

CSCE 970 Lecture 3: Regularization

Stephen Scott and Vinod Variyam

sscott@cse.unl.edu

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

Other
Performance
Measures

- Machine learning can generally be distilled to an optimization problem
- Choose a classifier (function, hypothesis) from a set of functions that minimizes an objective function
- Clearly we want part of this function to measure performance on the training set, but this is insufficient

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

Other
Performance
Measures

- Types of machine learning problems
- Loss functions
- Generalization performance vs training set performance
- Overfitting
- Regularization
- Estimating generalization performance

Machine Learning Problems

CSCE 970

Lecture 3:
RegularizationStephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
ProblemsMeasuring
Performance

Regularization

Estimating
Generalization
PerformanceComparing
Learning
AlgorithmsOther
Performance
Measures

- **Supervised Learning:** Algorithm is given labeled *training data* and is asked to infer a function (*hypothesis*) from a family of functions (e.g., set of all ANNs) that is able to predict well on new, unseen examples
 - **Classification:** Labels come from a finite, discrete set
 - **Regression:** Labels are real-valued
- **Unsupervised Learning:** Algorithm is given data without labels and is asked to model its structure
 - Clustering, density estimation
- **Reinforcement Learning:** Algorithm controls an agent that interacts with its environment and learns good actions in various situations

Measuring Performance

Loss

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Loss

Overfitting

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

Other

- In any learning problem, need to be able to quantify performance of an algorithm
- In supervised learning, we often use a **loss function** (or error function) \mathcal{J} for this task
- Given instance x with true label y , if the learner's prediction on x is \hat{y} , then

$$\mathcal{J}(y, \hat{y})$$

is the loss on that instance

Measuring Performance

Examples of Loss Functions

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Loss

Overfitting

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

Other

- **0-1 Loss:** $\mathcal{J}(y, \hat{y}) = 1$ if $y \neq \hat{y}$, 0 otherwise
- **Square Loss:** $\mathcal{J}(y, \hat{y}) = (y - \hat{y})^2$
- **Cross-Entropy:** $\mathcal{J}(y, \hat{y}) = -y \ln \hat{y} - (1 - y) \ln (1 - \hat{y})$
(y and \hat{y} are considered probabilities of a '1' label;
generalizes to multi-class.)
- **Hinge Loss:** $\mathcal{J}(y, \hat{y}) = \max(0, 1 - y\hat{y})$
(used sometimes for large margin classifiers like SVMs)

All non-negative

Measuring Performance

Training Loss

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Loss

Overfitting

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

Other

- Given a loss function \mathcal{J} and a training set \mathcal{X} , the total loss of the classifier h on \mathcal{X} is

$$\text{error}_{\mathcal{X}}(h) = \sum_{\mathbf{x} \in \mathcal{X}} \mathcal{J}(y_{\mathbf{x}}, \hat{y}_{\mathbf{x}}) \ ,$$

where $y_{\mathbf{x}}$ is \mathbf{x} 's label and $\hat{y}_{\mathbf{x}}$ is h 's prediction

Measuring Performance

Expected Loss

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Varyiam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Loss

Overfitting

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

Other

- More importantly, the learner needs to **generalize** well:
Given a new example drawn iid according to unknown probability distribution \mathcal{D} , we want to minimize h 's **expected loss**:

$$error_{\mathcal{D}}(h) = \mathbb{E}_{\mathbf{x} \sim \mathcal{D}} [\mathcal{J}(y_{\mathbf{x}}, \hat{y}_{\mathbf{x}})]$$

- Is minimizing training loss the same as minimizing expected loss?

