CyberMAGICS Workshop: Introduction to Machine Learning

Ken-ichi Nomura

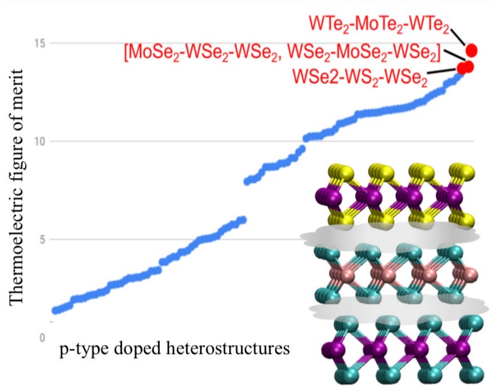
Collaboratory for Advanced Computing & Simulations
University of Southern California

Machine Learning hands-on: Anikeya Aditya, Nitish Baradwaj, Taufeq Razakh, Liqiu Yang

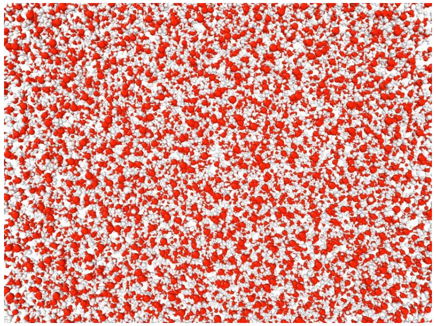
Supported by National Science Foundation, Award OAC-2118061

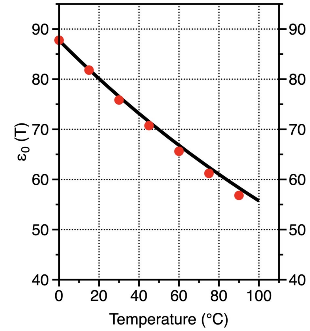
Material Modeling with Machine Learning

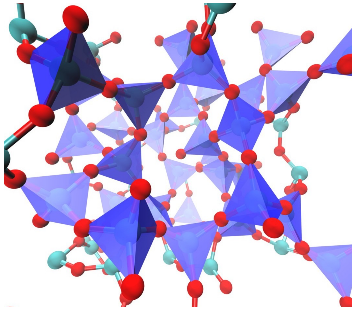
Active learning for accelerated material design

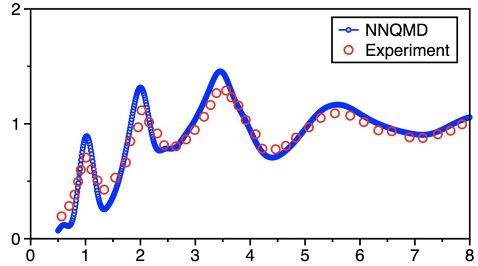


Large-scale and long-time neural network QMD simulations

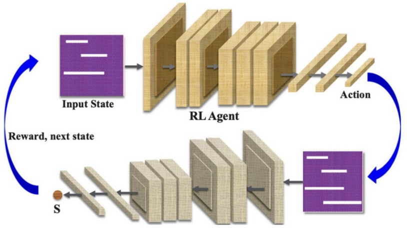


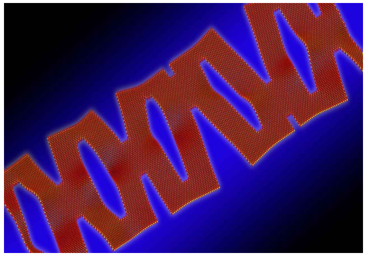


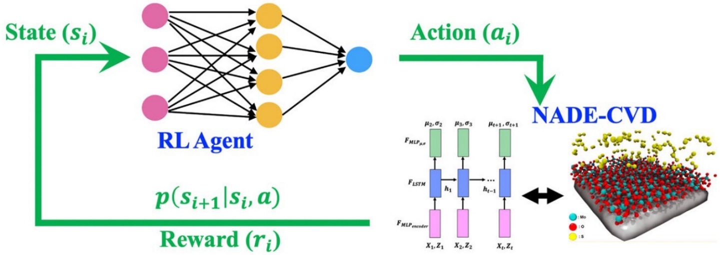




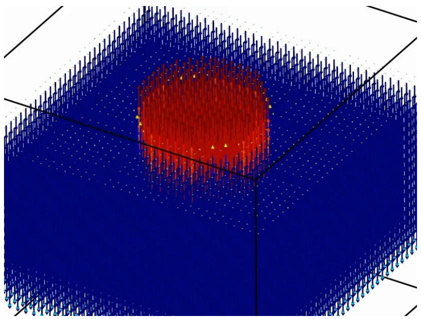
Reinforcement learning for quantum materials synthesis





