Introduction to Machine Learning and Artificial Intelligence: Lecture I

Michael Kagan

SLAC NATIONAL ACCELERATOR LABORATORY

2nd COFI Advanced Instrumentation and Analysis Techniques School

December 9, 2023

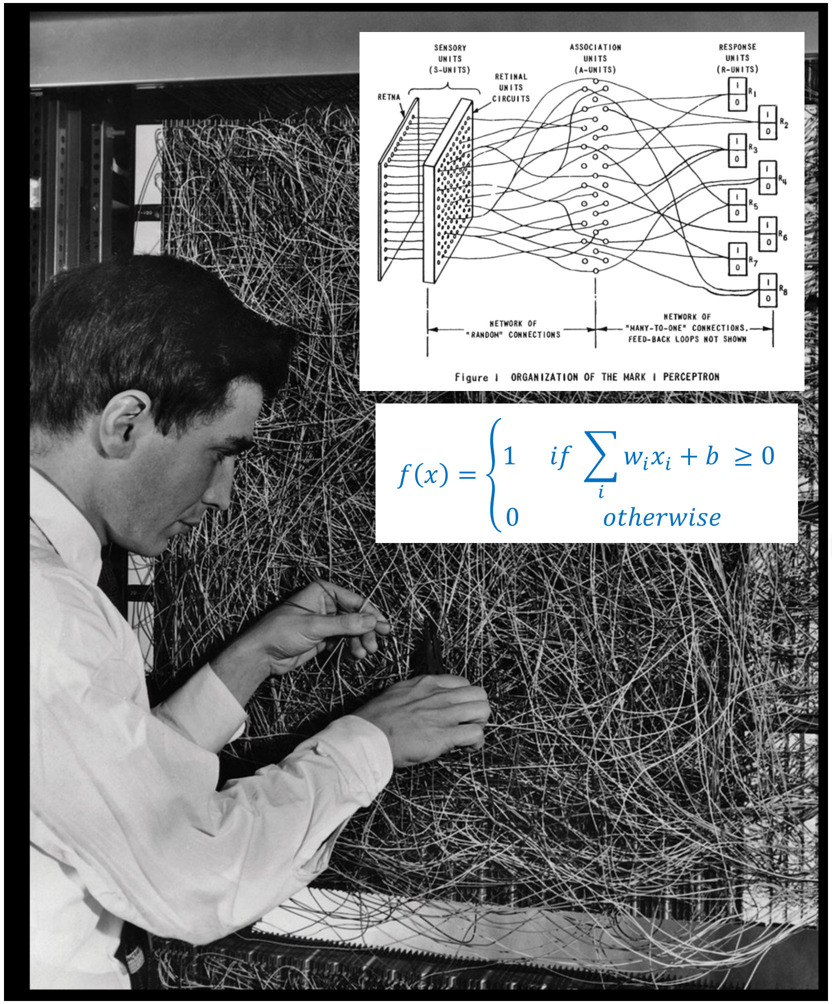
The Plan

- Lecture 1
  - Introduction to Machine Learning fundamentals
  - Linear Models

- Lecture 2
  - Neural Networks
  - Deep Neural Networks
  - Inductive Bias and Model Architectures

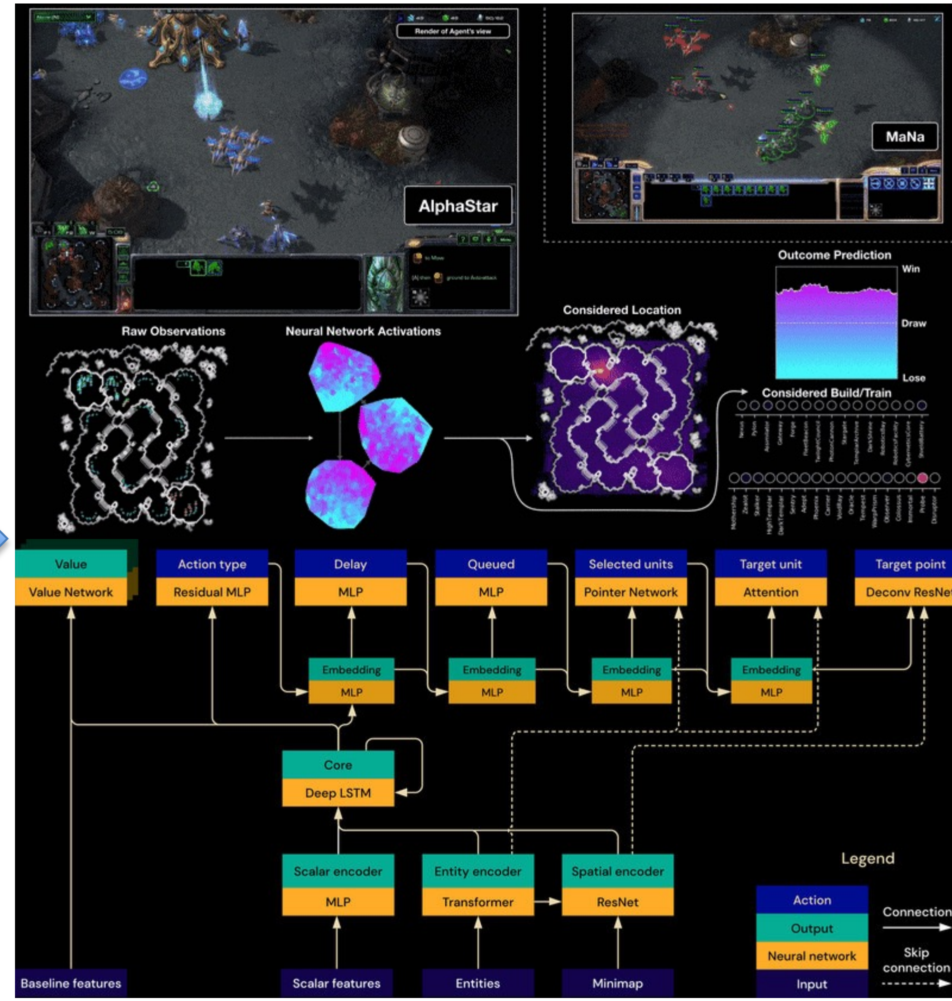
- Lecture 3
  - Unsupervised Learning
  - Autoencoders
  - Towards Generative Models: Variation Autoencoders

Long History of Machine Learning



$$f(x) = \begin{cases} 1 & \text{if } \sum_i w_i x_i + b \geq 0 \\ 0 & \text{otherwise} \end{cases}$$

Perceptron



AlphaStar

Rosenblatt [1958](), [1960]()

[Vinyals et. al. 2019]()

The Power of ML

street style photo of a woman selling pho at a Vietnamese street market, sunset, shot on fujifilm

generate low-level, high-dim data from high-level concepts

High-Level Concept

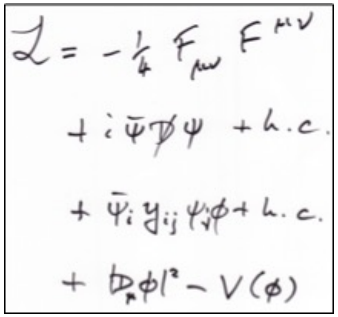
Low-Level Data

12

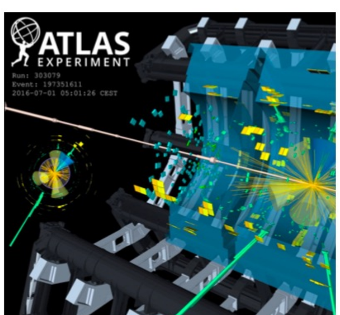
This is a picture of Barack Obama.
His foot is positioned on the right side of the scale.
The scale will show a higher weight.

reconstruct high level concepts from low-level, high-dim data

Particle Physics Has Similar Goals!



generate low-level, high-dim data from high-level concepts

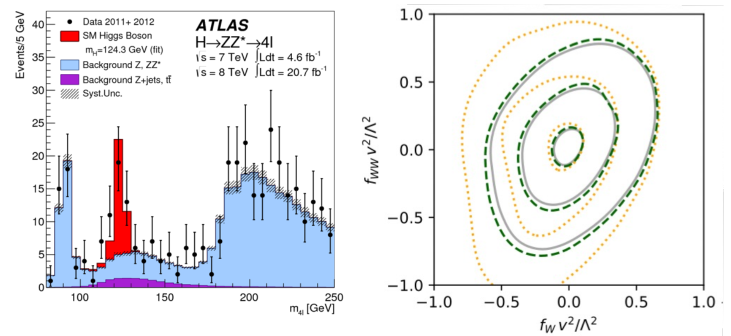


Simulation

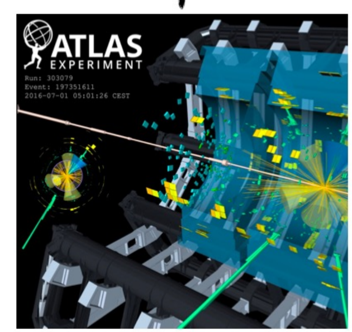
High-Level Concept

Low-Level Data

Data Analysis

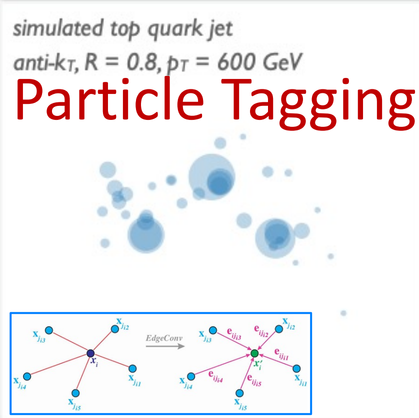


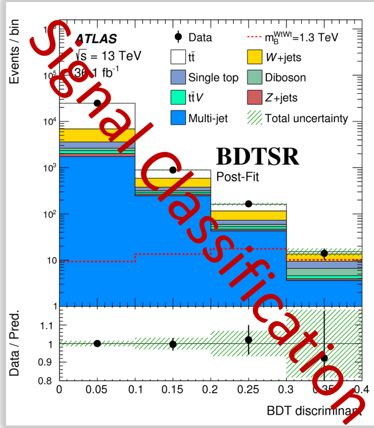
reconstruct high level concepts from low-level, high-dim data

