)$$

- Loss function: Mostly the quadratic loss, but there are also possibilities such as the ℓ_1 loss function which minimizes the absolute value of the errors.

Quadratic loss:

$$\sum_{t=1}^{T} (y_{t+h} - \beta' x_t)^2$$

ℓ^2 Penalty, the Ridge model

- The Ridge estimator is defined as:

$$arg\min_{\beta}\left[\sum_{t=1}^{T}(y_{t+h}-\beta' x_t)^2+\lambda\sum_{k=j}^{q}\beta_j^2\right]$$

where the second term in the equation above is the penalty function. λ is the regularization parameter which controls how much we penalize the βs.

- The Ridge penalizes the squared value of the coefficients. Less relevant variables are shrunk to zero but they will hardly be exactly zero.

ℓ^2 Penalty, the Ridge model

The Ridge has an analytical solution:

$$\hat{\beta} = (X'X + \lambda I)^{-1}X'y$$

where, $y = (y_{1+h}, \ldots, y_{T+h})'$ and $X = (x_1, \ldots, x_T)'$.

ℓ^1 Penalty, the LASSO model

Some notation:

- $x_i = (x'_{i,S}, x'_{i,S^c})'$, where $x'_{i,S}$ represents the relevant variables and x'_{i,S^c} the irrelevant ones.
- $\beta = (\beta'_S, \beta'_{S^c})'$.
- $\hat{\Sigma} = n^{-1}X'X$

ℓ^1 Penalty, the LASSO model

- The Least Absolute Shrinkage and Selection Operator (LASSO) is defined as:

$$arg \min_{\beta} \left[\sum_{t=1}^{T} (y_{t+h} - \beta' x_t)^2 + \lambda \sum_{k=j}^{q} |\beta_j| \right]$$

- "Irrelevant" variables are set exactly to 0.
- Does not have analytical solution. An algorithm is required.
- Cycling coordinate descent algorithm and the soft-thresholding operator

LASSO and Ridge in one picture

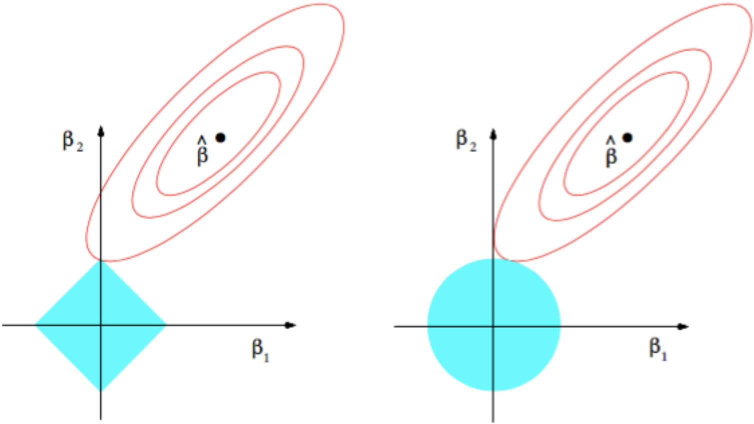


Figure 1: Lasso and Ridge Regularization

LASSO and Ridge regularization path

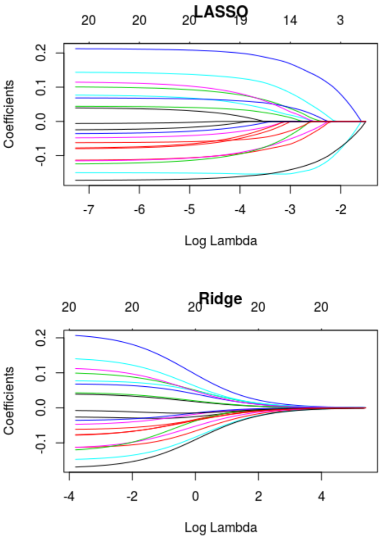


Figure 2: Lasso and Ridge Regularization Path

LASSO: Properties

- Can handle many more variables than observations,
- Under some restrictive conditions can select exactly the correct set of variables,
- Under less restrictive conditions has variable screening, i. e. selects the relevant variables butt some irrelevant variables are also selected,
- Not consistent, in general,
- Biased estimators for the non-zero parametes.

