Geospatial Data Science
Content Block II: *Techniques*
Lecture 8
Machine learing for geospatial data

Austin J. Brockmeier, Ph.D.

Monday, April 2nd, 2023

Outline

A one-class introduction to machine learning

Lab 8: scikit-learn, classification, convolutional neural networks

By The scikit-learn developers - github.com/scikit-learn/scikit-learn/blob/master/doc/logos/scikit-learn-logo.svg, BSD, https://commons.wikimedia.org/w/index.php?curid=71445288

Modeling

Predicting crop production:

Annual precipitation

Temperature

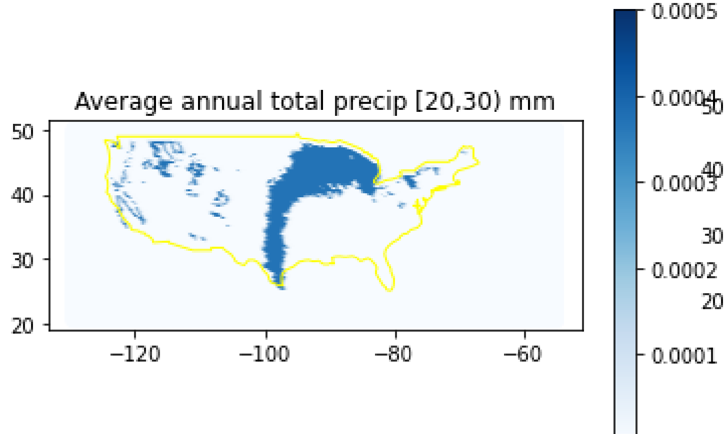
Soil type

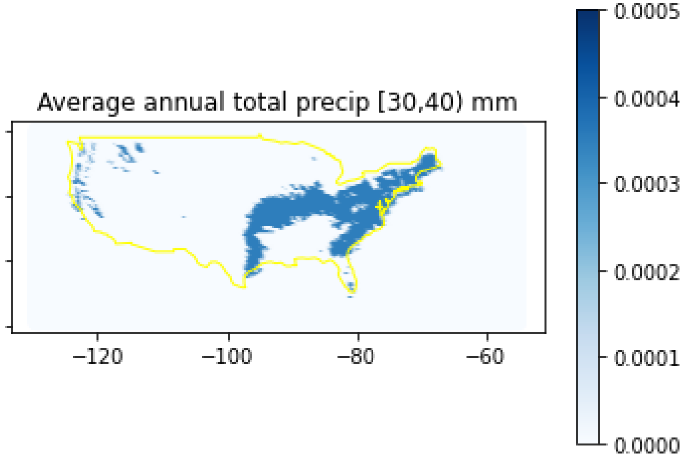
Topography

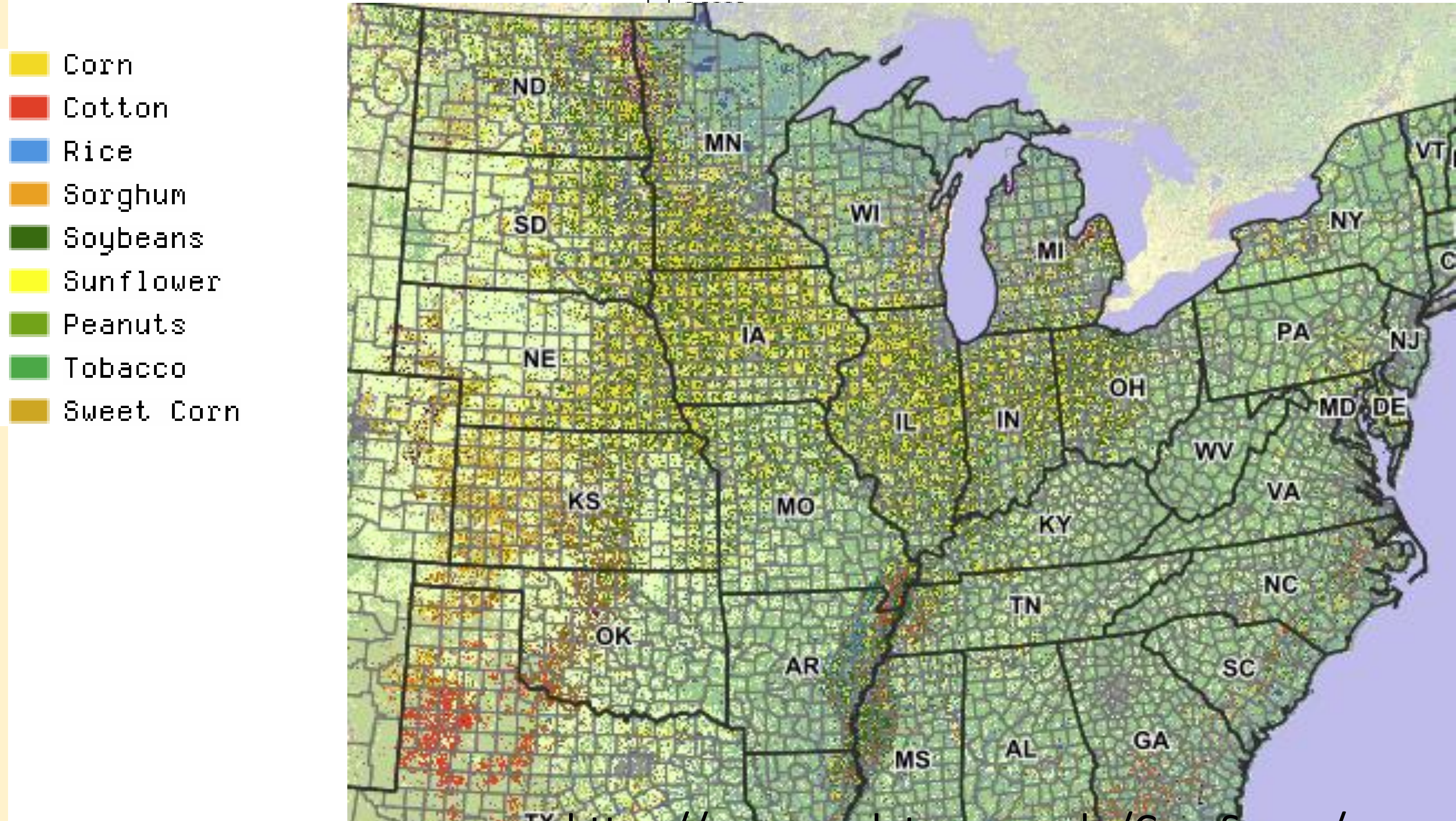
Sunlight

Latitude

Population density





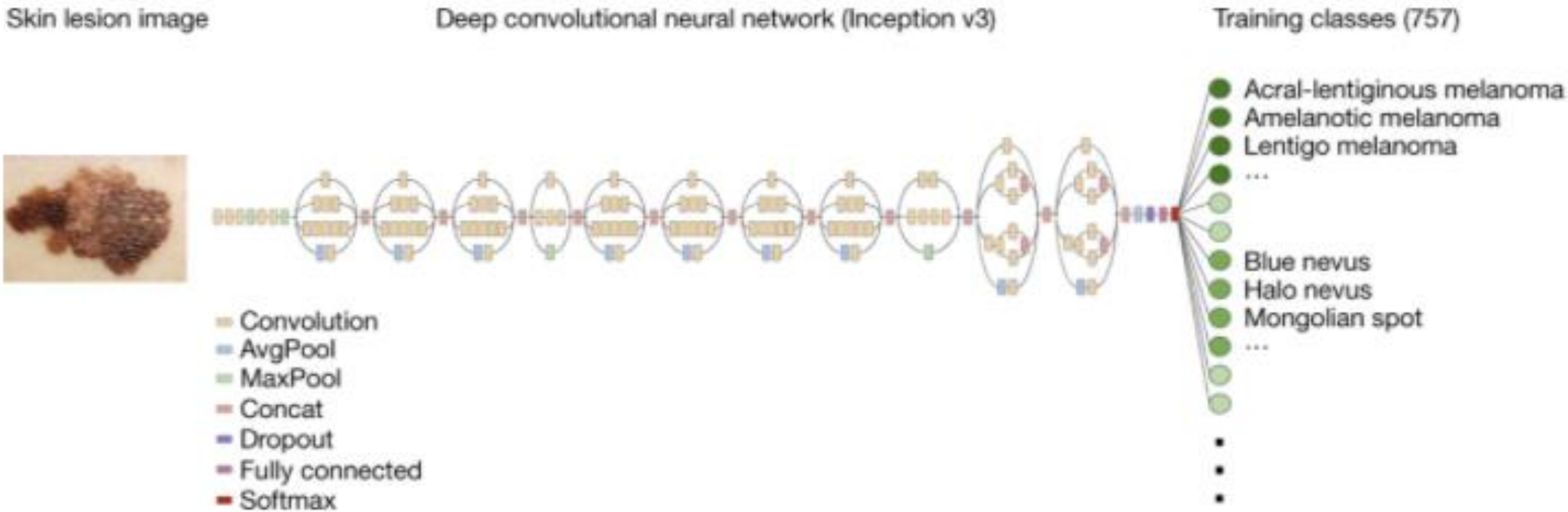


https://nassgeodata.gmu.edu/CropScape/

Esteva et al. "**Dermatologist-level classification of skin cancer with deep neural networks**", *Nature* 2017. https://doi.org/10.1038/nature21056

