CS 1675 Introduction to Machine Learning
Lecture 23

Learning with multiple models
Mixture of experts, Bagging

Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Learning with multiple models

We know how to build different classification or regression models from data

- **Question:**
  - Is it possible to learn and combine multiple (classification/regression) models and improve their predictive performance ?
- **Answer: yes**
- There are different ways of how to do it…

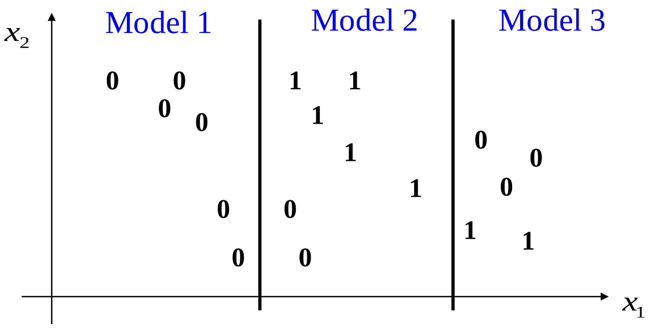
Learning with multiple models

- **Question:**
  - Is it possible to learn and combine multiple (classification/regression) models and improve their predictive performance ?
- There are different ways of how to do it…

- Assume you have models M1, M2, … Mk
- **Approach 1:** use different models (classifiers, regressors) to cover **the different parts of the input (x) space**
- **Approach 2:** use different models (classifiers, regressors) that cover **the complete input (x) space**, and combine their predictions

Approach 1

- Recall the decision tree:
  - **It partitions the input space to regions**
  - **picks the class independently**
- **What if we define a more general partitions of the input space and learn a model specific to these partitions**

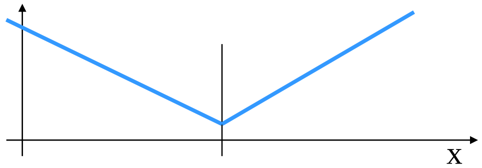


Learning with multiple models: Approach 1

Define a more general partitions of the input space and learn a model specific to these partitions

Example:

- 2 linear functions covering two regions of the input space



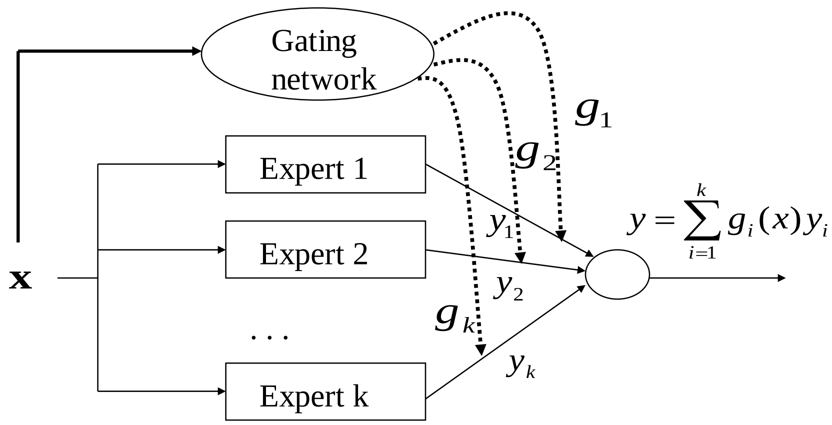
Mixture of expert model:

- Expert = learner (model)
- Different input regions covered with a different learner/model
- A "soft" switching between learners

