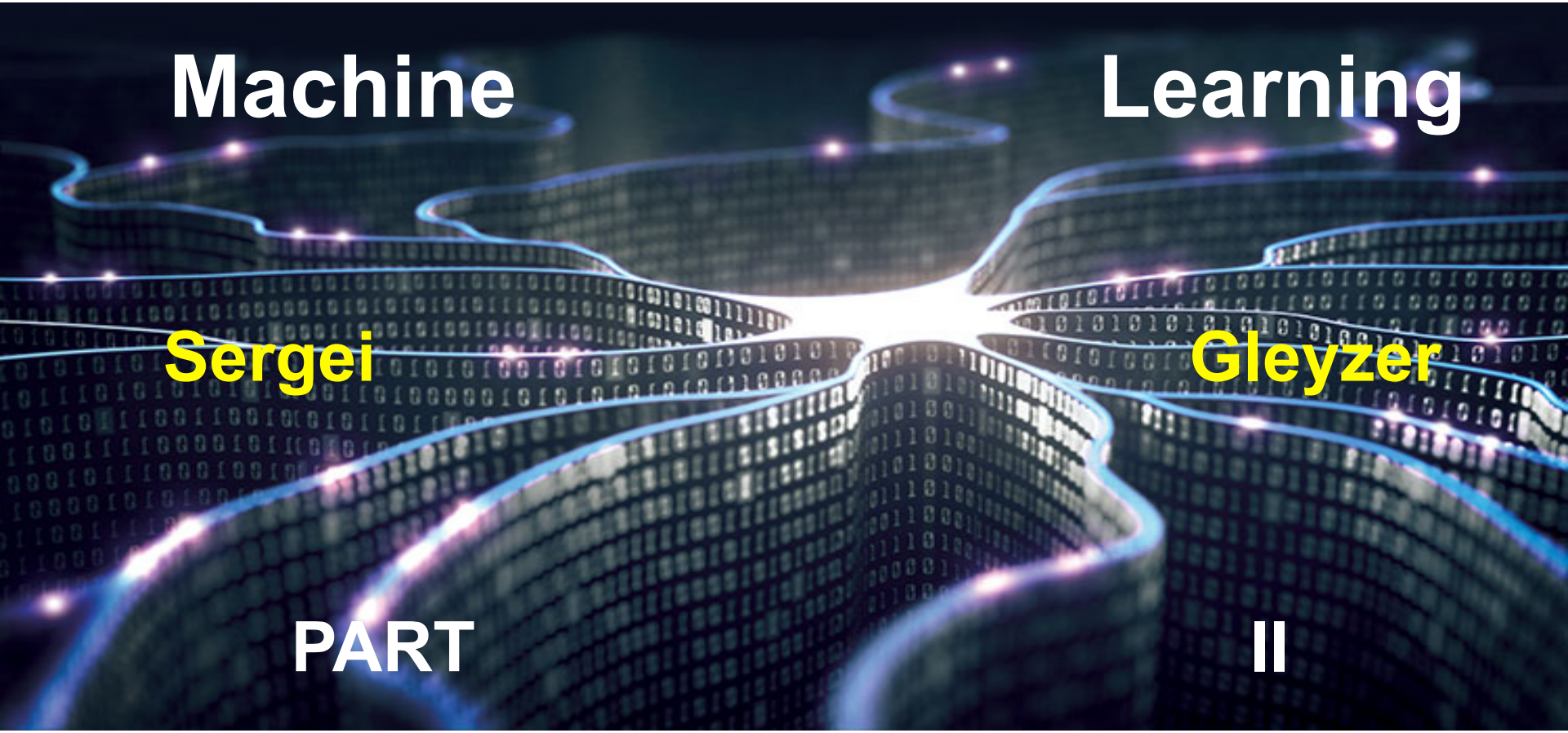
UF UNIVERSITY of FLORIDA



UPRM ML Lectures

Apr 25, 2019

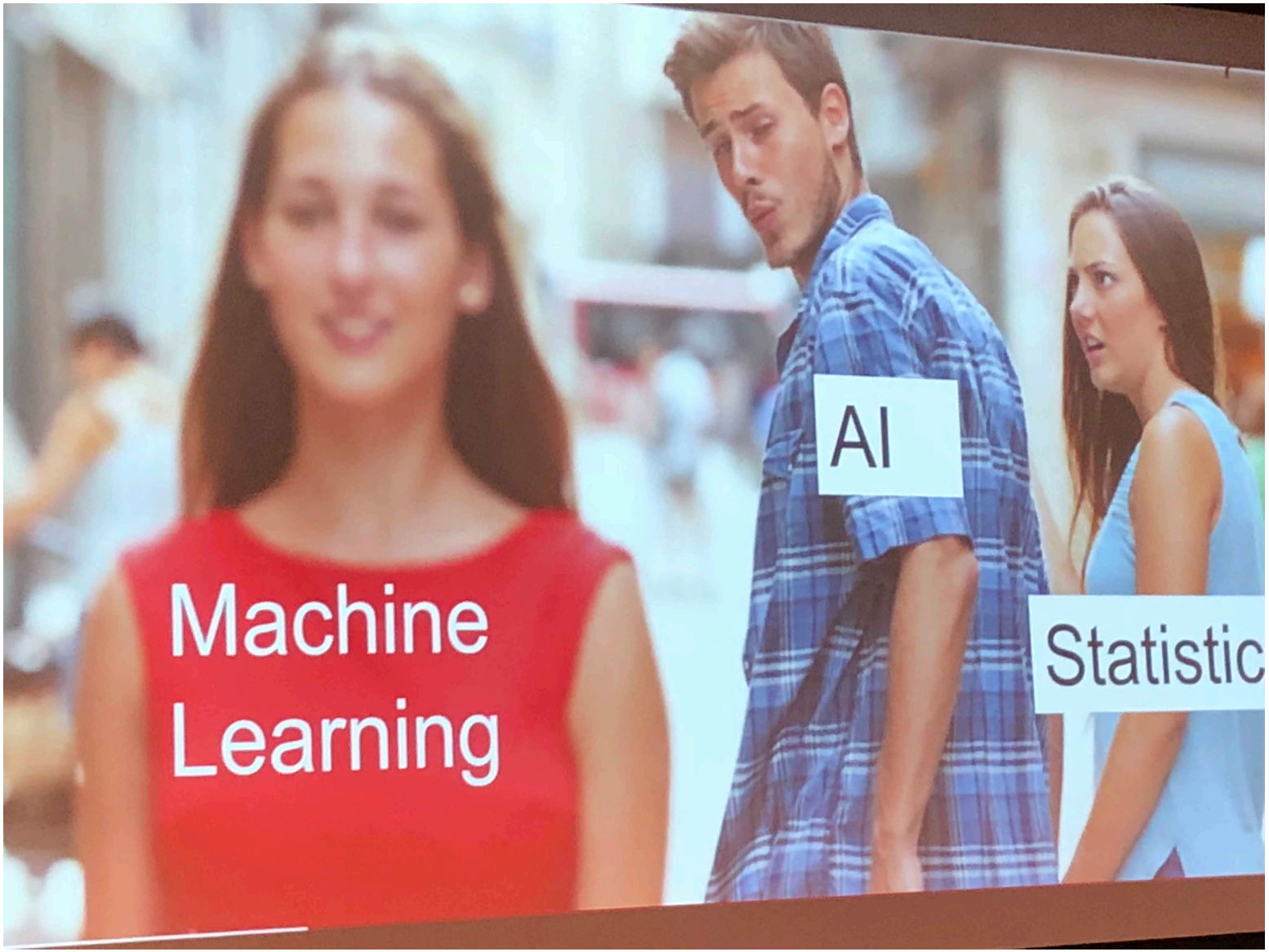
Recap

Machine Learning

What is Machine Learning?

- Study of algorithms that
  improve their performance **P**
  for a given task **T**
  with more experience **E**

Sample tasks: identifying faces, Higgs bosons


Machine
Learning
AI
Statistic

Machine Learning

Many methods (e.g., neural networks, boosted decision trees, rule-based systems, random forests,…) use the