Dermatologist-level classification of skin cancer with deep neural networks

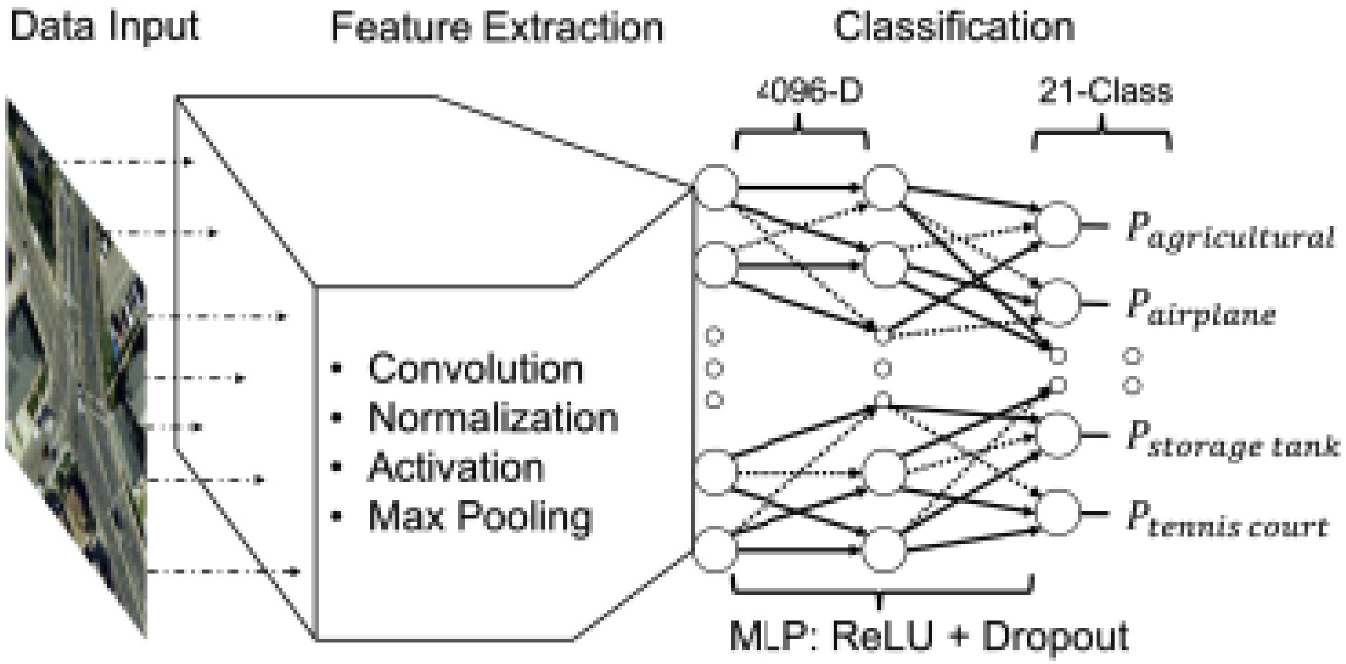
"used *129,450* clinical images of skin disease to train a deep convolutional neural network to classify skin lesions. […] accuracy of the system […] matched that of trained dermatologists."



Esteva et al. "Dermatologist-level classification of skin cancer with deep neural networks", *Nature* 2017.
https://doi.org/10.1038/nature21056

Avoiding starting from scratch: Transfer learning with fine-tuning and data augmentation for easy training.

"UC Merced data set to achieve the land–cover classification accuracies of 97.8 ± 2.3%, 97.6 ± 2.6%, and 98.5 ± 1.4% with CaffeNet, GoogLeNet, and ResNet, respectively."



Scott et al. "Training Deep Convolutional Neural Networks for Land–Cover Classification of High-Resolution Imagery" *IEEE Geoscience and Remote Sensing Letters,* 2017

Machine learning is useful for modeling relationships between patterns (visual patterns, attribute patterns, spatial patterns, temporal patterns, etc.)

Abstract examples:

- Predicting an attribute based on other attributes
- Predicting an attribute at a particular location based on other observations at other locations
- Predicting an attribute based on previous times
- Labeling land use/ objects/animals/structures from aerial image
- Detecting the type of vegetation based on remote sensing (electromagnetic spectral imaging)

Machine learning is useful for modeling relationships between patterns (visual patterns, attribute patterns, spatial patterns, temporal patterns, etc.)

Concrete examples:

- Predicting the voting outcome for a district based on demographics
- Predicting the sale price of a dwelling based on attributes and historical data in similar areas
- Predicting the temperature in Newark based on temperatures in Philadelphia and Baltimore
- Predicting the population of Newark in 2030
- Recognizing crops from satellite images
- Detecting the distribution of tree species in a forest

Data sources:

- Census data or surveys combined with poll results
- County property information combined with real estate listings
- NOAA data rasters sampled at points
- City records, census data
- USDA, satellite images, **ground truth**
- Hyperspectral images, **ground truth**

Machine learning is the optimization of a data-processing function in terms of a data-driven objective function

Data-processing function:

- **input (features)**
- **output (target value/label)**
- **formulation of a parametrized function**

Optimization is the process of searching (often by trial and error) for parameters to maximize a specified objective function under specified constraints

Data-driven objectives quantify the fitness of function’s output based on available data (the target output)

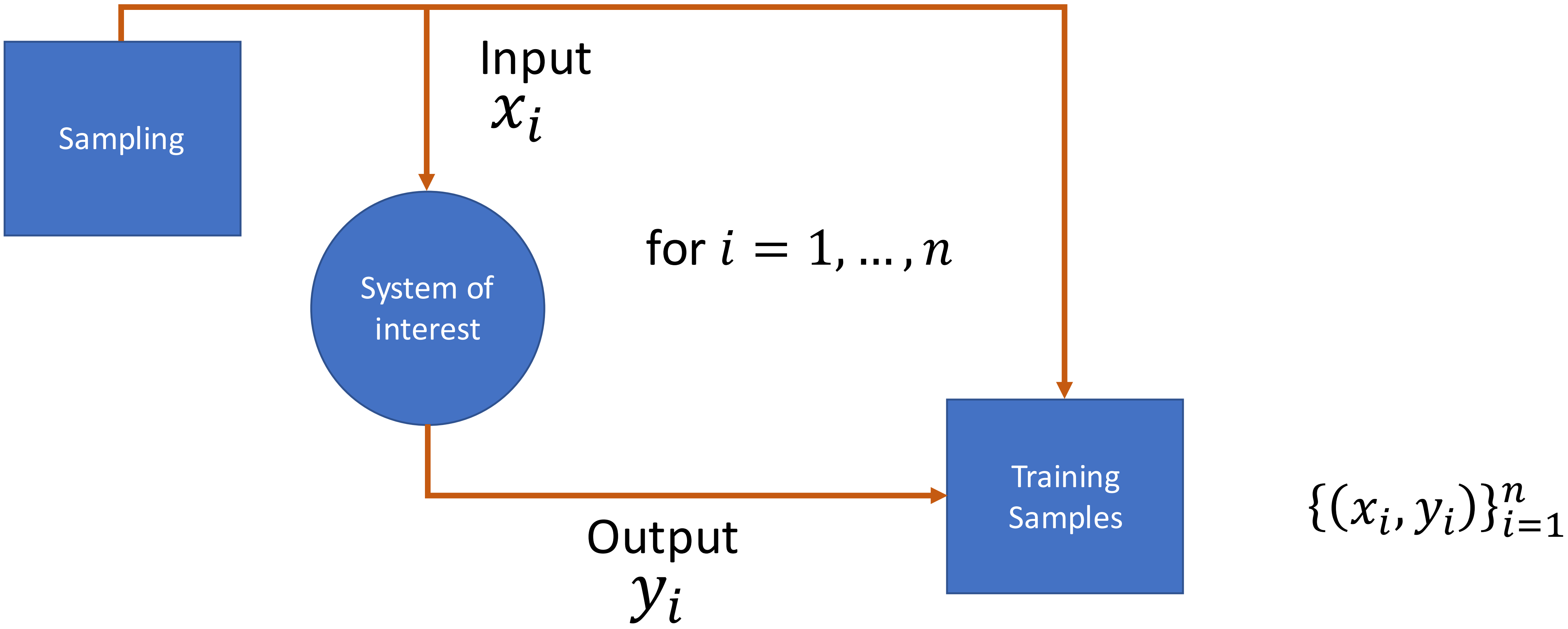
Machine learning is the process of creating artificial intelligence

Artificial intelligence is automated decision making that uses computer algorithms and models based on data and knowledge

