Machine Learning

A chemometric Perspective

Andreas Baum
Associate Professor
DTU Compute
Statistics and Data Analysis

What is Machine Learning?

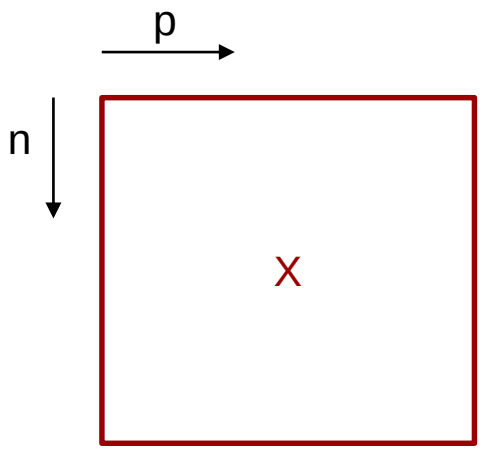
- *Arthur Samuel (1959)*
  Field of study that gives computers the ability to learn without being explicitly programmed.

- *Tom Mitchell (1998)*
  Well-posed learning Problem: A computer program is said to learn from **experience** E with respect to some **task** T and some **performance** measure P, if its performance on T, as measured by P, improves with experience E.

- **Experience** – training iterations/epoch
- **Performance** – Loss function
- **Task** – Learning objective

Machine Learning Paradigms

Unsupervised Learning



- Latent variable space (exploratory data analysis)
- Clustering
- Pattern recognition

Supervised Learning

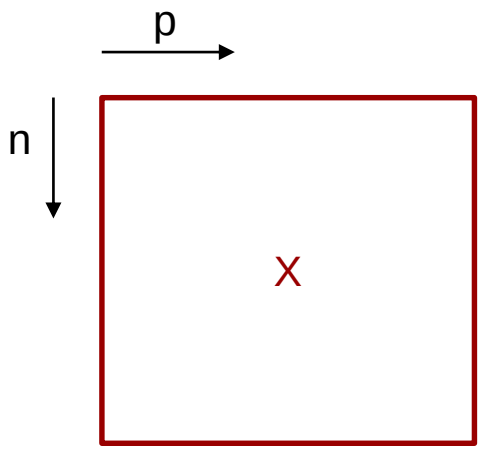
- Classification
- Regression

Reinforcement Learning

- Win a (computer) game
- Learn to perform a task while maximizing reward (sequential data → non i.i.d.)

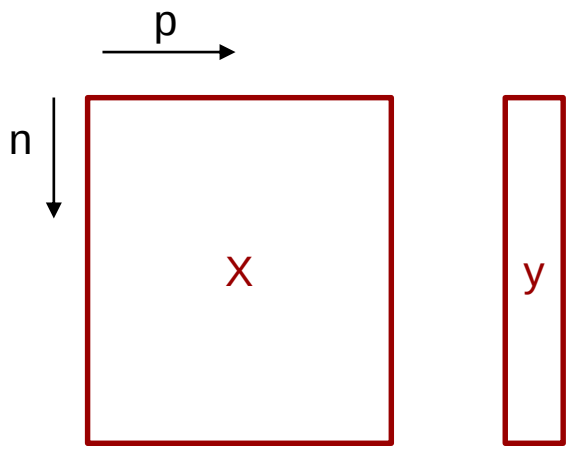
Machine Learning Paradigms

Unsupervised Learning



- Latent variable space (exploratory data analysis)
- Clustering
- Pattern recognition

Supervised Learning



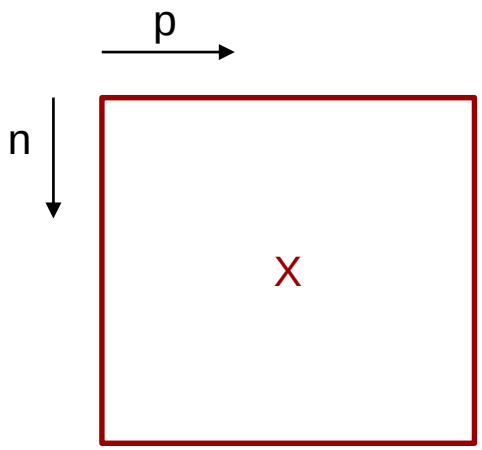
- Classification
- Regression

Reinforcement Learning

- Win a (computer) game
- Learn to perform a task while maximizing reward (sequential data → non i.i.d.)