quadratic loss

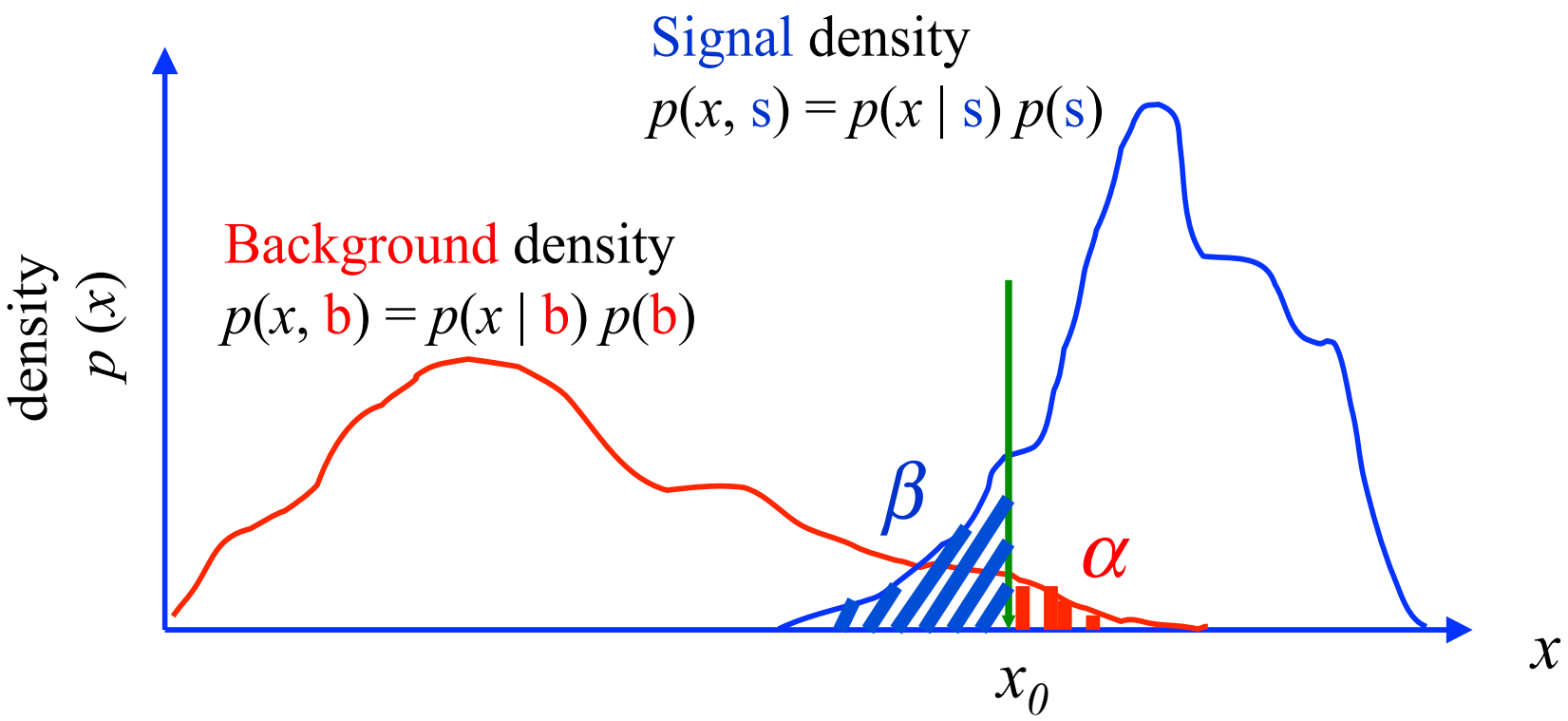
$$L(y, f(x, w)) = [y - f(x, w)]^2$$

and choose $f(x, w^*)$ by minimizing the

constrained mean square empirical risk

$$R[f_w] = \frac{1}{N}\sum_{i=1}^{N}[y_i - f(x_i, w)]^2 + C(w)$$

Classification Theory



Optimality criterion: minimize the error rate, $\alpha + \beta$

Classification Theory

The total loss L arising from classification errors is given by

$$L = L_b \int H(f)\, p(x,b)\, dx + L_s \int [1 - H(f)]\, p(x,s)\, dx$$

Cost of background misclassification

Cost of signal misclassification

where $f(x) = 0$ defines a decision boundary
such that $f(x) > 0$ defines the acceptance region

$H(f)$ is the Heaviside step function:

$H(f) = 1$ if $f > 0$, 0 otherwise

Decision Rules

Decision Rules:

- Deconstruct **Decision Tree**
- Set of **if – then – else** rules
  - Example of **"weak"** learners (better than random guessing)
  - In weighted ensemble become a competitive classifier
    - RuleFit: Friedman, Popescu, 2005

Ensemble Methods

Ensemble Methods

Suppose you have a **collection** of discriminants $f(x, w_k)$, which, individually, perform only **marginally** better than random guessing.

$$f(x) = a_0 + \sum_{k=1}^{K} a_k f(x, w_k)$$

From such discriminants, **weak learners**, it is possible to build highly effective ones by averaging over them:

Jerome Friedman & Bogdan Popescu (2008)

Ensemble Methods

Bagging (bootstrap aggregation)

- Each tree trained on **bootstrap sample** drawn from training set

Random Forest

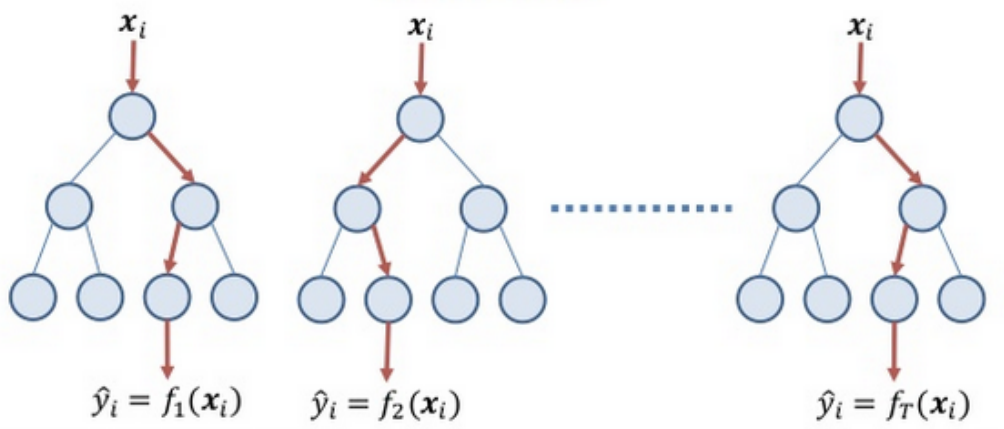
- Bagging with randomized trees
- Random subsets of features used at each split

Boosting

- Train on a **different weighting** of full training set, usually decision trees but is more general

Random Forests

Random Forest (L. Breinman, 2001)



- **Bagging** plus random subset of features for splitting at each node
- **Benefits:** excellent accuracy, avoids over-fitting

Boosting

Boosting (R. Shapire 1990)

- Turn **weak** learners into **strong** with weighted ensemble of **iterative** learners
- Boosting algorithms differ in how to weigh instances
  - Adaptation
  - **Benefits:** excellent accuracy

Adaptive Boosting

Adaptive Boosting

AdaBoost (Freund & Shapire 1997)

- **Train in stages:** adaptive weights
- **Misclassified** events get a larger weight going into the next training stage
  - Classify with a majority vote from all trees

- **Works** very well to improve classification power of "greedy" decision trees

Adaptive Boosting

Repeat *K* times:

1. Create a decision tree $f(x, \boldsymbol{w})$
2. Compute its error rate $\boldsymbol{\varepsilon}$ on the ***weighted*** training set
3. Compute $\boldsymbol{\alpha} = \ln(1-\boldsymbol{\varepsilon}) / \boldsymbol{\varepsilon}$
4. Modify training set: ***increase weight*** of ***incorrectly classified examples*** relative to the weights of those that are correctly classified

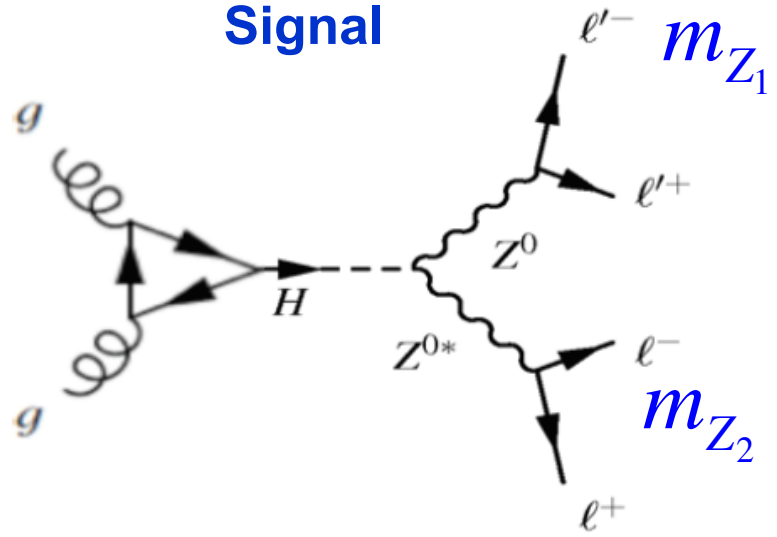
Then compute weighted average $f(x) = \sum \alpha_k \; f(x, w_k)$