Measuring Performance

Expected vs Training Loss

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Loss
Overfitting

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

Other

- Sufficiently sophisticated learners (decision trees, multi-layer ANNs) can often achieve arbitrarily small (or zero) loss on a training set
- A hypothesis (e.g., ANN with specific parameters) h *overfits* the training data \mathcal{X} if there is an alternative hypothesis h' such that

$$\text{error}_{\mathcal{X}}(h) < \text{error}_{\mathcal{X}}(h')$$

and

$$\text{error}_{\mathcal{D}}(h) > \text{error}_{\mathcal{D}}(h')$$

Measuring Performance

Overfitting

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Loss

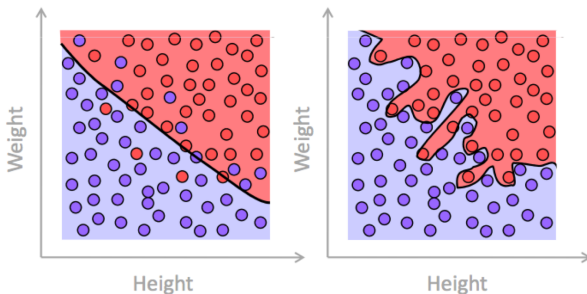
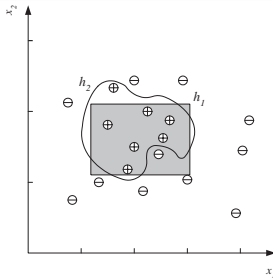
Overfitting

Regularization

Estimating
Generalization
Performance

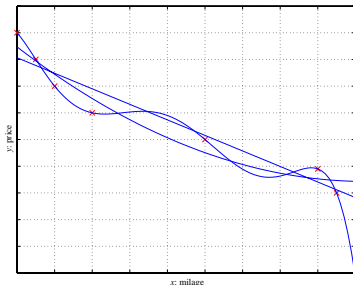
Comparing
Learning
Algorithms

Other/ 52

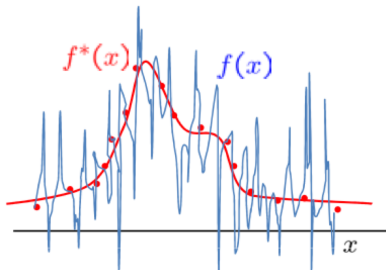


Measuring Performance

Overfitting



To generalize well, need to balance **training accuracy** with **simplicity**



CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Loss

Overfitting

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

Other/ 52

Regularization

Causes of Overfitting

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Varyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Causes of Overfitting

Early Stopping

Parameter Norm
Penalties

Data Augmentation

Multitask Learning

Dropout

Others

Estimating
Generalization
Performance

- Generally, if the set of functions \mathcal{H} the learner has to choose from is complex relative to what is required for correctly predicting the labels of \mathcal{X} , there's a larger chance of overfitting due to the large number of “wrong” choices in \mathcal{H}
 - Could be due to an overly sophisticated set of functions
 - E.g., can fit any set of n real-valued points with an $(n - 1)$ -degree polynomial, but perhaps only degree 2 is needed
 - E.g., using an ANN with 5 hidden layers to solve the logical AND problem
 - Could be due to training an ANN too long
 - Over-training an ANN often leads to weights deviating far from zero
 - Makes the function more non-linear, and more complex
- Often, a larger data set mitigates the problem

Regularization

Causes of Overfitting: Overtraining

CSCE 970

Lecture 3:

Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Causes of Overfitting

Early Stopping

Parameter Norm
Penalties

Data Augmentation

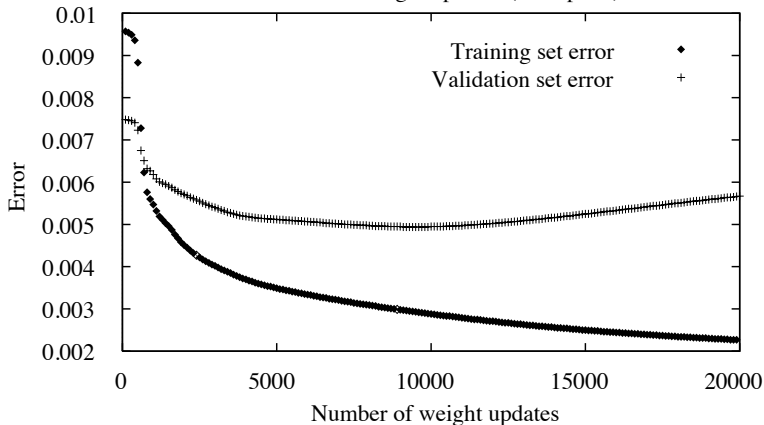
Multitask Learning

Dropout

Others

Estimating
Generalization
Performance

Error versus weight updates (example 1)