Irreprensentable Condition

- This is an important condition for the LASSO to have variable selection consistency.
- It dictates how the relevant variables may be correlated with irrelevant variables.

Strong Version: $\exists \eta > 0$ such that:

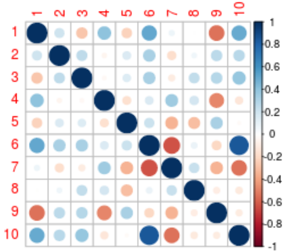
$$\left| \hat{\Sigma}_{S^c S} \hat{\Sigma}_{SS}^{-1} \mathrm{sign}(\beta_S) \right| \leq 1 - \eta$$

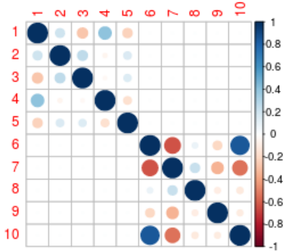
Week Version:

$$\left| \hat{\Sigma}_{S^c S} \hat{\Sigma}_{SS}^{-1} \mathrm{sign}(\beta_S) \right| \leq 1$$

Irreprensentable Condition

- Imagine a model with only 10 candidate variables where only the first five variables are relevant and consider the two covariance designs below:





Irreprensentable Condition

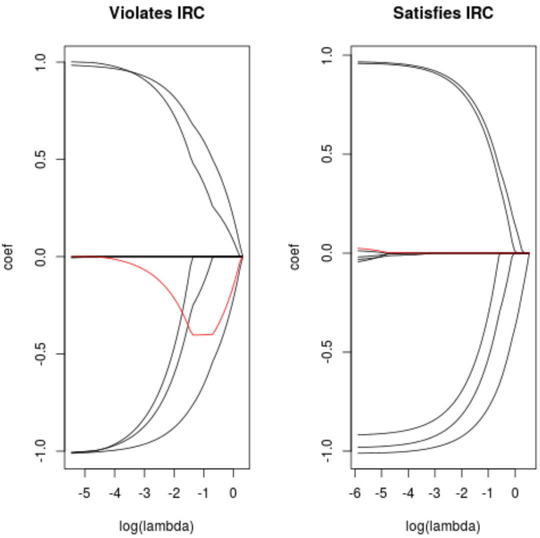


Figure 4: Regularization path

adaptive LASSO (adaLASSO)

The adaLASSO estimator is defined as:

$$arg \min_{\beta} \left[\sum_{t=1}^{T} (y_{t+h} - \beta' x_t)^2 + \lambda \sum_{k=j}^{q} w_j |\beta_j| \right]$$

where $w_j = |\beta_j^*|^{-\tau}$ and β_j^* are coefficients from a first step model.

- The first step model is normally the LASSO but other models such as Ridge, Elastic-Net and OLS are admitted.

adaptive LASSO (adaLASSO)

- Consistent under milder conditions than the LASSO,
- Consistent estimator for the non-zero parameters,
- Har the oracle property under very general conditions.

Irreprensentable Condition

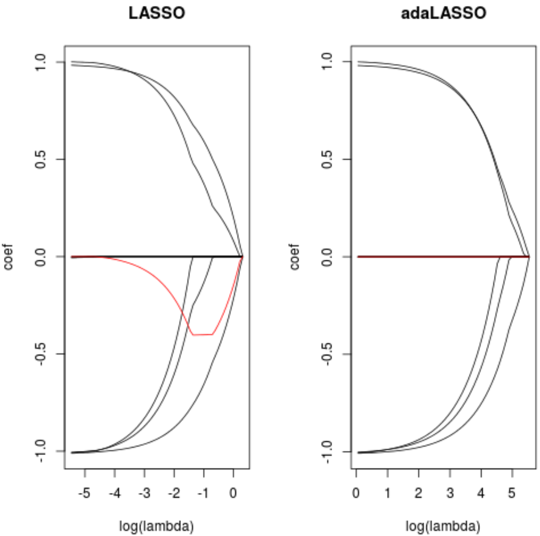


Figure 5: LASSO and adaLASSO

Forecasting Issue

- Is the adaLASSO better than the LASSO for forecasting?
- The answer is *NO*!
- There is absolutely nothing that ensures that the adaLASSO produces better out-of-sample forecasts. The results will deppend on the data.