Mixture of experts model

- **Gating network** : decides what expert to use

  $g_1, g_2, \ldots g_k$ - gating functions



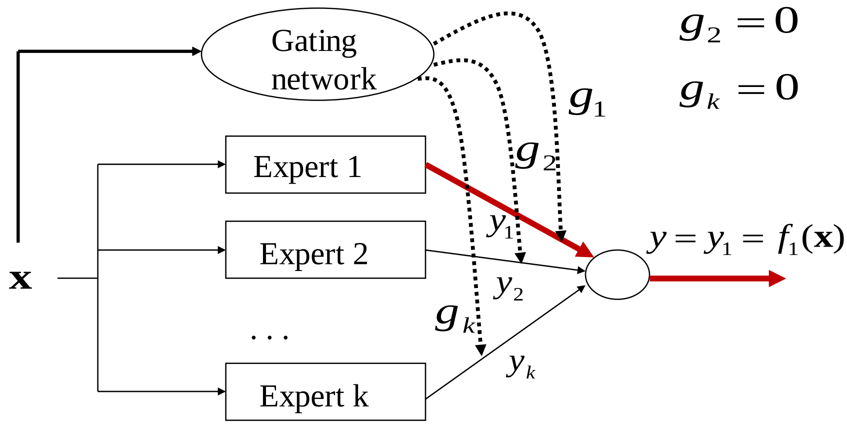
Mixture of experts model

- **Gating network** : decides what expert to use

  $g_1, g_2, \ldots g_k$ - gating functions

Assume

$$g_1 = 1$$
$$g_2 = 0$$
$$g_k = 0$$



Learning mixture of experts

- **Learning consists of two tasks:**
  - Learn the parameters of individual expert networks
  - Learn the parameters of the gating (switching) network
    - Decides where to make a split
- **Assume:** gating functions give probabilities

$$0 \le g_1(\mathbf{x}), g_2(\mathbf{x}), \ldots g_k(\mathbf{x}) \le 1 \qquad \sum_{u=1}^{k} g_u(\mathbf{x}) = 1$$

$$y = \sum_{u=1}^{k} g_u(\mathbf{x}) f_u(\mathbf{x})$$

- Based on the probability we partition the space
  - partitions belongs to different experts
- How to model the gating network?
  - **A multi-way classifier model:**
    - **softmax model**

Learning with multiple models: Approach 2

- **Approach 2:** use multiple models (classifiers, regressors) that cover the complete input (x) space and combine their outputs

- **Committee machines:**
  - Combine predictions of all models to produce the output
    - **Regression:** averaging
    - **Classification:** a majority vote
  - **Goal:** Improve the accuracy of the 'base' model

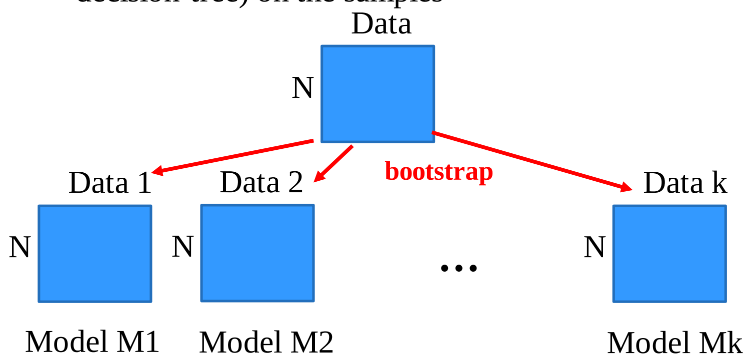
- **Methods:**
  - **Bagging (the same base models)**
  - **Boosting (the same base models)**
  - Stacking (different base model) not covered

Bagging (Bootstrap Aggregating)

- **Given:**
  - Training set of *N* examples
  - A base learning model (e.g. decision tree, neural network, …)
- **Method:**
  - Train multiple (k) base models on slightly different datasets
  - Predict (test) by averaging the results of k models
- **Goal:**
  - Improve the accuracy of one model by using its multiple copies
  - Average of misclassification errors on different data splits gives a better estimate of the predictive ability of a learning method

Bagging algorithm

- **Training**
- For each model M1, M2, … Mk
  - Randomly sample with replacement *N* samples from the training set (bootstrap)
  - Train a chosen "base model" (e.g. neural network, decision tree) on the samples