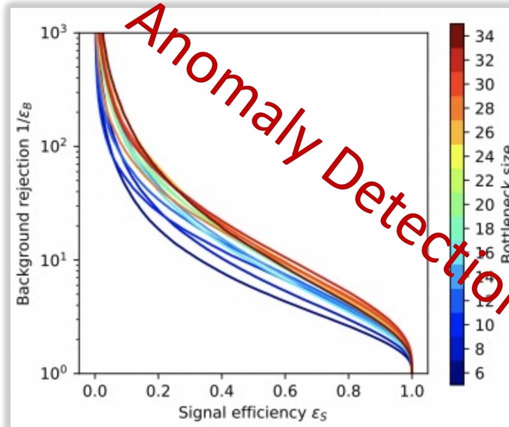


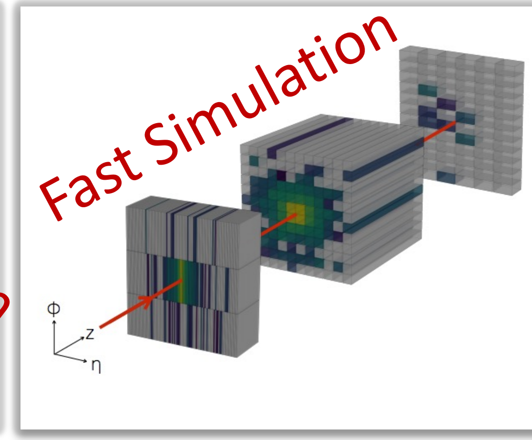
Slide credit: L. Heinrich

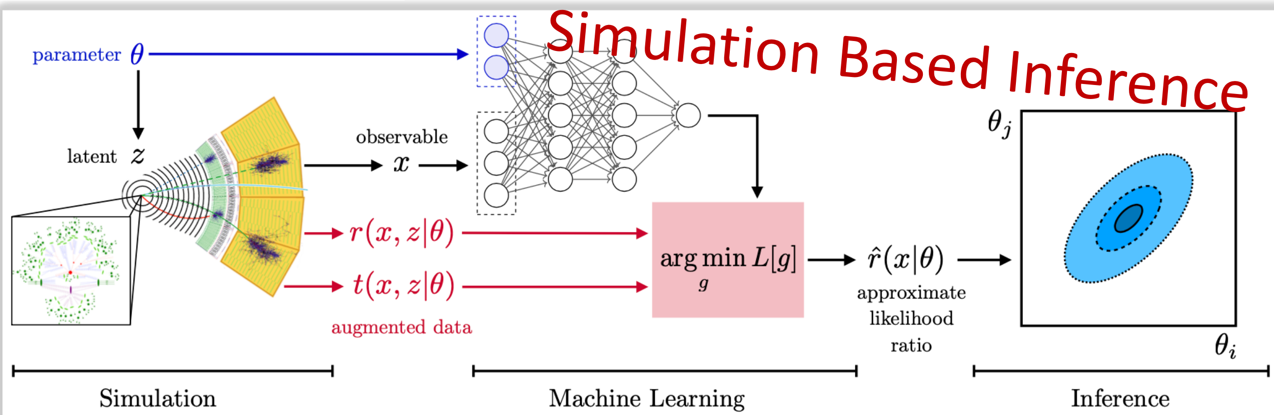
Machine Learning in HEP

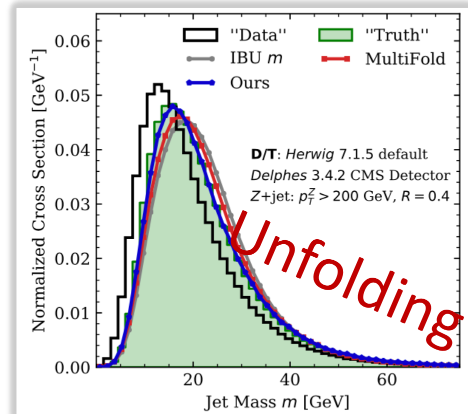


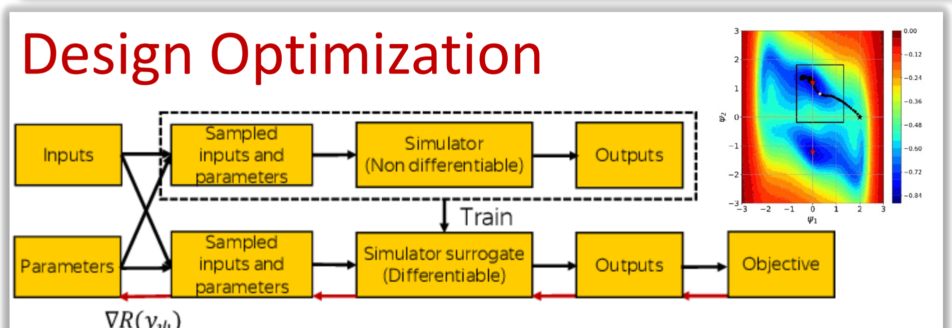


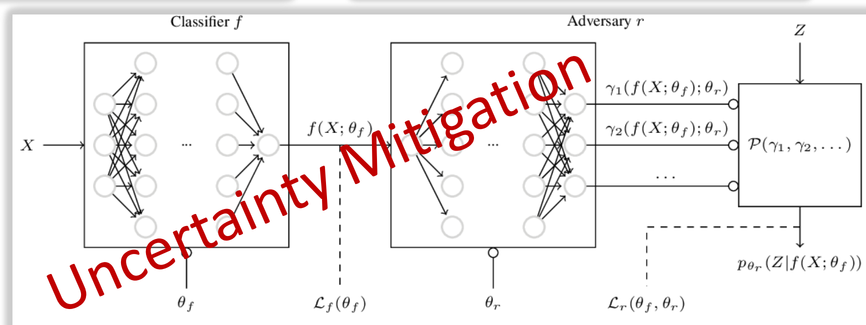










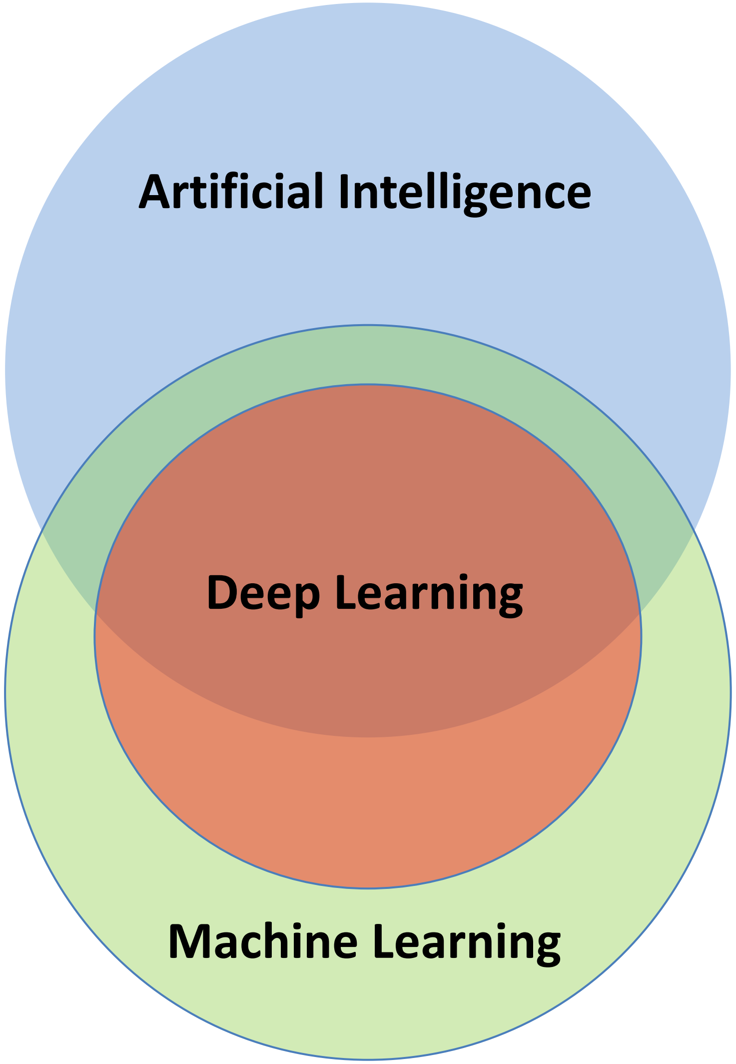


+ More! Check out The Living Review of ML in HEP

What is Machine Learning?

- Giving computers the ability to learn without explicitly programming them (Arthur Samuel, 1959)
- Statistics + Algorithms
- Computer Science + Probability + Optimization Techniques
- **Fitting data with complex functions**
- **Mathematical models** learnt from data that characterize the patterns, regularities, and relationships amongst variables in the system

Artificial Intelligence, Machine Learning, Deep Learning



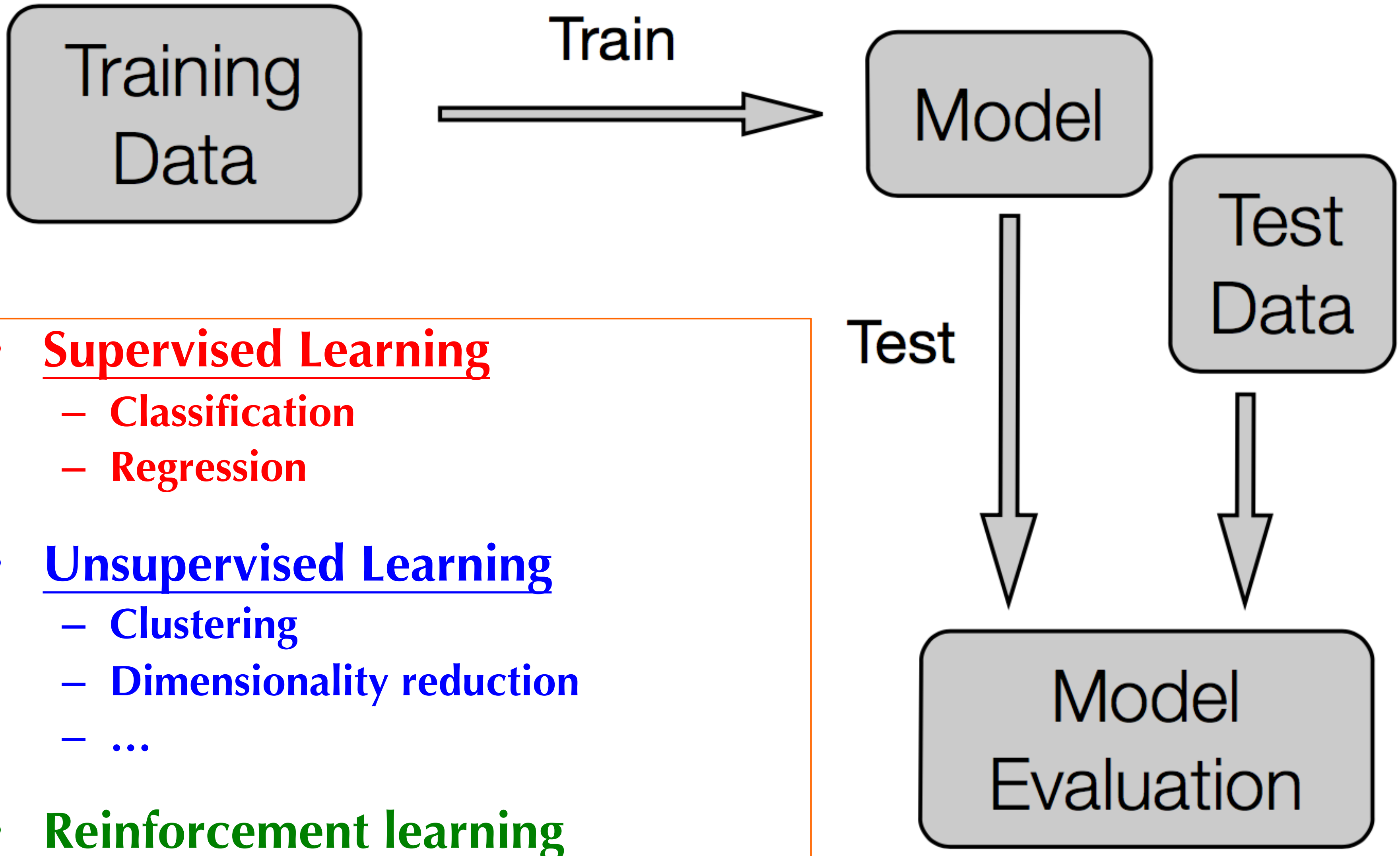
- **AI:** make computers act in an intelligent way
  - Rules, reasoning, symbol manipulation
- **ML:** Uses data to learn "intelligent" algorithms
- **Deep Learning**: Approach to ML that (often) uses complex pipelines to process low level data (e.g. pixels)

Slide credit: L. Heinrich

Machine Learning: Models

- Key element is a **mathematical model**
  - A mathematical characterization of system(s) of interest, typically via random variables
  - Chosen model depends on the task / available data
- **Learning**: estimate statistical model from data
  - Supervised learning
  - Unsupervised Learning
  - Reinforcement Learning
  - …
- **Prediction and Inference**: using statistical model to make predictions on new data points and infer properties of system(s)

Learning



- **Supervised Learning**
  - **Classification**
  - **Regression**
- **Unsupervised Learning**
  - **Clustering**
  - **Dimensionality reduction**
  - **…**
- **Reinforcement learning**

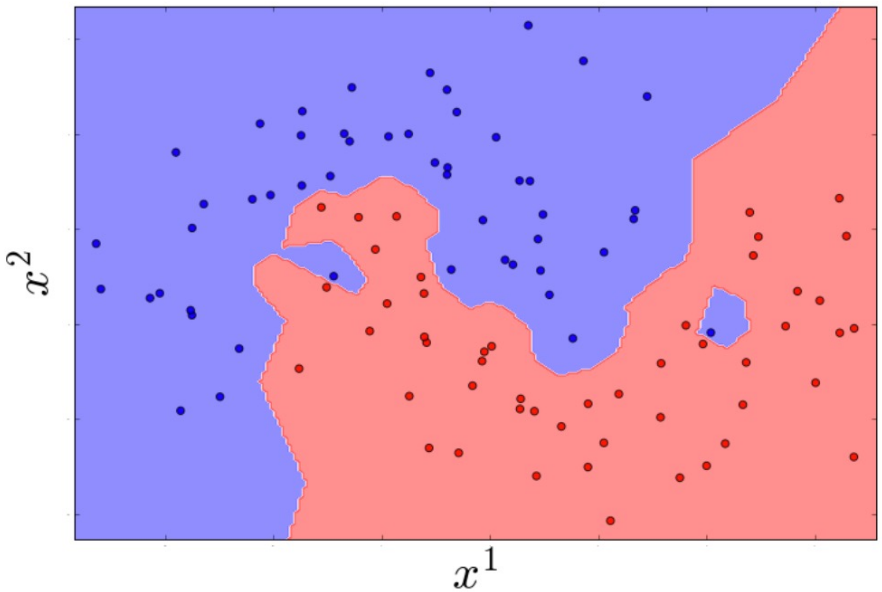
[Ravikumar]

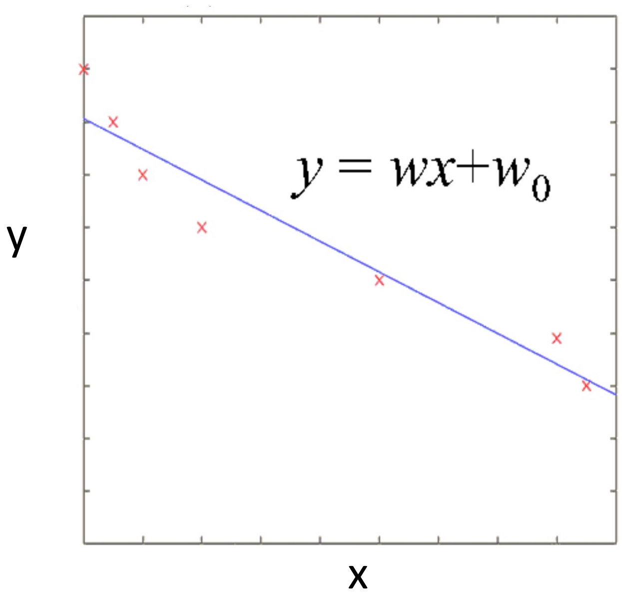
Supervised Learning

- Given N examples of observed features $\{x_i\}$ and prediction **targets** $\{y_i\}$, learn function mapping $\boldsymbol{h(x) = y}$

Classification:
Y is a finite set of **labels** (i.e. classes) denoted with integers

Regression:
Y is a real number



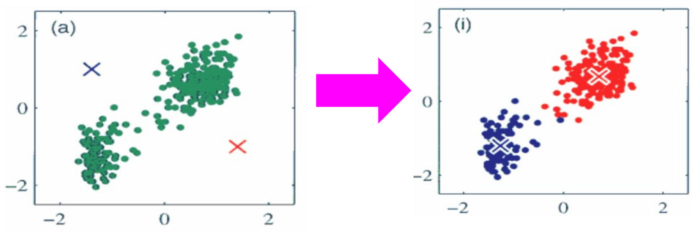


Unsupervised Learning

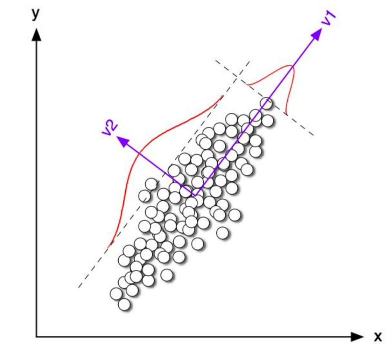
Given data $D = \{x_i\}$, but no labels, find structure in data

[Bishop]

Clustering: partition the data into groups $D = \{D_1 \cup D_2 \cup D_3 \ \ldots \cup \ D_k\}$



Dimensionality reduction: find a low dimensional (less complex) representation of the data with a mapping $Z = h(X)$



Density estimation and sampling: estimate density $p(x)$, and/or learn to draw new samples of x

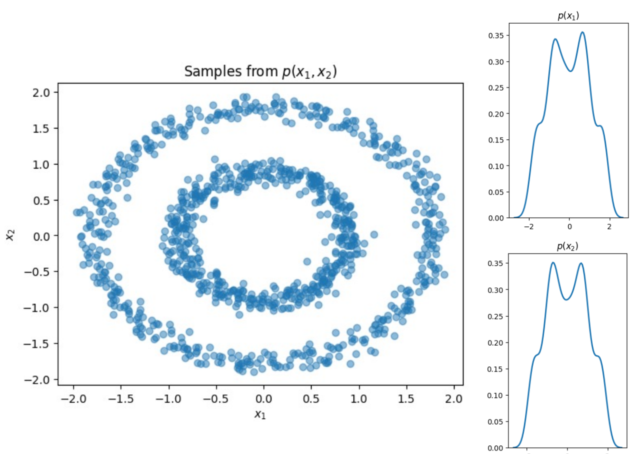
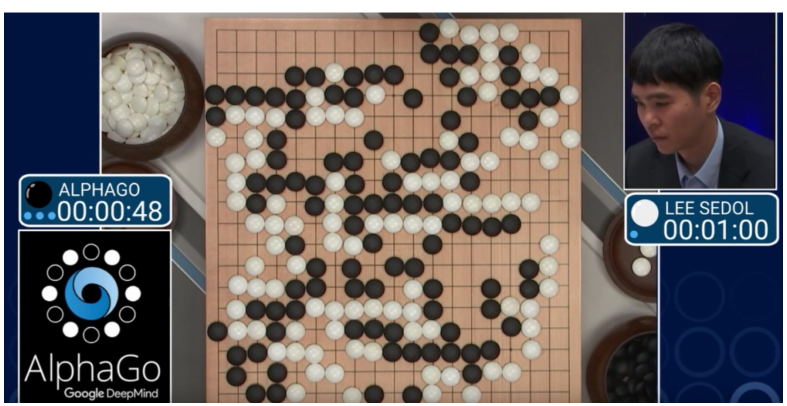