Choosing λ

- There is not a definitive way to choose λ. The process is usualy data-driven.
- Usual approaches:
    - Information criteria, such as the Bayesian Information Criterion (BIC),
    - Cross-Validation.

ℓ^1 and ℓ^2 regularizations combined: The Elastic-Net

- The Elastic-net estimator is defined as:

$$arg \min_{\beta} \left[\sum_{t=1}^{T} (y_{t+h} - \beta' x_t)^2 + (1-\alpha)\lambda \sum_{k=j}^{q} \beta_j^2 + \alpha\lambda \sum_{k=j}^{q} |\beta_j| \right]$$

- It combines the LASSO and the Ridge penalties.
- Shrinks “irrelevant” variables exactly to zero.

ℓ^1 and ℓ^2 regularizations combined: The Elastic-Net

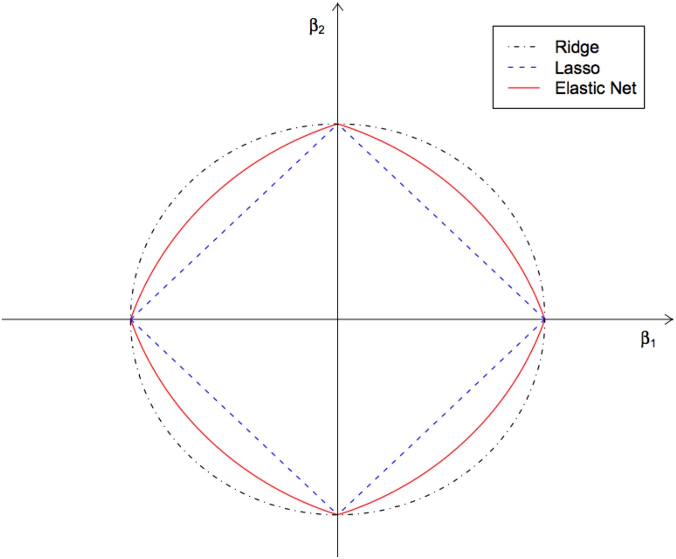


Figure 6: Ridge, LASSO and Elastic-Net

Large Bayesian VARs

Consider the following VAR model:

$$\boldsymbol{y_t} = \mathbf{c} + \sum_{k=1}^{p} \boldsymbol{A}_k \boldsymbol{y_{t-k}} + \varepsilon_{\boldsymbol{t}}$$

where, $\boldsymbol{y}_t$ is an n-dimensional vector with the VAR variables, $\boldsymbol{c}$ is an $n$dimensional vector of constants, $\boldsymbol{A}_k$ are the coefficient matrixs and $\varepsilon_{\boldsymbol{t}}$ is an n-dimensional vector of gaussian and covariance matrix $E[\varepsilon_{\boldsymbol{t}}\varepsilon_{\boldsymbol{t}}{}'] = \Sigma$. The equation above may be written as:

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{A} + \boldsymbol{\epsilon}$$

,

where, $\boldsymbol{Y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t)'$, $\boldsymbol{X} = (\boldsymbol{X}_1, \ldots, \boldsymbol{X}_t)'$ with $\boldsymbol{y}_t = (y_{1,t}, \ldots, y_{n,t})'$, $\boldsymbol{X}_t = (1, \boldsymbol{y}'_{t-1}, \ldots, \boldsymbol{y}'_{t-p})'$, $\boldsymbol{A} = (c, \boldsymbol{A}_1, \ldots, \boldsymbol{A}_p)'$ and $\boldsymbol{\epsilon} = (\varepsilon_1, \ldots \varepsilon_T)'$.

Large Bayesian VARs

The LBVAR inflates $\boldsymbol{Y}$ and $\boldsymbol{X}$ with dummy observations to replicate the Normal-Inverse-Wishart prior, which has the following moments:

$$E[(\boldsymbol{A}_k)_{i,j}] = \begin{cases} \delta_i, & j = i, k = 1 \\ 0, & \text{otherwise} \end{cases} , \quad V[(\boldsymbol{A}_k)_{i,j}] = \begin{cases} \frac{\lambda^2}{k^2}, & j = i \\ \frac{\lambda^2 \sigma_i^2}{k^2 \sigma_j^2}, & \text{c.c.} \end{cases}$$