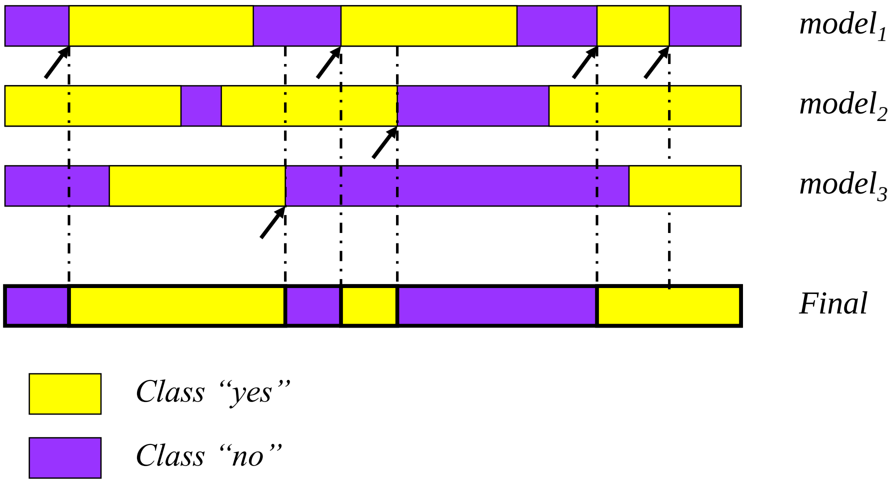


Bagging algorithm

- **Training**
- For each model M1, M2, … Mk
  - Randomly sample with replacement *N* samples from the training set
  - Train a chosen "base model" (e.g. a neural network, or a decision tree) on the samples
- **Test**
  - For each test example
    - Run all base models M1, M2, … Mk
    - Predict by combining results of all T trained models:
      - **Regression:** averaging
      - **Classification:** a majority vote

Class decision via majority voting

Test examples



Analysis of Bagging

- **Expected error= Bias+Variance**
  - *Expected error* is the expected discrepancy between the estimated and true function

$$E\left[\left(\hat{f}(X) - E[f(X)]\right)^2\right]$$

It decomposes to two terms *Bias* + *Varian*ce

  - *Bias* is a squared discrepancy between *averaged* estimated and true function

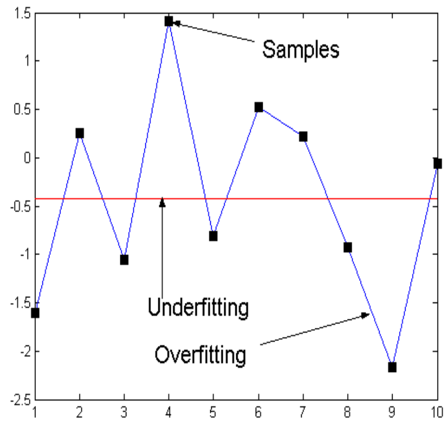
$$\left(E\left[\hat{f}(X)\right] - E[f(X)]\right)^2$$

  - *Variance* is an expected divergence of the estimated function vs. its average value

$$E\left[\left(\hat{f}(X) - E\left[\hat{f}(X)\right]\right)^2\right]$$

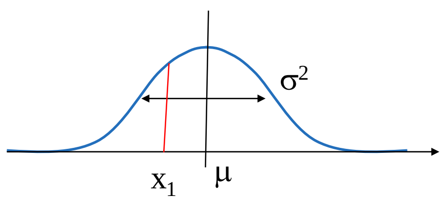
When Bagging works?
Under-fitting and over-fitting

- **Under-fitting:**
  - **High bias** (models are not accurate)
  - **Small variance** (smaller influence of examples in the training set)
- **Over-fitting:**
  - **Small bias** (models flexible enough to fit well to training data)
  - **Large variance** (models depend very much on the training set)



Averaging decreases variance

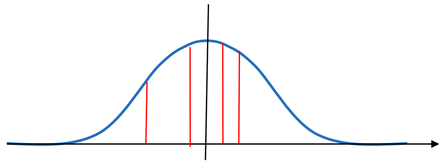
- **Example**
  - Assume a random variable x with a $N(\mu,\sigma^2)$ distribution



  - **Case 1:** we draw one example/measurement x_1 and use it to estimate the mean $\mu' = x_1$
    - The expected mean of the estimate $E[\mu'] = E[x_1] = \mu$
    - The variance of the mean estimate $Var(\mu') = Var(x_1) = \sigma^2$

Averaging decreases variance

- **Example** Assume a random variable x with a $N(\mu,\sigma^2)$ distribution



- **Case 2:** a variable x is measured independently K times $(x_1,x_2,\ldots x_k)$ and the mean is estimated as:

  $$\mu' = (x_1+x_2+\ldots+x_k)/K,$$

  - The expected mean of the estimate $E[\mu'] = \mu$
  - But, the variance of the mean estimate $Var(\mu')$is smaller:

    $$Var(\mu') = [Var(x_1)+\ldots Var(x_k)]/K^2 = K\sigma^2 / K^2 = \sigma^2/K$$

When Bagging works

Relation of the previous example to bagging:

- **Bagging is a kind of averaging!**

Main property of Bagging (proof omitted)

- Bagging **decreases variance** of the base model without changing the bias!!!
- Why? averaging!

Bagging typically helps

- When applied with an **over-fitted base model**
  - High dependency on actual training data
  - Example: fully grown decision trees

Bagging does not help much when

- Applied to models with a high bias. When the base model is robust to the changes in the training data (due to sampling)