Machine Learning
and
Applied Econometrics

Regression-Based Models

Machine Learning and Econometrics

- This introductory lecture is based on
  - Kevin P. Murphy, Machine Learning A Probabilistic Perspective, The MIT Press, 2017.
  - Darren Cook, Practical Machine Learning with H2O, O'Reilly Media, Inc., 2017.
  - Scott Burger, Introduction to Machine Learning with R: Rigorous Mathematical Analysis, O'Reilly Media, Inc., 2018.

Supervised Machine Learning

- Regression-based Methods
  - Generalized Linear Models
    - Linear Regression
    - Logistic Regression
  - Deep Learning (Neural Nets)
- Tree-based Ensemble Methods
  - Random Forest (Bagging: Bootstrap Aggregation)
    - Parallel ensemble to reduce variance
  - Gradient Boost Machine (Boosting)
    - Sequential ensemble to reduce bias

Regression-Based Models

- Generalized Linear Models
  - Linear Regression
  - Logistic Regression
- Deep Learning (Neural Nets)
  - Feed-Forward ANN with Back-Propagation

Generalized Linear Model

- Generalized Linear Model
  – Regression: OLS
  – Classification: Logit
- Regularization in GLM
  – Ridge Regression
  – LASSO
  – Elastic Net

Generalized Linear Model

- GLM is a flexible generalization of OLS.
- OLS restricts the regression coefficients to have a constant effect on the dependent variable. GLM allows for this effect to vary along the range of the explanatory variables.
- In particular, a nonlinear function links the linear parameterization to the expected value of the random variable.

Generalized Linear Model

- Let $\mu = E(Y|X)$ and $\eta = X\beta$. The basic structure of GLM is the link function $g(\mu) = \eta$:

$$\mu = E(Y \mid X) \xrightarrow{g} \eta = X\beta$$

- Therefore, $Y = g^{-1}(X\beta) + \varepsilon$
- The response variable Y may be continuous for a regression model or discrete for a classification model.

Generalized Linear Model

- The random component ε is assumed to follow a family of probability distribution (e.g., Gaussian, Gamma, Binomial, Poisson, etc..) which formulates the GLM log-likelihood to be maximized.
- The nonlinear invertible link function g transforms the expectation of the response to the linear predictor. The following link functions are considered: identity, log, inverse, logic etc..

Generalized Linear Model

- Standard GLM
  - $\max_{\beta}$ (GLM Log-likelihood)
- GLM with Variable Selection
  - $\max_{\beta,\beta 0}$ (GLM Log-likelihood – Regularization Penalty)
- The elastic net regularization penalty is the weighted sum of the L1 and L2 norms of the coefficients, with no penalty on the intercept term.

Generalized Linear Model

- Regularization Penalty
  - Elastic Net Regularization Penalty

    $$\lambda P(\beta) = \lambda\left[\alpha \|\beta\|_1 + (1-\alpha)\|\beta\|_2\right]$$

    $$\|\beta\|_1 = \sum_k |\beta_k|, \quad \|\beta\|_2 = \sum_k \beta_k^2$$

    $$0 < \alpha < 1, \quad \lambda > 0$$

  - Ridge Regression, as $\alpha = 0$.
  - LASSO (Least Absolute Shrinkage and Selection Operator), as $\alpha = 1$.

Generalized Linear Model

- Variable Selection
  - The intercept β_0 is not restricted
  - The predictors X_{ij} should be standardized
  - The tuning parameter λ is determined separately by cross-validation
  - The elastic net parameter α may be selected by grid search based on MSE critria
  - The LASSO model or as α -> 1 can perform variable selection to achieve a sparse model

Generalized Linear Modeling with H2O

- Basic Model
  - `h2o.glm (x, y, training_frame, model_id = NULL, …)`
- Model Specification Options
  - `family = c("gaussian", "binomial", "quasibinomial", "ordinal", "multinomial", "poisson", "gamma", "tweedie", "negativebinomial"),`
  - `tweedie_variance_power = 0,`
  - `tweedie_link_power = 1,`
  - `link = c("family_default", "identity", "logit", "log", "inverse", "tweedie", "ologit", "oprobit", "ologlog"),`