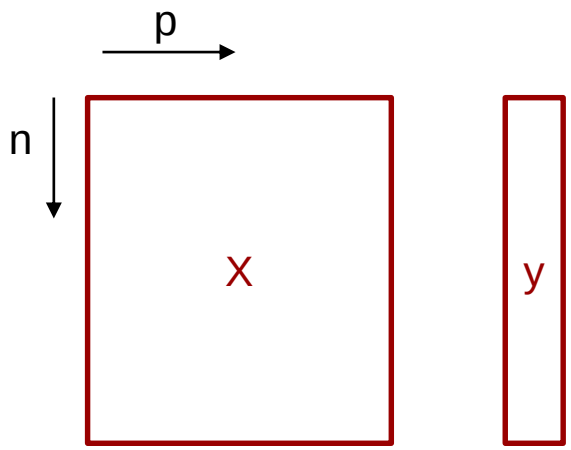
Machine Learning Paradigms

Unsupervised Learning



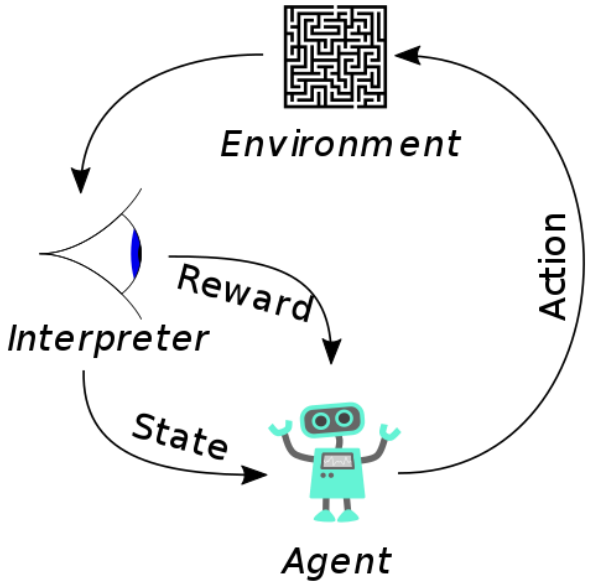
- Latent variable space (exploratory data analysis)
- Clustering
- Pattern recognition

Supervised Learning



- Classification
- Regression

Reinforcement Learning



- Win a (computer) game
- Learn to perform a task while maximizing reward (sequential data → non i.i.d.)

BIG versus SMALL DATA

Small Data

- ”chemometrics case”
- n << p
- OLS, Logistic regression,
  Ridge, Lasso
  PCA, PLS, PARAFAC, TUCKER
- Better model interpretability
- Fewer parameters
- Data pre-processing crucial

BIG versus SMALL DATA

Small Data

- ”chemometrics case”
- n << p
- OLS, Logistic regression, Ridge, Lasso
  PCA, PLS, PARAFAC, TUCKER
- Better model interpretability
- Fewer parameters
- Data pre-processing crucial

Big Data

- ”google case”
- n >> p
- Decision tree based ensembles (BRT, RF)
- ANNs, CNNs, RNNs
- Black boxes
- Many parameters
- End-to-end modeling

BIG versus SMALL DATA

Linearity assumption

Small Data

- ”chemometrics case”
- n << p
- OLS, Logistic regression, Ridge, Lasso
  PCA, PLS, PARAFAC, TUCKER
- Better model interpretability
- Fewer parameters
- Data pre-processing crucial

Big Data

- ”google case”
- n >> p
- Decision tree based ensembles (BRT, RF)
- ANNs, CNNs, RNNs
- Black boxes
- Many parameters
- End-to-end modeling

Handle non-linearity implicitly

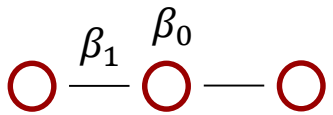
Bias / Variance trade-off (Regularization, Model Validation)

Let’s use a ANN approach to estimate model parameters for …

- Simple Linear Regression
- Multiple Linear Regression
- Logistic Regression
- Polynomial Regression
- PCA
- PLS

Simple Linear Regression

$$y_i = \beta_0 + \beta_1 x_i + e_i \quad , \quad i = 1 \ldots n$$



input layer (x_i) | hidden layer | output layer (y_i)

1. Choose random parameters β_0 and β_1
2. Forward pass (”Apply network”)
3. Backpropagate and update parameters using Gradient Descent
4. Repeat steps 2 and 3 until convergence

Objective (T)

$$\underset{\beta_0,\beta_1}{\operatorname{argmin}} \|\boldsymbol{e}\|_2^2$$

Loss (P)

$$\mathrm{L}(\beta_0, \beta_1) = \frac{\|\boldsymbol{e}\|_2^2}{n}$$

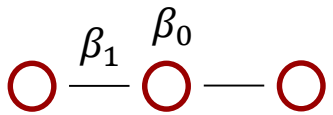
$$\beta_0 := \beta_0 - \gamma \frac{\partial}{\partial \beta_0} L(\beta_0, \beta_1)$$

$$\beta_1 := \beta_1 - \gamma \frac{\partial}{\partial \beta_1} L(\beta_0, \beta_1)$$

Learning rate

Simple Linear Regression

$$y_i = \beta_0 + \beta_1 x_i + e_i \quad , \quad i = 1 \ldots n$$



input layer (x_i) | hidden layer | output layer (y_i)

Objective (T)

$$\underset{\beta_0,\beta_1}{\operatorname{argmin}} \|\boldsymbol{e}\|_2^2$$

Loss (P)

$$\mathrm{L}(\beta_0, \beta_1) = \frac{\|\boldsymbol{e}\|_2^2}{n}$$

1. Choose random parameters β_0 and β_1
2. Forward pass (”Apply network”)
3. Backpropagate and update parameters using Gradient Descent
4. Repeat steps 2 and 3 until convergence

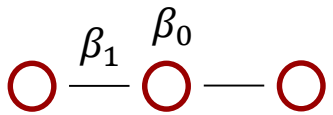
$$\beta_0 := \beta_0 - \gamma \frac{\partial}{\partial \beta_0} L(\beta_0, \beta_1)$$

$$\beta_1 := \beta_1 - \gamma \frac{\partial}{\partial \beta_1} L(\beta_0, \beta_1)$$