Y. Freund and R.E. Schapire (1997)

Illustrative Example

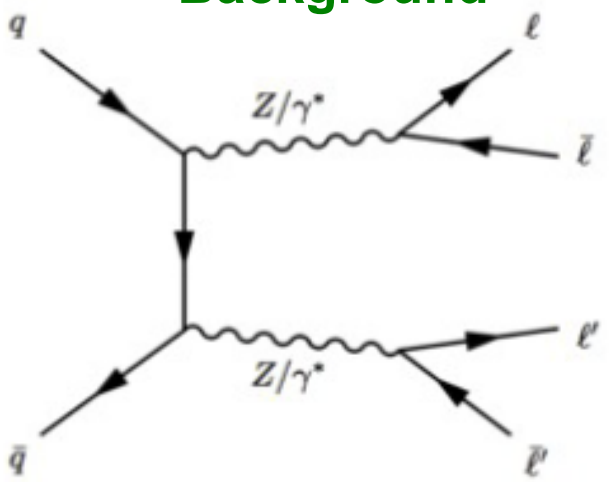
H → ZZ* → 4 leptons

Signal



$$pp \to H \to ZZ \to \ell^+\ell^-\ell'^+\ell'^-$$

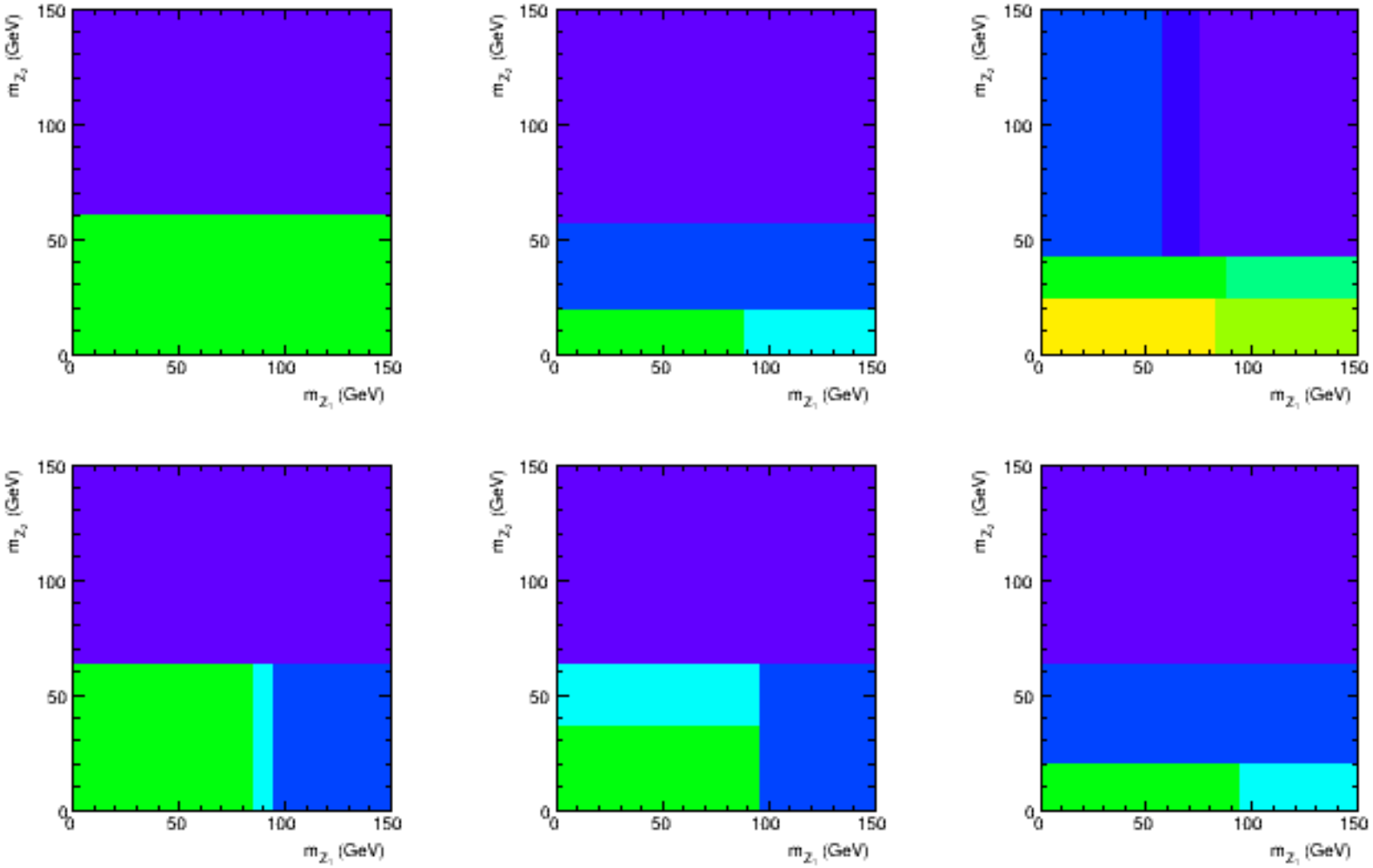
Background



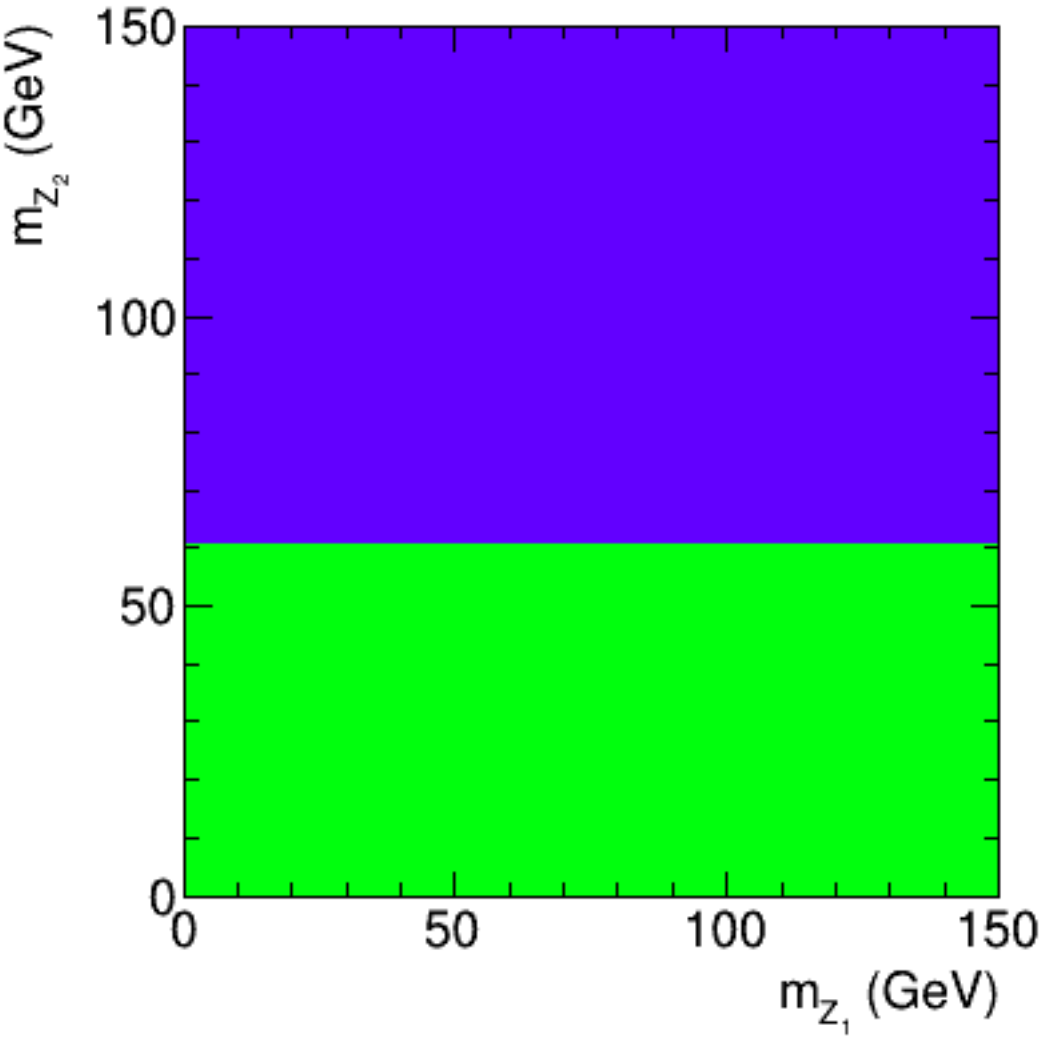
$$pp \to ZZ \to \ell^+\ell^-\ell'^+\ell'^-$$

x = (m_{Z1}, m_{Z2})