- λ controls the relative importance between the prior and the data:
  - $\lambda = 0$: posterior equals prior and the data is ignored,
  - $\lambda = \infty$: Model ignores the prior and we have the OLS estimates

Large Bayesian VARs

The Normal-Inverse-Wishart prior is defined as:

$$vec(\boldsymbol{A})|\Sigma \sim N(vec(\boldsymbol{A_0}), \Sigma \otimes \Omega_0)$$

,

$$\Sigma \sim iW(\boldsymbol{S}_0, \alpha_0)$$

,

where $\boldsymbol{A}_0$, Ω_0, $\boldsymbol{S}_0$ and α_0 hyperparameters chosen to obtain the moments from the previous slide and Σ is the Minnesota prior covariance matrix.

Large Bayesian VARs

The dummy observations follow the expressions below:

$$\boldsymbol{Y}_d = \left(\begin{array}{c} diag(\delta_1\sigma_1, \ldots, \delta_n\sigma_n)/\lambda \\ \mathbf{0}_{n(p-1)\times n} \\ \hdashline diag(\sigma_1, \ldots, \sigma_n) \\ \hdashline \mathbf{0}_{1\times n} \end{array} \right), \boldsymbol{X}_d = \left(\begin{array}{cc} \boldsymbol{J}_p \otimes diag(\sigma_1, \ldots, \sigma_n)/\lambda & \mathbf{0}_{np\times 1} \\ \hdashline \mathbf{0}_{n\times np} & \mathbf{0}_{n\times 1} \\ \hdashline \mathbf{0}_{1\times np} & \rho \end{array} \right)$$

where, $\boldsymbol{J}_p = diag(1, 2, \ldots, p)$ and ρ is a small value. σ_i^2 is the variance of the ith variable.

Large Bayesian VARs

Finally, we plug the dummy observations in the data:

- $\boldsymbol{Y}_* = (\boldsymbol{Y}'\boldsymbol{Y}_d')'$,
- $\boldsymbol{X}_* = (\boldsymbol{X}'\boldsymbol{X}_d')'$,
- $\boldsymbol{\epsilon}_* = (\boldsymbol{\epsilon}'\boldsymbol{\epsilon}_d')'$.

Resulting on the following VAR model:

$$\boldsymbol{Y}_* = \boldsymbol{X}_*\boldsymbol{A} + \boldsymbol{\epsilon}_*$$

Large Bayesian VARs

The model posterior will be:

$$vec(\boldsymbol{A})|\Sigma, \boldsymbol{Y} \sim N(vec(\tilde{\boldsymbol{A}}), \Sigma \otimes (\boldsymbol{X}_*'\boldsymbol{X}_*)^{-1})$$

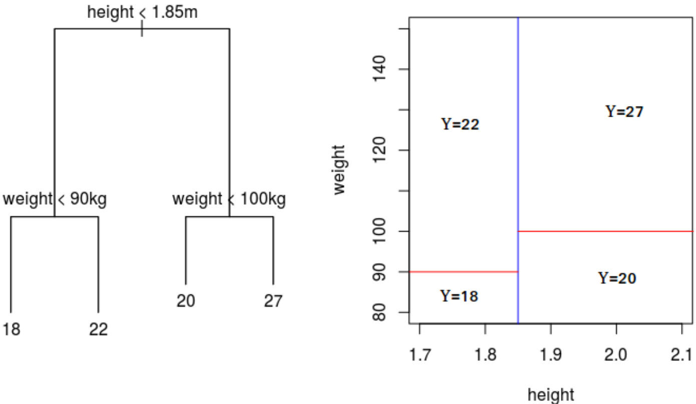
$$\Sigma|\boldsymbol{Y} \sim iW(\tilde{\Sigma}, T_d + 1 + T - np)$$

where $\tilde{\boldsymbol{A}} = (\boldsymbol{X}_*'\boldsymbol{X}_*)^{-1}\boldsymbol{X}_*'\boldsymbol{Y}_*$, which is the OLS estimator of the inflated VAR and the posterior mean of the Minnesota prior VAR.

- The posterior covariance matrix will deppend on: $\tilde{\Sigma} = (\boldsymbol{Y}_* - \boldsymbol{X}_*\tilde{\boldsymbol{A}})'(\boldsymbol{Y}_* - \boldsymbol{X}_*\tilde{\boldsymbol{A}})$.