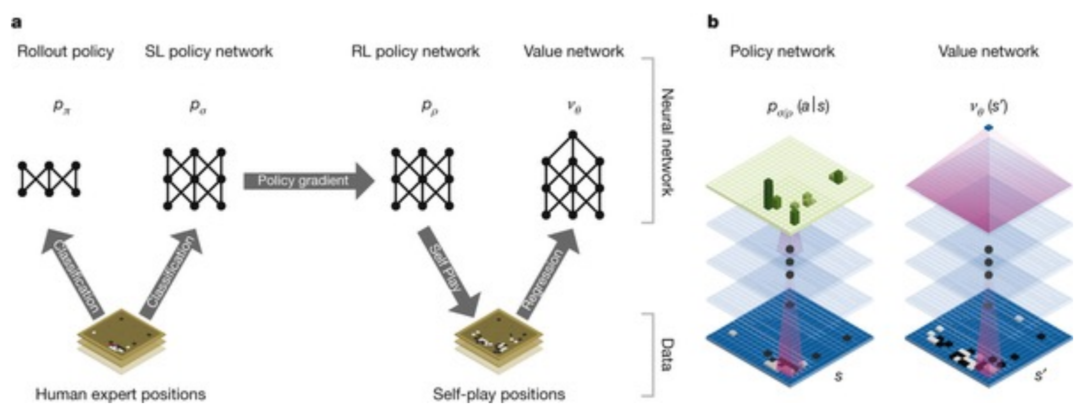
Image Credit - Link

Reinforcement Learning

[Ravikumar]

- Learn to make the best sequence of decisions to achieve a given goal when feedback is often delayed until you reach the goal

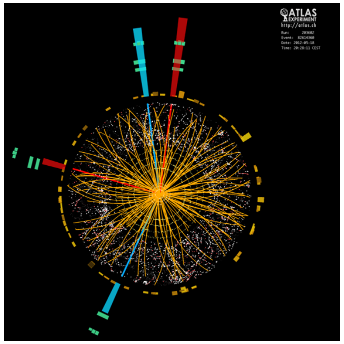




Nature 529, 484–489 (28 January 2016)

Supervised Learning: How does it work?

Supervised Learning: How does it work?



$h(\boldsymbol{x};\ \boldsymbol{w})$
Function with adjustable parameters

Loss Function

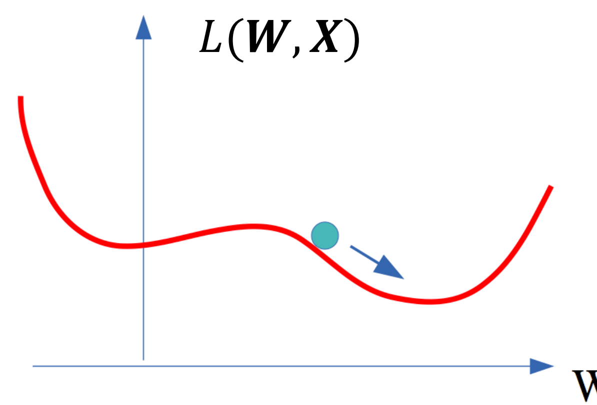
Compare prediction with true label

Loss

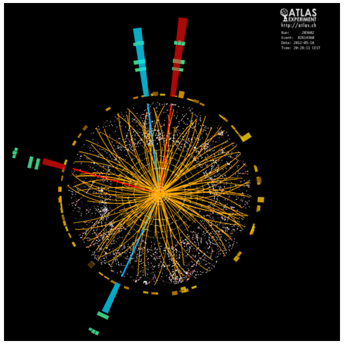
<u>True labels:</u>
Higgs = 1
Bkg = 0

- Design function with adjustable parameters
- Design a Loss function
- Find best parameters which minimize loss

Y. Le Cun



Supervised Learning: How does it work?



$h(\boldsymbol{x};\ \boldsymbol{w})$
Function with adjustable parameters

Loss Function

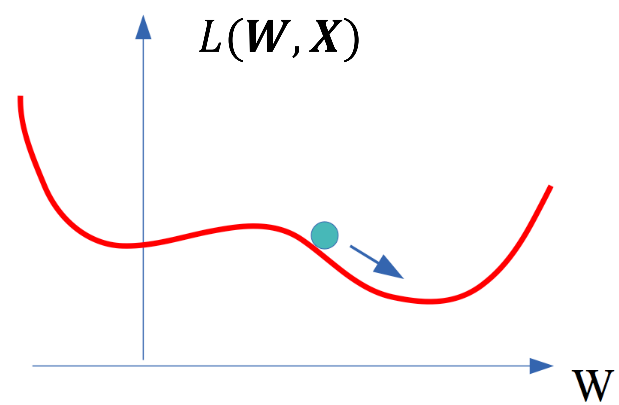
Compare prediction with true label

Loss

True labels:
Higgs = 1
Bkg = 0

- Design function with adjustable parameters
- Design a Loss function
- Find best parameters which minimize loss
  - Use a labeled *training-set* to compute loss
  - Adjust parameters to reduce loss function
  - Repeat until parameters stabilize

Y. Le Cun



Empirical Risk Minimization

$$\arg\min_{\mathbf{w}} \frac{1}{N} \sum_{i=1}^{N} L(h(\mathbf{x}_i; \mathbf{w}), y_i) + \lambda \Omega(\mathbf{w})$$

Average expected loss | Model regularization

- Framework to design learning algorithms
- ***L*** is loss function: compare prediction $h(\cdot)$ to label y
- $\Omega(\boldsymbol{w})$ is a regularizer, penalizing certain values of $\boldsymbol{w}$
  - λ controls how much penalty. Hyperparameter we tune
- Learning is cast as an optimization problem

Example Loss Functions

- Square Error Loss:
  - Often used in regression

$$L(h(\mathbf{x};\mathbf{w}),y) = \big(h(\mathbf{x};\mathbf{w}) - y\big)^2$$

- Cross entropy:
  - With y ∈ {0,1}
  - Often used in classification

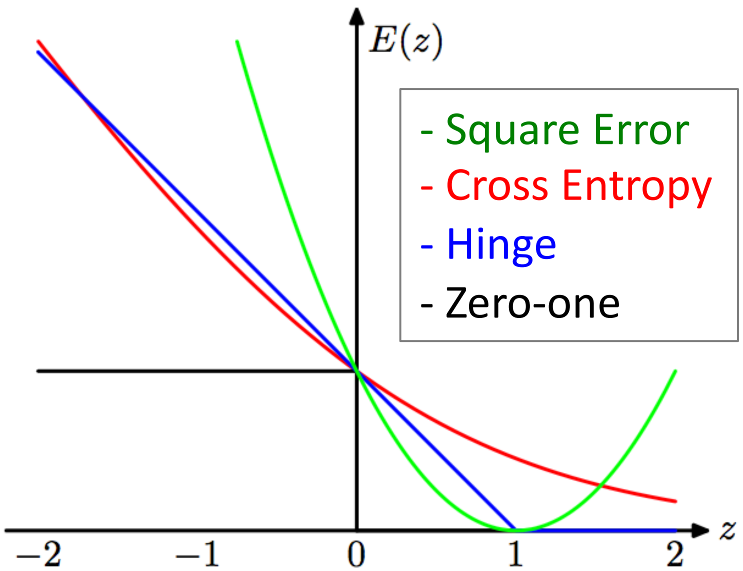
$$L(h(\mathbf{x};\mathbf{w}),y) = -\,y\log h(\mathbf{x};\mathbf{w}) - (1-y)\log(1-h(\mathbf{x};\mathbf{w}))$$

- Hinge Loss:
  - With y ∈ {-1,1}

$$L(h(\mathbf{x};\mathbf{w}),y) = \max(0, 1 - yh(\mathbf{x};\mathbf{w}))$$

- Zero-One loss
  - h(**x**; **w**) predicting label

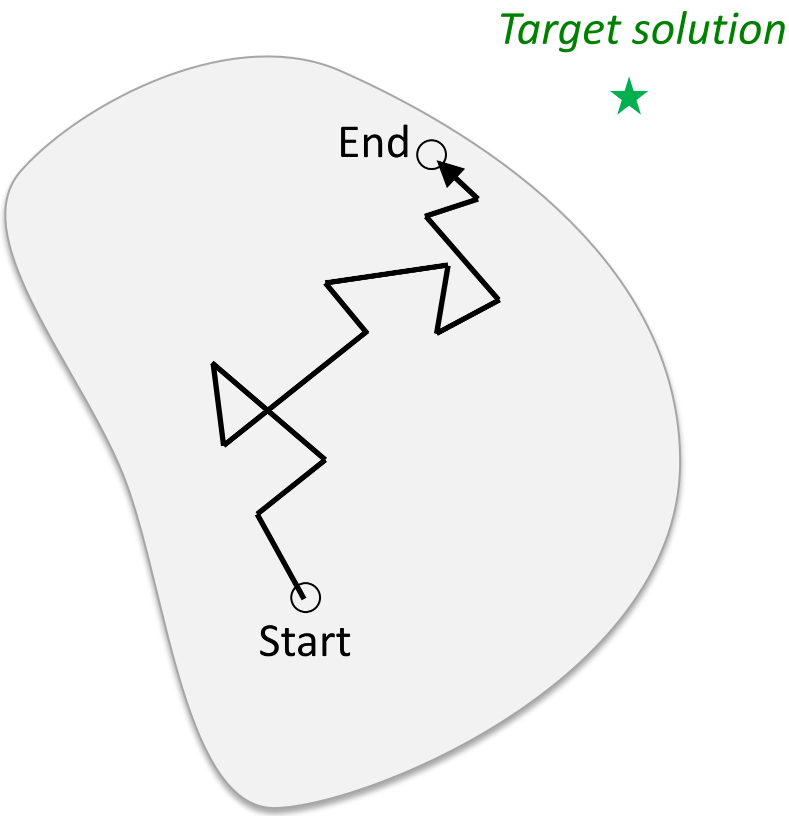
$$L(h(\mathbf{x};\mathbf{w}),y) = 1_{y\neq h(\mathbf{x};\mathbf{w})}$$



[Bishop]

Model Space and Learning Algorithms

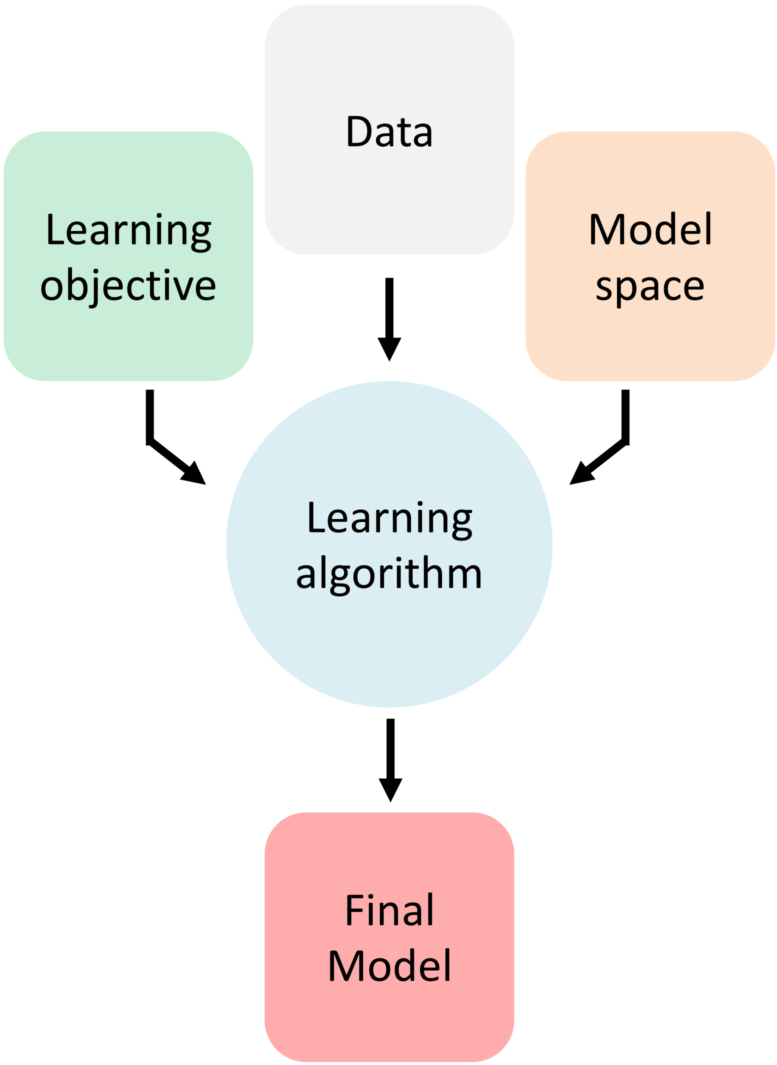
- Choose type of model
  - Each set of parameters is a point in space of models
- Need to find the best model parameters for loss
- **Learning is like a search** through space of models, **guided by the data**
- Various possibilities
  - Exhaustive search
  - Closed for solutions (rare)
  - Iterative optimization



Space of Possible Models

Putting It All Together

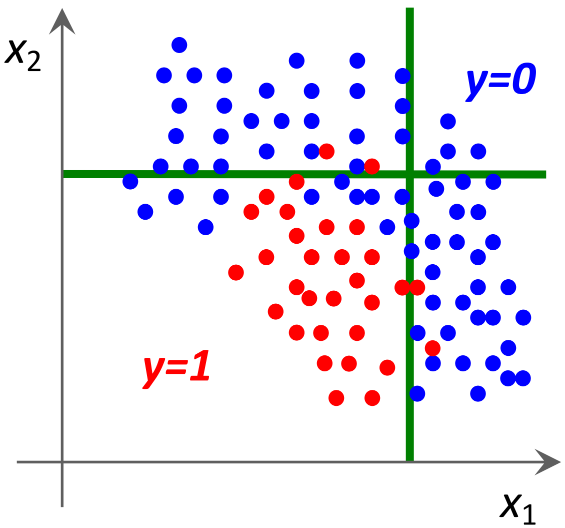
- Gather data to be used
- Propose a space of possible models
- Define what "good" means with loss function / learning objective
- Use learning algorithm to find best model