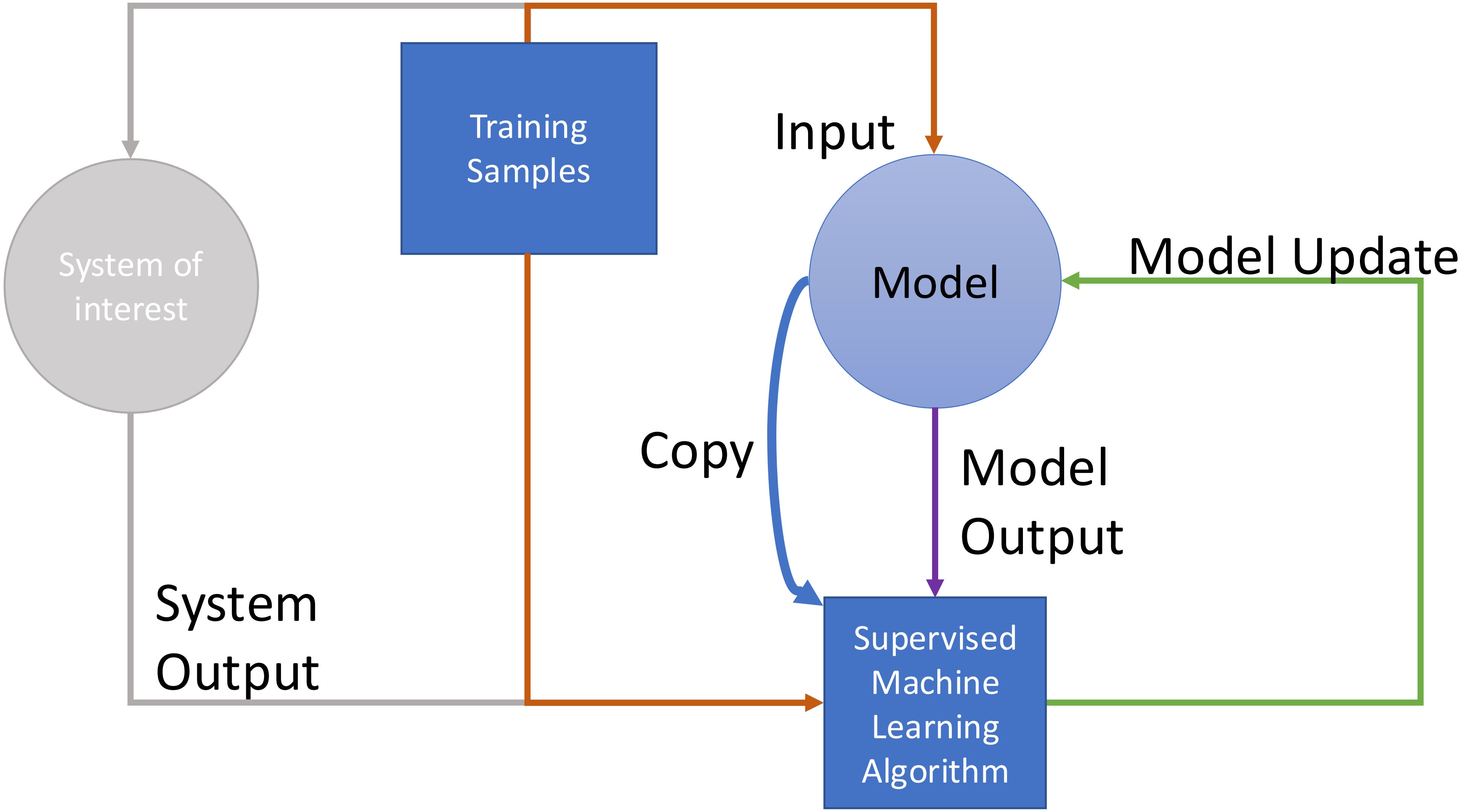
Perception and Action

- Observe and learn
  - Gather data
    - **Decide what/how to measure**
  - Characterize this data
    - **Decide how to represent it**
  - Update belief/knowledge
    - **Decide how to store/update belief**
- Act
  - Gather new data/explore the system
    - **Decide how to sample or do**
- Evaluate

Observing a system (gathering data)