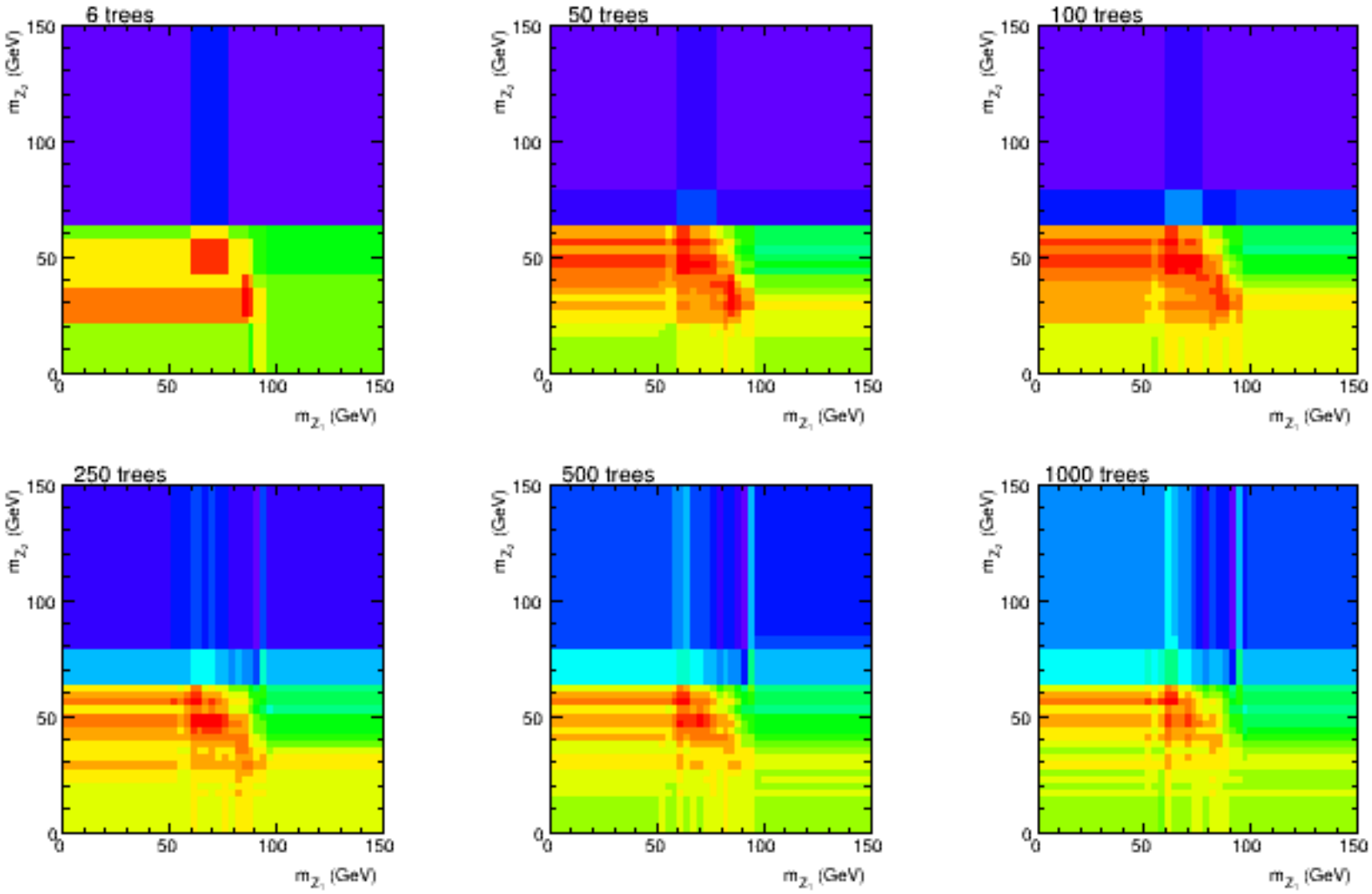
First 6 Decision Trees



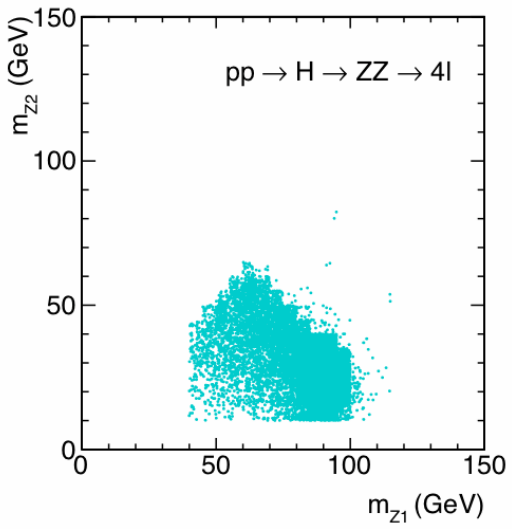
First 100 Decision Trees

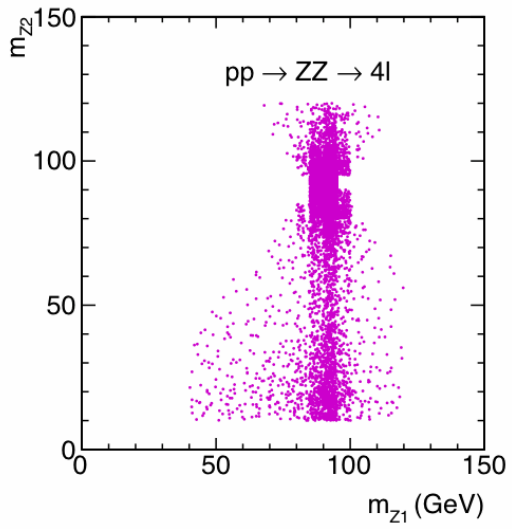


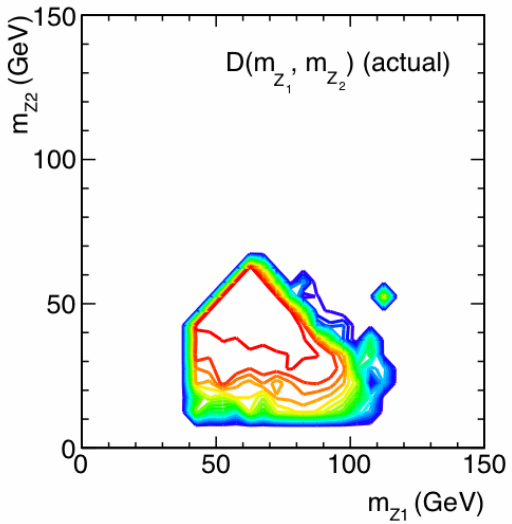
Averaging over a Forest

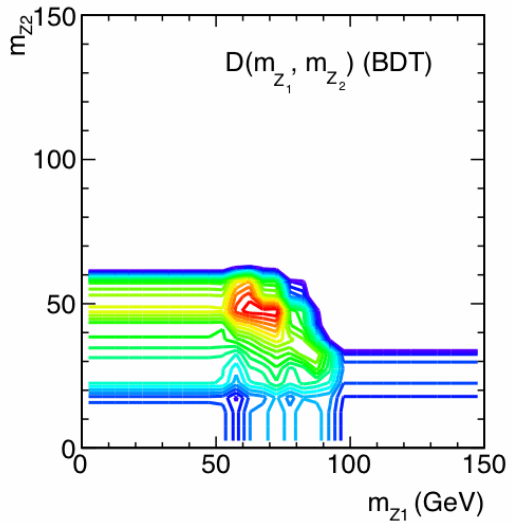


H to ZZ to 4Leptons







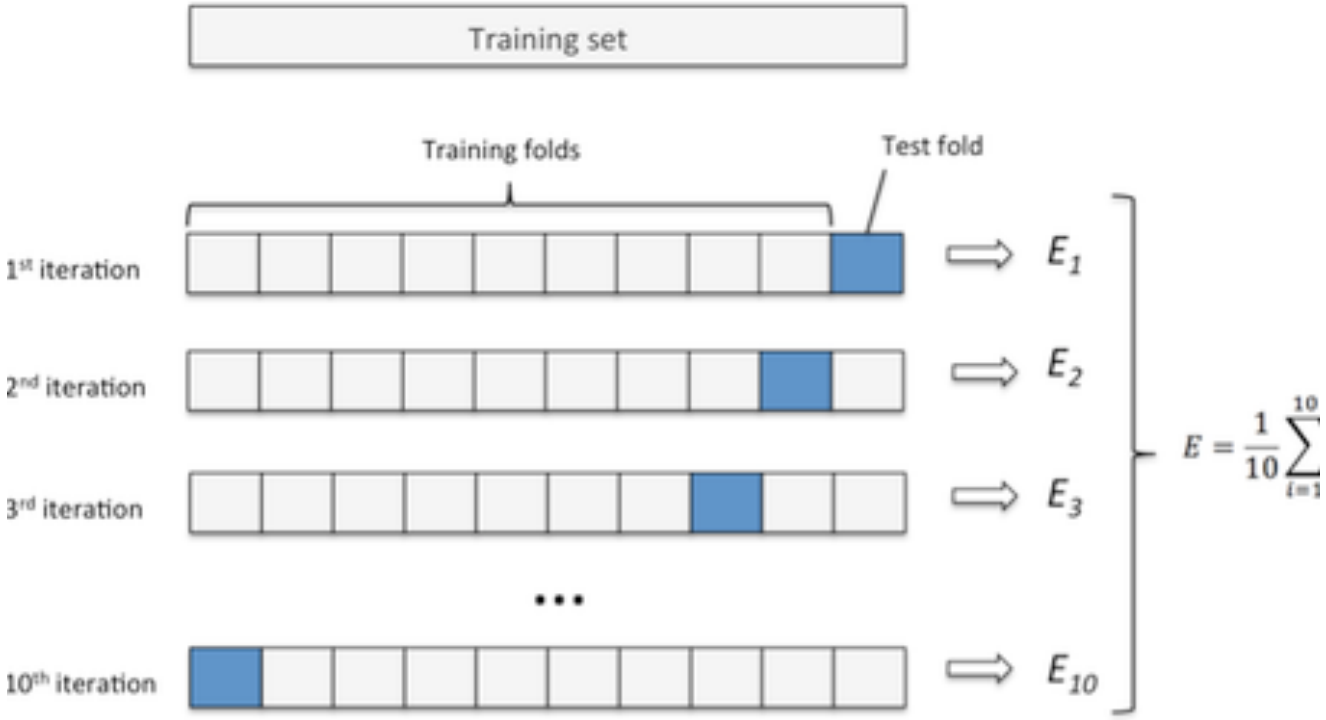


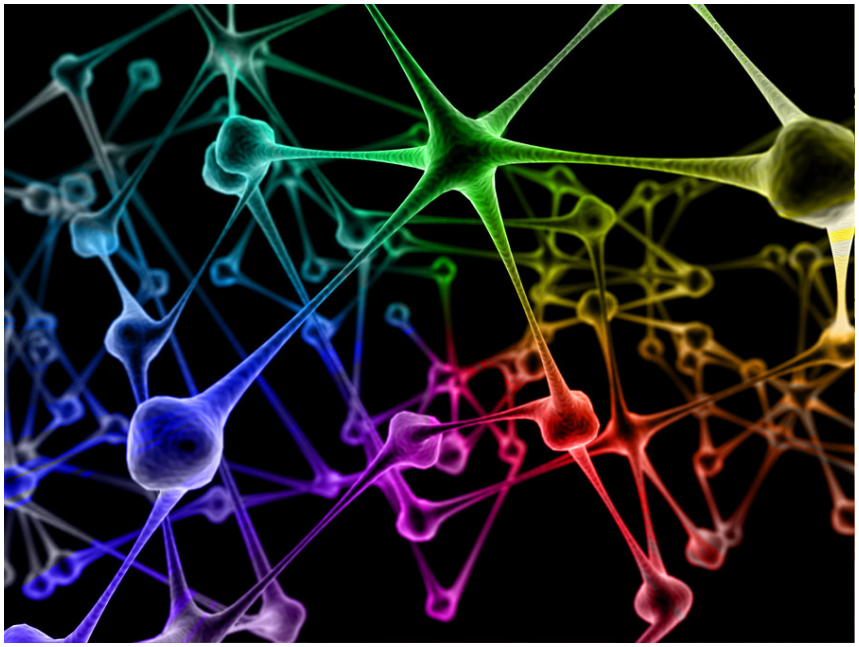
Cross Validation

Generalization of train-test split for more accurate evaluation of classifier performance

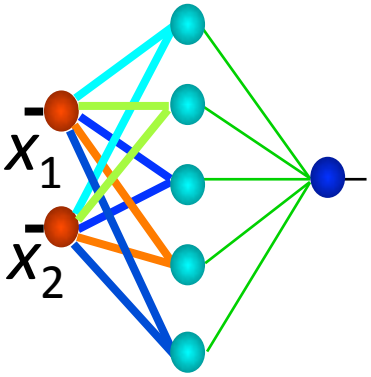
- Randomly split dataset into K equal partitions
- In each fold use K-1 samples to train, leftover to test

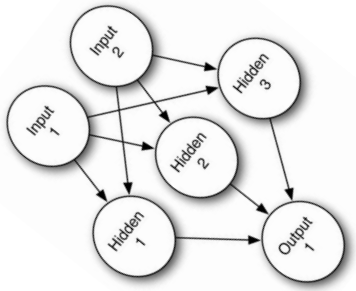
Cross Validation



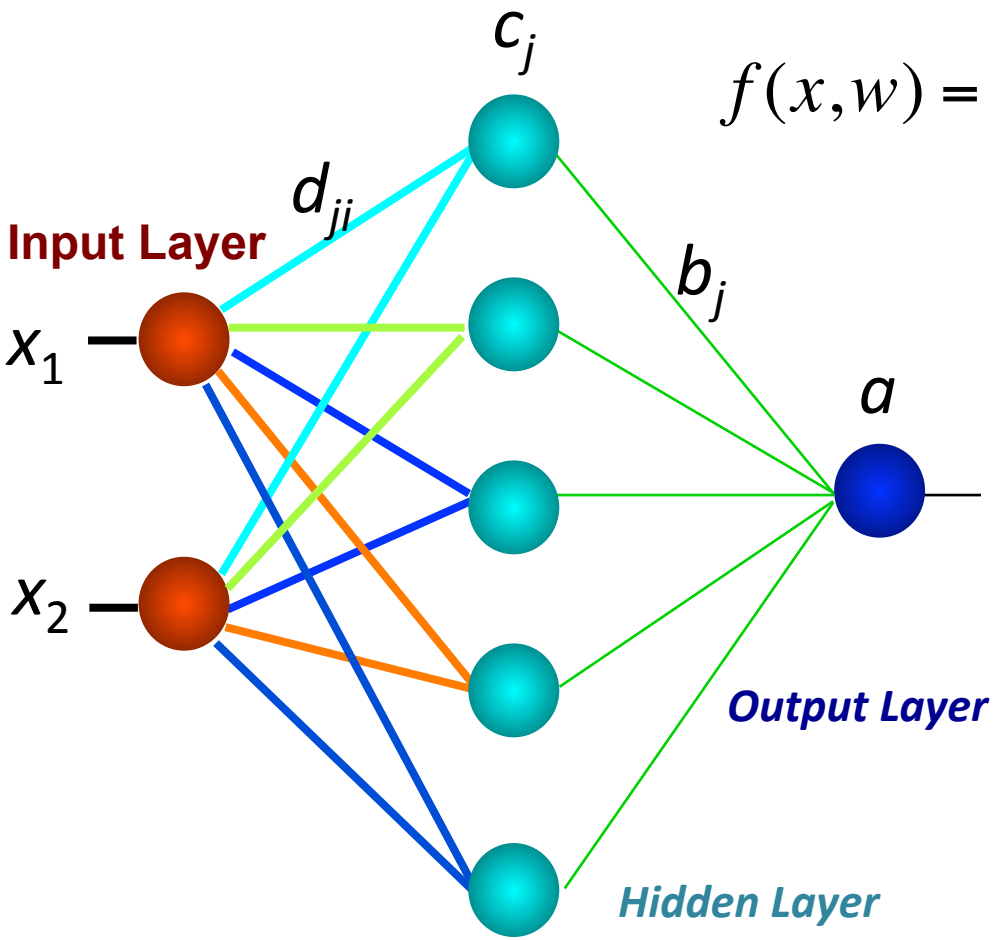


Artificial Neural Networks





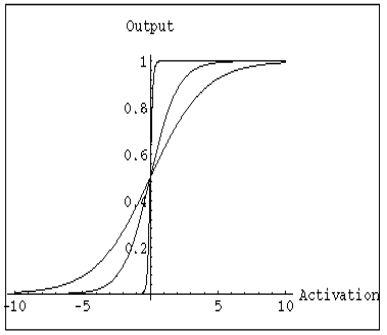
Graphical Representation



$$f(x,w) = a + \sum_{j=1}^{H} b_j \tanh\left[c_j + \sum_{i=1}^{I} d_{ji} x_i\right]$$