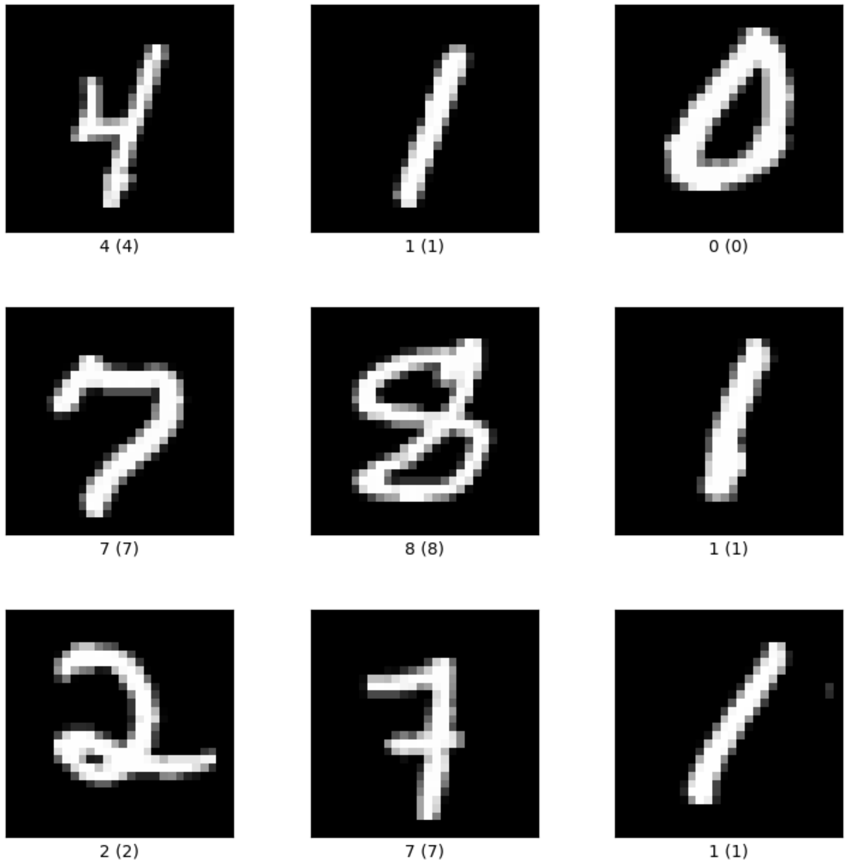


Deep generative model for ferroelectrics



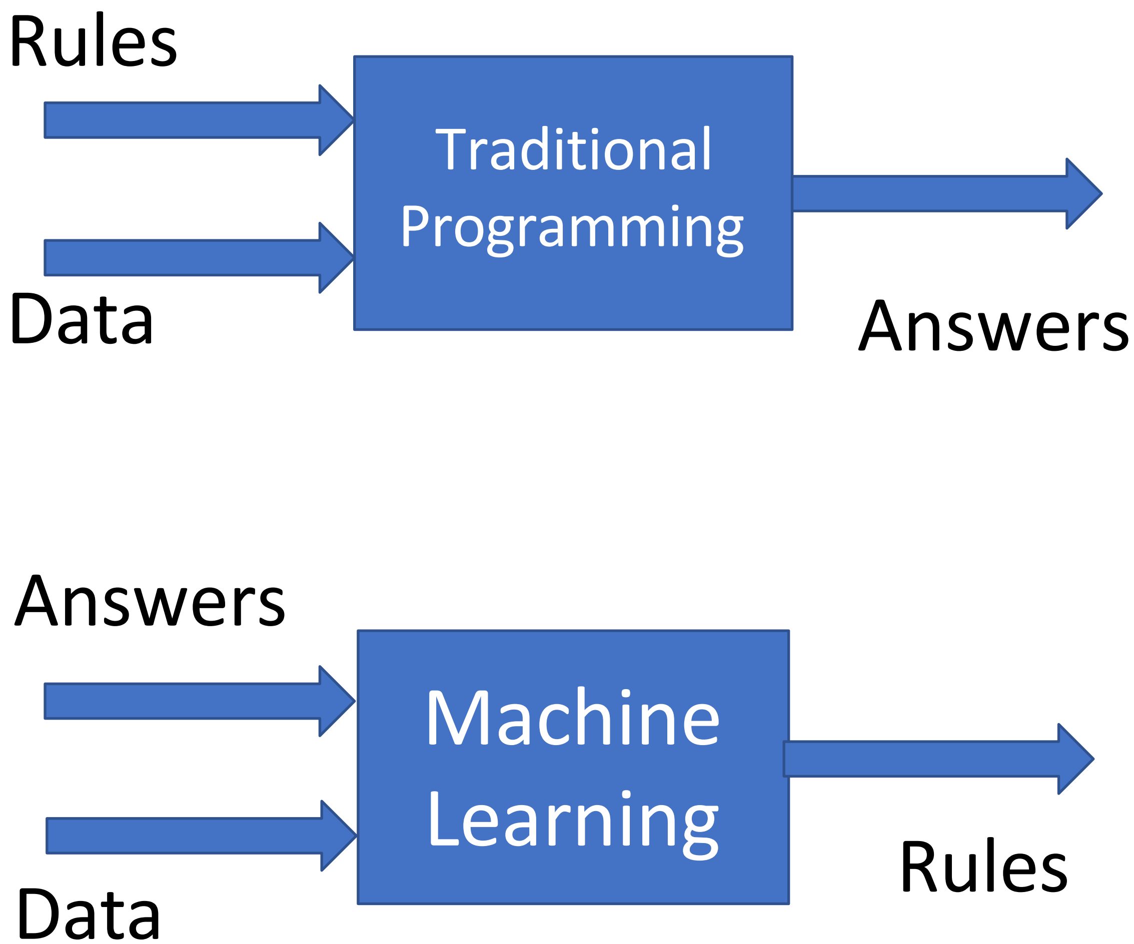
What is Machine Learning?