Training a system



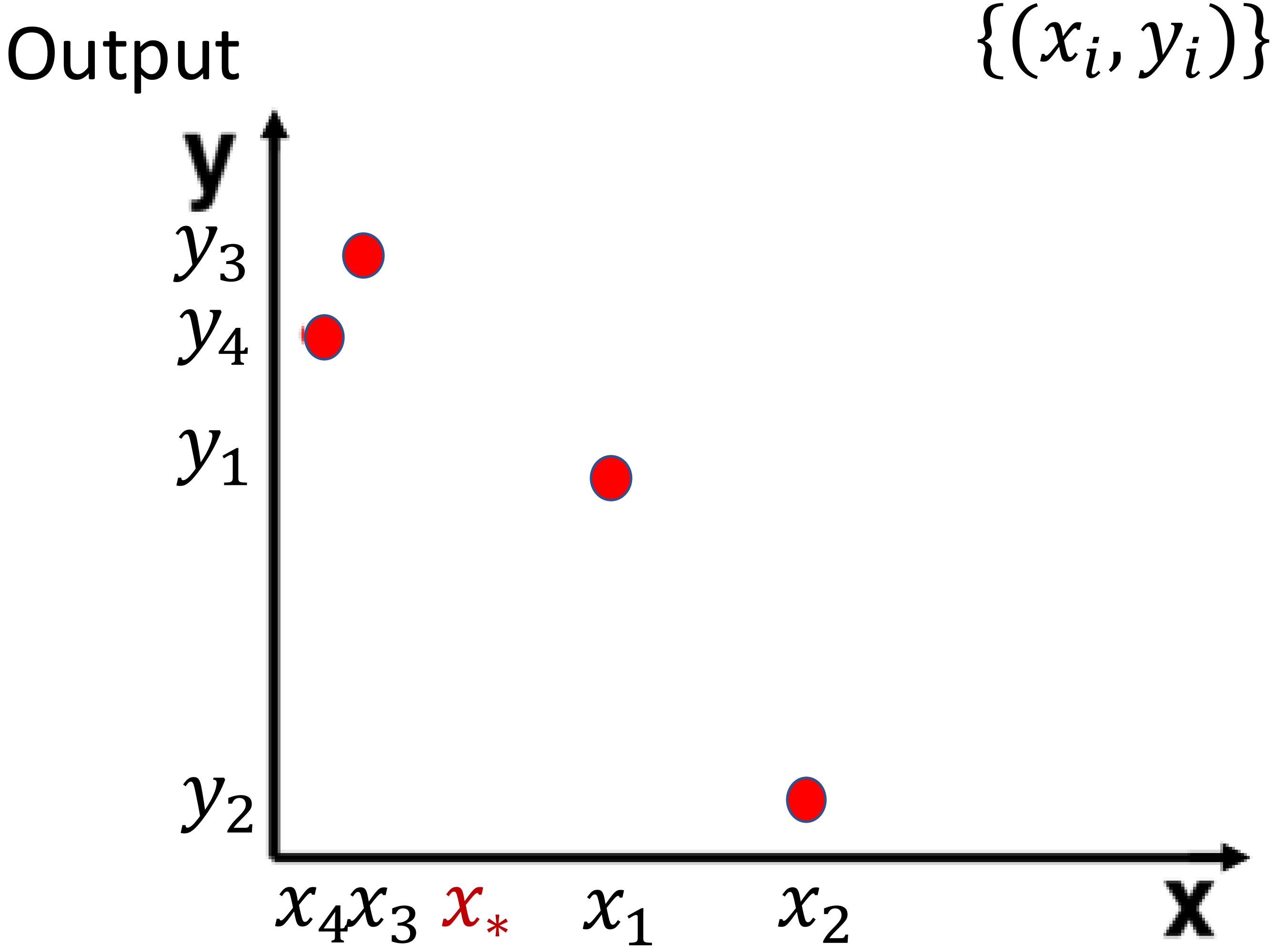
Objective

Input, processing function, and output

Optimization

Modeling a relationship between two attributes

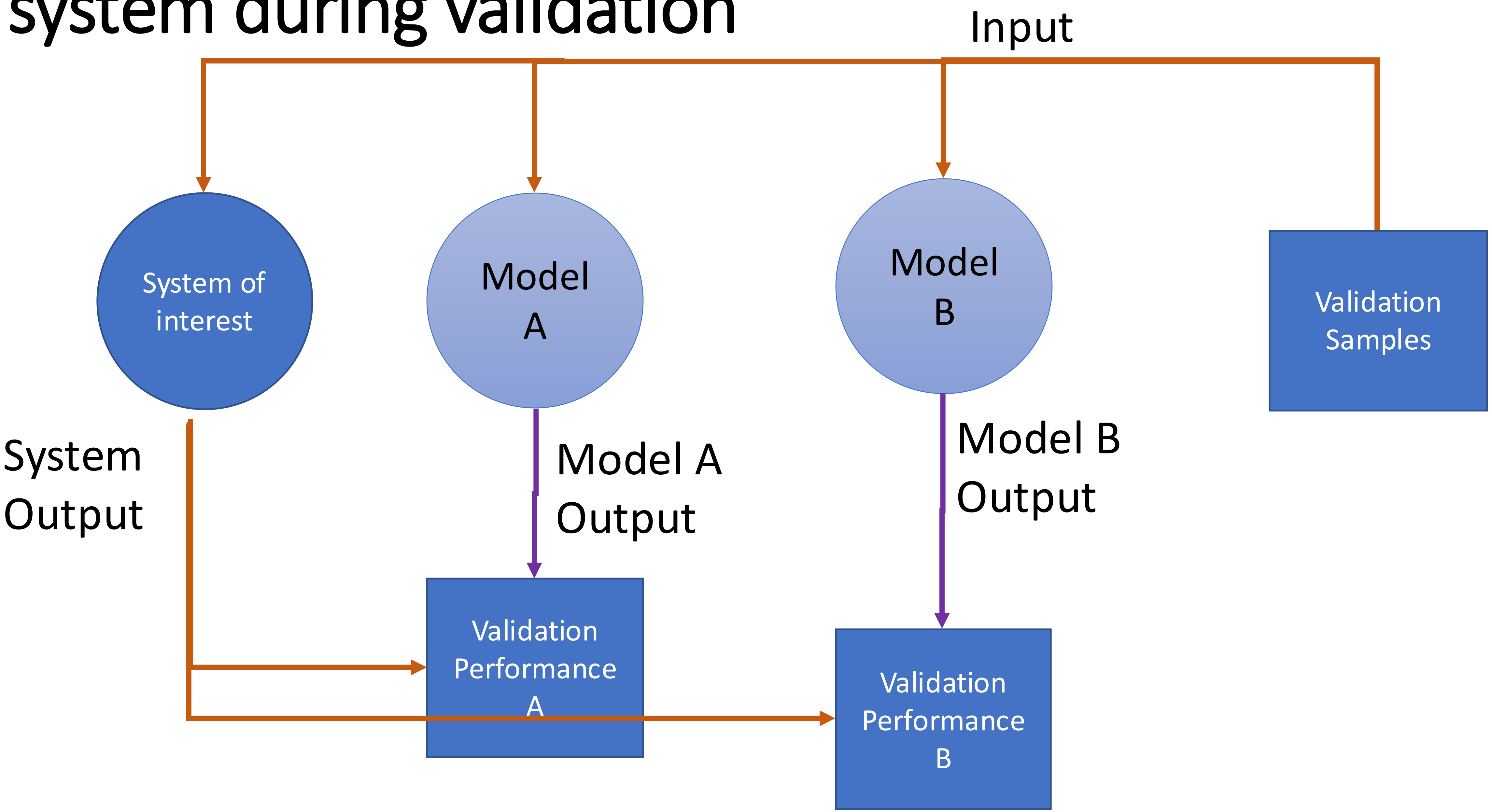
$\{(x_i, y_i)\}_{i=1}^4$



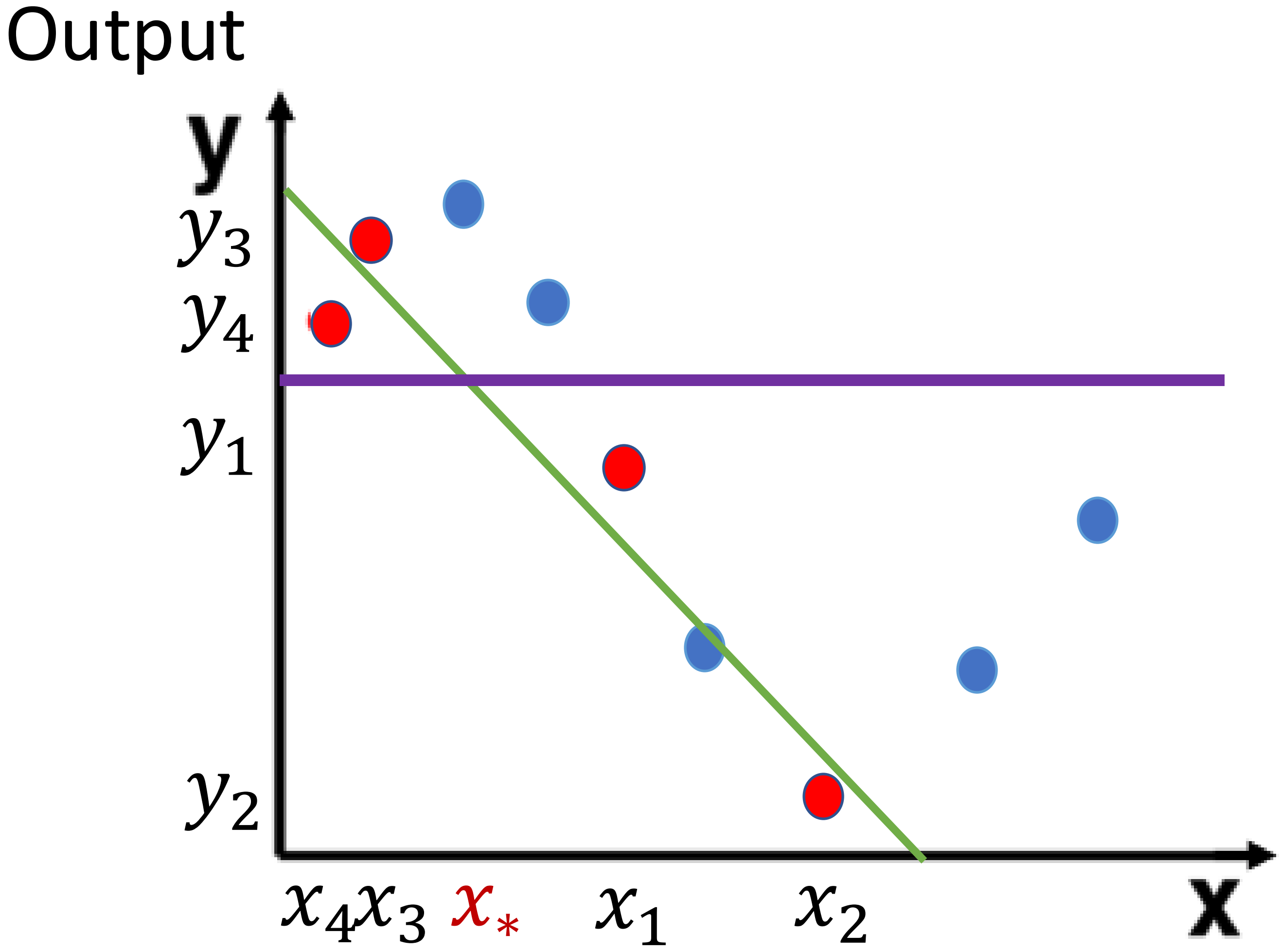
	$\vec{x}_1$	$\vec{x}_2$	$\vec{x}_3$	$\vec{x}_4$	$\vec{x}_*$
x	1.0	1.4	0.4	0.2	0.7
y	*3.0*	0.0	4.0	3.5	?
	y_1	y_2	y_3	y_4	y_*

Given x_* and $\{(x_i, y_i)\}_{i=1}^4$ what is y_*?

Choosing hyper-parameters and evaluating a system during validation



Which model performs the best on the validation data?



Multivariate regression

	$\vec{x}_1$	$\vec{x}_2$	…	$\vec{x}_n$
$x^{(1)}$				
$x^{(2)}$				
$x^{(3)}$				
$\vdots$				
$x^{(d)}$				
y	y_1	y_2	…	y_n

Instance 1

Instance 2

Instance n

$$\hat{y} = w \cdot \vec{\boldsymbol{x}} + b = \sum_{j=1}^{d} x^{(j)} w_j + b$$

Linear model for regression

	$\vec{\boldsymbol{x}}_1$	$\vec{\boldsymbol{x}}_2$	…	$\vec{\boldsymbol{x}}_n$
$\boldsymbol{x}^{(1)}$				
$\boldsymbol{x}^{(2)}$				
$\boldsymbol{x}^{(3)}$				
$\vdots$				
$\boldsymbol{x}^{(d)}$				
$\boldsymbol{y}$	$\boldsymbol{y}_1$	$\boldsymbol{y}_2$	…	$\boldsymbol{y}_n$

Data:

$\{(\vec{\boldsymbol{x}}_i, y_i)\}_{i=1}^{n}$

$$\vec{\boldsymbol{x}}_i = \begin{bmatrix} x_i^{(1)} \\ \vdots \\ x_i^{(d)} \end{bmatrix} \in \mathbb{R}^d, \quad y_i \in \mathbb{R} \ \text{ for } i = 1,\ldots,n$$

Model:

$$f_\theta(\ \vec{\boldsymbol{x}}\ ) = w \cdot \vec{\boldsymbol{x}} + b = \hat{y}, \qquad \theta = [w_1, \ldots, w_d, b] \in \mathbb{R}^{d+1}$$

Loss: $\quad J_\mu(\theta) = \sum_{i=1}^{n} \frac{1}{n} |y_i - f_\theta(x_i)|^2 + \mu \sum_{j=1}^{d} w_j^2$

Optimal solution: $\quad \theta^* = \underset{\theta}{\operatorname{argmin}}\, J_\mu(\theta)$

Linear model for regression

$$f_{\theta^*}(\vec{\boldsymbol{x}}) = w^* \cdot (\vec{\boldsymbol{x}} - \overline{\boldsymbol{x}}) + b^*$$

$$\theta^* = [w_1^*, \dots, w_d^*, b] \in \mathbb{R}^{d+1}$$

$$b^* = \sum_{i=1}^{n} \frac{1}{n} y_i \ ,$$

$$w^* = (\Sigma + \frac{\mu}{n} I)^{-1} \vec{\boldsymbol{p}}$$

$$\Sigma = \mathrm{Cov}(\vec{\boldsymbol{x}})$$

$$\Sigma_{kl} = \mathrm{Cov}(x^{(k)}, x^{(l)})$$

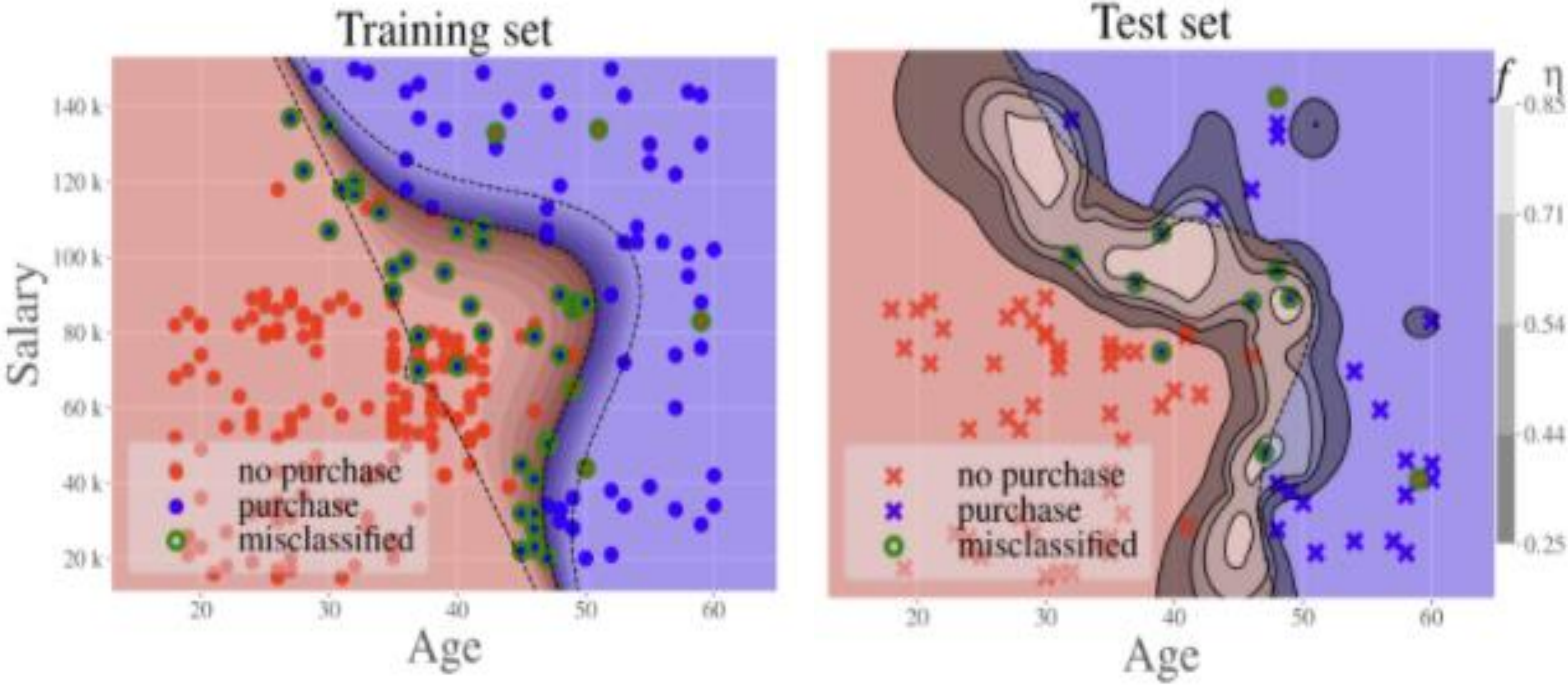
$$\vec{\boldsymbol{p}} = \mathrm{Cov}(\vec{\boldsymbol{x}}, y)$$