Regression Trees - Intuition

- A regular regression tree is a nonparametric model that approximates a nonlinear function with local predictions using recursive partitioning of the space of the predictor variables.
- A tree may be represented by a graph as in the left side of figure below, which is equivalent as the partitioning in the right side of the figure for this bi-dimensional case.



Regression Trees

- Let $\boldsymbol{x}_i = (x_{i,1}, \dots, x_{i,m})'$ be a set of m explanatory variables for the response variable y_i. The relation between y_i and $\boldsymbol{x_i}$ is mapped by a unknown function f such that:

$$y_i \approx f(\boldsymbol{x_i}) + \varepsilon_i, \qquad i = 1, \dots, N$$

- A Regression Tree model with K terminal nodes (leaves) is a recursive partitioning model that approximates $f(\cdot)$ by a general nonlinear function $H(\boldsymbol{x}_i, \boldsymbol{\psi})$, where $\boldsymbol{\psi}$ is a vector of parameters.
- $H(\cdot)$ is a piecewise function with K subregions that are orthogonal to the axis of the predictor variables.

Regression Trees

- Each subregion represents one terminal node, and they are defined by $k_j(\boldsymbol{\theta}_j)$, $j = 0, \ldots, K-1$. The parameter $\boldsymbol{\theta_j}$ defines each subregion such that:

$$f(\boldsymbol{x}_i) = \sum_{j=0}^{K-1} \beta_j I_j(\boldsymbol{x}_i; \boldsymbol{\theta}_j)$$

where $I_j(\boldsymbol{x}_i; \boldsymbol{\theta}_j)$ is an indicator function such that:

$$I_j(\boldsymbol{x}_i; \boldsymbol{\theta}_j) = \begin{cases} 1 & \text{if } \boldsymbol{x}_i \in k_j(\boldsymbol{\theta}_j), \\ 0 & \text{otherwise.} \end{cases}$$

Regression Trees

- Each new partition in the tree is created by solving an optimization problem.
- The objective is to find the partition that has the biggest contribution in reducing the model's squared error.
- The optimal partition is found by searching every possible variable and observation.

Random Forests

- Random Forest is an algorithm that combines regression trees with bootstrap aggregating (Bagging) techniques.
- Regression trees alone are know to be very unstable models. A small change in the data may drasticaly change the predicted values.
- Bagging benefits from this instability to generate stable models.

Random Forests

The Random Forest estimation follows these steps:

- 1. Generate a bootstrap sample b from the data with replacement. Normally b has the same number of observations as the data.
- 2. In the bootstrap sample, grow a regression tree (normally big trees).
- 3. Repeat steps 1) and 2) B times and compute the forecast as the average forecast across all estimated trees.
- OBS: In every new node in a tree it is usual to select only a subset of the potential variables to determine where to make the split. This adds more instability to individual trees but the final Random Forest results improve.

Boosting

- Boosting is an iterative algorithm that combines models in an aditive way.
- Although the algorithm is very general and accepts several types of models, the most usual is to use boosting on regression trees.
- Has the advantage of growing smaller trees than the Random Forest, which makes the algorithm faster in some cases.
- More succetible to over-fitting if poorly tuned than the Random Forest.
- More parameters to tune than the Random Forest.

Boosting

The Boosting algorithm follows these steps:

- Set $\phi_0 = \bar{y}$ and for $m = 1, \ldots, M$ do:

1. Estimate the residuals $u_m = y - \phi_m$,
2. Grow a regression tree in $u_{i,m} = H_m(\boldsymbol{x}_t, \boldsymbol{\psi}_m) + \varepsilon$,
3. Make $\hat{\rho}_m = arg \min_\rho \sum_{t=1}^{T} [u_{t,m} - \rho H_m(\boldsymbol{x}_t, \boldsymbol{\psi}_m)]^2$
4. Update $\phi_{m+1} = \phi_m + v\rho H_m(\boldsymbol{x}, \boldsymbol{\psi}_m)$

Artificial Neural Networks

- Nonlinear models,
- Regression and classification,
- Components:
  - Input layer,
  - Weights w,
  - Activation Function,
  - Hidden layers,
  - Output layer.