Learning rate

Simple Linear Regression

$$y_i = \beta_0 + \beta_1 x_i + e_i \quad , \quad i = 1 \ldots n$$



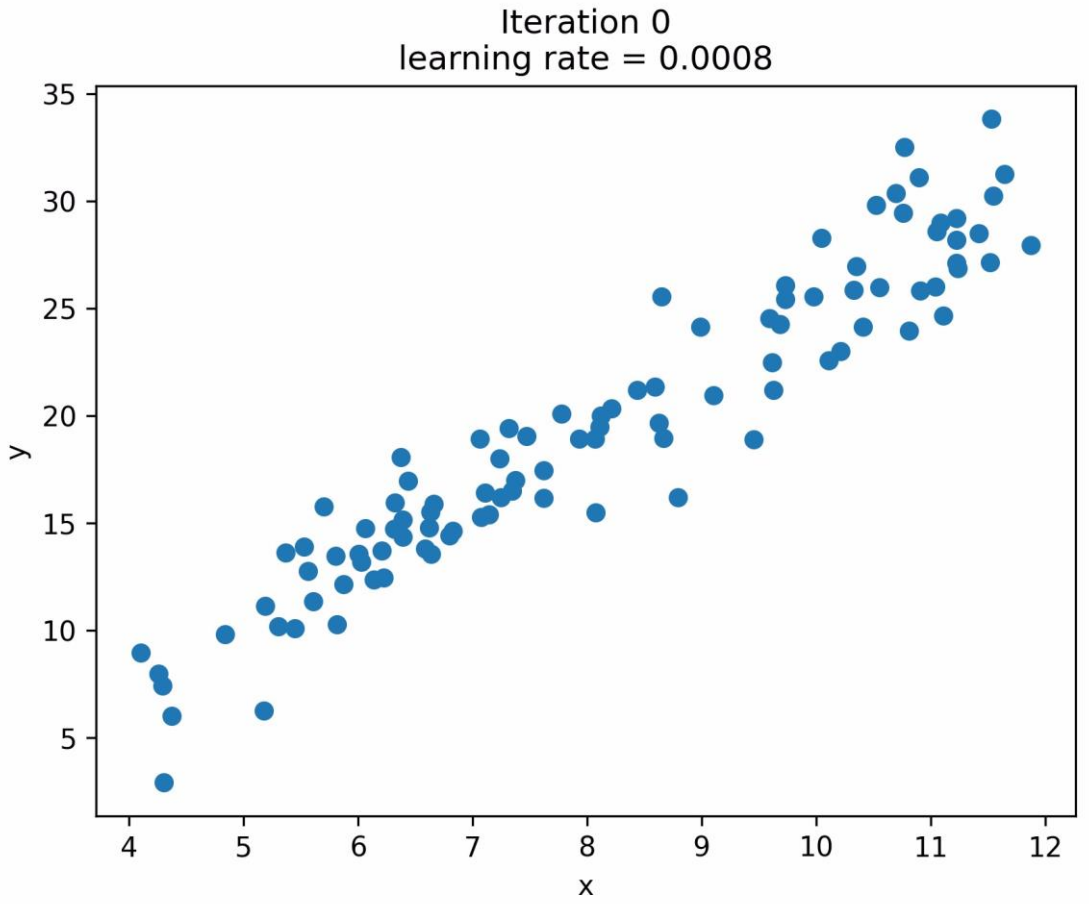
input layer (x_i) | hidden layer | output layer (y_i)

Objective (T)

$$\underset{\beta_0,\beta_1}{\operatorname{argmin}} \|\boldsymbol{e}\|_2^2$$

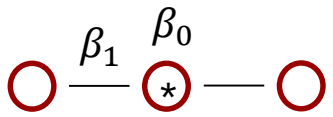
Loss (P)

$$\mathrm{L}(\beta_0, \beta_1) = \frac{\|\boldsymbol{e}\|_2^2}{n}$$



Logistic Regression

$$y_i = g(\beta_0 + \beta_1 x_i) + e_i \quad , \quad i = 1 \ldots n$$



input layer (x_i) | hidden layer | output layer (y_i)

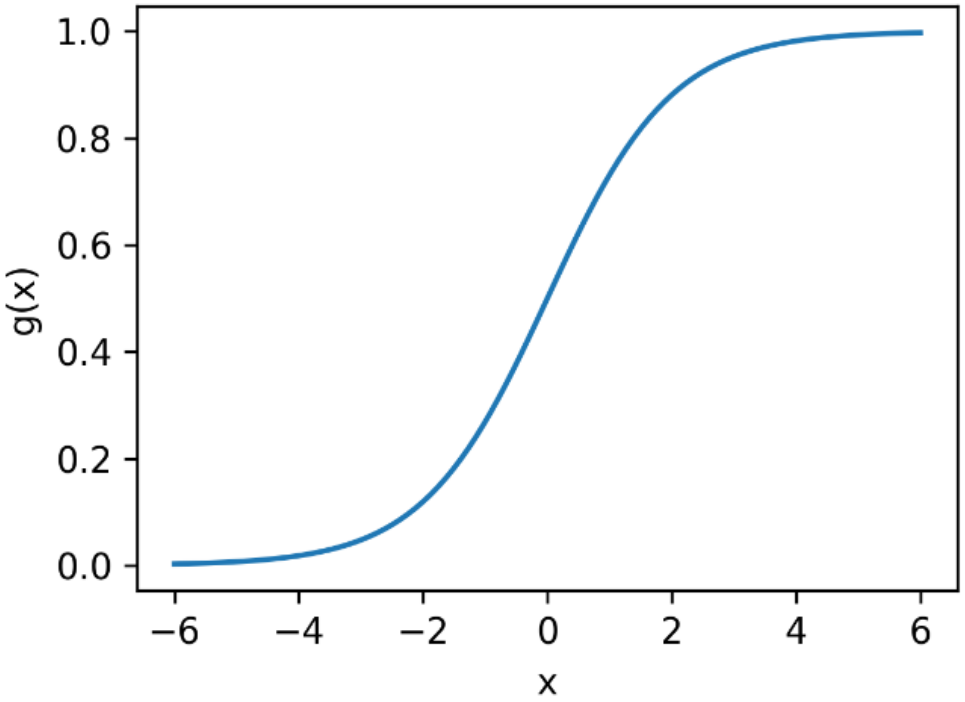
Objective (T)