$$p_k = \mathrm{Cov}\left(x^{(k)}, y\right)$$

$$= n^{-1} \sum_i y_i (x_i^{(k)} - \bar{x}^{(k)})$$

Σ	$\boldsymbol{x}^{(1)}$	$\boldsymbol{x}^{(2)}$	...	$\boldsymbol{x}^{(d)}$
$\mathbf{x}^{(1)}$	σ_1^2	$\rho_{12}\sigma_2\sigma_1$		$\rho_{1d}\sigma_d\sigma_1$
$\boldsymbol{x}^{(2)}$	$\rho_{12}\sigma_1\sigma_2$	σ_2^2		
$\vdots$			$\ddots$	
$\boldsymbol{x}^{(d)}$	$\rho_{1d}\sigma_1\sigma_d$			σ_d^2

Example (sales) with linear decision boundary versus nonlinear



Designing a Machine Learning System

1. **Goal**: What is the task?
2. **Data**: How is the data represented? Define the characteristics of the input(features) and output(predictions).
3. **Model**: What are the possible relationships or ways to process the data? Define the set/family of functions that map input to output, how are they parametrized (linear model, neural network, etc.)?
4. **Fit**: What is the training objective (*Loss/cost)* and is there a separate performance metric for validation/testing?
   a) How is the data divided between training and validation and test?
   b) Should all instances/cases/classes have equal influence?
5. **Train**: What method will be used to adjust the parameters of a model during training?
6. **Select:** Hyper-parameter choices create different trained models. How will the best combination be chosen and/or the space of hyper-parameters searched?

When is a model/belief is good enough?

“Decision makers can […] either [find] optimum solutions for a simplified world, or [find] satisfactory solutions for a more realistic world. Neither approach, in general, dominates the other...”

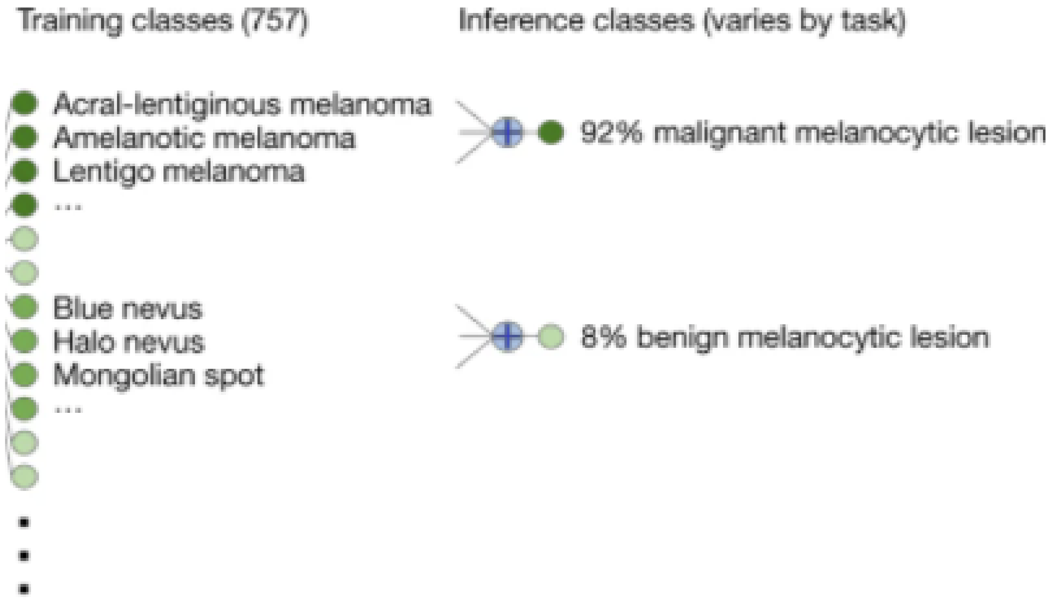
—Herbert A. Simon, Nobel Prize in Economics

Model: Popular choices

- Linear model
- k-nearest neighbor
- Decision trees
- Random forest, gradient boosting
- Neural networks
- Convolutional neural network
- Kernel ridge regression
- Support vector machine
- Gaussian processes

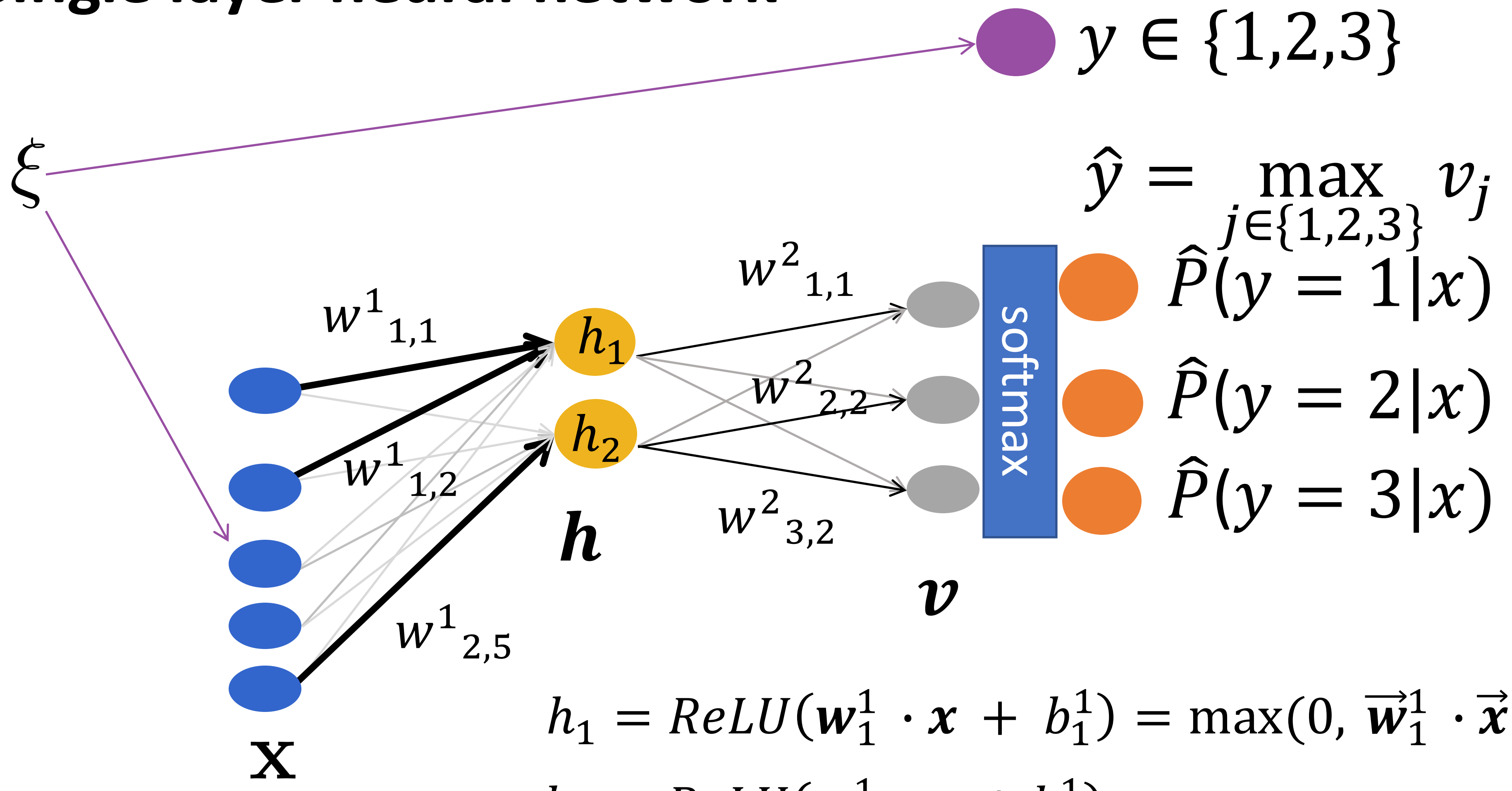
Fit: Quantifying performance with a confusion matrix

		Predicted condition	
	Total population = P + N	Positive (PP)	Negative (PN)
Actual condition	Positive (P)	True positive (TP)	False negative (FN)
	Negative (N)	False positive (FP)	True negative (TN)



Artificial neural networks consist of layers of processing connected together

Single layer neural network



$$h_1 = ReLU(\boldsymbol{w}_1^1 \cdot \boldsymbol{x} + b_1^1) = \max(0, \vec{\boldsymbol{w}}_1^1 \cdot \vec{\boldsymbol{x}} + b_1^1)$$

$$h_2 = ReLU(\boldsymbol{w}_2^1 \cdot \boldsymbol{x} + b_2^1)$$

$$v_j = \boldsymbol{w}_j^2 \cdot \boldsymbol{h} + b_j^2, \text{ for } j = 1, \dots, 3$$

$$\hat{P}(y=j|x) = \frac{e^{v_j}}{\sum_k e^{v_k}}, \text{ for } j = 1, \dots, 3$$

CNN: Where’s Waldo?
(Prediction yes or no for each image patch

?