Image classification using MNIST dataset



What is Machine Learning?

What is Machine Learning?

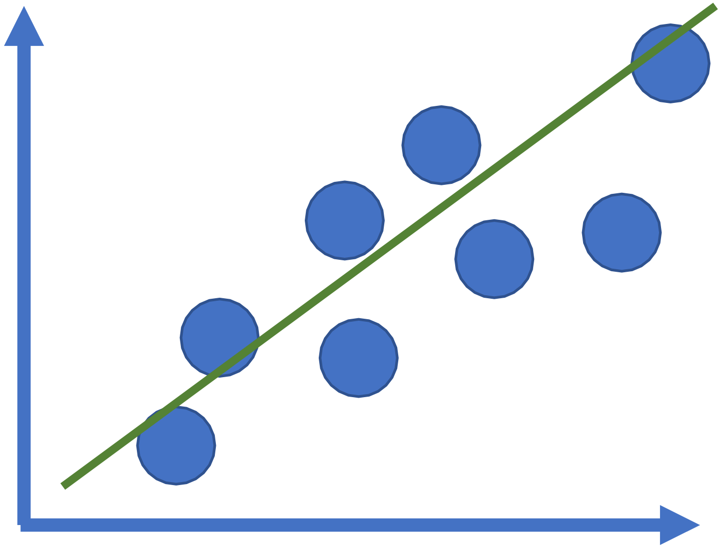


Google I/O 2019

Classification vs Regression

Classification
Predict class label

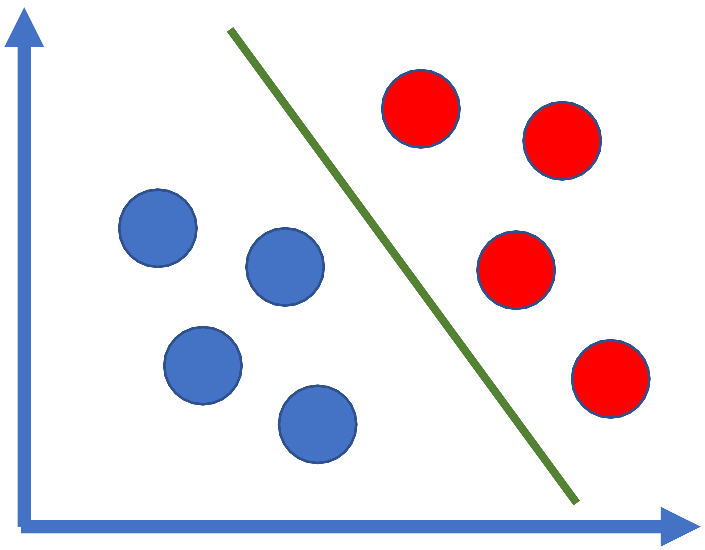
Regression
Predict real value

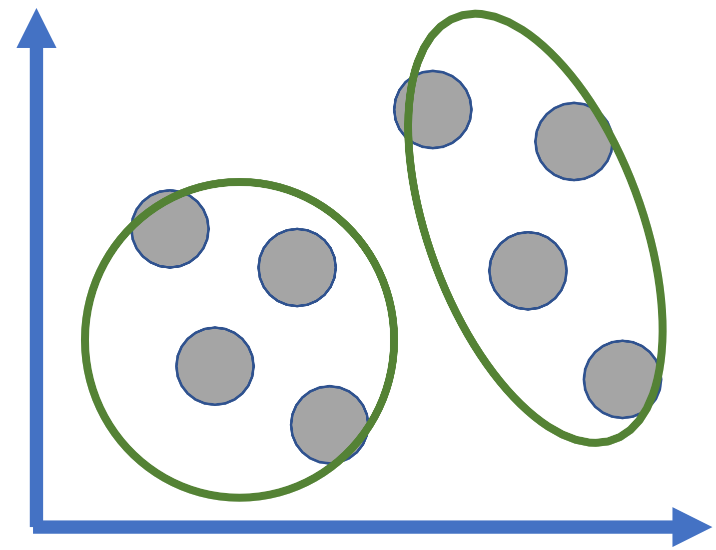


Supervised vs Unsupervised

Supervised:
Data are ”labeled”
classification, regression

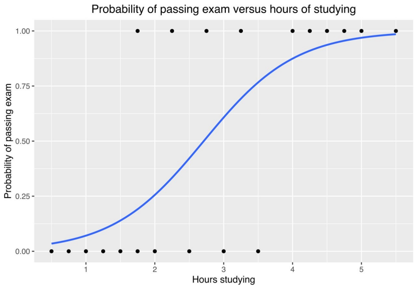
Unsupervised:
Data are “not labeled”
Clustering, dimensionality reduction

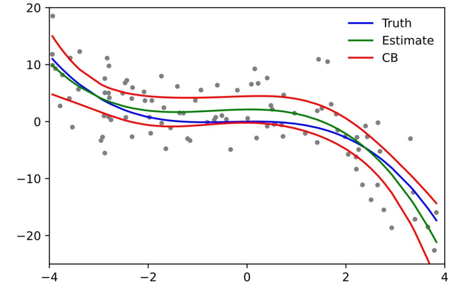


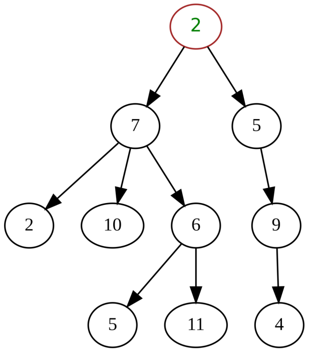


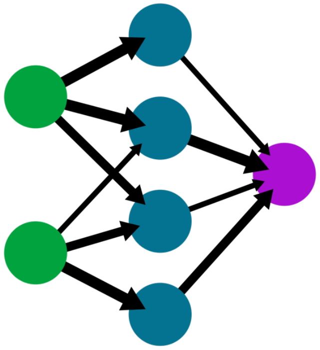