Regularization

Early Stopping

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

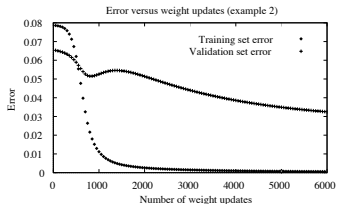
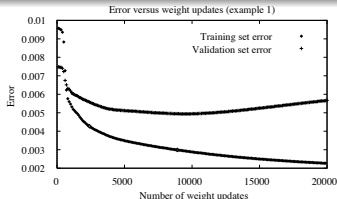
Causes of Overfitting

Early Stopping

Parameter Norm
Penalties

Data Augmentation
Multitask Learning
Dropout
Others

Estimating
Generalization
Performance



- Danger of stopping too soon
 - “Patience” parameter in Algorithm 7.1
- Can re-start with small, random weights (Algorithm 7.2)

Regularization

Parameter Norm Penalties

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Causes of Overfitting
Early Stopping

Parameter Norm
Penalties

Data Augmentation
Multitask Learning
Dropout
Others

Estimating
Generalization
Performance

- Still want to minimize training loss, but balance it against a complexity penalty on the parameters used:

$$\tilde{\mathcal{J}}(\boldsymbol{\theta}; \mathcal{X}, \mathbf{y}) = \mathcal{J}(\boldsymbol{\theta}; \mathcal{X}, \mathbf{y}) + \alpha \Omega(\boldsymbol{\theta})$$

- $\alpha \in [0, \infty)$ weights loss \mathcal{J} against penalty Ω

Regularization

Parameter Norm Penalties: L^2 Norm

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Causes of Overfitting
Early Stopping

Parameter Norm
Penalties

Data Augmentation
Multitask Learning
Dropout
Others

Estimating
Generalization
Performance

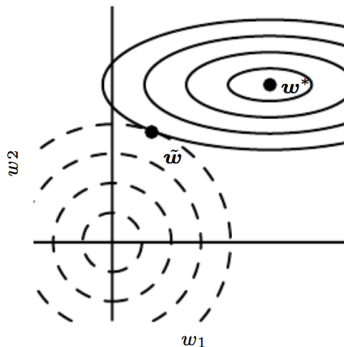
- $\Omega(\theta) = (1/2)\|\theta\|_2^2$, i.e., sum of squares of network's weights
- Since $\theta = \mathbf{w}$, this becomes

$$\tilde{\mathcal{J}}(\mathbf{w}; \mathcal{X}, \mathbf{y}) = (\alpha/2)\mathbf{w}^\top \mathbf{w} + \mathcal{J}(\mathbf{w}; \mathcal{X}, \mathbf{y})$$

- As weights deviate from zero, activation functions become more nonlinear, which is higher risk of overfitting

Regularization

Parameter Norm Penalties: L^2 Norm

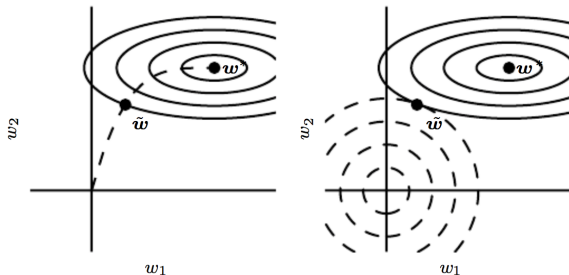


- w^* is optimal for \mathcal{J} , $\mathbf{0}$ optimal for regularizer
- \mathcal{J} less sensitive to w_1 , so \tilde{w} (optimal for $\tilde{\mathcal{J}}$) closer to 0 than w_2

Regularization

Parameter Norm Penalties: L^2 Norm

Related to early stopping: For linear model and square loss, get trajectory of weight updates that, on average, lands in a similar result