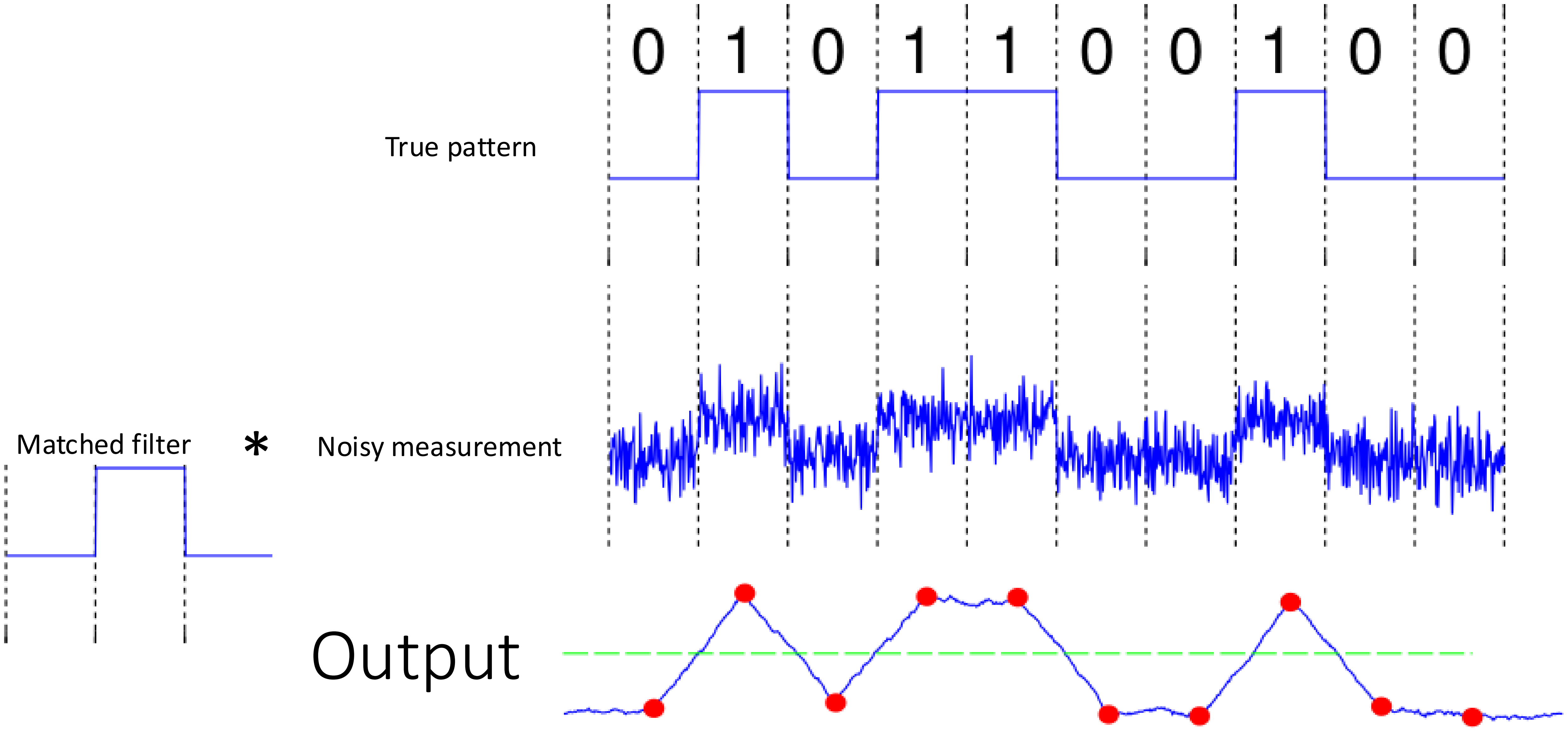
?

?

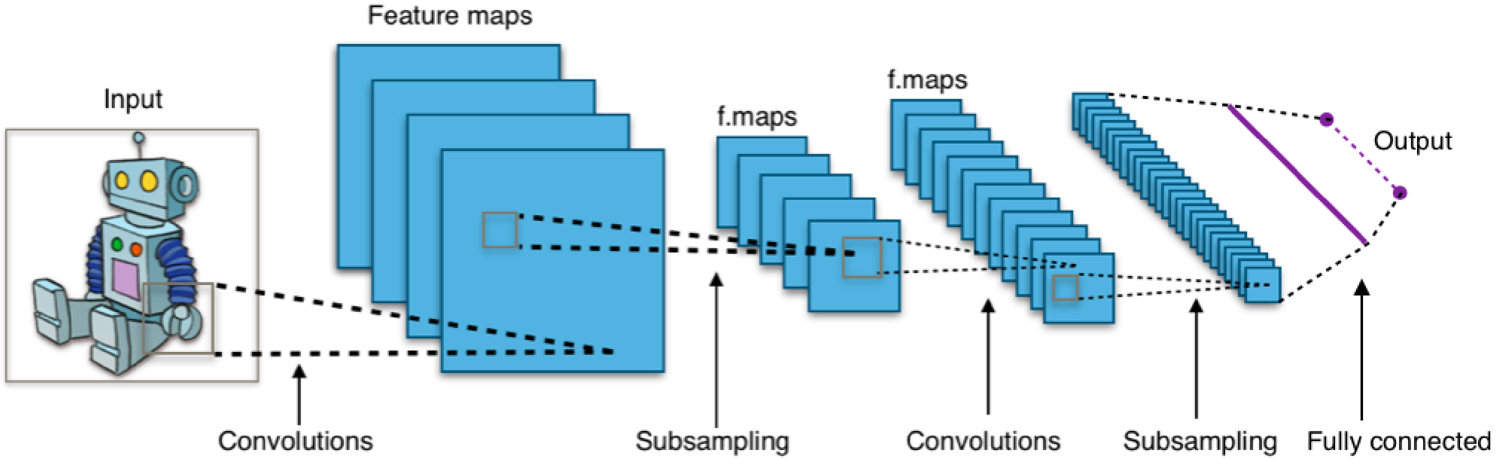
?

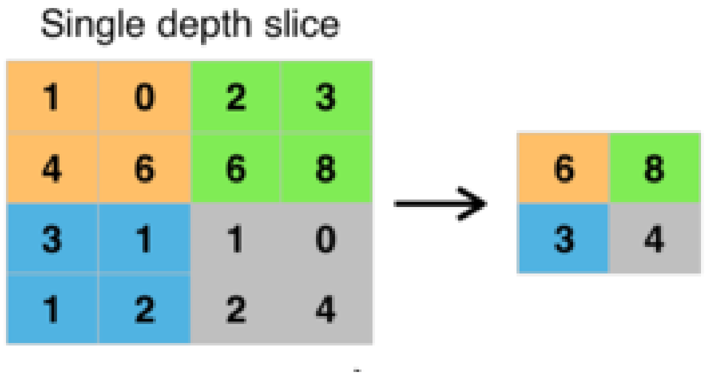
https://eng.libretexts.org/Bookshelves/Electrical_Engineering/Signal_Processing_and_Modeling/Signals_and_Systems_%28Baraniuk_et_al.%29/13%3A_Capstone_Signal_Processing_Topics/13.04%3A_Matched_Filter_Detector

Convolution or matched filtering

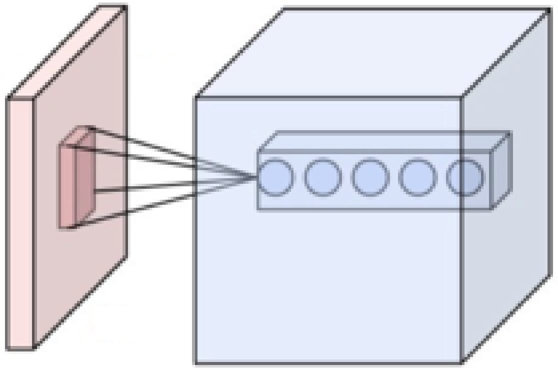


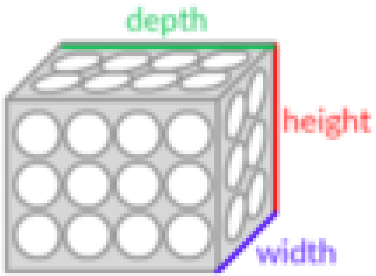
Depth is the number of channels/attributes/layers