Generalized Linear Modeling with H2O

- Cross-Validation Parameters
  - `validation_frame = NULL,`
  - `nfolds = 0, seed = -1,`
  - `keep_cross_validation_models = TRUE,`
  - `keep_cross_validation_predictions = FALSE,`
  - `keep_cross_validation_fold_assignment = FALSE,`
  - `fold_assignment = c("AUTO", "Random", "Modulo", "Stratified"),`
  - `fold_column = NULL,`

Generalized Linear Modeling with H2O

- Regularization Options
  - `alpha = NULL,`
  - `lambda = NULL,`
  - `lambda_search = FALSE,`
  - `nlambdas = -1,`
- Early Stopping
  - `early_stopping = TRUE,`
  - `max_active_predictors = -1,`
  - `max_iterations = -1,`

Generalized Linear Modeling with H2O

- Other Important Control Parameters
  - `solver = c("AUTO", "IRLSM", "L_BFGS", "COORDINATE_DESCENT_NAIVE", "COORDINATE_DESCENT")`
  - `standardize = TRUE`
  - `intercept = TRUE`
  - `missing_values_handling = c("MeanImputation", "Skip"),`

Grid Search of Models

- `h2o.grid(algorithm, grid_id, x, y, training_frame, ..., hyper_params = list(), is_supervised = NULL, do_hyper_params_check = FALSE, search_criteria = NULL)`

Deep Learning (Neural Nets)