$$\underset{\beta_0,\beta_1}{\operatorname{argmin}} \|\boldsymbol{e}\|_2^2$$

Loss (P)

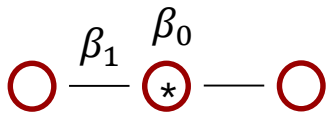
$$\mathrm{L}(\beta_0, \beta_1) = \frac{\|\boldsymbol{e}\|_2^2}{n}$$

* Activation function (sigmoidal)



Logistic Regression

$$y_i = g(\beta_0 + \beta_1 x_i) + e_i \quad , \quad i = 1 \ldots n$$



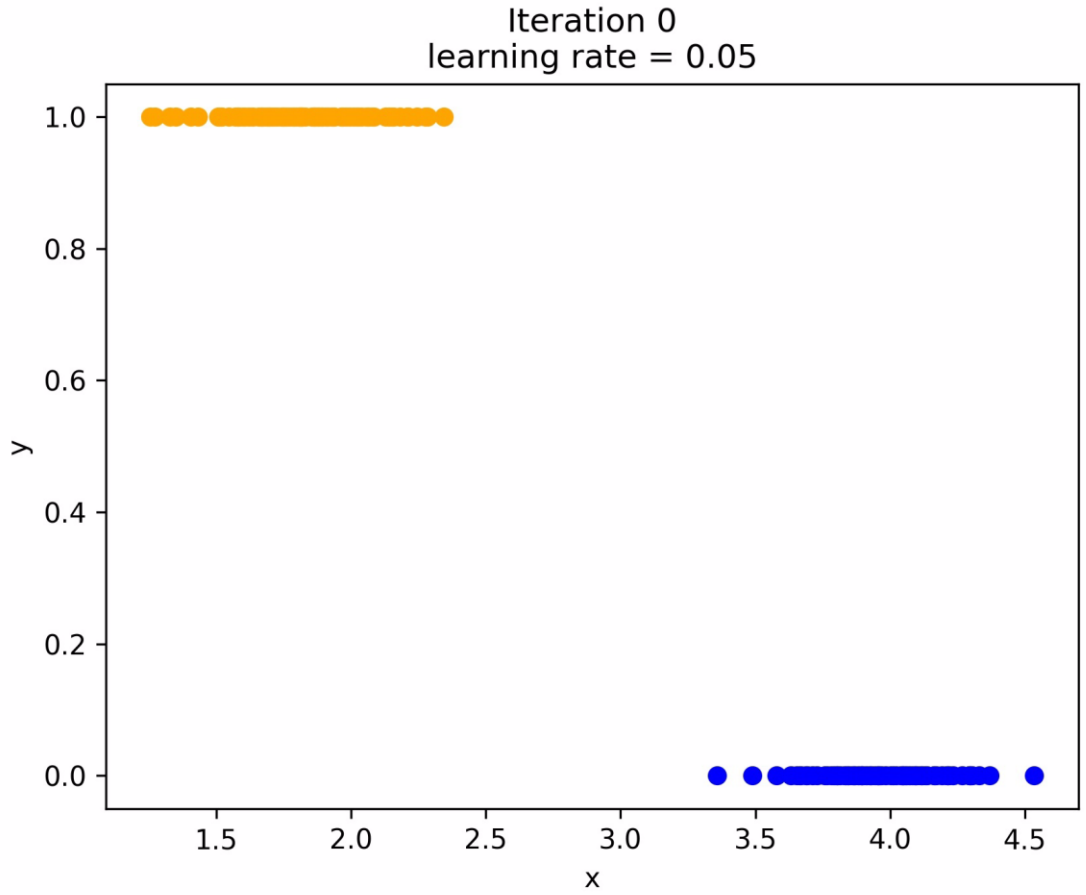
input layer (x_i) | hidden layer | output layer (y_i)

Objective (T)

$$\underset{\beta_0,\beta_1}{\operatorname{argmin}} \|\boldsymbol{e}\|_2^2$$

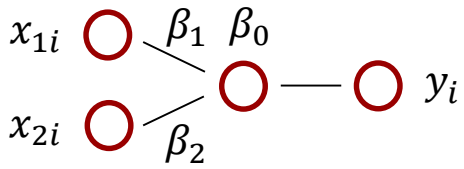
Loss (P)

$$\mathrm{L}(\beta_0, \beta_1) = \frac{\|\boldsymbol{e}\|_2^2}{n}$$



Multiple Linear Regression (MLR)