Length τ of trajectory related to weight α in L^2 regularizer

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Varyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Causes of Overfitting
Early Stopping

Parameter Norm
Penalties

Data Augmentation
Multitask Learning
Dropout
Others

Estimating
Generalization
Performance

Regularization

Parameter Norm Penalties: L^1 Norm

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Causes of Overfitting
Early Stopping

Parameter Norm
Penalties

Data Augmentation
Multitask Learning
Dropout
Others

Estimating
Generalization
Performance

- $\Omega(\theta) = \|\theta\|_1$, i.e., sum of absolute values of network's weights

$$\tilde{\mathcal{J}}(\mathbf{w}; \mathcal{X}, \mathbf{y}) = \alpha \|\mathbf{w}\|_1 + \mathcal{J}(\mathbf{w}; \mathcal{X}, \mathbf{y})$$

- As with L^2 regularization, penalizes large weights
- Unlike L^2 regularization, can drive some weights to zero
 - **Sparse** solution
 - Sometimes used in **feature selection** (e.g., LASSO algorithm)

Regularization

Data Augmentation

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Causes of Overfitting
Early Stopping
Parameter Norm
Penalties

Data Augmentation

Multitask Learning
Dropout
Others

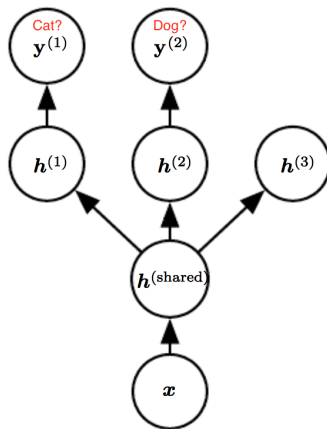
Estimating
Generalization
Performance

- If \mathcal{H} is powerful and \mathcal{X} is small, then a learner can choose some $h \in \mathcal{H}$ that fits the idiosyncrasies or noise in the data
- Deep ANNs would like to have at least thousands or tens of thousands of data points
- In classification of high-dimensional data (e.g., image classification), expect the classifier to tolerate transformations and noise
 - ⇒ Can artificially enlarge data set by duplicating existing instances and applying transformations
 - Translating, rotating, scaling
 - Be careful to to change the class, e.g., “b” vs “d” or “6” vs “9”
 - ⇒ Can also apply noise injection to input or even hidden layers

Regularization

Multitask Learning

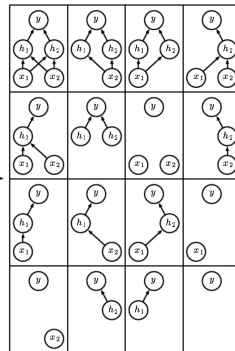
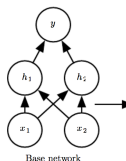
- If multiple tasks share **generic parameters**, initially process inputs via shared nodes, then do final processing via task-specific nodes
- Backpropagation works as before with multiple output nodes
- Serves as a regularizer since parameter tuning of shared nodes is based on backpropagated error from multiple tasks



Regularization

Dropout

- Imagine if, for a network, we could average over all networks with each subset of nodes deleted
- Analogous to **bagging**, where we average over ANNs trained on random samples of \mathcal{X}
- In each training iteration, sample a random bit vector μ , which determines which nodes are used (e.g., $P(\mu_i = 1) = 0.8$ for input unit, 0.5 for hidden unit)



- Make predictions by sampling new vectors μ and averaging

Regularization

Other Approaches

- **Parameter Tying:** If two learners are learning the same task but different distributions, can tie their parameters together
 - If $\mathbf{w}^{(A)}$ are weights for task A and $\mathbf{w}^{(B)}$ are weights for task B , then can use regularization term

$$\Omega(\mathbf{w}^{(A)}, \mathbf{w}^{(B)}) = \|\mathbf{w}^{(A)} - \mathbf{w}^{(B)}\|_2^2$$
- **Parameter Sharing:** When detecting objects in an image, the same recognizer should apply invariant to translation
 - Can train a single detector (subnetwork) for the object (e.g., cat) by training full network on multiple images with translated cats, where the cat detector subnetworks share parameters (single copy, used multiple times)