ML Algorithms

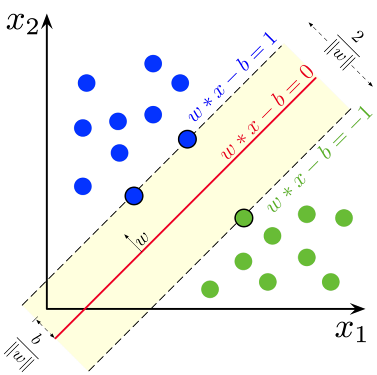
- **Linear/polynomial Regressions**
- **Logistic Regression**
- **K-Nearest Neighbors**
- **Decision Trees**
- **Random Forests**
- **Support Vector Machines**
- **Neural Networks**
- **Bayesian Networks**
- **PCA & t-SNE**

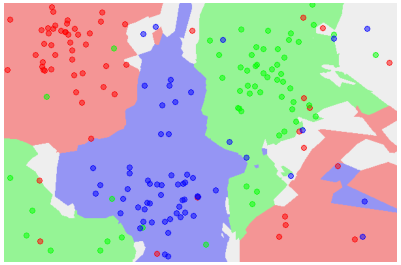








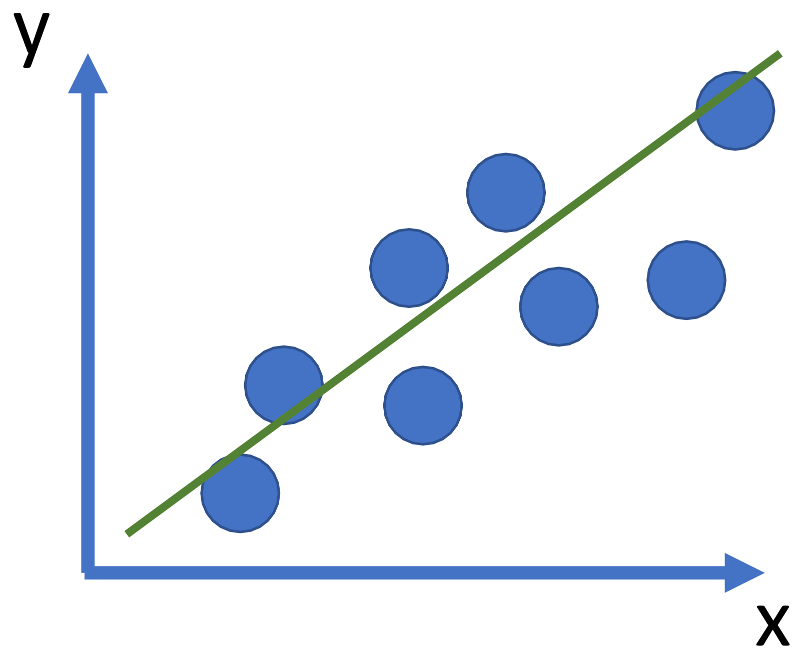




Linear Regression

- **Assumes a linear relationship between the input variable(s) and the output variable (y)**
- **Can be univariate, multivariate, polynomial, logarithmic, …**
- **Coefficients (*b*$_i$) are obtained by minimizing the sum of the difference between all data and line**