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + e_i \quad , \quad i = 1 \ldots n$$



input layer | hidden layer | output layer

Objective (T)

$$\underset{\beta_0,\beta_1}{\operatorname{argmin}} \|\boldsymbol{e}\|_2^2$$

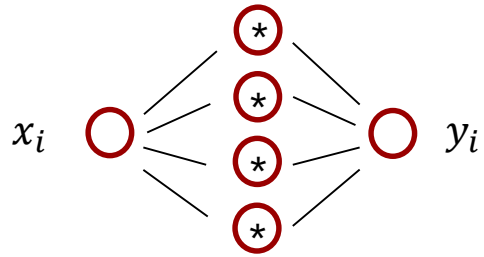
Loss (P)

$$\mathrm{L}(\beta_0, \beta_1) = \frac{\|\boldsymbol{e}\|_2^2}{n}$$

Polynomial Regression

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \beta_4 x_i^4 + e_i$$

$$i = 1 \ldots n$$

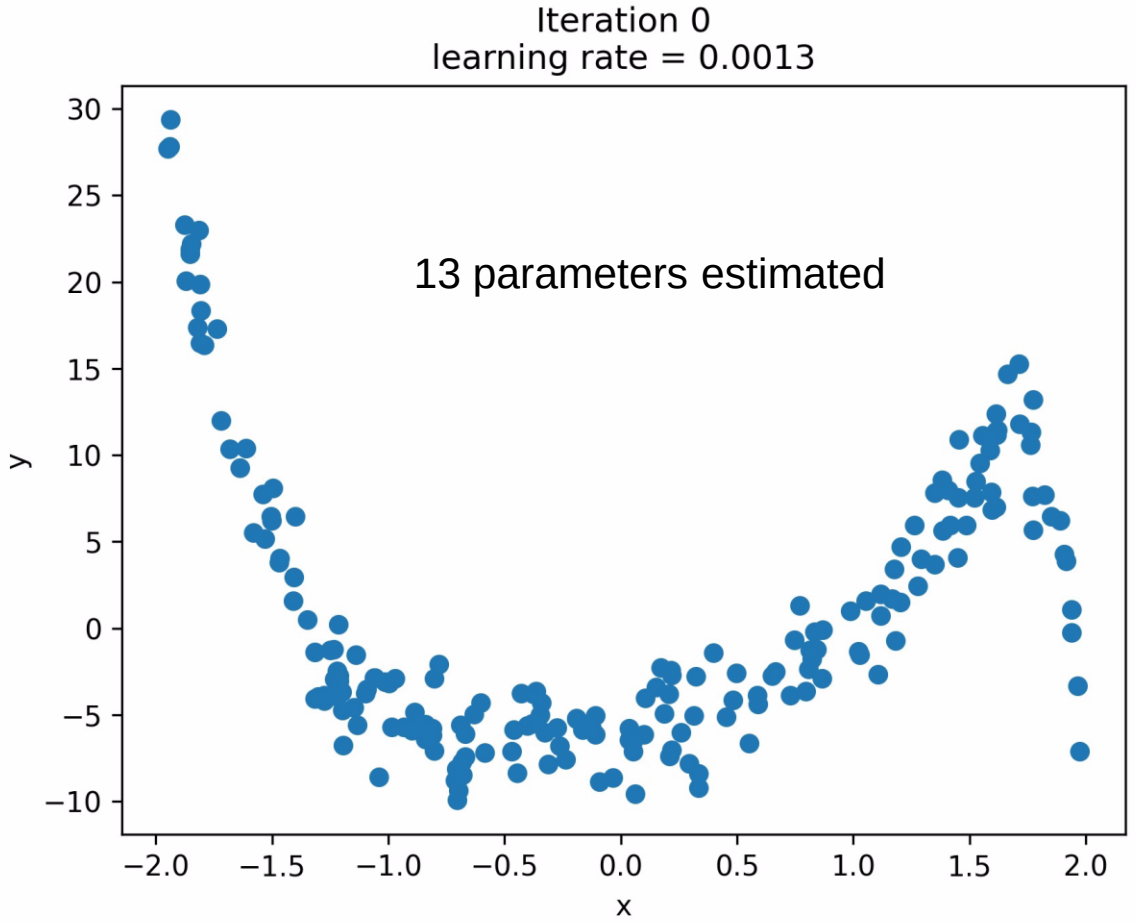


Objective (T)

$$\underset{\beta_0,\beta_1}{\operatorname{argmin}} \|\boldsymbol{e}\|_2^2$$

Loss (P)

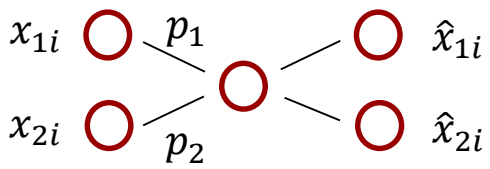
$$\mathrm{L}(\beta_0, \beta_1) = \frac{\|\boldsymbol{e}\|_2^2}{n}$$



Principal Component Analysis (PCA)

$$\boldsymbol{X} = \boldsymbol{t}\boldsymbol{p}^T + \boldsymbol{E} \qquad \boldsymbol{X} \in \mathbb{R}^{n \times 2}$$