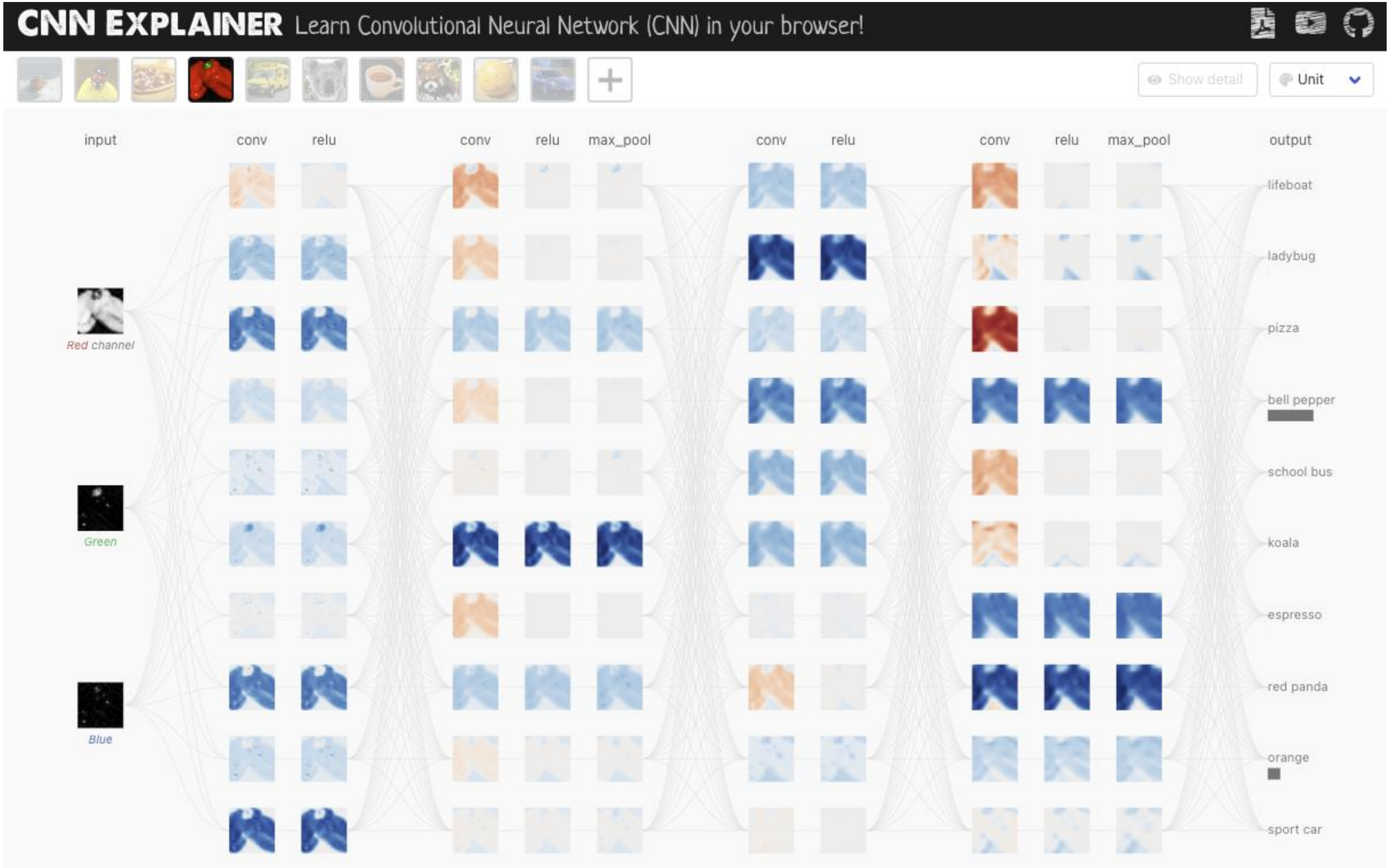


Subsampling via
max pooling with a 2x2 filter and stride = 2



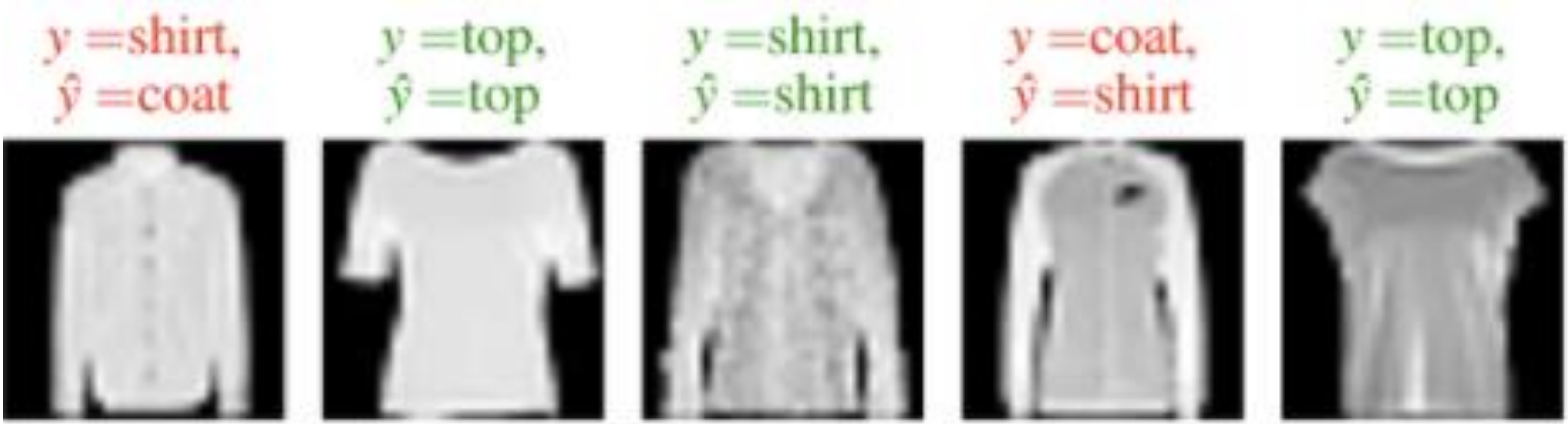


https://poloclub.github.io/cnn-explainer

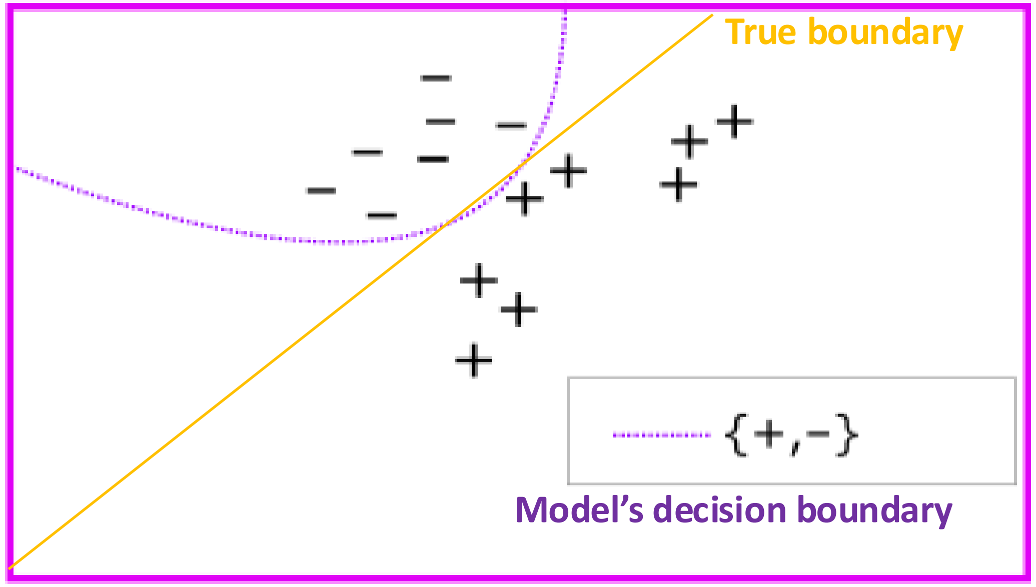


Example of CNN classifier on Fashion MNIST

- Error rate is 273/2646

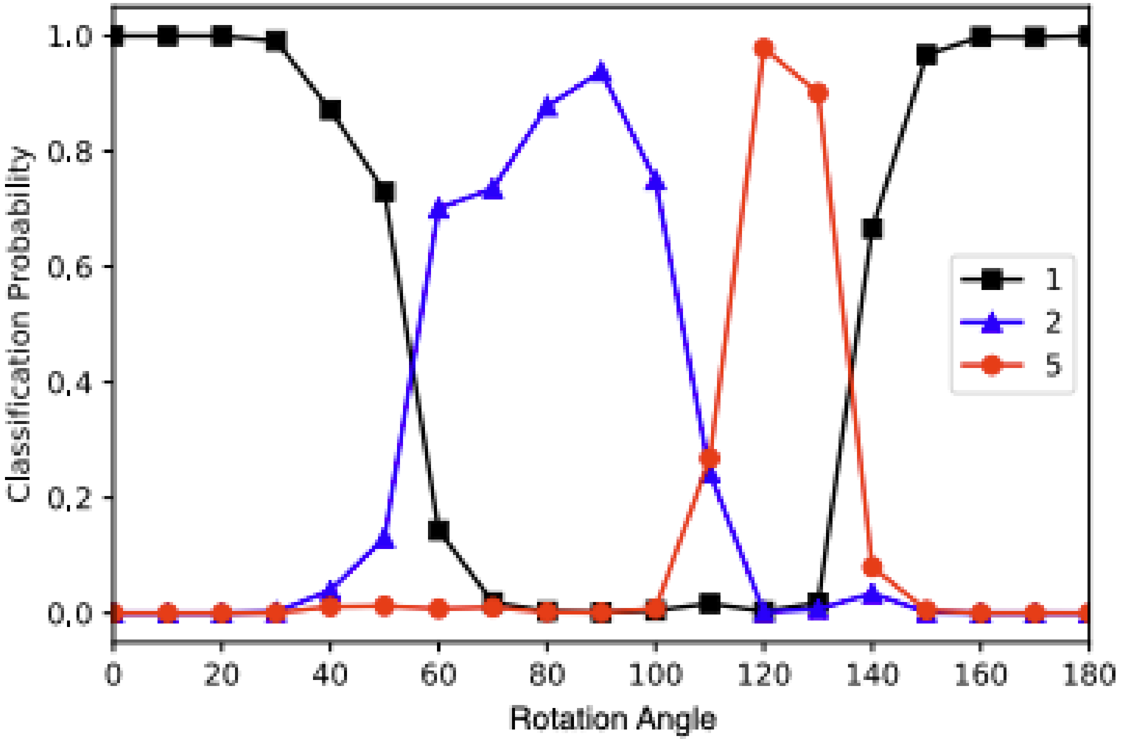


Classifiers tend to learn isolate class with less diversit



Rotated satellite images have same land use label (invariant to rotation)

Machine learning models may fail to recognize what they don't know



Input image (rotated 1)

- Sensoy et al. *NeurIPS* 2018 https://proceedings.neurips.cc/paper/2018/file/a981f2b708044d6fb4a71a1463242520-Paper.pdf