$$\hat{y} = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n = \boldsymbol{x}_i^T \boldsymbol{\beta}$$

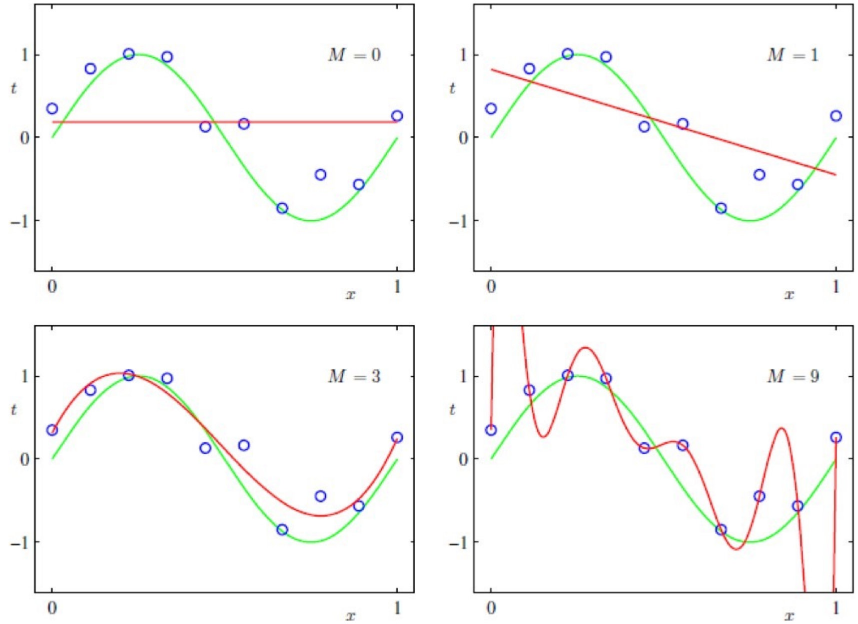


$$L = \frac{1}{N} \Sigma_{i=1}^{N} \left(\widehat{Y}_i - Y_i\right)^2$$

Overfitting and Regularization

- **A good ML model should accurately predict existing training data as well as “unseen” (out-of-sample) data**
- **A model with many parameters tends to pick up noise in data and poorly perform on unseen data, i.e. overfitting**
- **Regularizations, such as Ridge and LASSO**

$$\hat{y} = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n = \boldsymbol{x}_i^T \boldsymbol{\beta}$$

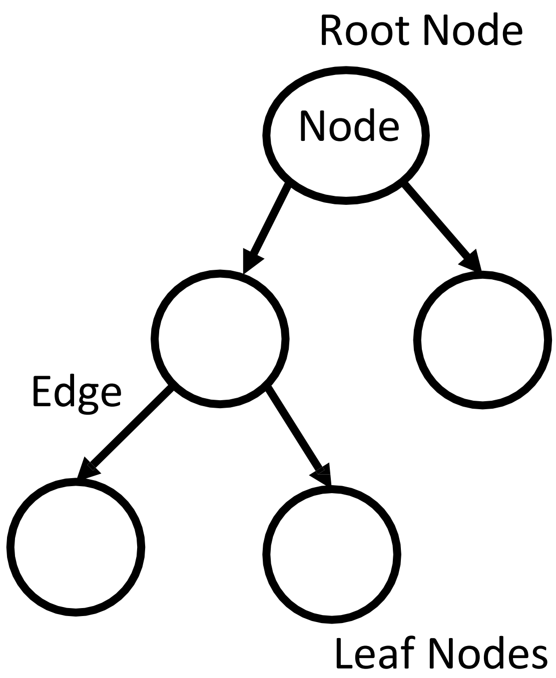


True function
Training data with noise
Model predictions

$$\min_{\beta,\beta_0} \left\{ \frac{1}{N} \|y - X\beta\|^n \right\} \; with \; \|\beta\| \le t \;\; or \;\; \|\beta\|^2 \le t$$

Decision Tree and Random Forest

- **Used for classification or regression**
- **Starting from root node, "ask a question and select an answer" until a leaf node is reached**
- **Tree construction based on information theory**
  - **Gini index/entropy for classification**
  - **Variance/RMSE for regression**
- **Easy to construct and interpret, but also overfit**