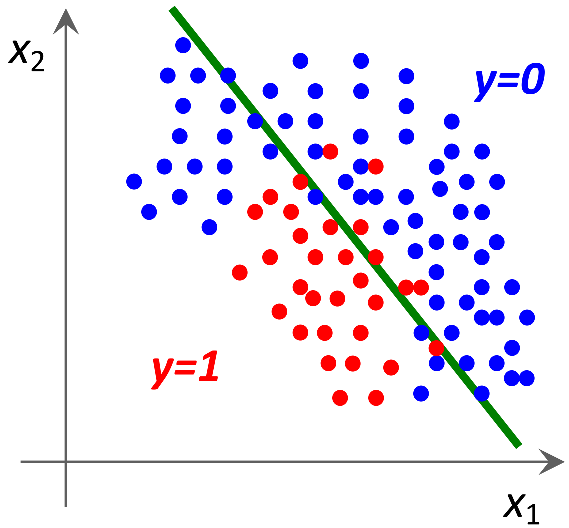
Linear Classification

Classification

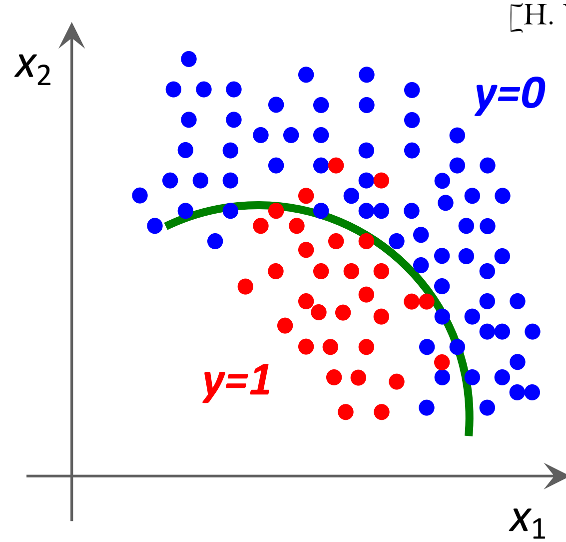
[H. Voss]



Rectangular cuts

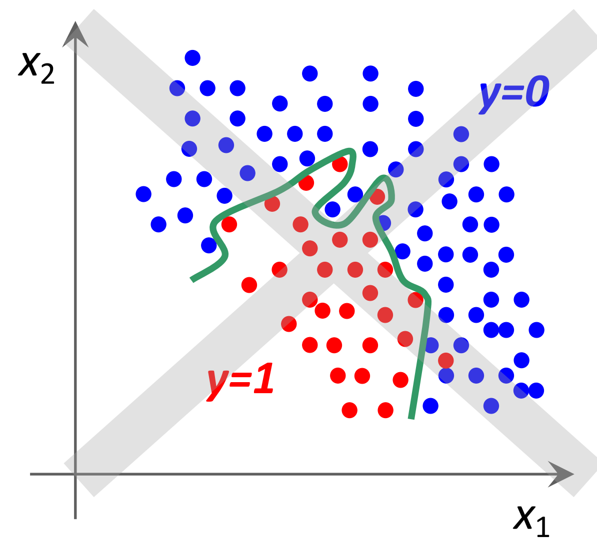


Linear discriminant



Nonlinear discriminant

- Learn a function to separate different classes of data
- Avoid over-fitting:
  - Learning too fine details about training sample that will not generalize to unseen data

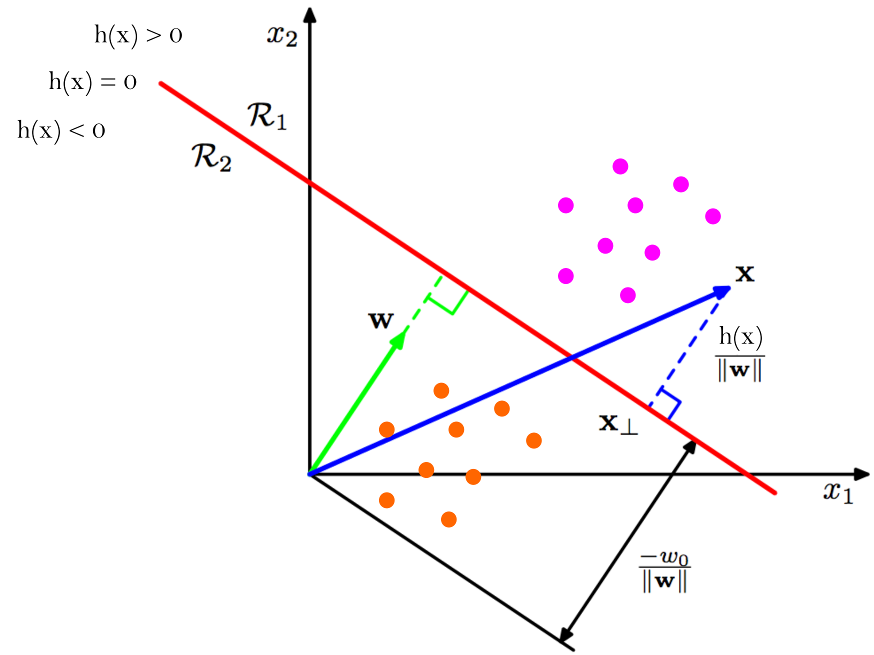


Linear Decision Boundaries

- Separate two classes:
  - $\boldsymbol{x}_i \in \mathbb{R}^m$
  - $y_i \in \{-1,1\}$

- Linear discriminant model

$$h(\boldsymbol{x};\ \boldsymbol{w}) = \boldsymbol{w}^T\boldsymbol{x} + b$$



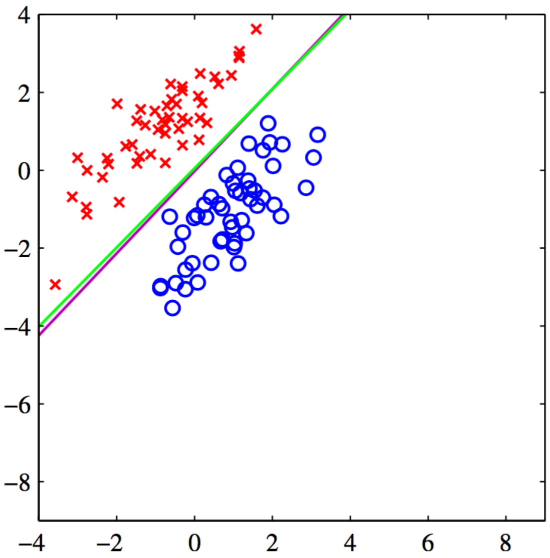
[Bishop]

- **Decision boundary** defined by hyperplane

$$h(\boldsymbol{x};\ \boldsymbol{w}) = \boldsymbol{w}^T\boldsymbol{x} + b = 0$$

- ***Class predictions:*** Predict class 0 if $h(\boldsymbol{x}_i\ ;\ \boldsymbol{w}) < 0$, else class 1

Linear Classifier with Least Squares?

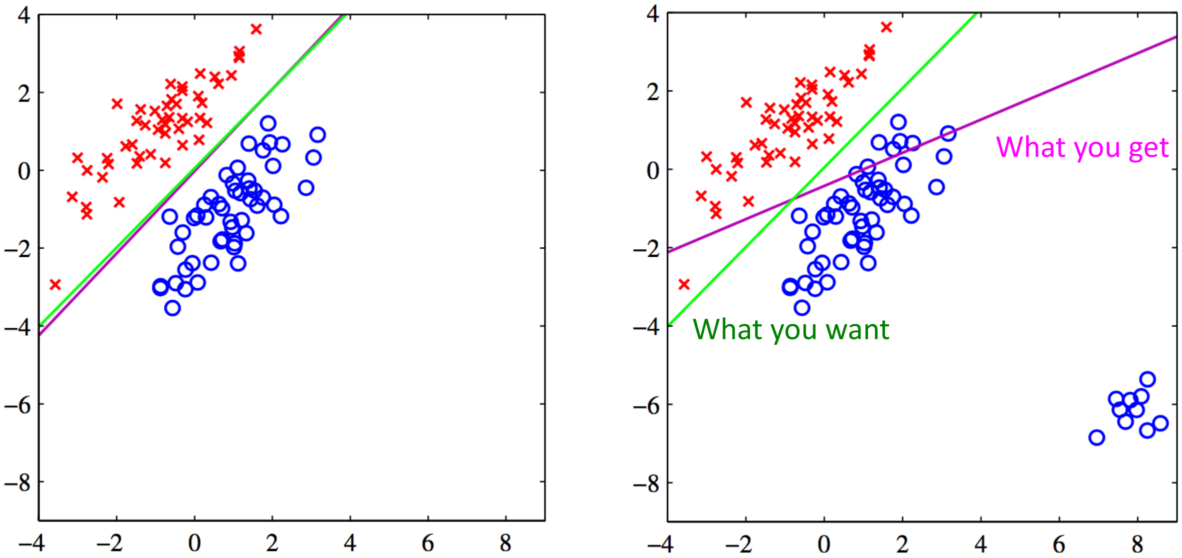


$$L(\mathbf{w}) = \frac{1}{2} \sum_i (y_i - \mathbf{w}^T \mathbf{x}_i)^2$$

[Bishop]

- Why not use least squares loss with binary targets?

Linear Classifier with Least Squares?



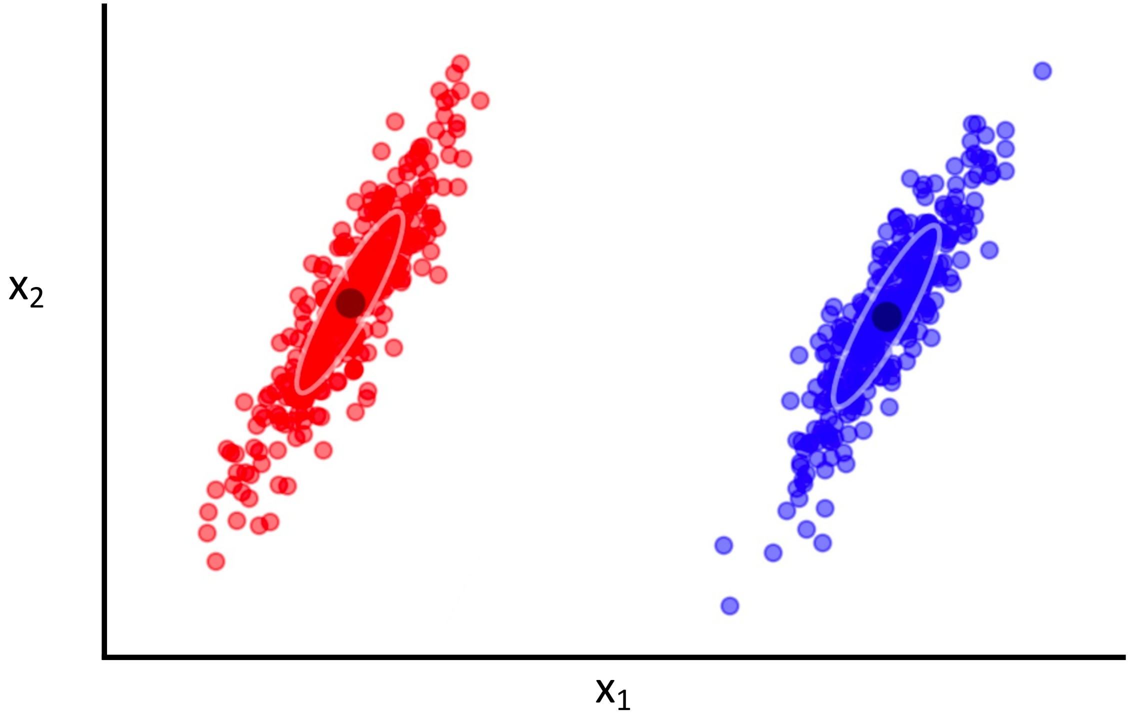
$$L(\mathbf{w}) = \frac{1}{2}\sum_i (y_i - \mathbf{w}^T\mathbf{x}_i)^2$$

[Bishop]

- Why not use least squares loss with binary targets?
  - Penalized even when predict class correctly
  - Least squares is very sensitive to outliers

Linear Discriminant Analysis

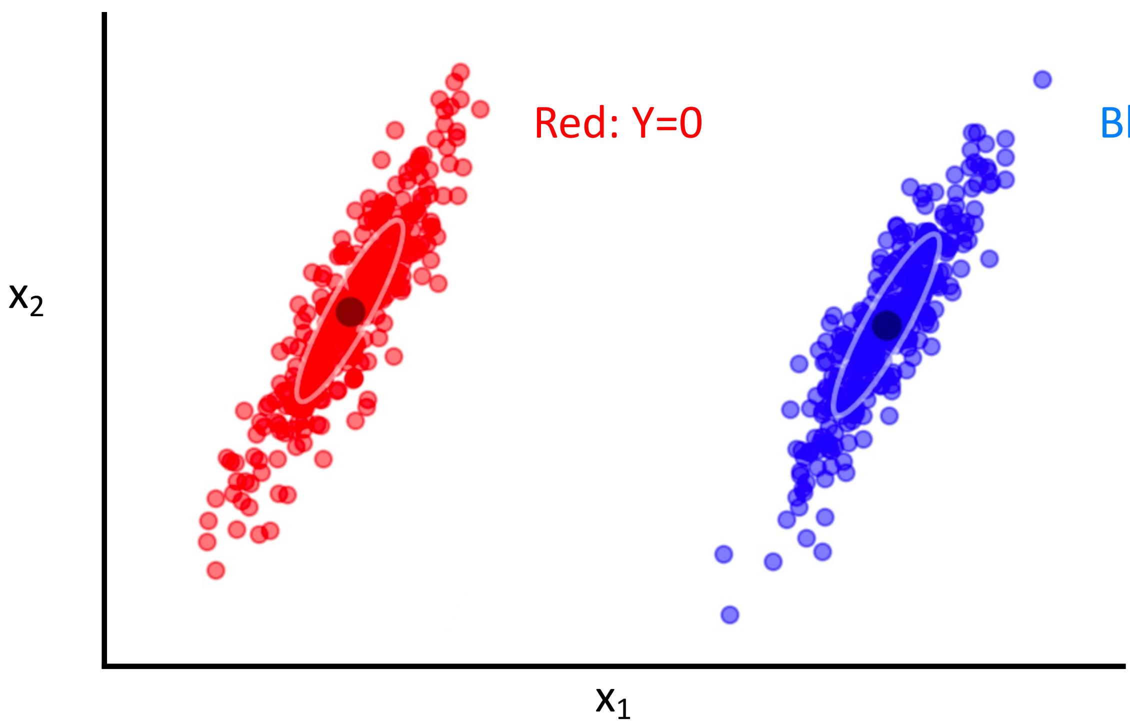
- Goal: Separate data from two classes / populations