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Causes of Overfitting

Early Stopping

Parameter Norm
Penalties

Data Augmentation

Multitask Learning

Dropout

Others

Estimating
Generalization
Performance

Regularization

Other Approaches (cont'd)

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Causes of Overfitting
Early Stopping
Parameter Norm
Penalties

Data Augmentation
Multitask Learning
Dropout

Others

Estimating
Generalization
Performance

- **Sparse Representations:** Instead of penalizing large weights, penalize large outputs of hidden nodes:

$$\tilde{\mathcal{J}}(\theta; \mathcal{X}, \mathbf{y}) = \mathcal{J}(\theta; \mathcal{X}, \mathbf{y}) + \alpha \Omega(\mathbf{h}) \quad ,$$

where \mathbf{h} is the vector of hidden unit outputs

Estimating Generalization Performance

Setting Goals

- Before setting up an experiment, need to understand exactly what the goal is
 - Estimate the generalization performance of a hypothesis
 - Tuning a learning algorithm's parameters
 - Comparing two learning algorithms on a specific task
 - Etc.
- Will never be able to answer the question with 100% certainty
 - Due to variances in training set selection, test set selection, etc.
 - Will choose an **estimator** for the quantity in question, determine the probability distribution of the estimator, and bound the probability that the estimator is way off
 - Estimator needs to work regardless of distribution of training/testing data

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Estimating
Generalization
Performance

Setting Goals

Confidence Intervals

Comparing
Learning
Algorithms

Other/ 52

Estimating Generalization Performance

Setting Goals

CSCE 970

Lecture 3:
RegularizationStephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
ProblemsMeasuring
Performance

Regularization

Estimating
Generalization
Performance

Setting Goals

Confidence Intervals

Comparing
Learning
Algorithms

Other/ 52

- Need to note that, in addition to statistical variations, what we determine is limited to the application that we are studying
 - E.g., if ANN_1 better than ANN_2 on speech recognition, that means nothing about video analysis
- In planning experiments, need to ensure that training data not used for evaluation
 - I.e., **don't test on the training set!**
 - Will bias the performance estimator
 - Also holds for **validation set** used for early stopping, tuning parameters, etc.
 - Validation set serves as part of training set, but not used for model building

Confidence Intervals

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Varyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Estimating
Generalization
Performance

Setting Goals
Confidence Intervals

Comparing
Learning
Algorithms

12/17/52

Let $error_{\mathcal{D}}(h)$ be 0-1 loss of h on instances drawn according to distribution \mathcal{D} . If

- \mathcal{V} contains N examples, drawn independently of h and each other
- $N \geq 30$

Then with approximately 95% probability, $error_{\mathcal{D}}(h)$ lies in

$$error_{\mathcal{V}}(h) \pm 1.96 \sqrt{\frac{error_{\mathcal{V}}(h)(1 - error_{\mathcal{V}}(h))}{N}}$$

E.g. hypothesis h misclassifies 12 of the 40 examples in test set \mathcal{V} :

$$error_{\mathcal{V}}(h) = \frac{12}{40} = 0.30$$

Then with approx. 95% confidence, $error_{\mathcal{D}}(h) \in [0.158, 0.442]$

Confidence Intervals (cont'd)

Let $error_{\mathcal{D}}(h)$ be 0-1 loss of h on instances drawn according to distribution \mathcal{D} . If

- \mathcal{V} contains N examples, drawn independently of h and each other
- $N \geq 30$

Then with approximately $c\%$ probability, $error_{\mathcal{D}}(h)$ lies in

$$error_{\mathcal{V}}(h) \pm z_c \sqrt{\frac{error_{\mathcal{V}}(h)(1 - error_{\mathcal{V}}(h))}{N}}$$

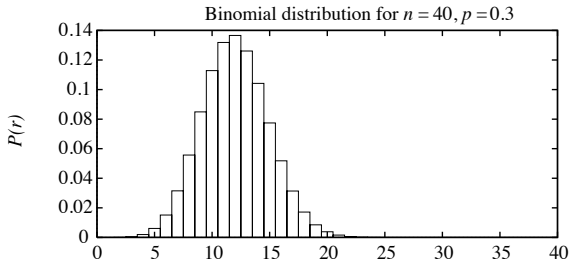
$N\%:$	50%	68%	80%	90%	95%	98%	99%
$z_c:$	0.67	1.00	1.28	1.64	1.96	2.33	2.58

Why?