$$i = 1 \ldots n$$



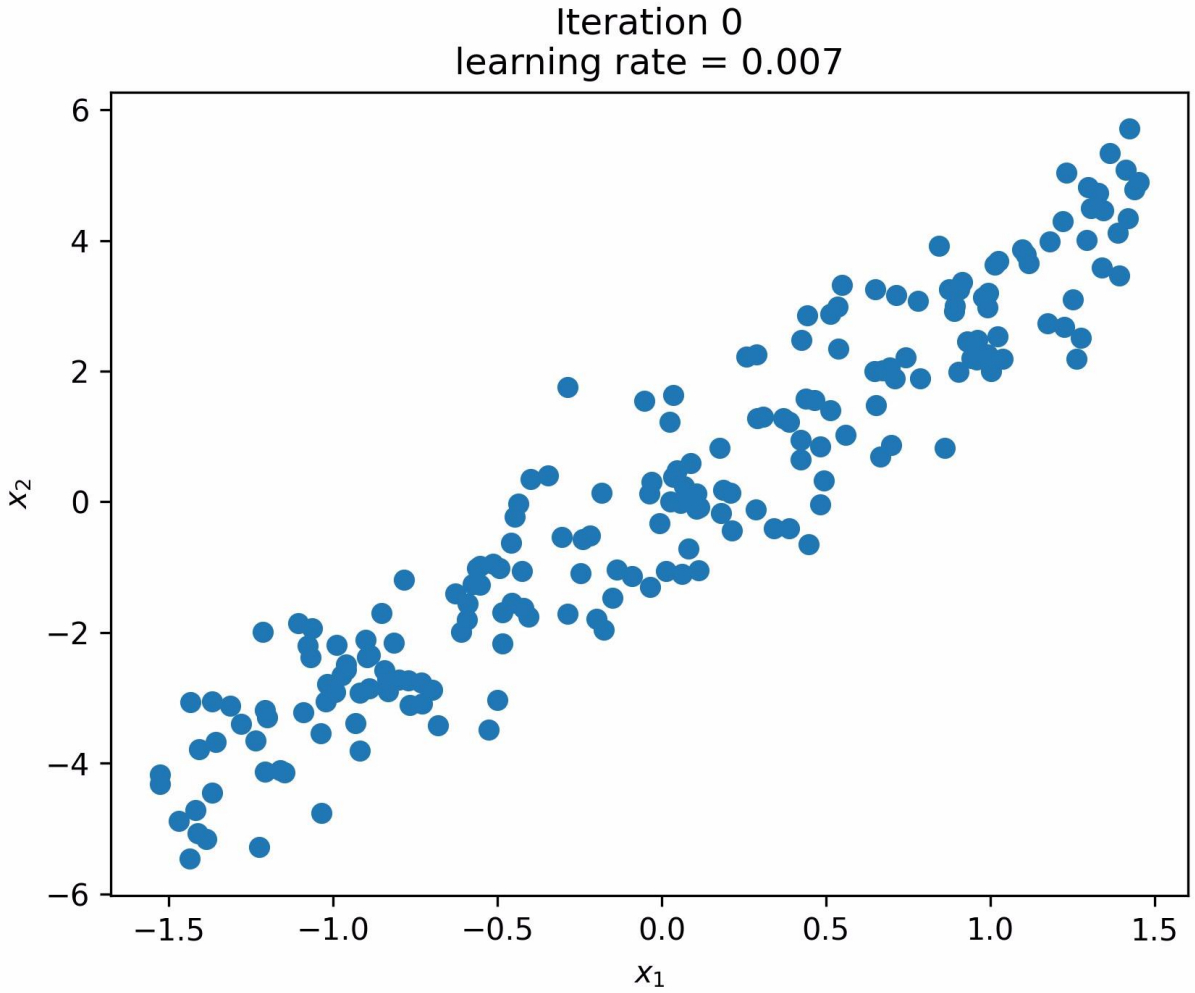
input layer | hidden layer | output layer

Objective (T)

$$\underset{p_1,p_2}{\operatorname{argmin}} \left\| \boldsymbol{X} - \hat{\boldsymbol{X}} \right\|_2^2$$

Loss (P)

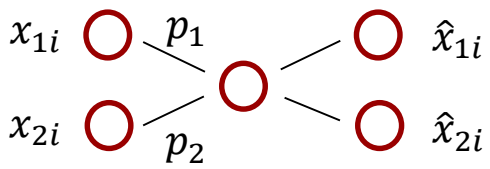
$$\mathrm{L}(p_1, p_2) = \frac{\|\boldsymbol{E}\|_2^2}{2n}$$



Patial Least Squares Regression (PLS)

$$\boldsymbol{X} = \boldsymbol{t}\boldsymbol{p}^T + \boldsymbol{E} \qquad \boldsymbol{X} \in \mathbb{R}^{n \times 2}$$