Linear Discriminant Analysis

- Goal: Separate data from two classes / populations

- Data from joint distribution $(\boldsymbol{x}, y) \sim p(\boldsymbol{X}, Y)$
  - Features: $\boldsymbol{x} \in \mathbb{R}^m$
  - Labels: $y \in \{0,1\}$



Linear Discriminant Analysis

- Goal: Separate data from two classes / populations

- Data from joint distribution $(\boldsymbol{x}, y) \sim p(\boldsymbol{X}, Y)$
  - Features: $\boldsymbol{x} \in \mathbb{R}^m$
  - Labels: $y \in \{0,1\}$

- Breakdown the joint distribution:

$$p(x, y) = p(x|y)p(y)$$

Likelihood:
Distribution of features
for a given class

Prior:
Probability of each class

Linear Discriminant Analysis

- Goal: Separate data from two classes / populations
- Data from joint distribution $(\boldsymbol{x}, y) \sim p(\boldsymbol{X}, Y)$
  - Features: $\boldsymbol{x} \in \mathbb{R}^m$
  - Labels: $y \in \{0,1\}$
- Breakdown the joint distribution:

$$p(x, y) = p(x|y)p(y)$$

- Assume likelihoods are Gaussian

$$p(x|y) = \frac{1}{\sqrt{(2\pi)^m|\Sigma|}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_y)^T \Sigma^{-1}(\boldsymbol{x} - \boldsymbol{\mu}_y)\right)$$

Predicting the Class

- Separating classes → Predict the class of a point **x**

$$p(y = 1|\mathbf{x})$$

Want to build classifier to predict label y given input **x**

Predicting the Class

- Separating classes → Predict the class of a point **x**

$$p(y=1|\mathbf{x}) = \frac{p(\mathbf{x}|y=1)p(y=1)}{p(\mathbf{x})}$$

Bayes Rule

Predicting the Class

- Separating classes → Predict the class of a point **x**

$$p(y=1|\mathbf{x}) = \frac{p(\mathbf{x}|y=1)p(y=1)}{p(\mathbf{x})}$$

Bayes Rule

$$= \frac{p(\mathbf{x}|y=1)p(y=1)}{p(\mathbf{x}|y=0)p(y=0) + p(\mathbf{x}|y=1)p(y=1)}$$

Marginal definition

Predicting the Class

- Separating classes → Predict the class of a point **x**

$$p(y=1|\mathbf{x}) = \frac{p(\mathbf{x}|y=1)p(y=1)}{p(\mathbf{x})}$$

Bayes Rule

$$= \frac{p(\mathbf{x}|y=1)p(y=1)}{p(\mathbf{x}|y=0)p(y=0) + p(\mathbf{x}|y=1)p(y=1)}$$

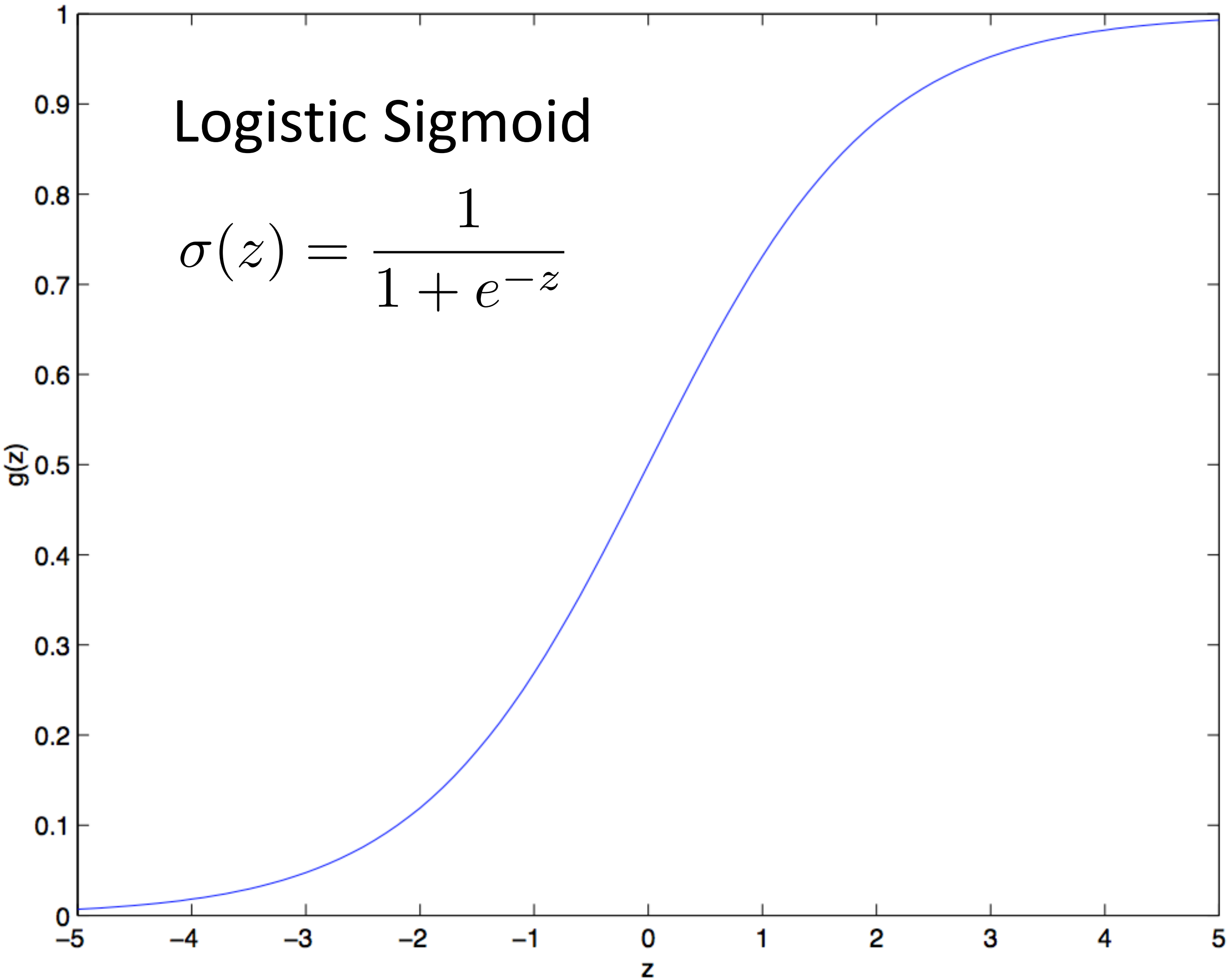
Marginal definition

$$= \frac{1}{1 + \frac{p(\mathbf{x}|y=0)p(y=0)}{p(\mathbf{x}|y=1)p(y=1)}}$$

$$= \frac{1}{1 + \exp\left(\log \frac{p(\mathbf{x}|y=0)p(y=0)}{p(\mathbf{x}|y=1)p(y=1)}\right)}$$

Why?

Logistic Sigmoid Function



Predicting Classes with Gaussian Likelihoods

$$p(y=1|\mathbf{x}) = \sigma\Big(\log\frac{p(\mathbf{x}|y=1)}{p(\mathbf{x}|y=0)} + \log\frac{p(y=1)}{p(y=0)}\Big)$$

Log-likelihood ratio

Constant w.r.t. **x**

Predicting Classes with Gaussian Likelihoods

$$p(y=1|\mathbf{x}) = \sigma\Big(\log\frac{p(\mathbf{x}|y=1)}{p(\mathbf{x}|y=0)} + \log\frac{p(y=1)}{p(y=0)}\Big)$$

- For our Gaussian data:

$$= \sigma\Big(\log p(\mathbf{x}|y=1) - \log p(\mathbf{x}|y=0) + const.\Big)$$

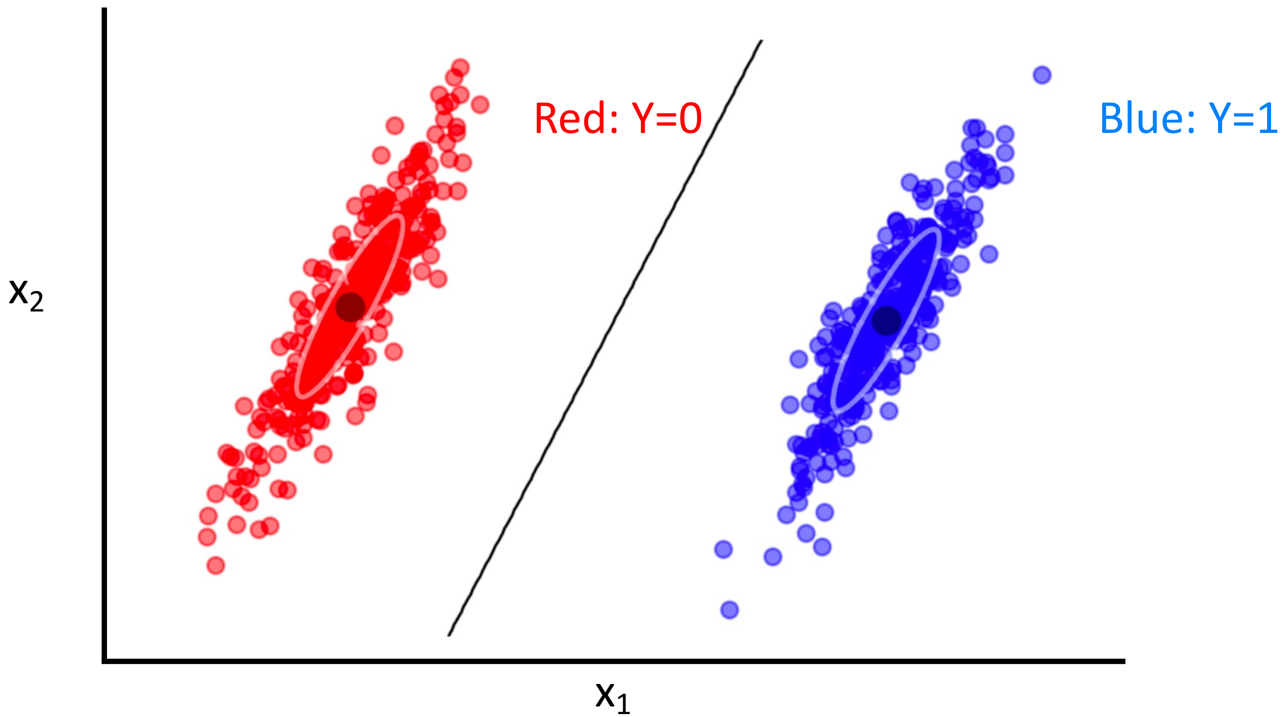
$$= \sigma\Big(-\frac{1}{2}(\mathbf{x}-\mu_1)^T\Sigma^{-1}(\mathbf{x}-\mu_1) + \frac{1}{2}(\mathbf{x}-\mu_0)^T\Sigma^{-1}(\mathbf{x}-\mu_0) + const.\Big)$$

$$= \sigma\Big(\mathbf{w}^T\mathbf{x} + b\Big)$$

Collect terms

What did we learn?

- For this data, the log-likelihood ratio is linear!
  - Line defines boundary to separate the classes
  - Sigmoid turns distance from boundary to probability



Logistic Regression

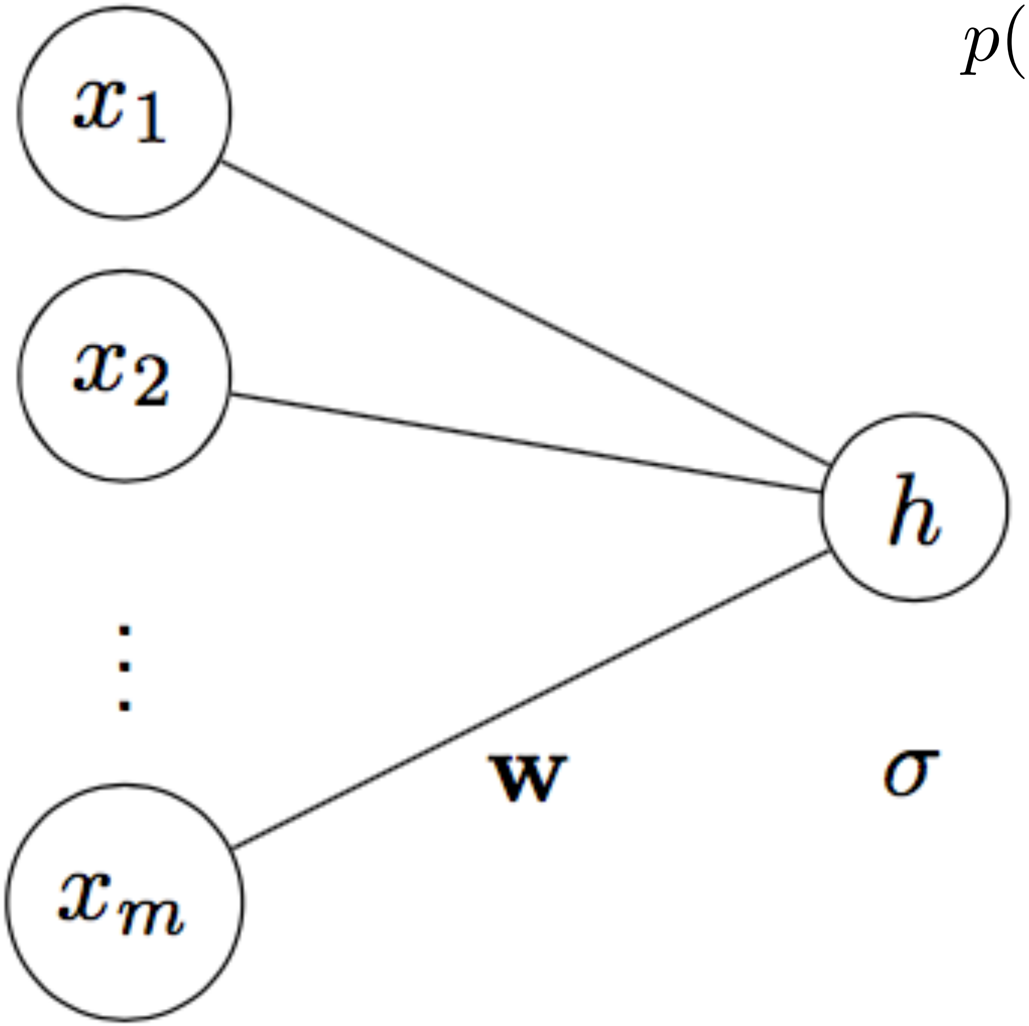
- What if we ignore Gaussian assumption on data?