$$n(x,w) = \frac{1}{1+\exp[-f(x,w)]}$$

sigmoid

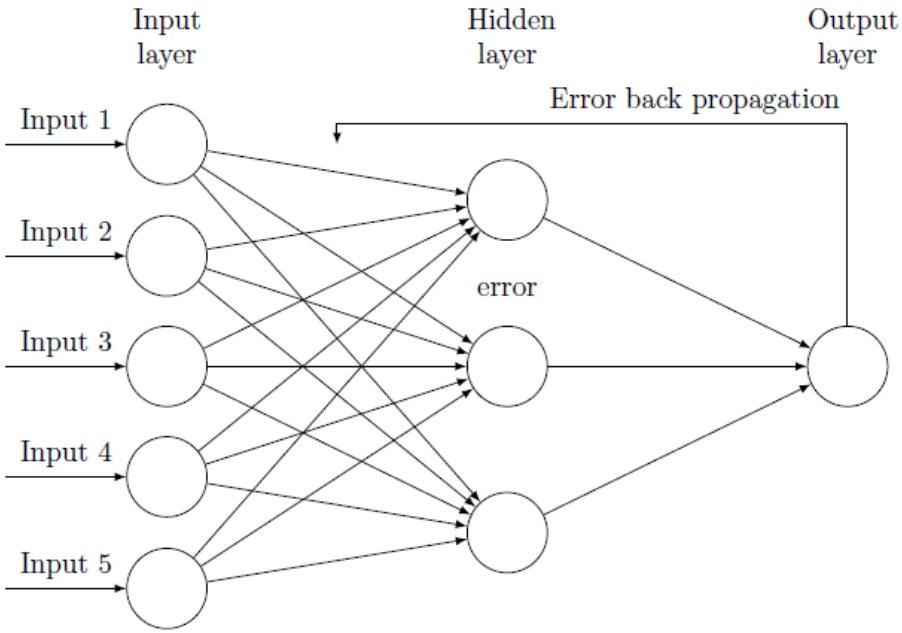


f **is used for regression**
n **is used for classification**
w* = *a*, *b*, *c*, *d

Adjustable Weights

Compute network weights with

- Error gradients

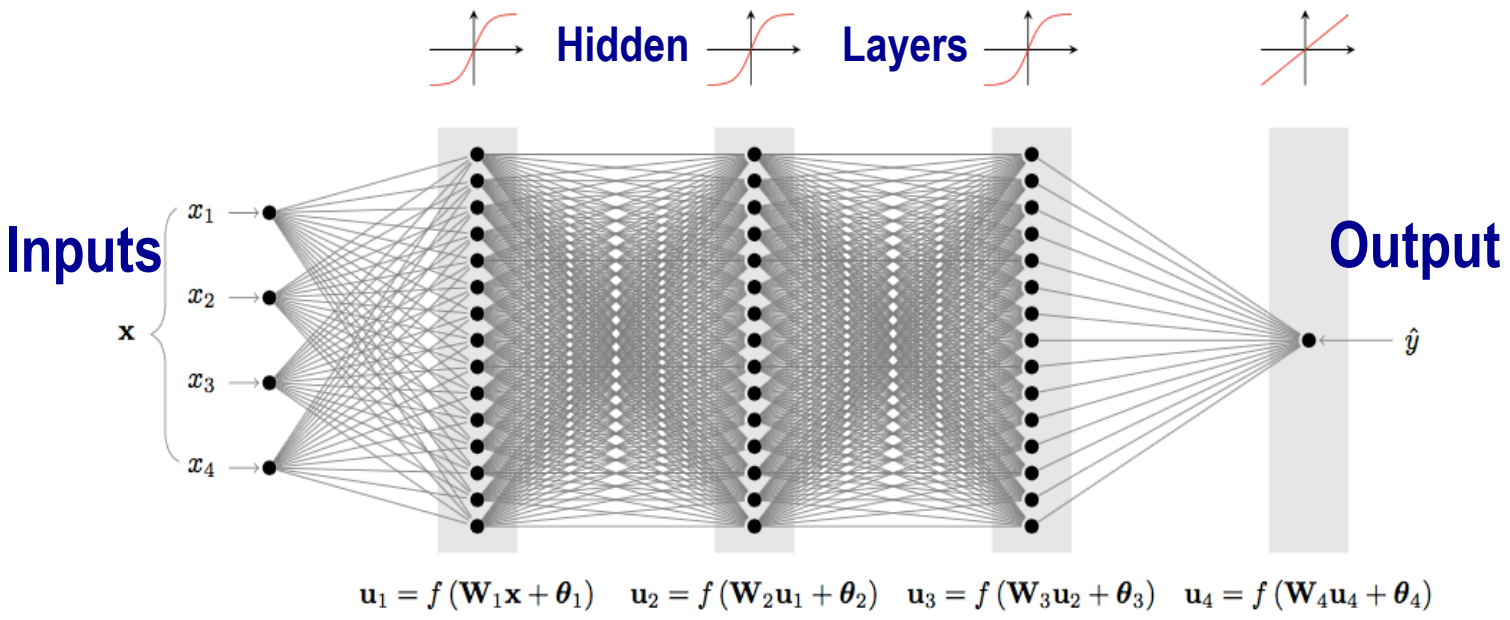


Inputs forward

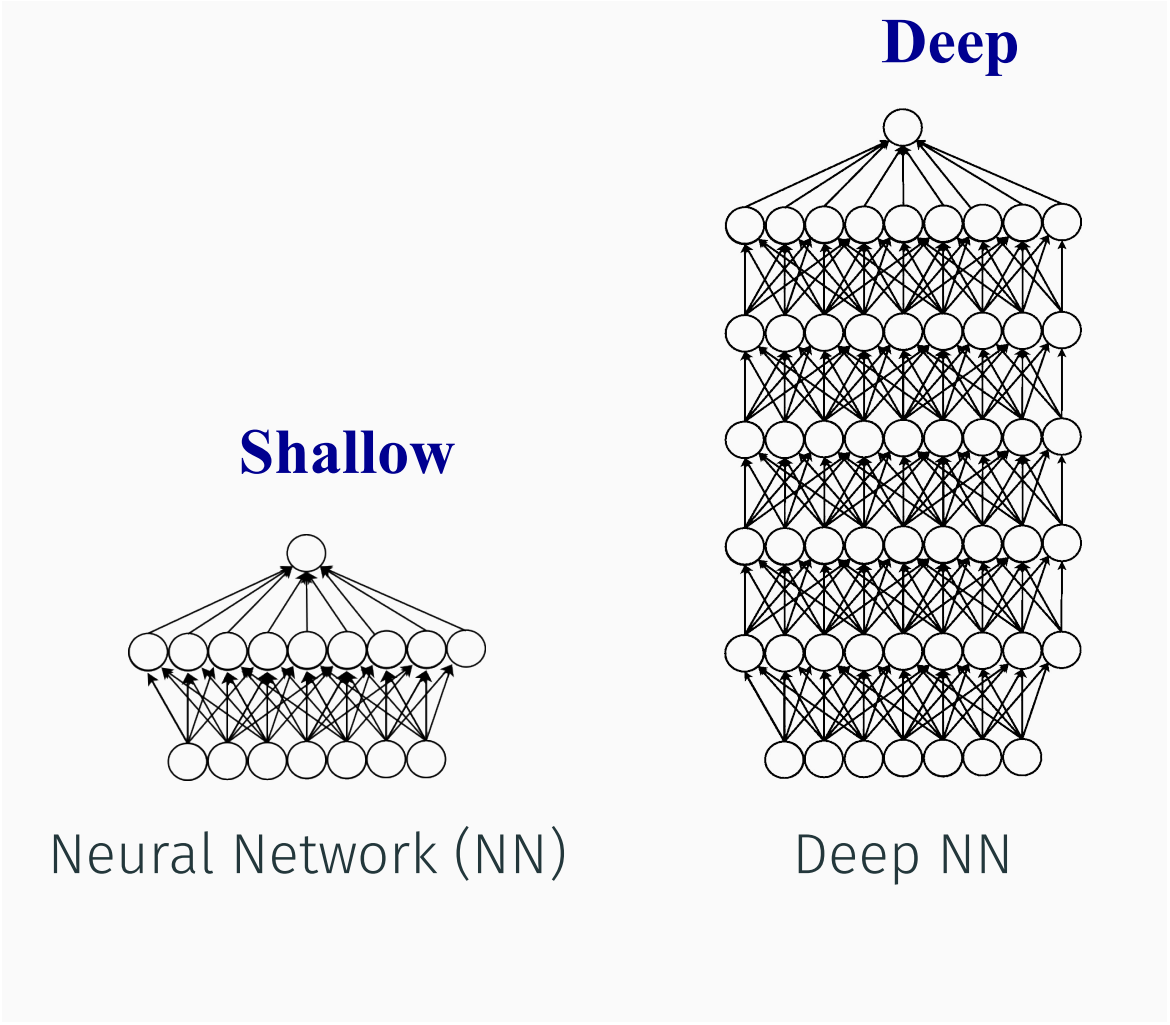
Errors go backward!