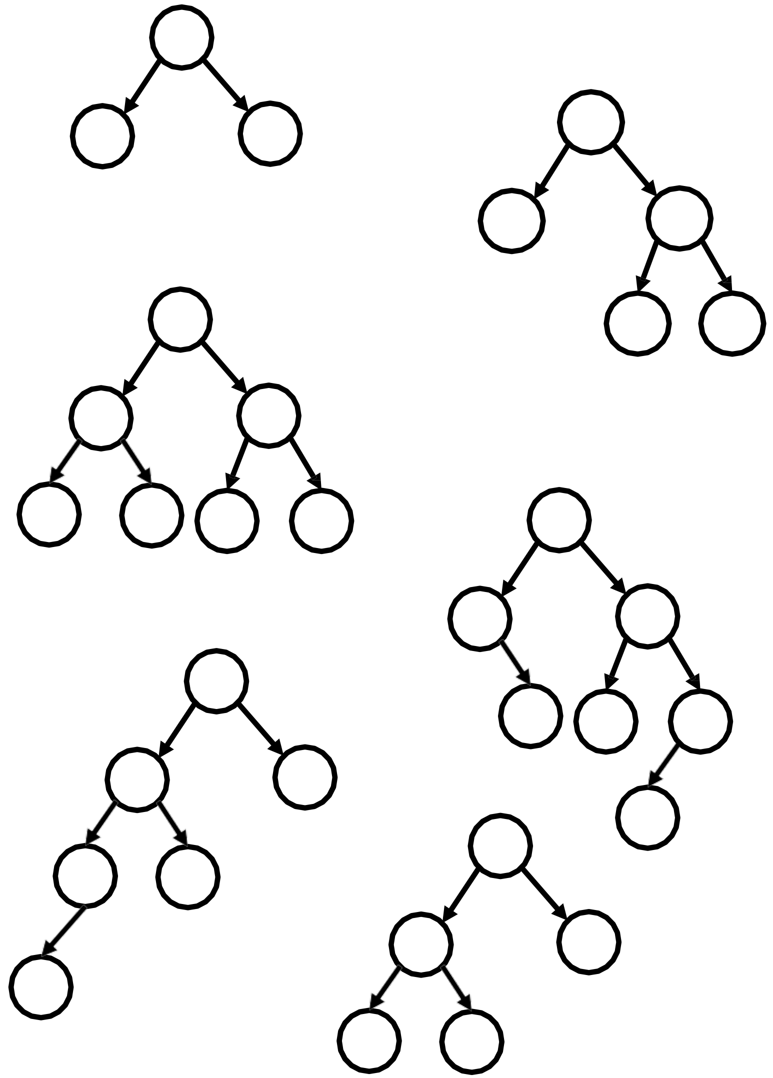


$$I_{Gini} = 1 - \Sigma_j p_j^2$$

$$I_{entropy} = -\Sigma_j p_j \log(p_j)$$

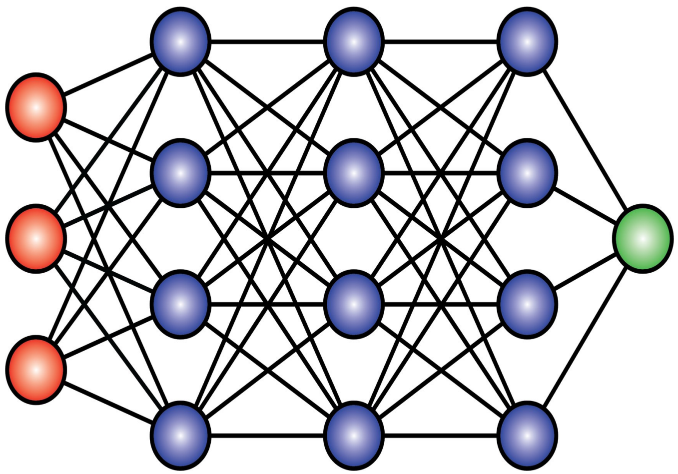
Decision Tree and Random Forest

- Ensemble of decision trees
- Aggregate predictions from each tree as the model prediction
- Good prediction accuracy, generalizability, robust to overfitting
- Less interpretability to single decision tree