When human experts fail

- Biased
  - Becomes rare with enough training
- Ambiguous cases
  - Need second opinion or more data
- Out of distribution
  - Outside of expertise

When statistical models fail

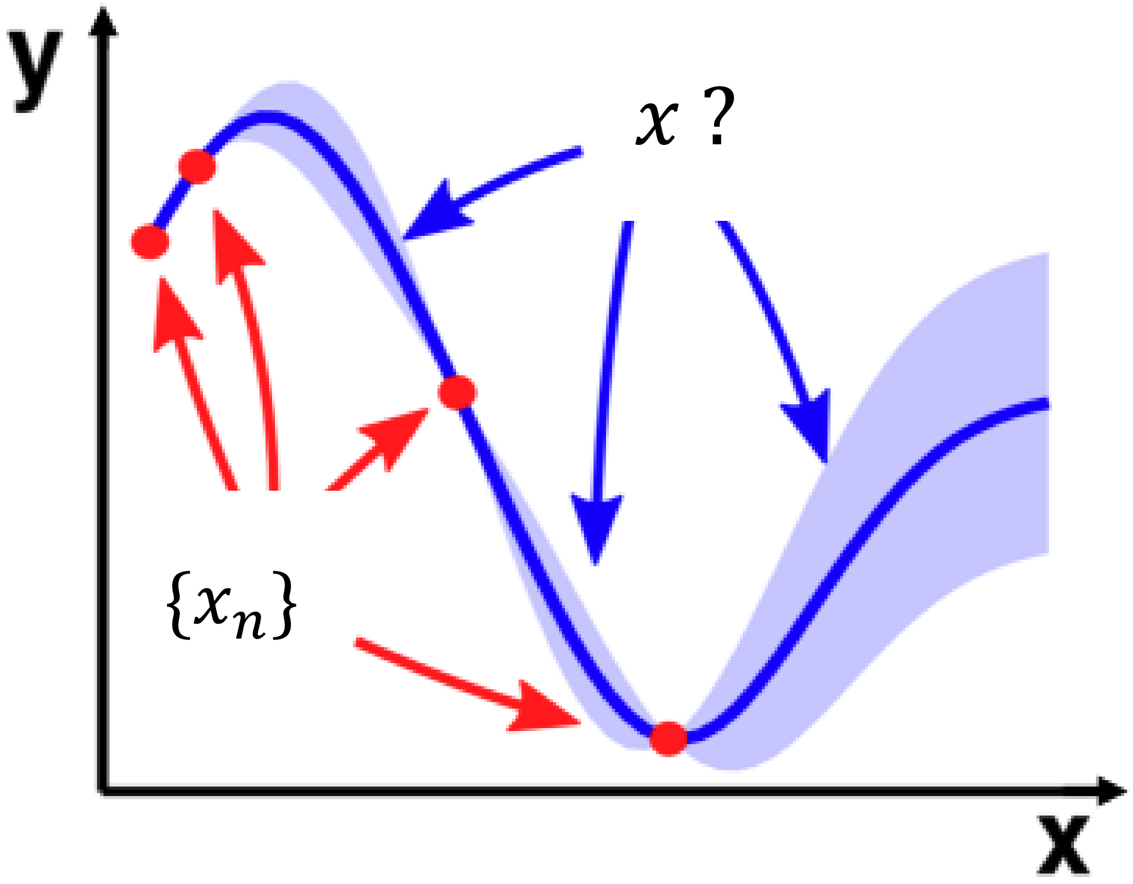
- Systematic error
  - Becomes rare with enough **correct** training data
- Ambiguous cases
  - Expert also needs to be careful
- Out of distribution
  - Corrupted or unseen case
  - Expert can easily recognize

Human-machine systems fail when

- they propagate/exacerbate biases
  - machine as a productivity multiplier
- excessive trust is given to machine
  - loss of vigilance
- insufficient data is collected to improve future versions
  - acquire more unbiased labeled data to validate

Non-linear models

- k-nearest neighbor
- Decision trees, random forests, gradient boosting
- Neural networks
- Kernel ridge regression
- Gaussian process

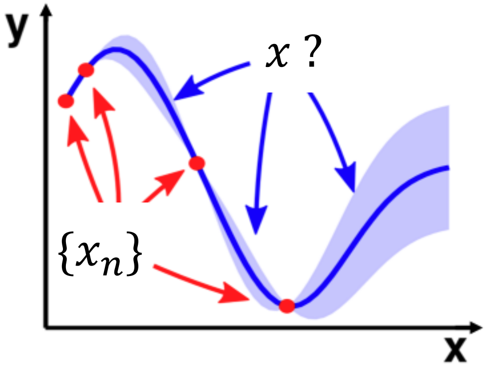


Phase 1. Fit relationship
Phase 2: Find x that gives a specific y with high confidence (near seen data) and fits constraints!

- Kernel regression
  - Advanced by Prof. Grace Wahba at UW-Madison

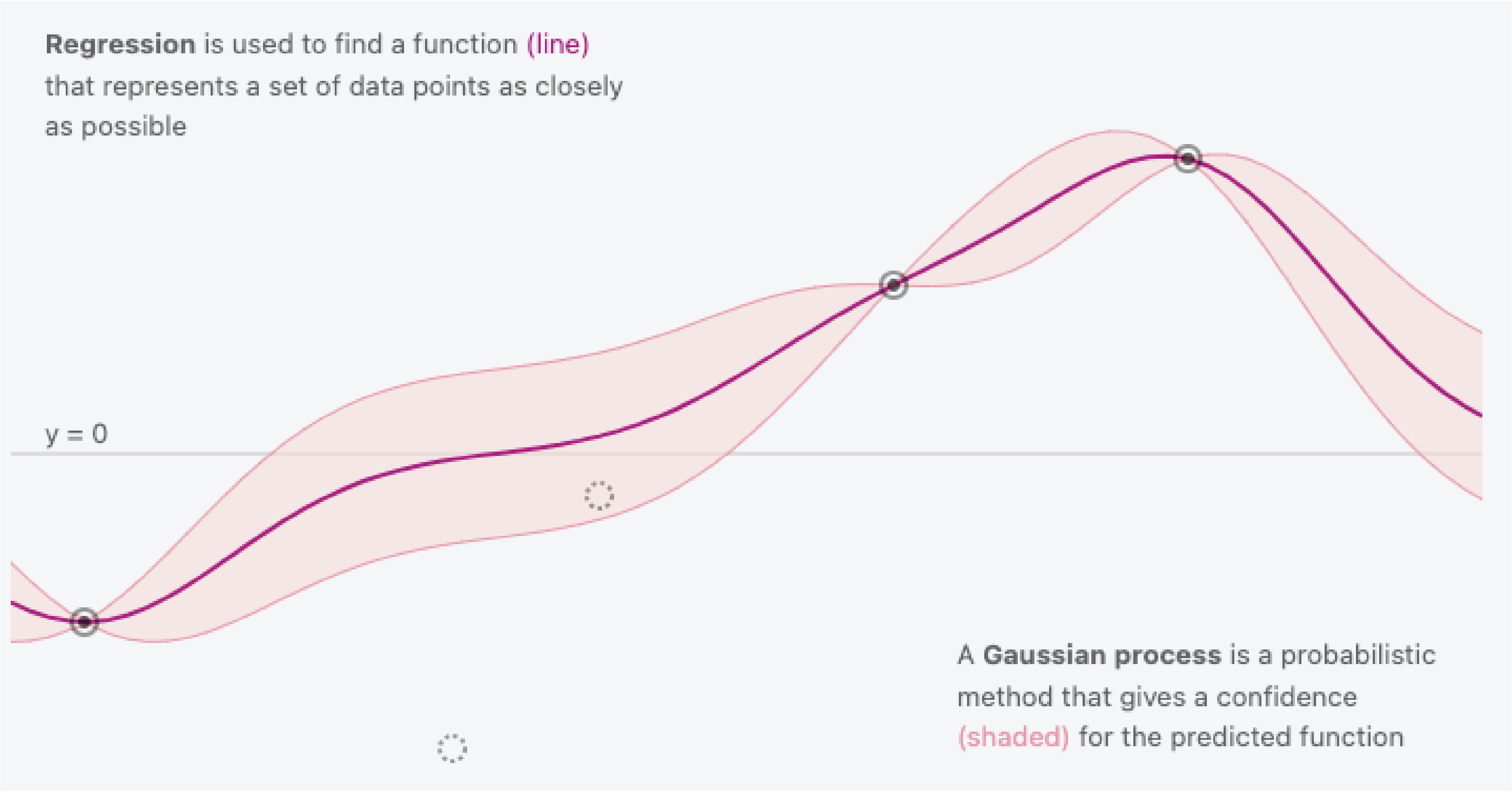
$$E[Y|x,\{(x_i,y_i)\}_{i=1}^{n}] = \bar{f}(x) = [\kappa(x,x_1),\ldots,\kappa(x,x_n)]\mathbf{K}^{-1}\vec{\mathbf{y}} = \boldsymbol{K}\vec{\boldsymbol{\alpha}}$$

```
krr.fit(X,y).predict(x)
```



By The scikit-learn developers - github.com/scikit-learn/scikit-learn/blob/master/doc/logos/scikit-learn-logo.svg, BSD, https://commons.wikimedia.org/w/index.php?curid=71445288

Gaussian process (Kriging)



The predicted value at x is normally distributed with mean $f(x)$, and variance σ_x^2

$$\mathcal{N}(f(x), \sigma_x^2)$$

$$\sigma_x^2 = \operatorname{cov}\big(f(x), f(x)\big) = \kappa(x, x) - [\kappa(x, x_1), \dots, \kappa(x, x_N)]\mathbf{K}^{-1}[\kappa(x, x_1), \dots, \kappa(x, x_N)]$$

https://distill.pub/2019/visual-exploration-gaussian-processes/