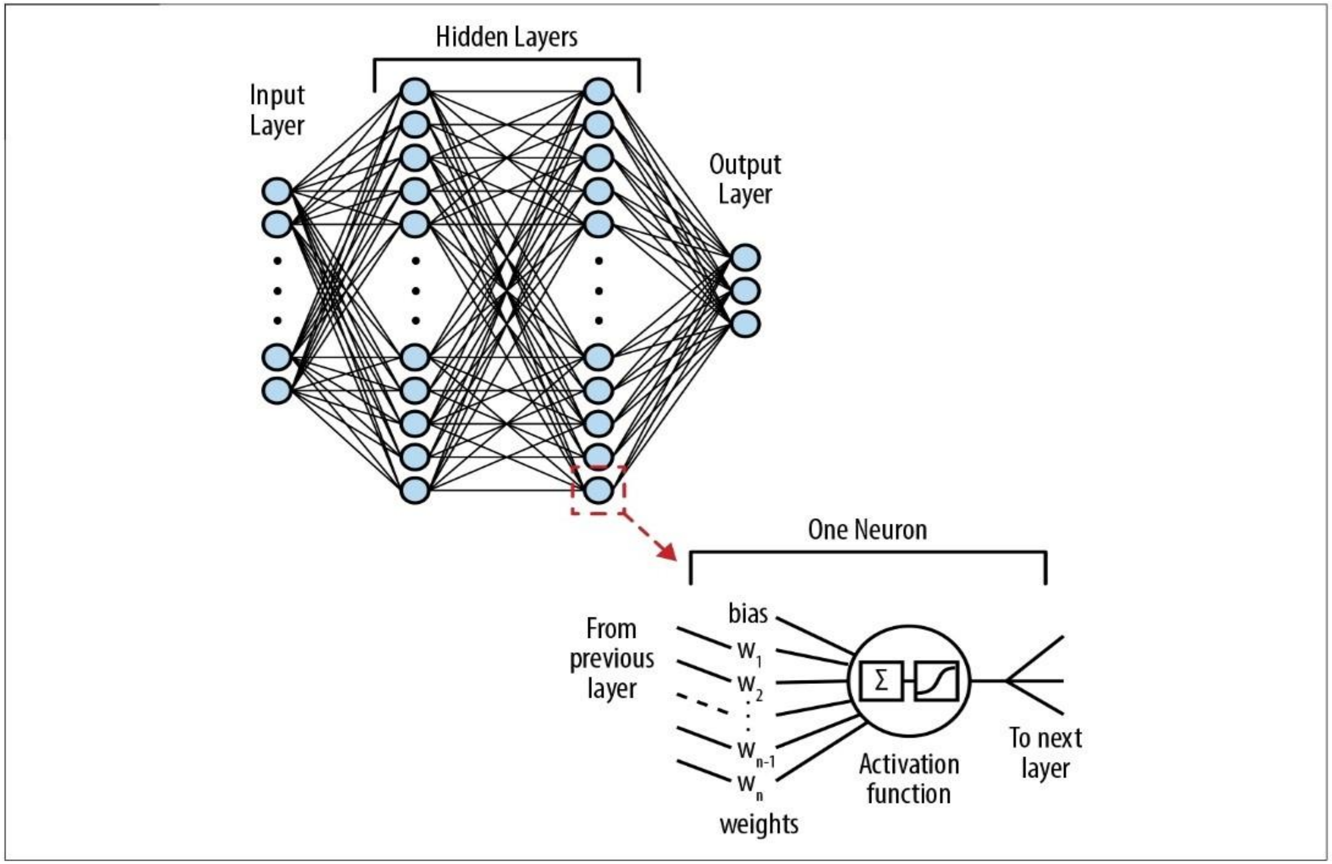

Figure 8-1. Network, layers, neurons

Deep Learning (Neural Nets)

- Multi-layer feed-forward ANN trained with stochastic gradient descent using back-propagation
  - A larger number of hidden layers consisting of neurons with tanh, rectifier, and maxout activation functions
- Generalize from GLM $\hat{y} = f\left(\sum_{i=1}^{m} w_i x_i + w_0\right)$

Deep Learning (Neural Nets)

$$\hat{y}_j = f^{L}\left(\sum_{i=1}^{m^L} w_i^L z_i^{L-1} + w_0^L\right)$$

$$where\ z_i^0 = x_i\ and\ z_k^l = f^{l}\left(\sum_{i=1}^{m^l} w_i^l z_i^{l-1} + w_0^l\right)$$