Deep Learning

Deep Neural Networks (DNN) achieve significant performance improvements



Deep Learning

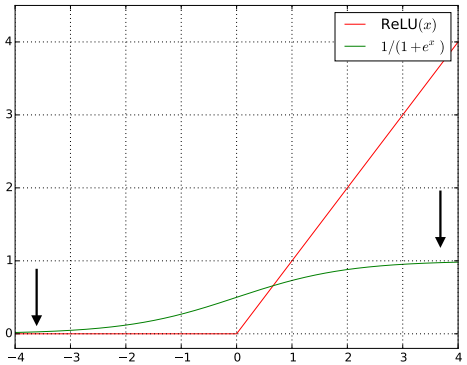


Deep Learning

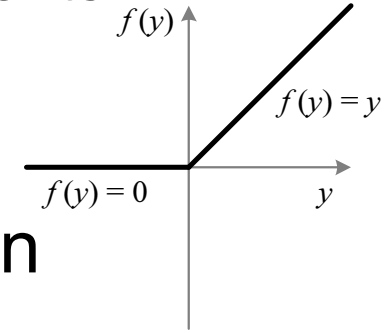
- **Training more complex models**
  - Increased Depth/Width
  - Specialized Architectures: Convolutional, Recurrent, Graphs
  - Novel activation functions: ReLU
- **Effective strategies avoiding over-fitting**
  - Regularization: L1/2, Data Augmentation, DropOut

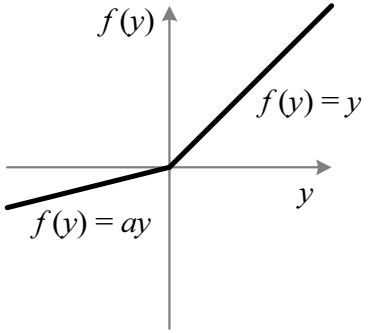
ReLU

Rectified Linear Unit (ReLU)



- Rectified neuron
- Faster training convergence
  - Better solutions than sigmoids
    - Vanishing gradients
  - Trained by back-propagation

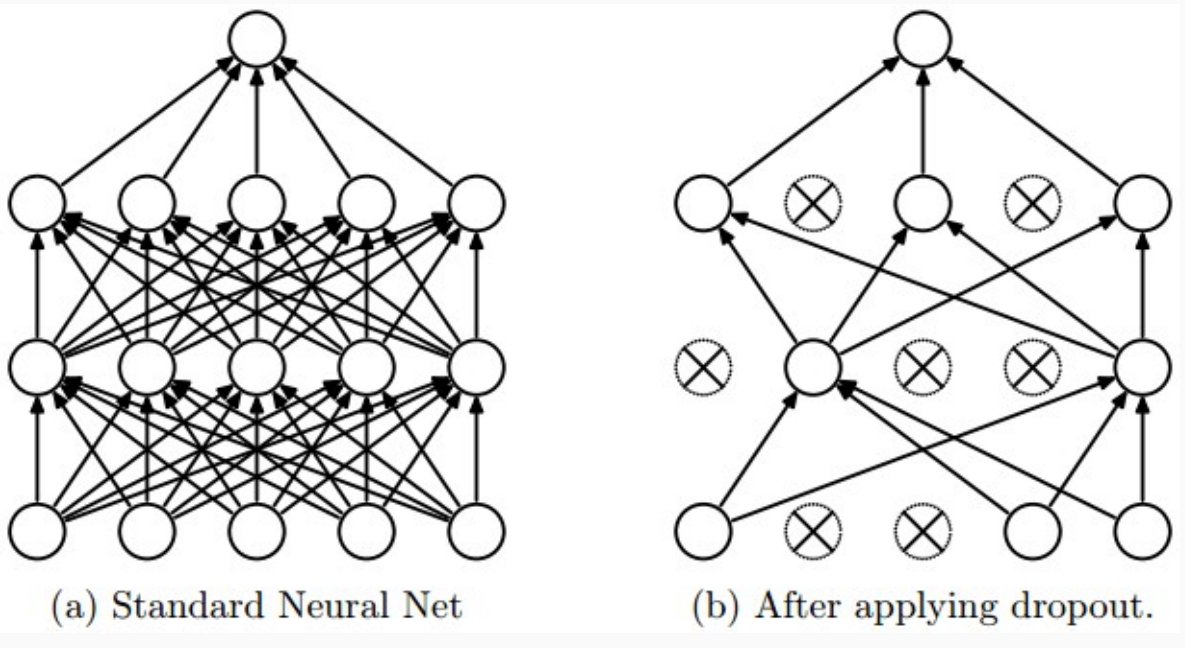




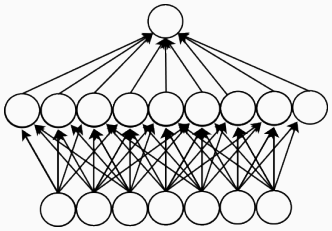
ReLU and Parametric PReLU

Regularization

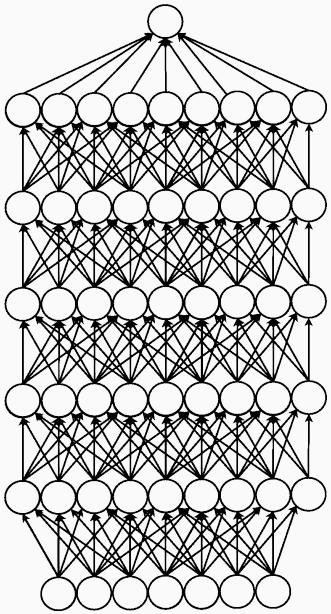
- i.e. Drop-Out



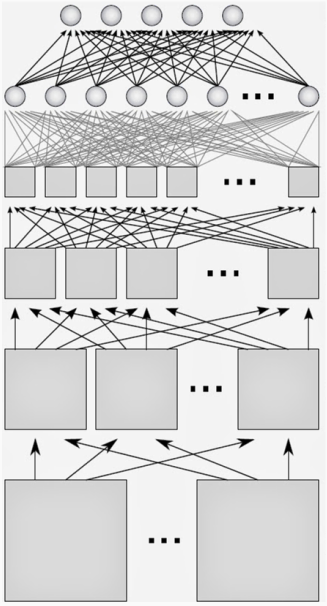
Convolution



Neural Network (NN)

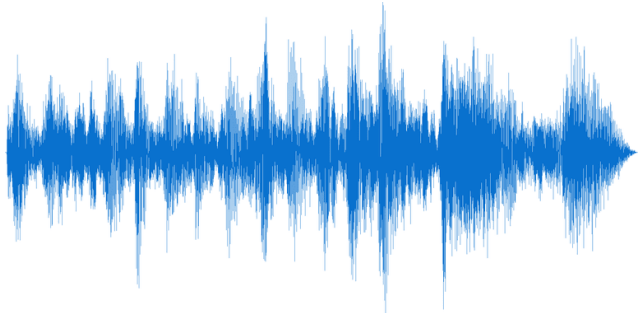


Deep NN



Convolutional NN

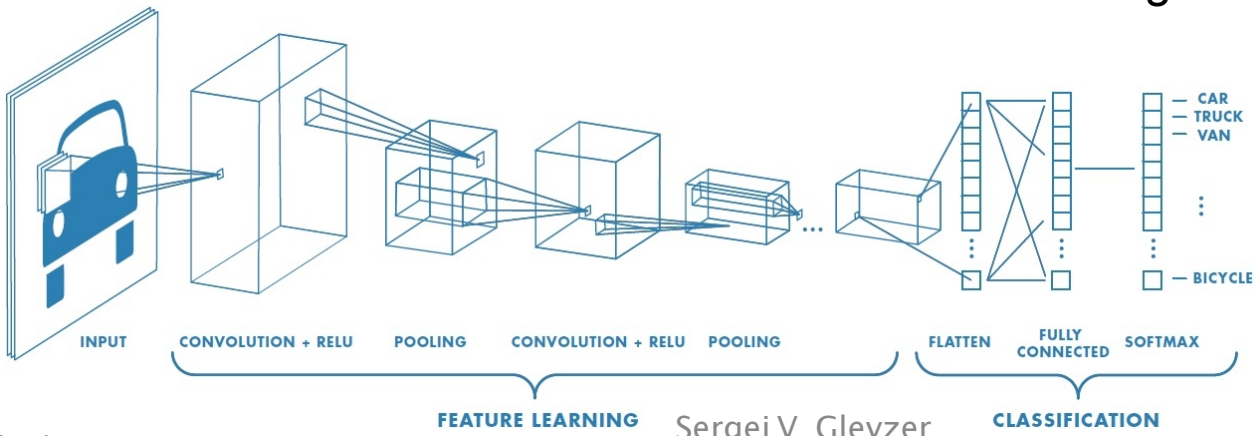
Convolutional NN



$[x^1\ x^2\ x^3 \ldots x^i]^T$
waveform heights

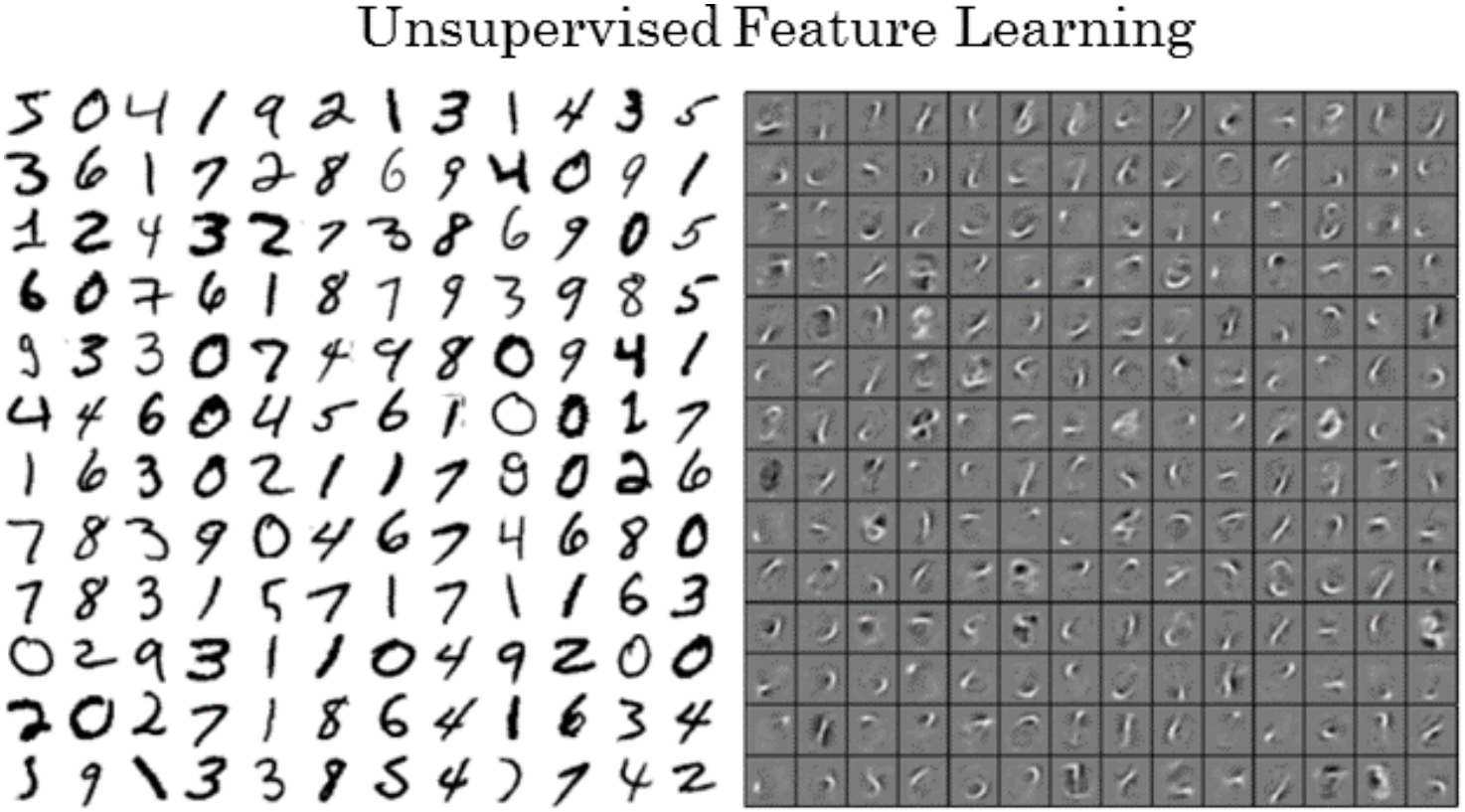
$[x^{11}\ x^{12} \ldots x^{1n}\ x^{21}\ x^{22} \ldots]^T$
pixel intensities