$error_{\mathcal{V}}(h)$ is a Random Variable

Repeatedly run the experiment, each with different randomly drawn \mathcal{V} (each of size N)

Probability of observing r misclassified examples:



$$P(r) = \binom{N}{r} error_{\mathcal{D}}(h)^r (1 - error_{\mathcal{D}}(h))^{N-r}$$

I.e., let $error_{\mathcal{D}}(h)$ be probability of heads in biased coin, then $P(r)$ = prob. of getting r heads out of N flips

Binomial Probability Distribution

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Varyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Estimating
Generalization
Performance

Setting Goals
Confidence Intervals

Comparing
Learning
Algorithms

Other/ 52

$$P(r) = \binom{N}{r} p^r (1-p)^{N-r} = \frac{N!}{r!(N-r)!} p^r (1-p)^{N-r}$$

Probability $P(r)$ of r heads in N coin flips, if $p = \Pr(\text{heads})$

- Expected, or mean value of X , $E[X]$ (= # heads on N flips = # mistakes on N test exs), is

$$E[X] \equiv \sum_{i=0}^N iP(i) = Np = N \cdot \text{error}_{\mathcal{D}}(h)$$

- Variance of X is

$$\text{Var}(X) \equiv E[(X - E[X])^2] = Np(1-p)$$

- Standard deviation of X , σ_X , is

$$\sigma_X \equiv \sqrt{E[(X - E[X])^2]} = \sqrt{Np(1-p)}$$

Approximate Binomial Dist. with Normal

$error_V(h) = r/N$ is binomially distributed, with

- mean $\mu_{error_V(h)} = error_D(h)$ (i.e., unbiased est.)
- standard deviation $\sigma_{error_V(h)}$

$$\sigma_{error_V(h)} = \sqrt{\frac{error_D(h)(1 - error_D(h))}{N}}$$

(increasing N decreases variance)

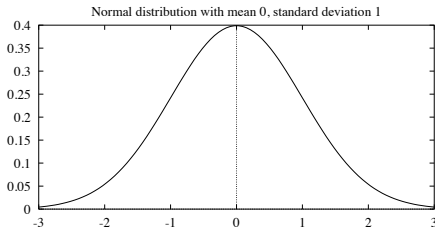
Want to compute confidence interval = interval centered at $error_D(h)$ containing $c\%$ of the weight under the distribution

Approximate binomial by **normal** (Gaussian) dist:

- mean $\mu_{error_V(h)} = error_D(h)$
- standard deviation $\sigma_{error_V(h)}$

$$\sigma_{error_V(h)} \approx \sqrt{\frac{error_V(h)(1 - error_V(h))}{N}}$$

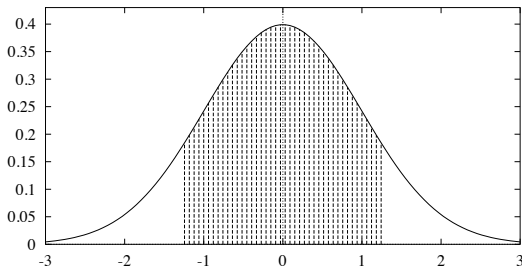
Normal Probability Distribution



$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right)$$

- The probability that X will fall into the interval (a, b) is given by $\int_a^b p(x) dx$
- Expected, or mean value of X , $E[X]$, is $E[X] = \mu$
- Variance is $Var(X) = \sigma^2$, standard deviation is $\sigma_X = \sigma$