$$i = 1 \ldots n$$

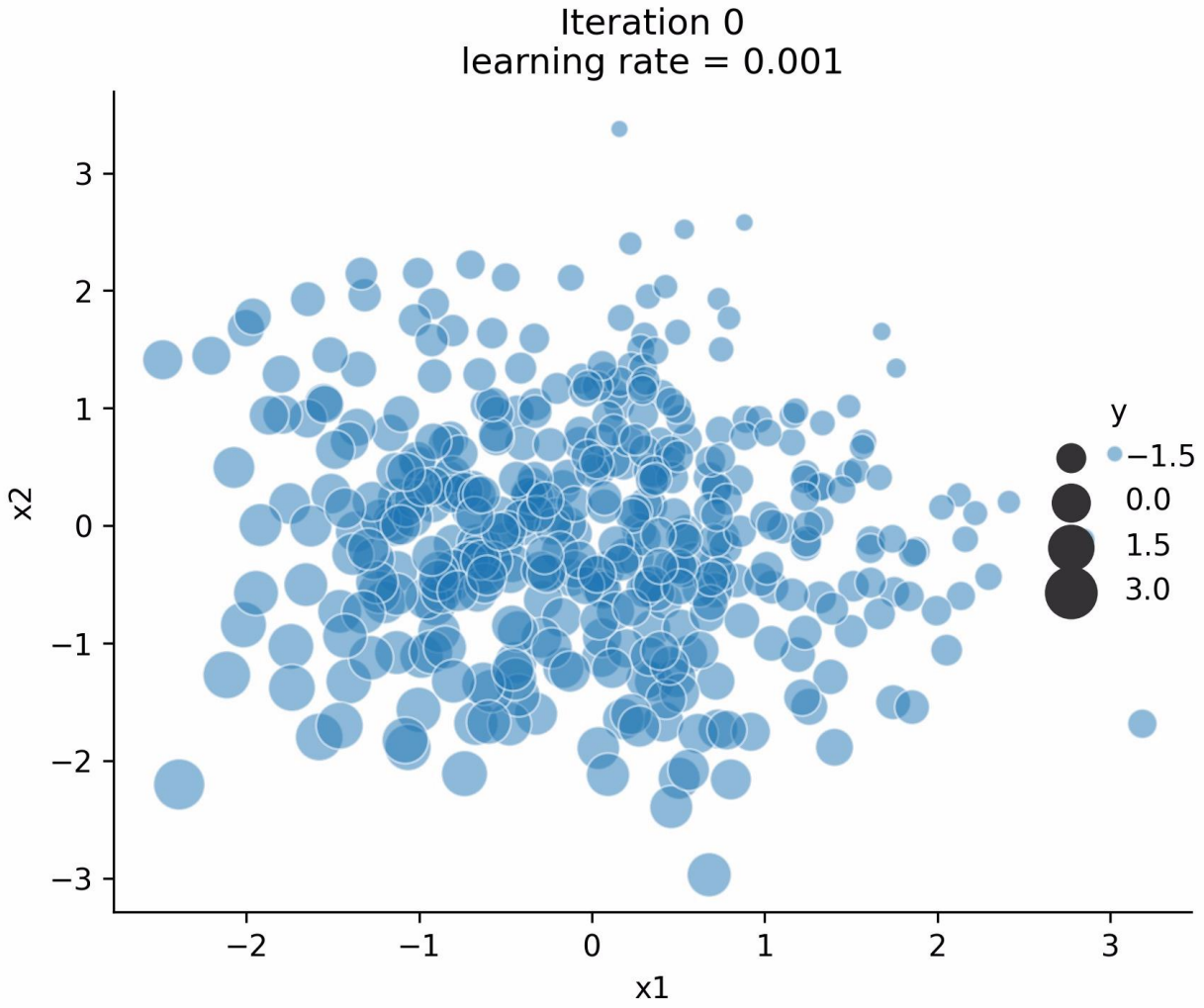


Objective (T)

$$\underset{p_1,p_2}{\operatorname{argmax}}\, cov(\boldsymbol{t}, \boldsymbol{y})$$

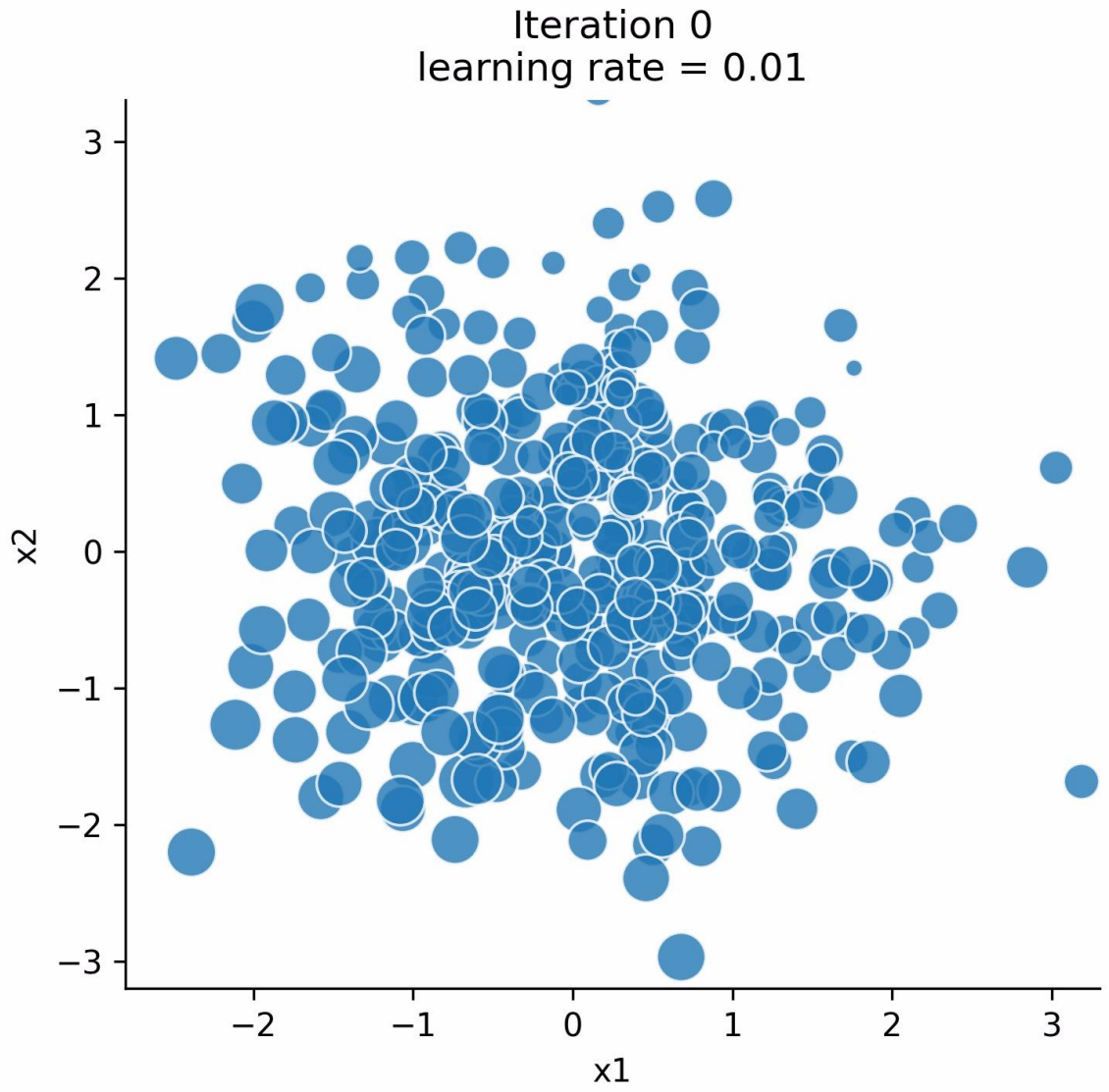
Loss (P)