Model: $$p(y=1|\mathbf{x}) = \sigma\left(\mathbf{w}^T\mathbf{x} + b\right) \equiv h(\mathbf{x};\mathbf{w})$$

- Farther from boundary $\boldsymbol{w}^T\boldsymbol{x} + b = 0$, more certain about class

- Sigmoid converts distance to class probability

Logistic Regression



$$p(y=1|\mathbf{x}) = \sigma\left(\mathbf{w}^T\mathbf{x} + b\right)$$
$$= \frac{1}{1 + e^{-\mathbf{w}^T\mathbf{x}\text{-b}}}$$

This unit is the main building block of Neural Networks!

Logistic Regression

- What if we ignore Gaussian assumption on data?

Model: $$p(y=1|\mathbf{x}) = \sigma\left(\mathbf{w}^T\mathbf{x} + b\right) \equiv h(\mathbf{x};\mathbf{w})$$

- With $p_i \equiv p(y_i = y|\boldsymbol{x}_i)$

$$P(y_i = y|x_i) = \text{Bernoulli}(p_i) = (p_i)^{y_i}(1-p_i)^{1-y_i} = \begin{cases} p_i & \text{if } y_i = 1 \\ 1 - pi & \text{if } y_i = 0 \end{cases}$$

- **Goal**:
  - Given i.i.d. dataset of pairs $(\boldsymbol{x}_i, y_i)$ find **w** and b that maximize likelihood of data

Logistic Regression

- Negative log-likelihood

$$-\ln \mathcal{L} = -\ln \prod_i (p_i)^{y_i} (1 - p_i)^{1-y_i}$$

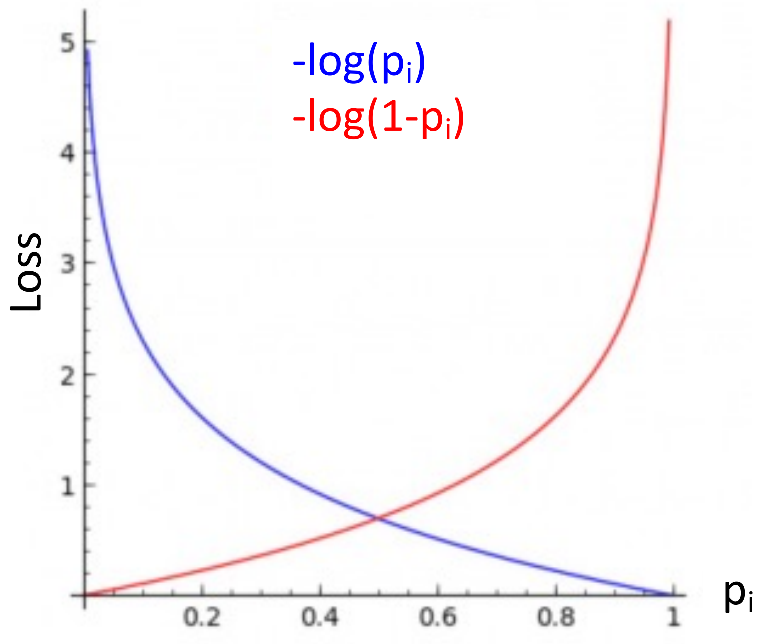
Logistic Regression

- Negative log-likelihood

$$-\ln \mathcal{L} = -\ln \prod_i (p_i)^{y_i} (1-p_i)^{1-y_i}$$

$$= -\sum_i y_i \ln(p_i) + (1-y_i)\ln(1-p_i)$$

binary cross entropy loss function!



Logistic Regression

- Negative log-likelihood

$$
\begin{aligned}
-\ln \mathcal{L} &= -\ln \prod_i (p_i)^{y_i}(1-p_i)^{1-y_i} \\
&= -\sum_i y_i \ln(p_i) + (1-y_i)\ln(1-p_i) \\
&= \sum_i y_i \ln(1+e^{-\mathbf{w}^T\mathbf{x}}) + (1-y_i)\ln(1+e^{\mathbf{w}^T\mathbf{x}})
\end{aligned}
$$

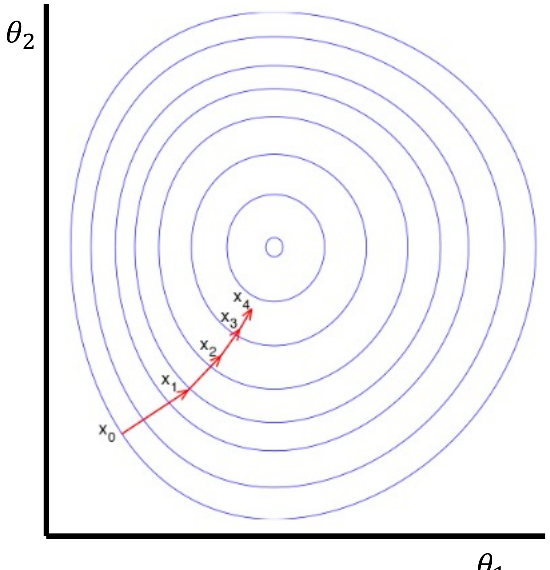
binary cross entropy loss function!

- No closed form solution to $w^* = \arg\min_w -\ln \mathcal{L}(w)$

- How to solve for $\mathbf{w}$?

Gradient Descent

- Minimize loss by repeated gradient steps
  - Compute gradient w.r.t. current parameters: $\nabla_{\theta_i}\mathcal{L}(\theta_i)$
  - Update parameters: $\theta_{i+1} \leftarrow \theta_i - \eta\nabla_{\theta_i}\mathcal{L}(\theta_i)$
  - η is the *learning rate*, controls how big of a step to take



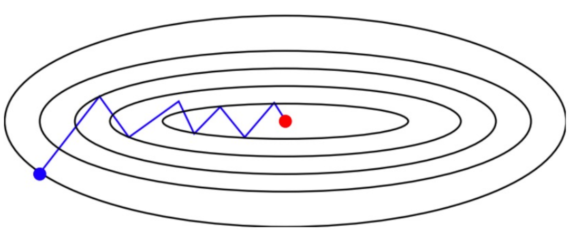
Stochastic Gradient Descent

- Loss is composed of a sum over samples:

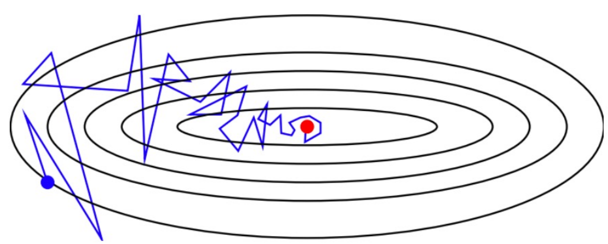
$$\nabla_\theta \mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \nabla_\theta \mathcal{L}\big(y_i, h(x_i; \theta)\big)$$

  – Computing gradient grows linearly with N!

- **(Mini-Batch) Stochastic Gradient Descent**
  – Compute gradient update using 1 random sample (small size batch)
  – Gradient is unbiased → on average it moves in correct direction
  – Tends to be much faster the full gradient descent



Batch gradient descent



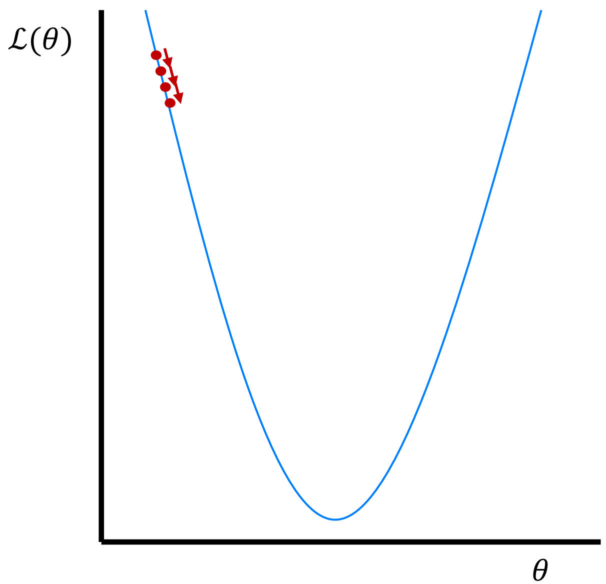
Stochastic gradient descent

Step Sizes

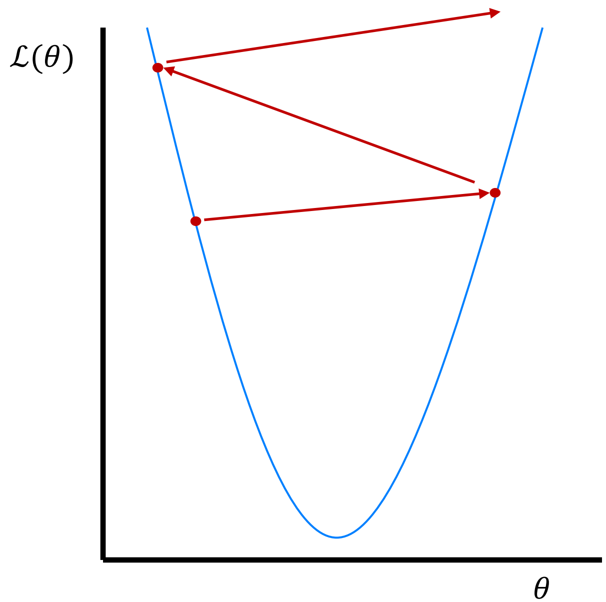
- Too small a learning rate, convergence very slow