Feature learning

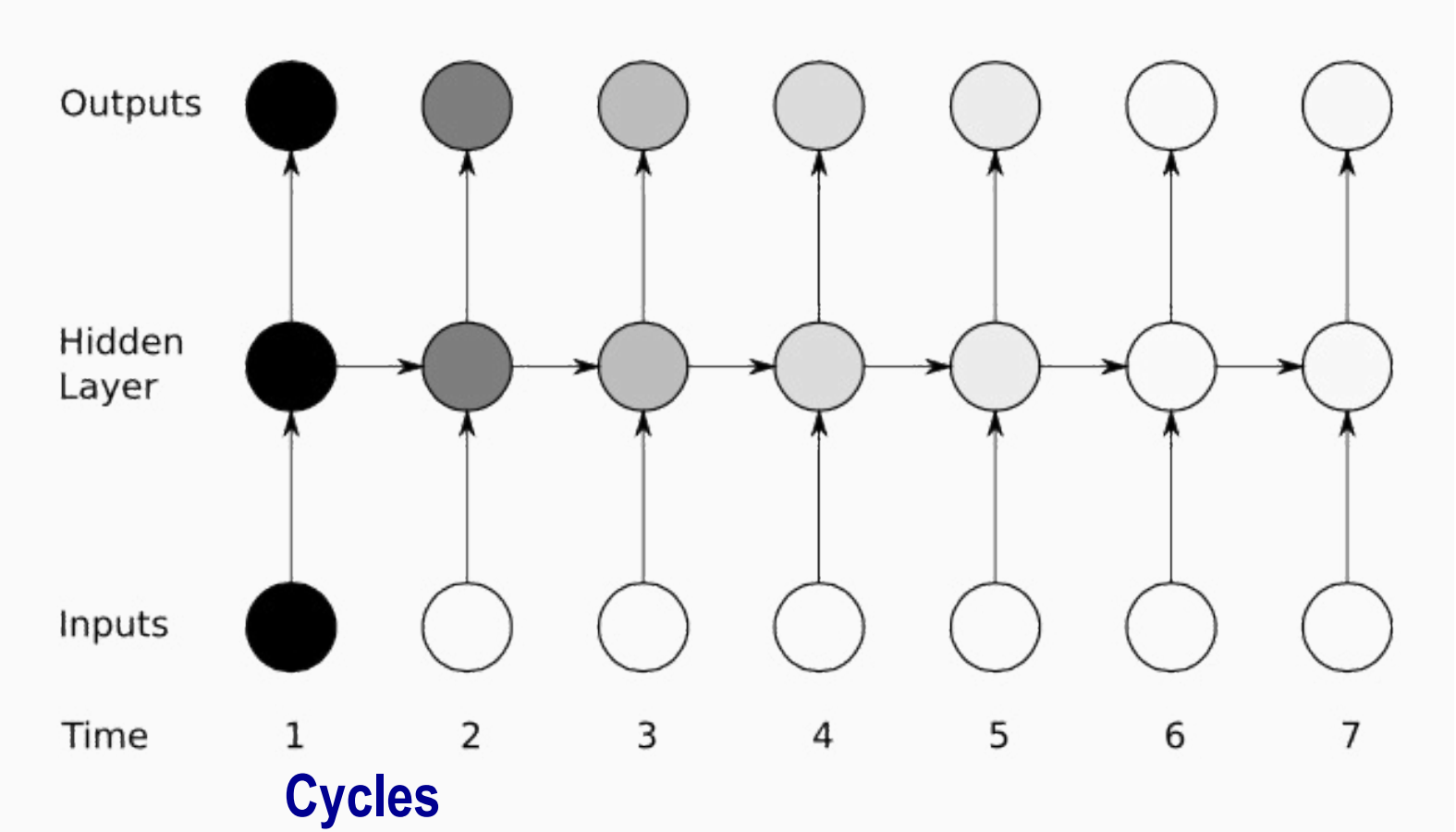


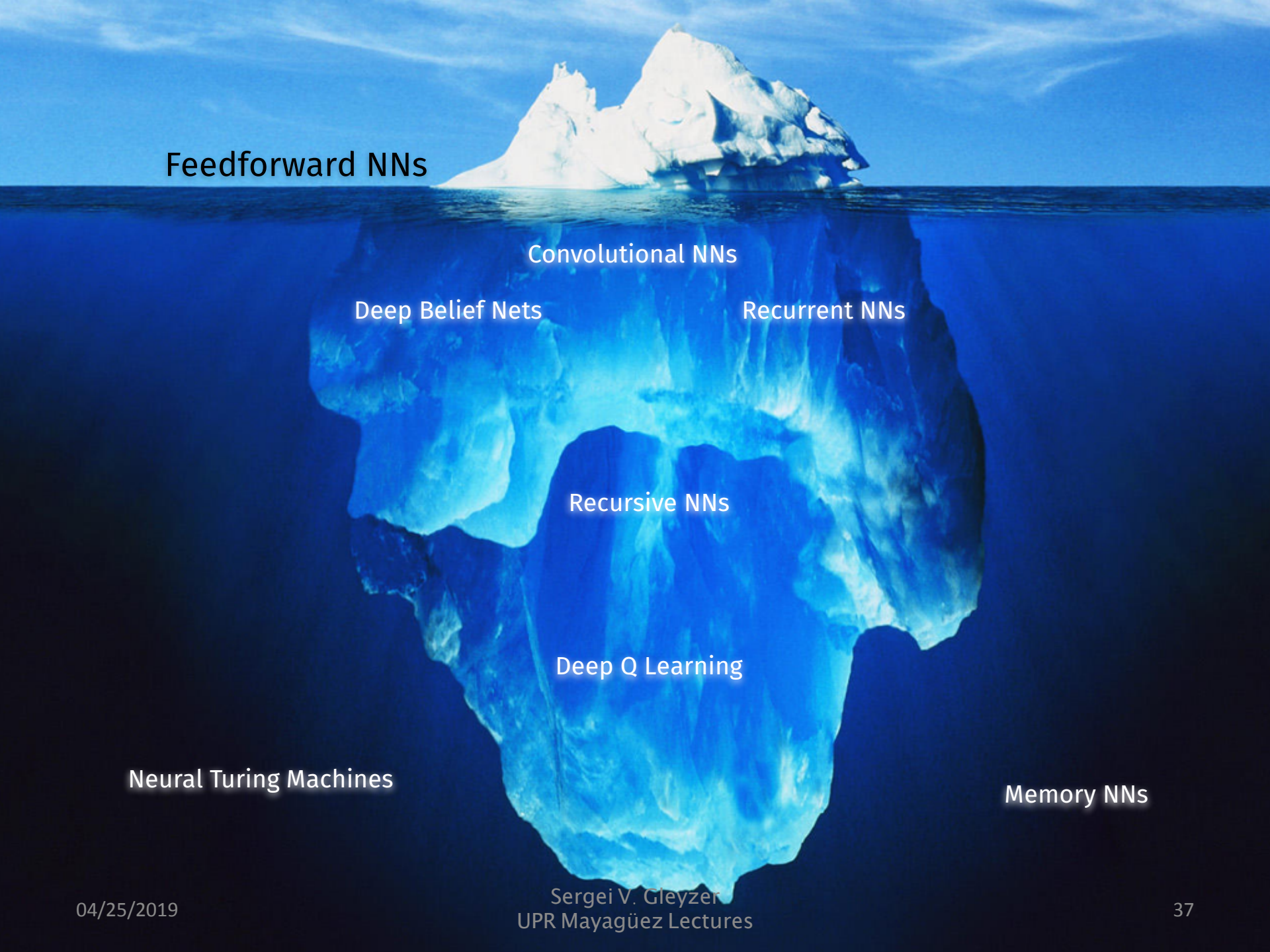
Filters

Convolutional Neural Networks:

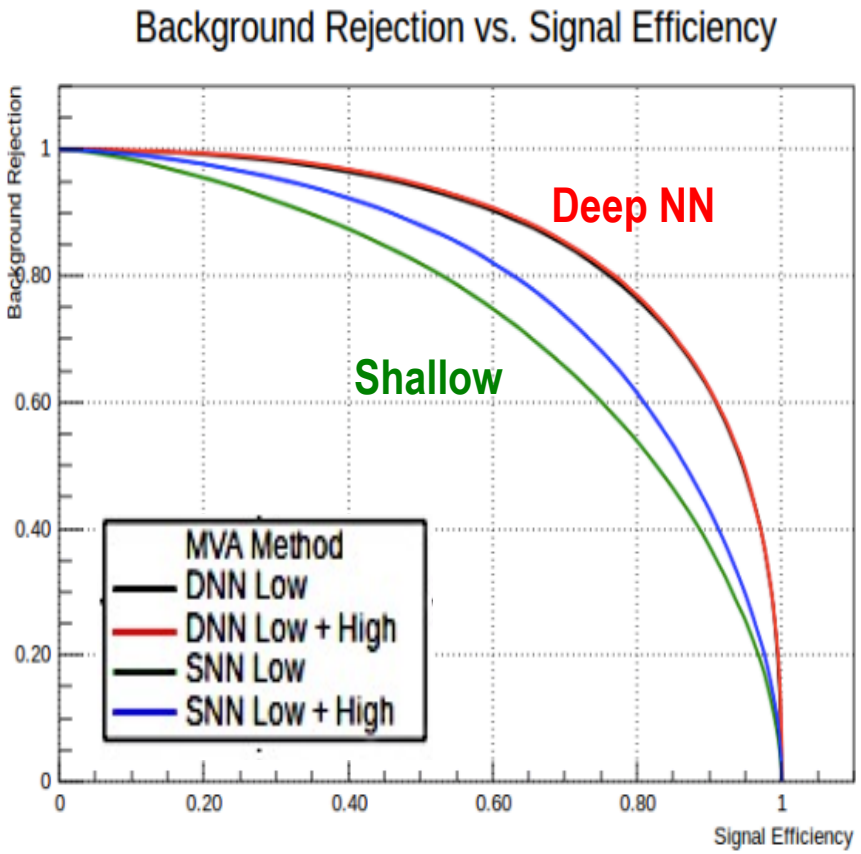


Recurrent NN




Feedforward NNs
Convolutional NNs
Deep Belief Nets
Recurrent NNs
Recursive NNs
Deep Q Learning
Neural Turing Machines
Memory NNs