Normal Probability Distribution (cont'd)



80% of area (probability) lies in $\mu \pm 1.28\sigma$

$c\%$ of area (probability) lies in $\mu \pm z_c \sigma$

$c\%$:	50%	68%	80%	90%	95%	98%	99%
z_c :	0.67	1.00	1.28	1.64	1.96	2.33	2.58

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Estimating
Generalization
Performance

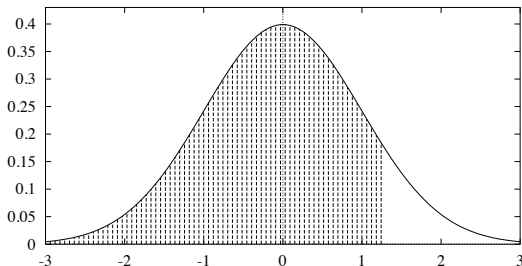
Setting Goals
Confidence Intervals

Comparing
Learning
Algorithms

Other/ 52

Normal Probability Distribution (cont'd)

Can also have **one-sided** bounds:



$c\%$ of area lies $< \mu + z'_c \sigma$ or $> \mu - z'_c \sigma$, where

$$z'_c = z_{100-(100-c)/2}$$

$c\%$:	50%	68%	80%	90%	95%	98%	99%
z'_c :	0.0	0.47	0.84	1.28	1.64	2.05	2.33

Confidence Intervals Revisited

If \mathcal{V} contains $N \geq 30$ examples, indep. of h and each other

Then with approximately 95% probability, $error_{\mathcal{V}}(h)$ lies in

$$error_{\mathcal{D}}(h) \pm 1.96 \sqrt{\frac{error_{\mathcal{D}}(h)(1 - error_{\mathcal{D}}(h))}{N}}$$

Equivalently, $error_{\mathcal{D}}(h)$ lies in

$$error_{\mathcal{V}}(h) \pm 1.96 \sqrt{\frac{error_{\mathcal{D}}(h)(1 - error_{\mathcal{D}}(h))}{N}}$$

which is approximately

$$error_{\mathcal{V}}(h) \pm 1.96 \sqrt{\frac{error_{\mathcal{V}}(h)(1 - error_{\mathcal{V}}(h))}{N}}$$

(One-sided bounds yield upper or lower error bounds)

Central Limit Theorem

How can we justify approximation?

Consider set of iid random variables Y_1, \dots, Y_N , all from **arbitrary** probability distribution with mean μ and finite variance σ^2 . Define sample mean $\bar{Y} \equiv (1/N) \sum_{i=1}^N Y_i$

\bar{Y} is itself a random variable, i.e., result of an experiment (e.g., $error_S(h) = r/N$)

Central Limit Theorem: As $N \rightarrow \infty$, the distribution governing \bar{Y} approaches normal distribution with mean μ and variance σ^2/N

Thus the distribution of $error_S(h)$ is approximately normal for large N , and its expected value is $error_D(h)$

(Rule of thumb: $N \geq 30$ when estimator's distribution is binomial; might need to be larger for other distributions)

Calculating Confidence Intervals

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Estimating
Generalization
Performance

Setting Goals
Confidence Intervals

Comparing
Learning
Algorithms

Other/ 52

- 1 Pick parameter to estimate: $error_{\mathcal{D}}(h)$
(0-1 loss on distribution \mathcal{D})
- 2 Choose an estimator: $error_{\mathcal{V}}(h)$
(0-1 loss on independent test set \mathcal{V})
- 3 Determine probability distribution that governs estimator: $error_{\mathcal{V}}(h)$ governed by binomial distribution, approximated by normal when $N \geq 30$
- 4 Find interval (L, U) such that $c\%$ of probability mass falls in the interval
 - Could have $L = -\infty$ or $U = \infty$
 - Use table of z_c or z'_c values (if distribution normal)