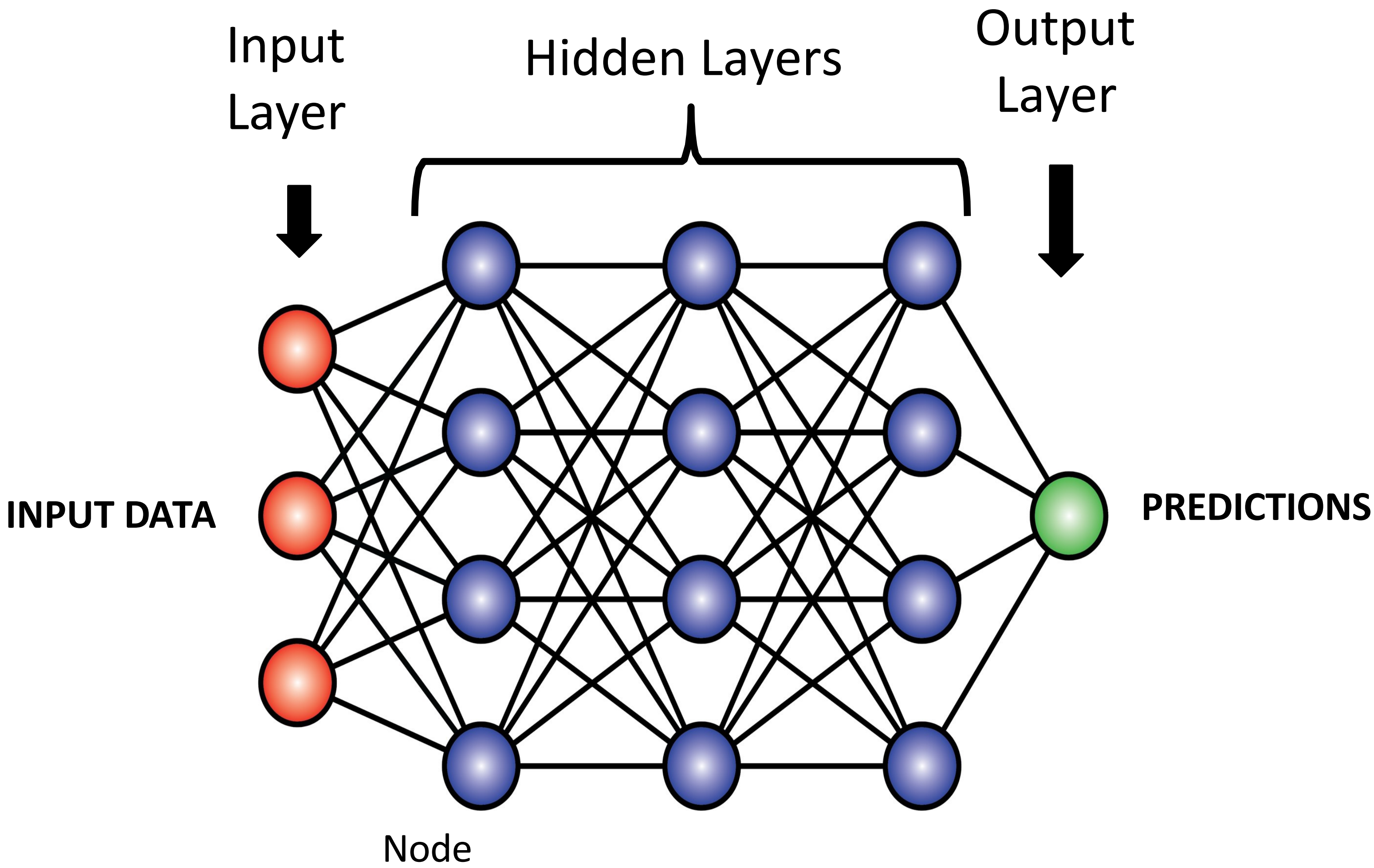


Neural Network

- Inspired by biological brain
- A universal function approximator
- A key component in other deep learning algorithms
- Hyperparameters
  - Number of nodes
  - Degree of connectivity of nodes
  - Number of layers in network

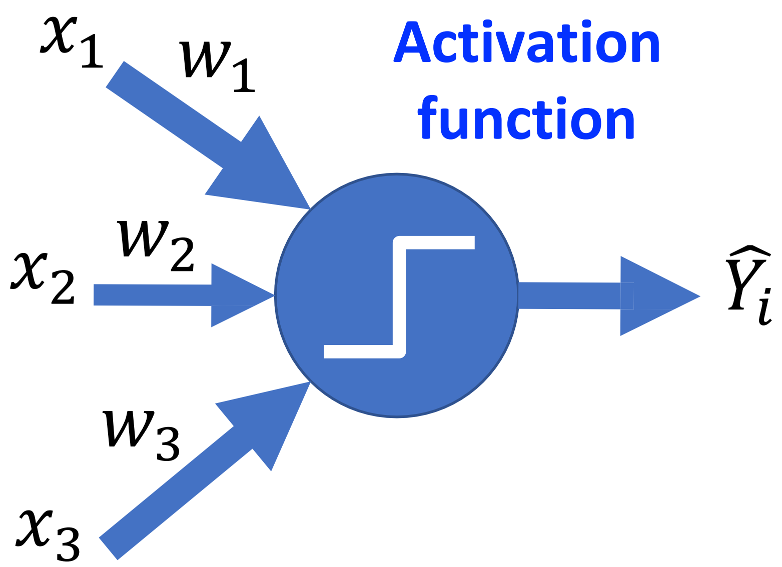


Neural Network



Neural Network

- On each node, outputs (x) from previous layer are aggerated with weights (w)
- A non-linear activation function transforms the aggregated inputs and pass it to next layer
- Compute Loss function (difference between model prediction and given true value) after the output layer