$$L(p_1, p_2) = -\boldsymbol{t}^T \boldsymbol{y}$$



Stochastic Gradient Descent – PLS example

- Train over randomized batches
- Can be mapped to several workers (parallel computing)
- get out of local minima (non-convex problems)
- Handle very large amounts of data



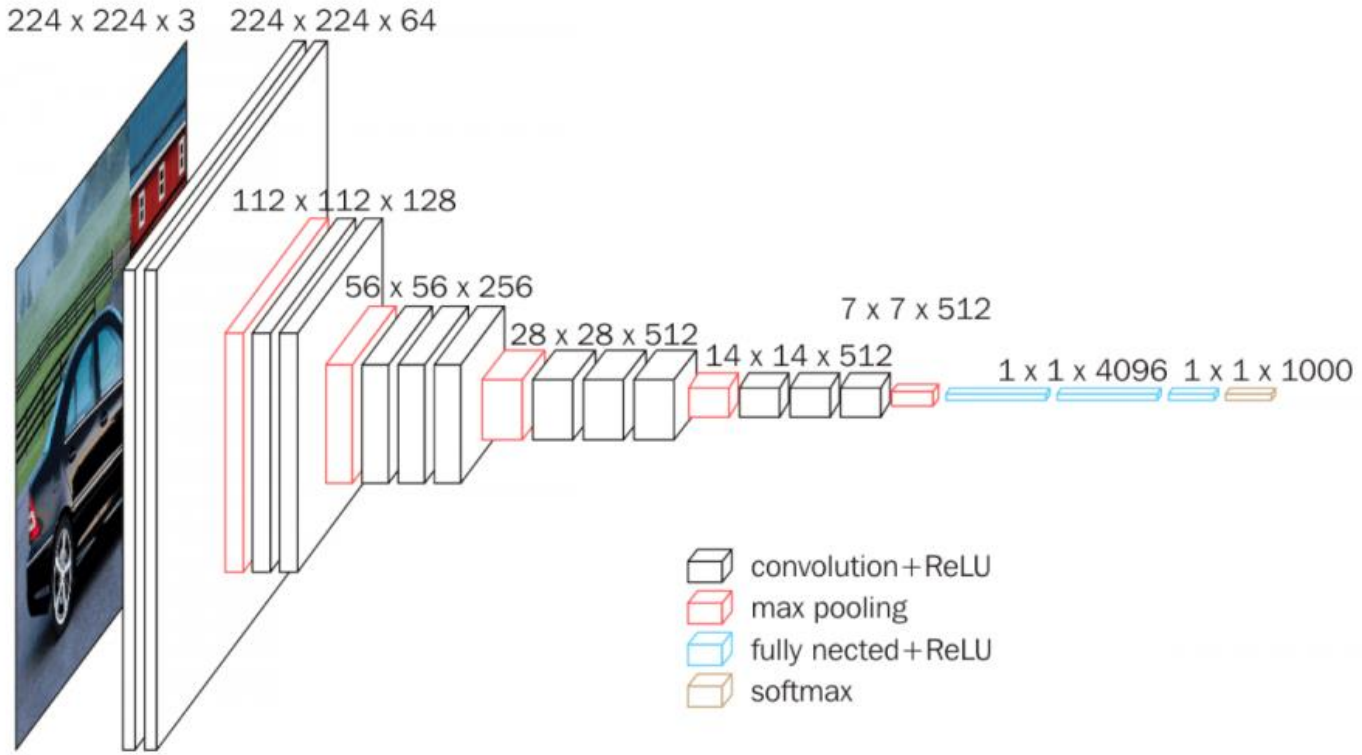
Popular Deep Learning Applications

- Natural Language processing
  - Translation
  - Sentiment analysis
  - Topic modeling
  - Text-to-speech
- Computer vision
  - Face recognition
  - Object detection
  - Image classification
  - Autonomous driving

Popular Deep Learning Applications

VGG16 pre-trained model

- 92.7% test accuracy classifying ImageNet data, which contains over 14 million images belonging to 1000 classes
- Image source: https://arxiv.org/pdf/1409.1556.pdf



Deep Learning Applications in Chemometrics?

Let’s see what the day brings…

Thank you for your attention!