- Too large a learning rate, algorithm diverges

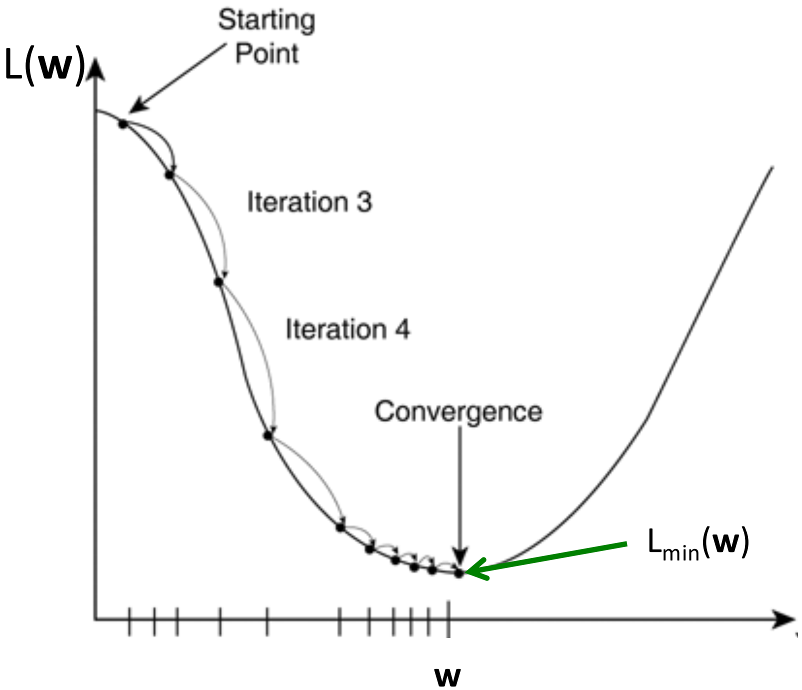
Small Learning rate

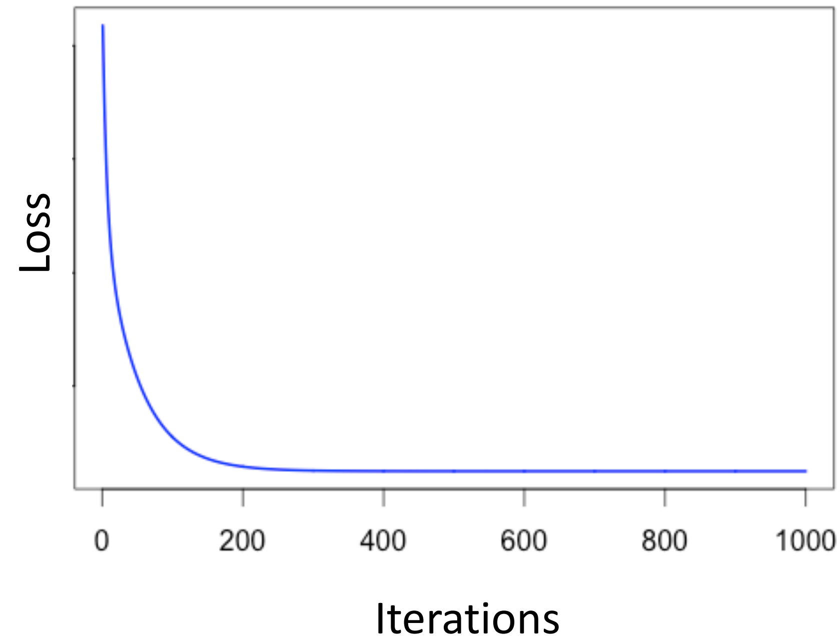


Large Learning rate



Gradient Descent





- Logistic Regression Loss is convex
  - Single global minimum

- Iterations lower loss and move toward minimum

Logistic Regression Example

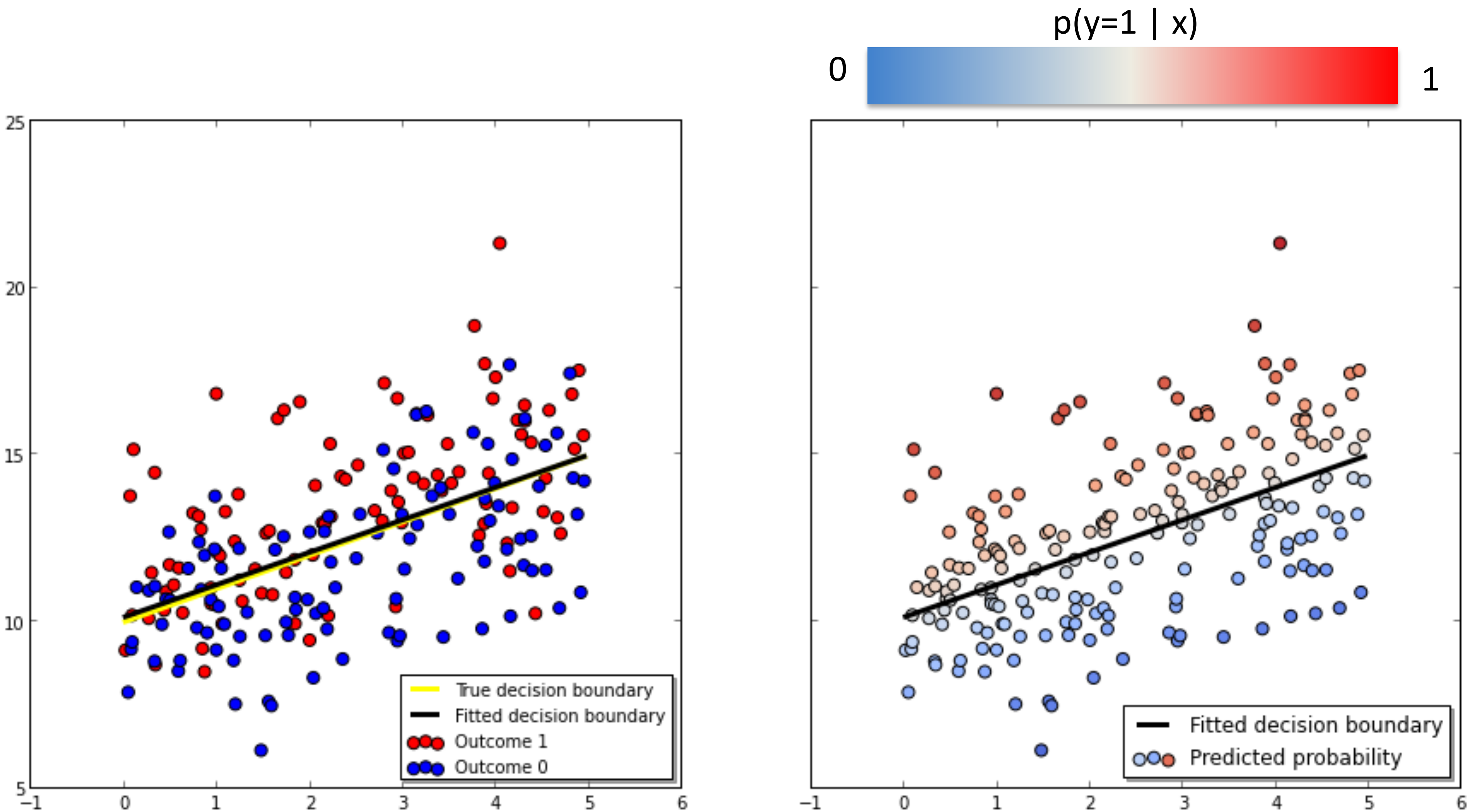
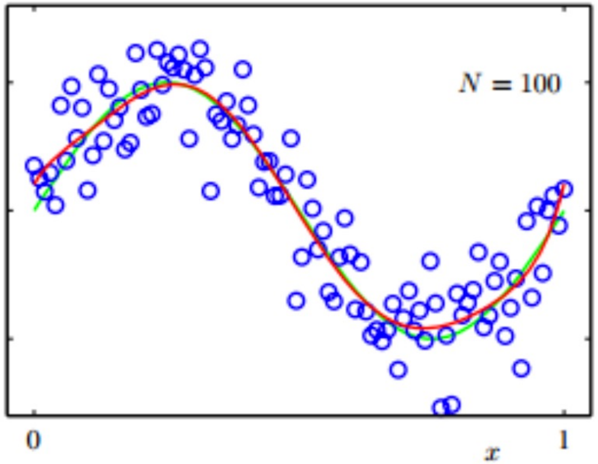


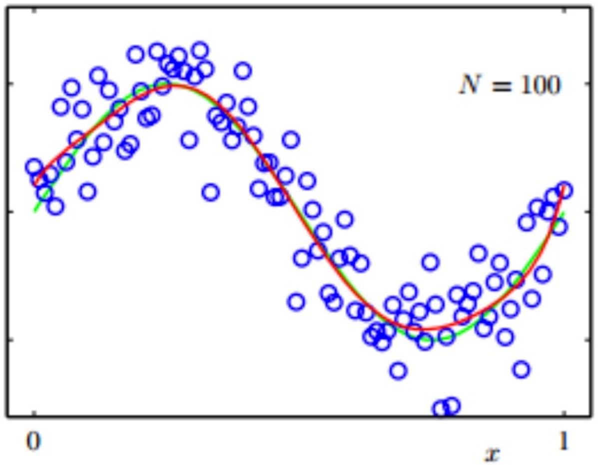
Image source

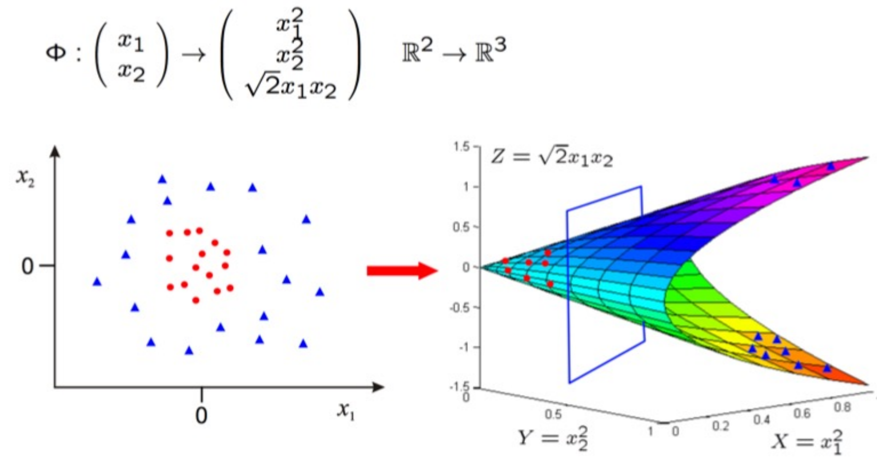
Basis Functions



- What if non-linear relationship between **y** and **x**?

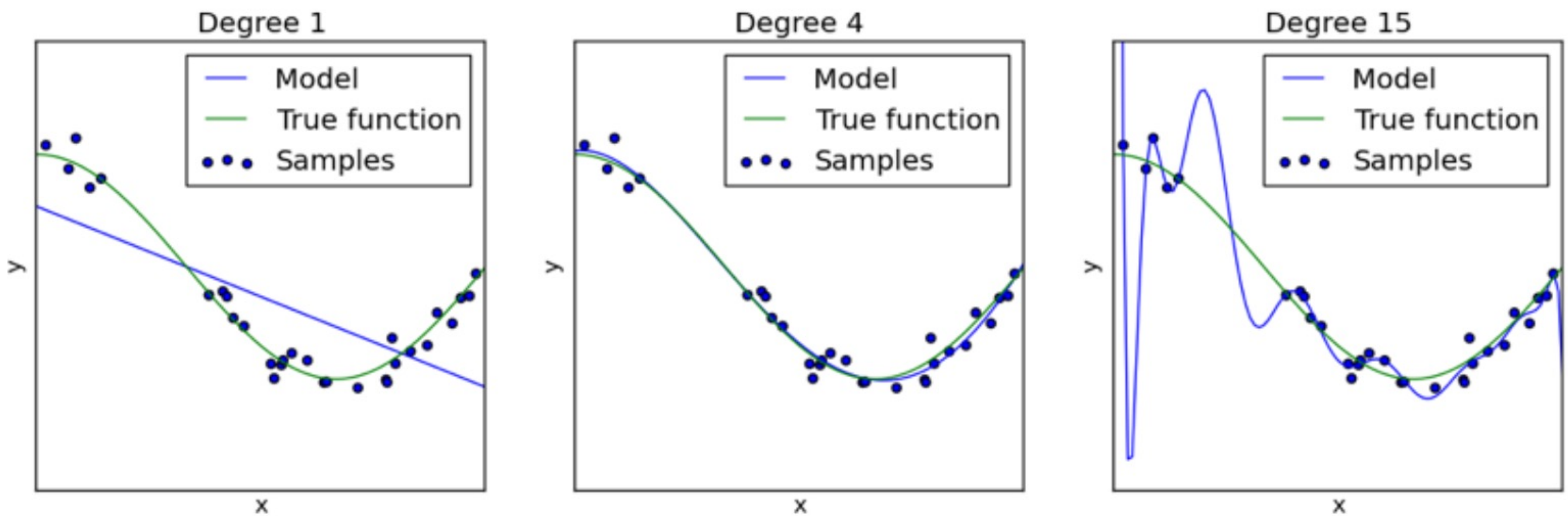
Basis Functions





- What if non-linear relationship between **y** and **x**?
- Choose **basis functions $\phi(x)$** to form new features
  - Example: Polynomial basis $\phi(x) \sim \{1, x, x^2, x^3, \ldots\}$
  - Logistic regression on new features: $h(x; w) = \sigma\left(w^T \phi(x)\right)$
- What basis functions to choose? *Overfit* with too much flexibility?

What is Overfitting



Underfitting

Overfitting

http://scikit-learn.org/

- Models allow us to **generalize** from data
- Different models generalize in different ways

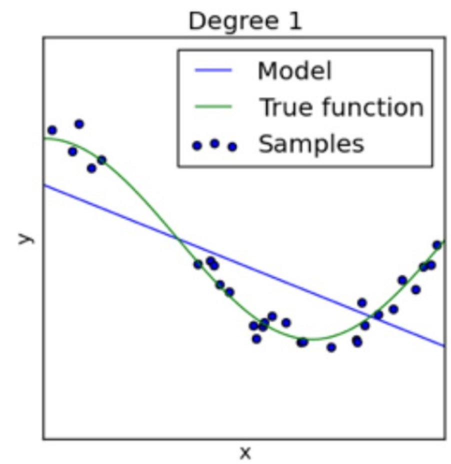
Bias Variance Tradeoff

- generalization error = systematic error + sensitivity of prediction
  (bias) (variance)

Bias Variance Tradeoff

- generalization error = systematic error + sensitivity of prediction
  (bias) (variance)

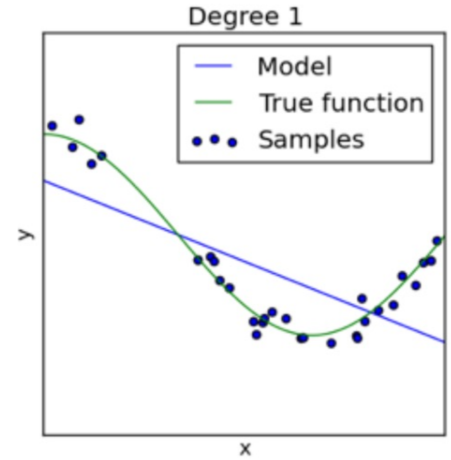
- Simple models under-fit:
  will deviate from data (high bias)
  but will not be influenced by
  peculiarities of data (low variance).



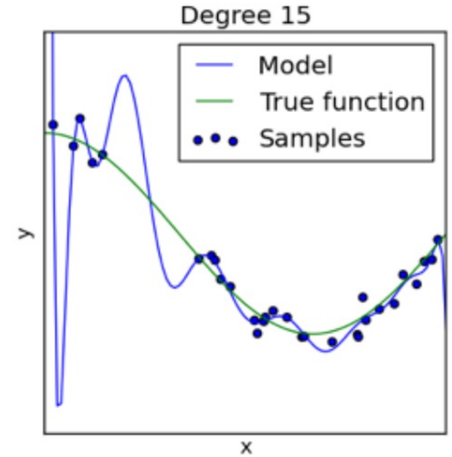
Bias Variance Tradeoff

- generalization error = systematic error + sensitivity of prediction
  (bias) (variance)

- Simple models under-fit:
  will deviate from data (high bias)
  but will not be influenced by
  peculiarities of data (low variance).



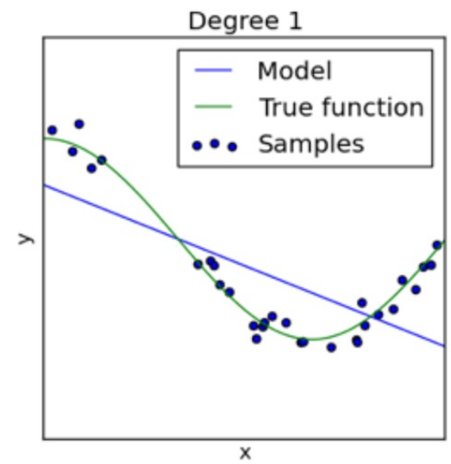
- Complex models over-fit:
  will not deviate systematically from
  data (low bias) but will be very
  sensitive to data (high variance).



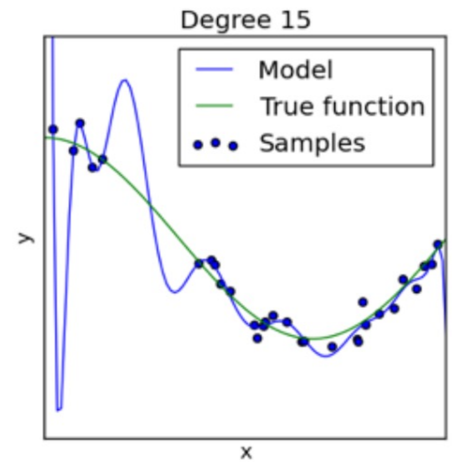
Bias Variance Tradeoff

- generalization error = systematic error + sensitivity of prediction
  (bias) (variance)