Feature Extraction



Inherent Feature Extraction

Baldi et al., 2014

Feature Selection

Feature Selection

- **In data analysis one of the most crucial decisions is which features to use**
  - Garbage In = Garbage Out
- **Main Ingredients:**
  - Relevance to the problem
  - How well feature is understood
  - Its power and relationship with others

Typical Initial Set

Basic measurements covering phase space of problem:

- Functions made from them

More complex features using domain knowledge to help discriminate among classes

- 1-D discriminants

Feature Engineering

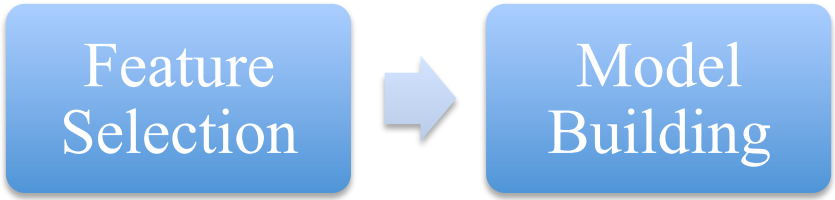
By combining features with each other this set can grow quickly

- Still small compared to 100k features of cancer or image recognition datasets

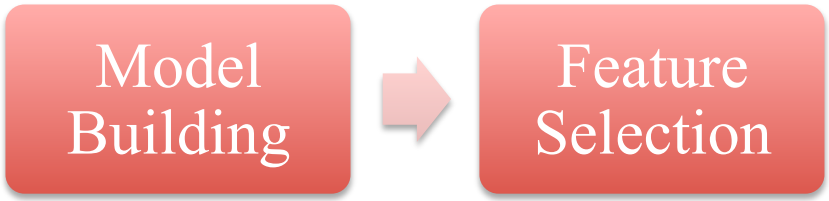
- Balance between Occam's razor and need for additional performance

Selection Methods

Filters



Wrappers



Embedded-Hybrid

Wrappers

Selection tied to a model:

- More accurate
- Assess feature interactions
- Search for optimal subset of features

Types:

- Methodical
- Probabilistic (random hill-climbing)
- Heuristic (forward backward elimination)

Example Wrapper

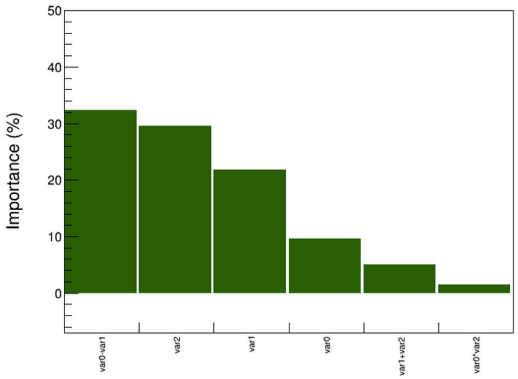
Feature Importance → proportional to **classifier performance** in which feature participates

$$FI(X_i) = \sum_{S \subseteq V : X_i \in S} F(S) \times W_{X_i}(S)$$

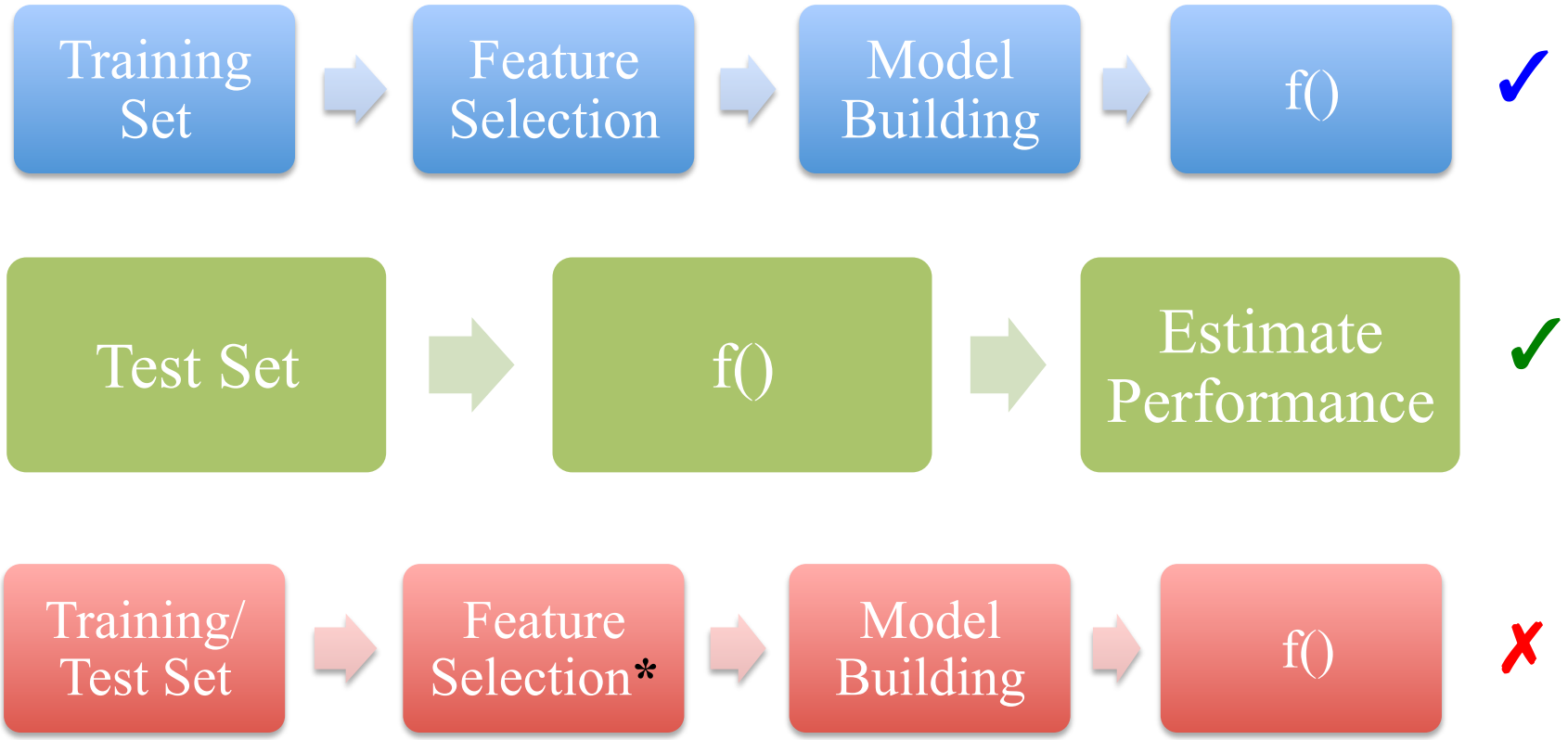
- **Full feature set {V}**
- **Feature subsets {S}**
- **Classifier performance F(S)**

$$W_{X_i}(S) \equiv 1 - \frac{F(S - \{X_i\})}{F(S)}$$

- Fast stochastic version uses random subset seeds



Practicum



*Feature Selection Bias

Embedded Methods

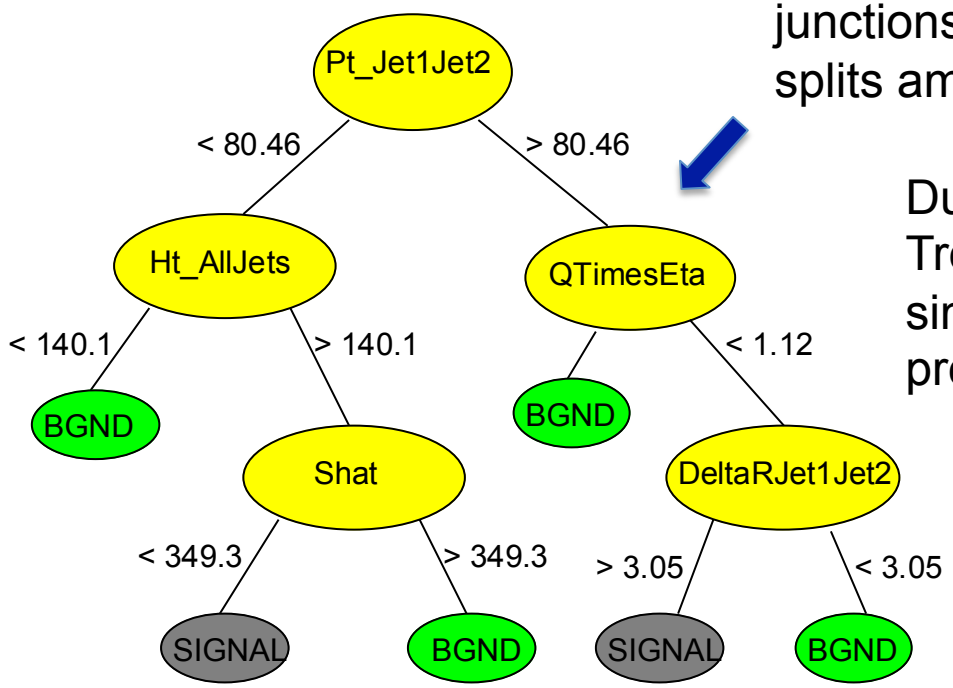
Incorporate feature importance in the model-building process

- Penalize features in the classification or regression process
  - Regularization
    - LASSO, Tibshirani, 1996
    - Regularized Trees

Regularized Trees

Inspired by J. Friedman and Popescu, 2008 work on rules regularization

Decision Tree:



Votes taken at decision junctions on possible splits among the features

During voting Regularized Trees **penalize** features similar to those used in previous decisions

End up with a **high quality** feature set

Summary

- **Machine Learning methods: neural networks and boosted decision trees**
  - Apply variants depending on the applications
  - More applications next time