$$w_i^l = weight\ of\ node\ i\ in\ layer\ l,\ z_i^l$$

$$m^l = nodes\ in\ layer\ l = 1, 2, \ldots, L$$

$$x_i = input; \quad y_j = output$$

$$f = activation\ \ function\ (Tanh, \mathrm{Re}ctifier, Maxout)$$

Activation Functions

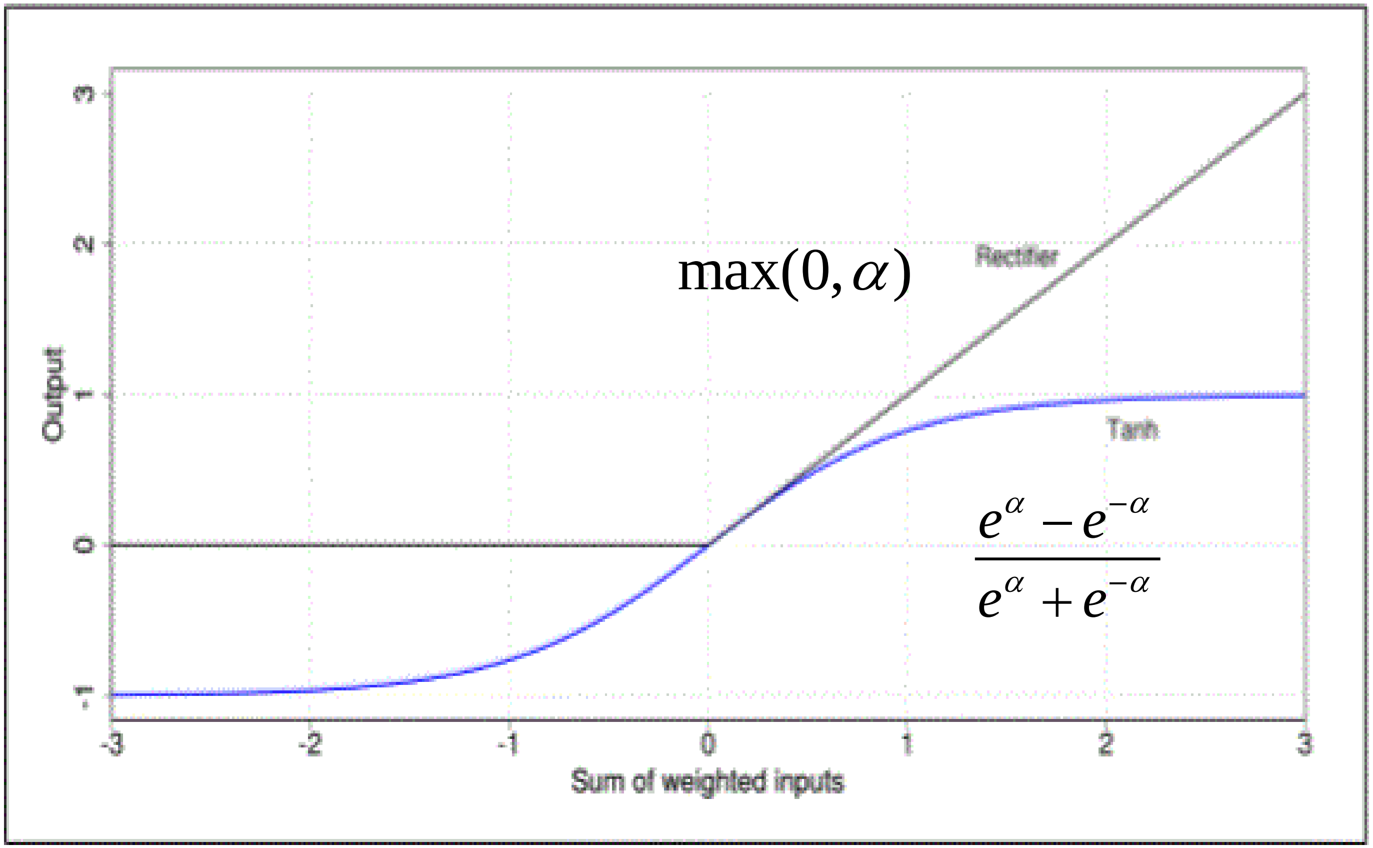

Figure 8-2. Rectifier and Tanh activation functions

Deep Learning with H2O

- Basic Model
  - `h2o.deeplearning (x, y, training_frame, model_id = NULL …)`
- Model Specification Options
  - `hidden = c(200, 200), epochs = 10, seed = -1,`
  - `activation = c("Tanh", "TanhWithDropout", "Rectifier", "RectifierWithDropout", "Maxout", "MaxoutWithDropout"),`
  - `loss = c("Automatic", "CrossEntropy", "Quadratic", "Huber", "Absolute", "Quantile"),`

Deep Learning with H2O

- Model Specification Options (Continued)
  - `distribution = c("AUTO", "bernoulli", "multinomial", "gaussian", "poisson", "gamma", "tweedie", "laplace", "quantile", "huber"),`
  - `quantile_alpha = 0.5,`
  - `tweedie_power = 1.5,`
  - `huber_alpha = 0.9,`
  - `ignore_const_cols = TRUE,`
  - `weights_column = NULL,`
  - `offset_column = NULL,`
  - `standardize = TRUE,`
  - `checkpoint = NULL,`

Deep Learning with H2O

- Cross-Validation Parameters
  - `validation_frame = NULL,`
  - `nfolds = 0, seed = -1,`
  - `keep_cross_validation_models = TRUE,`
  - `keep_cross_validation_predictions = FALSE,`
  - `keep_cross_validation_fold_assignment = FALSE,`
  - `fold_assignment = c("AUTO", "Random", "Modulo", "Stratified"),`
  - `fold_column = NULL,`

Deep Learning with H2O

- Regularization Options
  - `l1 = 0, l2 = 0, max_w2 = 3.4028235e+38,`
  - `input_dropout_ratio = 0,`
  - `hidden_dropout_ratios = NULL,`
- Early Stopping
  - `classification_stop = 0,`
  - `regression_stop = 1e-06,`
  - `stopping_rounds = 5, stopping_tolerance = 0,`
  - `max_runtime_secs = 0,`
  - `stopping_metric = c("AUTO", "deviance", "logloss", "MSE", "RMSE", "MAE", "RMSLE", "AUC", "lift_top_group", "misclassification", "mean_per_class_error", "custom", "custom_increasing"),`

Deep Learning with H2O

- Advanced Optimization Options
  - Adaptive Learning
    - `adaptive_rate = TRUE,`
    - `rho = 0.99, epsilon = 1e-08,`
    - `rate = 0.005,`
  - Rate Annealing
    - `rate_annealing = 1e-06, rate_decay = 1,`
  - Momentum Training
    - `momentum_start = 0, momentum_ramp = 1e+06,`
    - `momentum_stable = 0,`
    - `nesterov_accelerated_gradient = TRUE,`

Deep Learning with H2O

- Other Important Control Parameters
  - `train_samples_per_iteration = -2,`
  - `target_ratio_comm_to_comp = 0.05,`
  - `pretrained_autoencoder = NULL,`
  - `score_interval = 5,`
  - `score_training_samples = 10000,`
  - `score_validation_samples = 0,`
  - `score_duty_cycle = 0.1,`
  - …