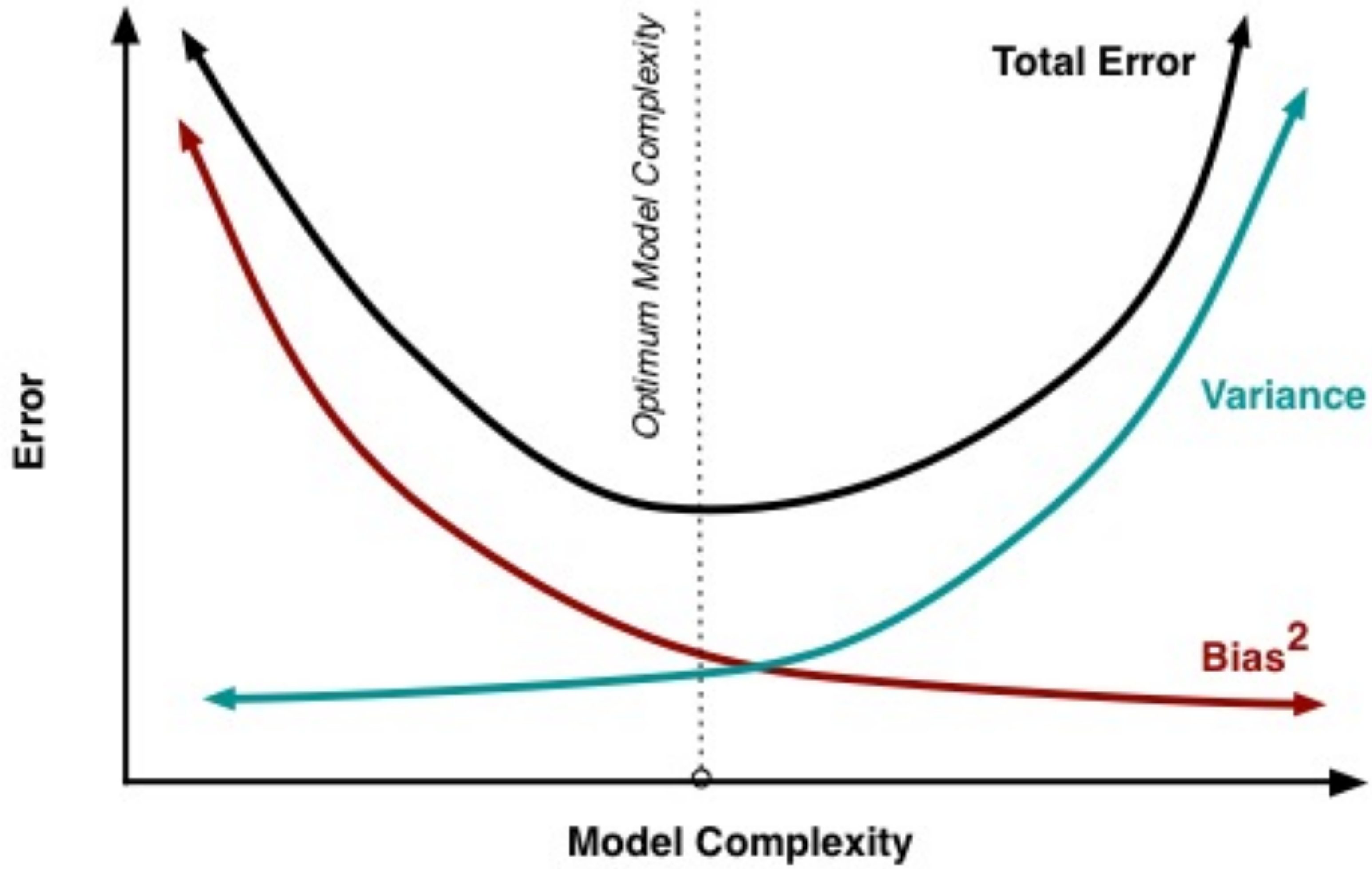
- Simple models under-fit:
  will deviate from data (high bias)
  but will not be influenced by
  peculiarities of data (low variance).



- Complex models over-fit:
  will not deviate systematically from
  data (low bias) but will be very
  sensitive to data (high variance).
  - **As dataset size grows, can reduce variance! Use more complex model**



Bias Variance Tradeoff

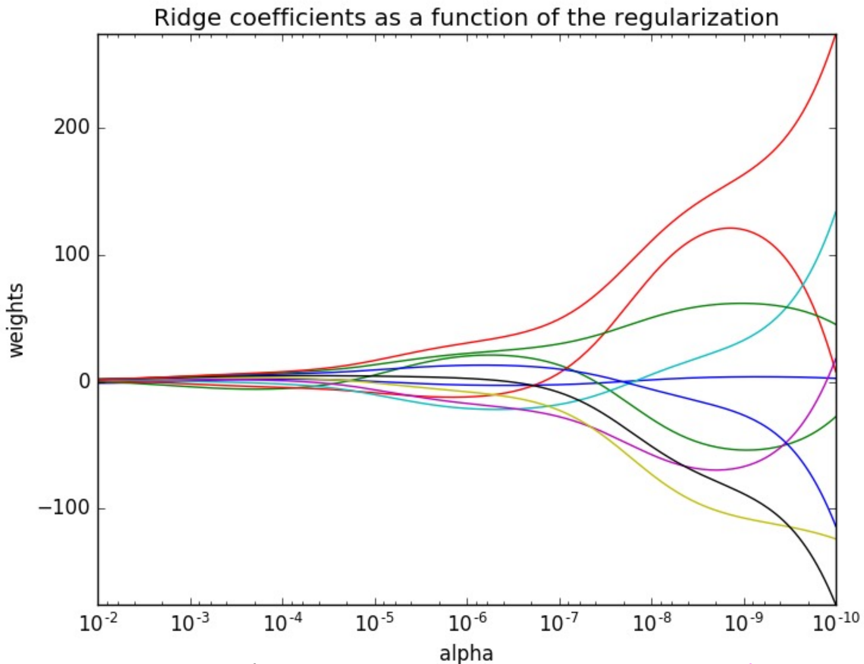


Regularization – Control Complexity

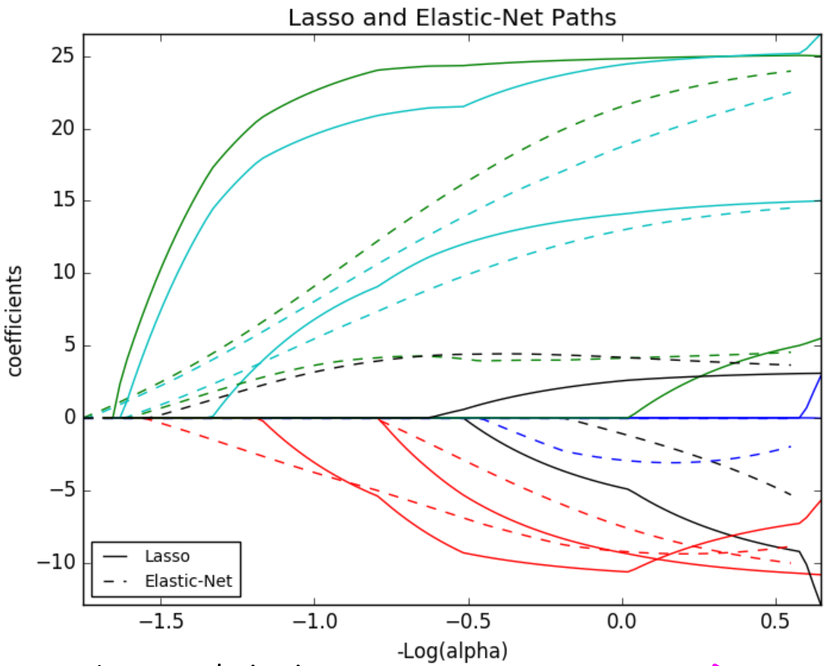
$$L(\mathbf{w}) = \frac{1}{2}(\mathbf{y} - \mathbf{X}\mathbf{w})^2 + \alpha\Omega(\mathbf{w})$$

$$L2: \quad \Omega(\mathbf{w}) = ||\mathbf{w}||^2$$

$$L1: \quad \Omega(\mathbf{w}) = ||\mathbf{w}||$$



Less regularization →



Less regularization →

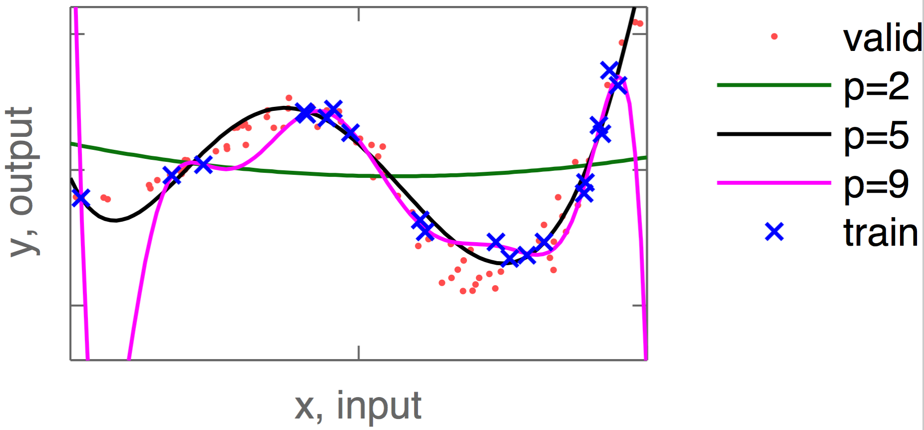
- L2 keeps weights small, L1 keeps weights sparse!
- But how to choose hyperparameter α?

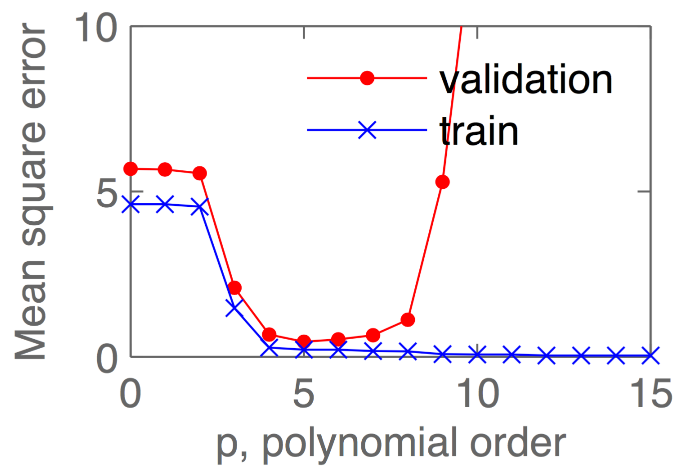
http://scikit-learn.org/

How to Measure Generalization Error?

Training set | Validation set | Test set

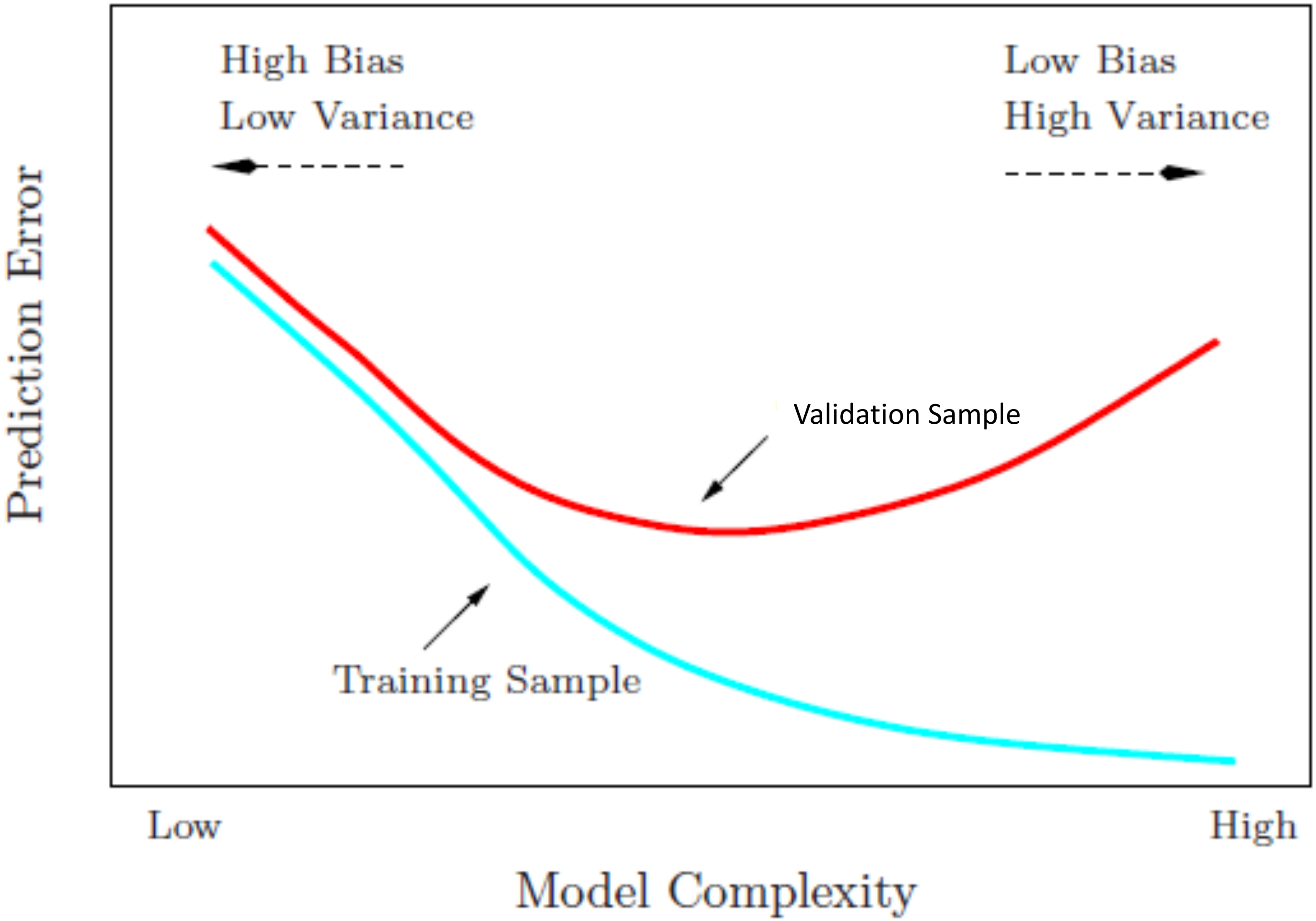
- Split dataset into multiple parts
- **Training set**
  - Used to fit model parameters
- **Validation set**
  - Used to check performance on independent data and tune hyper parameters
- **Test set**
  - final evaluation of performance after all hyper-parameters fixed
  - Needed since we tune, or "peek", performance with validation set





[Murray]

How to Measure Generalization Error?



Summary

- Machine learning uses mathematical & statistical models learned from data to characterize patterns and relations between inputs, and use this for inference / prediction

- Machine learning comes in many forms, much of which has probabilistic and statistical foundations and interpretations (i.e. *Statistical Machine Learning*)

- Machine learning is a powerful toolkit to analyze data

  – Linear methods can help greatly in understanding data

  – Choosing a model for a given problem is difficult, keep in mind the bias-variance tradeoff when building an ML mode

References

- http://scikit-learn.org/
- [Bishop] Pattern Recognition and Machine Learning, Bishop (2006)
- [ESL] Elements of Statistical Learning (2nd Ed.) Hastie, Tibshirani & Friedman 2009
- [Murray] Introduction to machine learning, Murray
  - http://videolectures.net/bootcamp2010_murray_iml/
- [Ravikumar] What is Machine Learning, Ravikumar and Stone
  - http://www.cs.utexas.edu/sites/default/files/legacy_files/research/documents/MLSS-Intro.pdf
- [Parkes] CS181, Parkes and Rush, Harvard University
  - http://cs181.fas.harvard.edu
- [Ng] CS229, Ng, Stanford University
  - http://cs229.stanford.edu/
- [Rogozhnikov] Machine learning in high energy physics, Alex Rogozhnikov
  - https://indico.cern.ch/event/497368/
- [Fleuret] Francois Fleuret, EE559 Deep Learning, EPFL, 2018
  - https://documents.epfl.ch/users/f/fl/fleuret/www/dlc/