Artificial Neural Networks

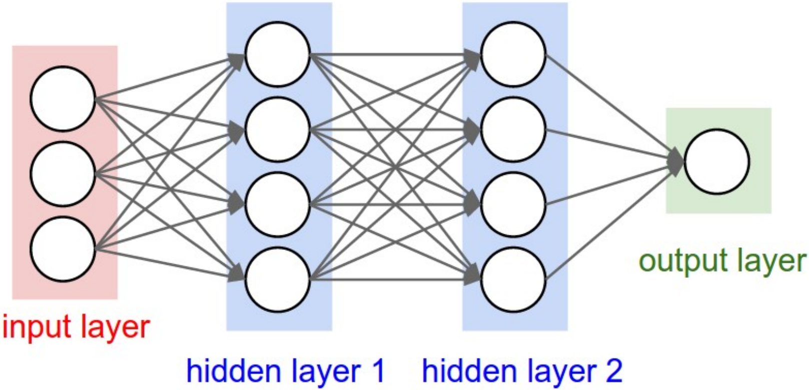


Figure 8: General ANN

Artificial Neural Networks

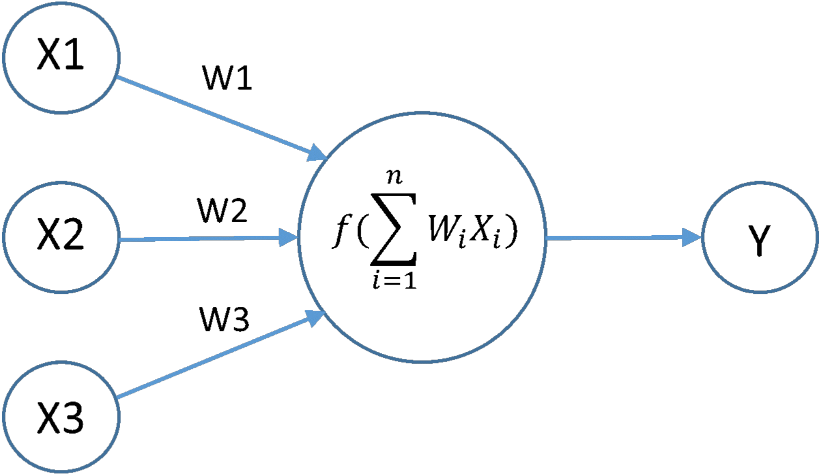
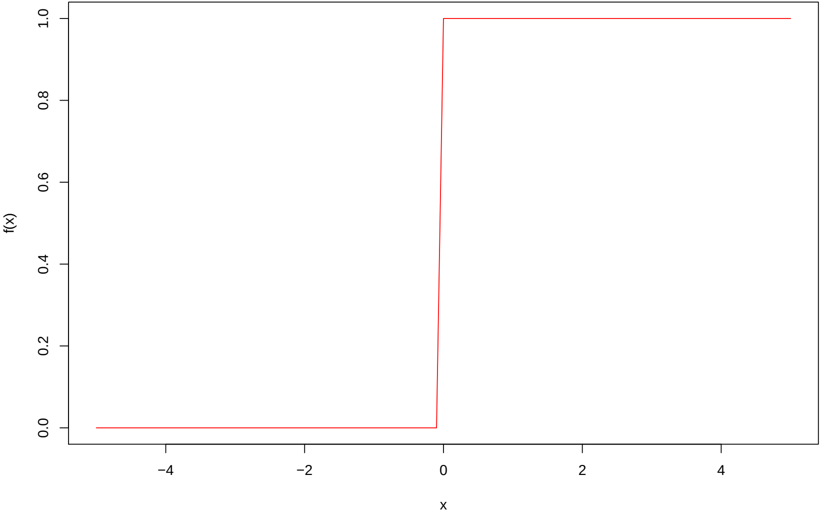