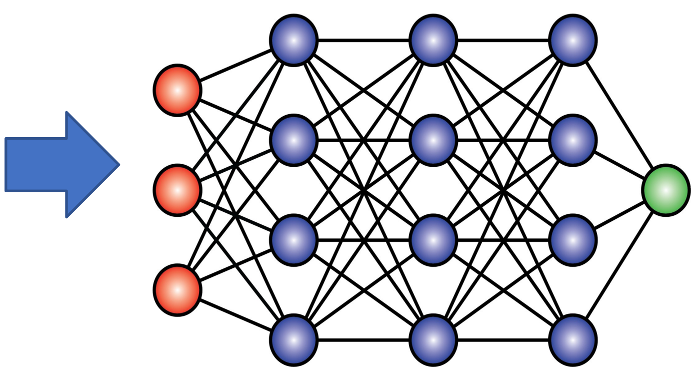


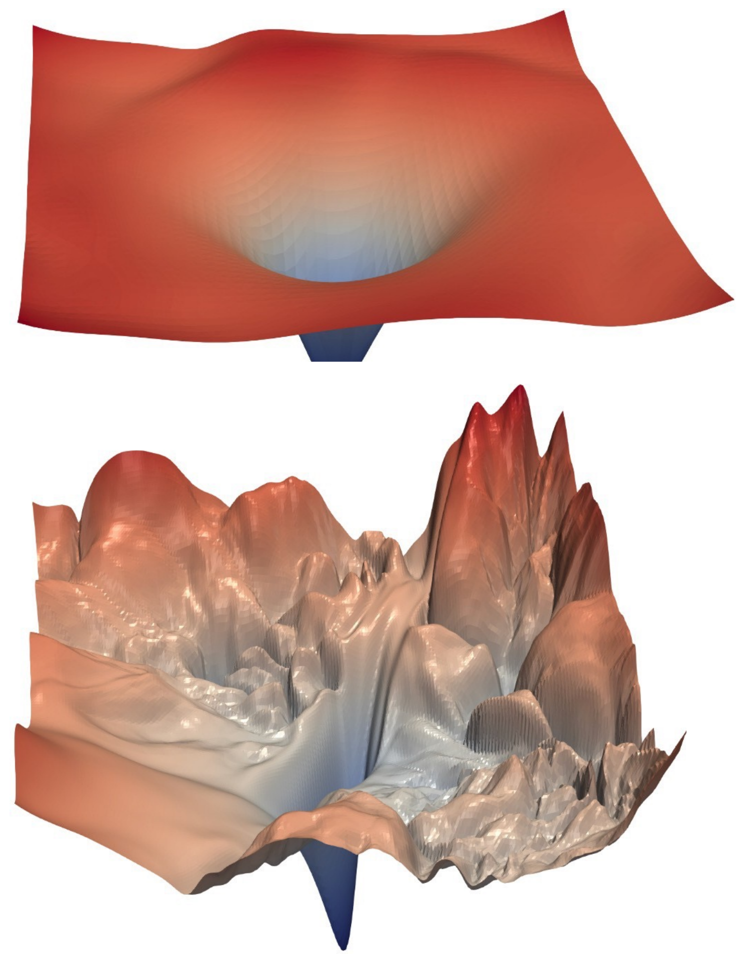
$$L = \frac{1}{N}\Sigma_{i=1}^{N}\left(\widehat{Y}_i - Y_i\right)^2$$

Neural Network Training

- **Network parameters are "trained" by minimizing loss function**
- **Stochastic gradient decent is commonly used**

$$\Delta w = -\partial L/\partial w$$





Loss function landscape

H. Li et al., *"Visualizing the Loss Landscape of Neural Nets,"* arXiv:1712.09913v3

Moving Forward

- **Linear algebra, Statistics & Probability, Python**
- **Online courses**
  https://www.coursera.org/browse/data-science/machine-learning
- **Textbooks**
  Deep Learning
  Ian Goodfellow and Yoshua Bengio and Aaron Courville
  https://www.deeplearningbook.org/
  The Elements of Statistical Learning
  Trevor Hastie, Robert Tibshirani, Jerome Friedman
  https://hastie.su.domains/ElemStatLearn/
- **Python Programming**
  Scikit-learn https://scikit-learn.org
  Pytorch https://pytorch.org
  Tensorflow https://www.tensorflow.org