Figure 9: Simple ANN

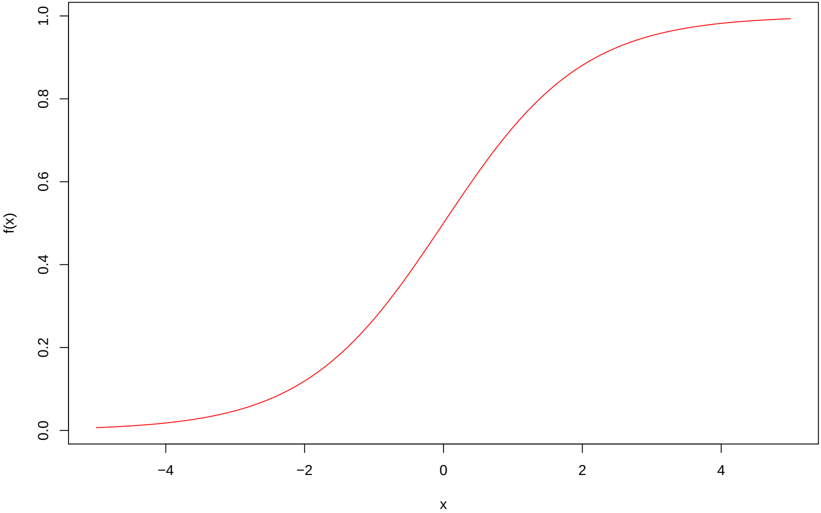
Activation Functions: Indicator

$$f(x) = 1[x > 0]$$



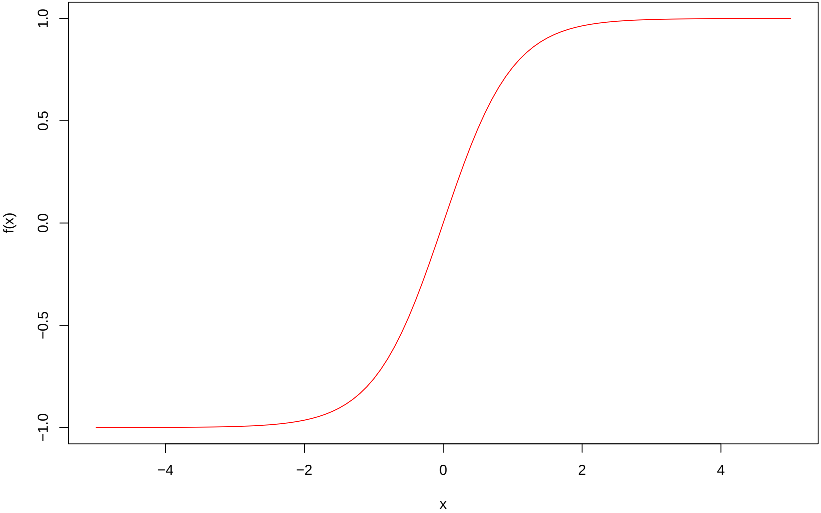
Activation Functions: Logistic

$$f(x) = \frac{1}{1 + e^{-x}}$$



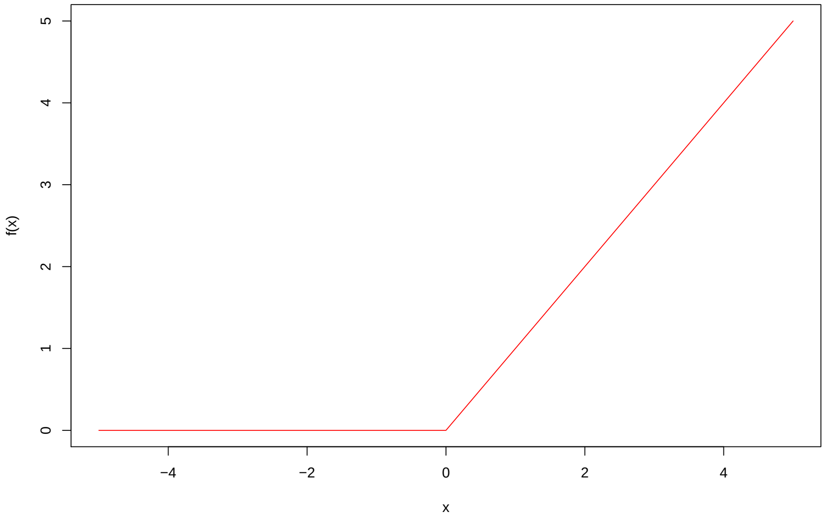
Activation Functions: Hyperbolic Tangent

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$



Activation Functions: Rectifier

$$f(x) = \max(x, 0)$$



0
1
2
3
4
5
f(x)
−4
−2
0
2
4
x