Comparing Learning Algorithms

CSCE 970

Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

K -Fold CV

Student's t
Distribution

38 / 52

Other

- What if we want to compare two learning algorithms L^1 and L^2 (e.g., two ANN architectures, two regularizers, etc.) on a specific application?
- Insufficient to simply compare error rates on a single test set
- Use **K -fold cross validation** and a **paired t test**

K-Fold Cross Validation

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Varyiam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

K-Fold CV

Student's t
Distribution

39 / 52

- ➊ Partition data set \mathcal{X} into K equal-sized subsets $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_K$, where $|\mathcal{X}_i| \geq 30$
- ➋ For i from 1 to K , do
(Use \mathcal{X}_i for testing, and rest for training)
 - ➊ $\mathcal{V}_i = \mathcal{X}_i$
 - ➋ $\mathcal{T}_i = \mathcal{X} \setminus \mathcal{X}_i$
 - ➌ Train learning algorithm L^1 on \mathcal{T}_i to get h_i^1
 - ➍ Train learning algorithm L^2 on \mathcal{T}_i to get h_i^2
 - ➎ Let p_i^j be error of h_i^j on test set \mathcal{V}_i
 - ➏ $p_i = p_i^1 - p_i^2$
- ➌ Error difference estimate $p = (1/K) \sum_i^K p_i$

K-Fold Cross Validation (cont'd)

- Now estimate confidence that true expected error difference < 0

⇒ Confidence that L^1 is better than L^2 on learning task

- Use **one-sided test**, with confidence derived from **student's t distribution** with $K - 1$ **degrees of freedom**
- With approximately $c\%$ probability, true difference of expected error between L^1 and L^2 is at most

$$p + t_{c,K-1} s_p$$

where

$$s_p \equiv \sqrt{\frac{1}{K(K-1)} \sum_{i=1}^K (p_i - p)^2}$$

Student's t Distribution (One-Sided Test)

df	0.600	0.700	0.800	0.900	0.950	0.975	0.990	0.995
1	0.325	0.727	1.376	3.078	6.314	12.706	31.821	63.657
2	0.289	0.617	1.061	1.886	2.920	4.303	6.965	9.925
3	0.277	0.584	0.978	1.638	2.353	3.182	4.541	5.841
4	0.271	0.569	0.941	1.533	2.132	2.776	3.747	4.604
5	0.267	0.559	0.920	1.476	2.015	2.571	3.365	4.032
6	0.265	0.553	0.906	1.440	1.943	2.447	3.143	3.707
7	0.263	0.549	0.896	1.415	1.895	2.365	2.998	3.499
8	0.262	0.546	0.889	1.397	1.860	2.306	2.896	3.355
9	0.261	0.543	0.883	1.383	1.833	2.262	2.821	3.250
10	0.260	0.542	0.879	1.372	1.812	2.228	2.764	3.169
11	0.260	0.540	0.876	1.363	1.796	2.201	2.718	3.106
12	0.259	0.539	0.873	1.356	1.782	2.179	2.681	3.055
13	0.259	0.538	0.870	1.350	1.771	2.160	2.650	3.012

If $p + t_{c,K-1} s_p < 0$ our assertion that L^1 has less error than L^2 is supported with confidence c

So if K -fold CV used, compute p , look up $t_{c,K-1}$ and check if $p < -t_{c,K-1} s_p$

One-sided test; says nothing about L^2 over L^1

CSCE 970

Lecture 3:

Regularization

Stephen Scott
and Vinod
Varyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

K -Fold CV

Student's t
Distribution

42 / 52

- Say you want to show that learning algorithm L^1 performs better than algorithms L^2, L^3, L^4, L^5
- If you use K -fold CV to show superior performance of L^1 over each of L^2, \dots, L^5 at 95% confidence, there's a 5% chance each one is wrong
- ⇒ There's an over 18.5% chance that at least one is wrong
- ⇒ Our overall confidence is only just over 81%
 - Need to account for this, or use more appropriate test

More Specific Performance Measures

CSCE 970

Lecture 3:
RegularizationStephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
ProblemsMeasuring
Performance

Regularization

Estimating
Generalization
PerformanceComparing
Learning
AlgorithmsOther
Performance
Measures

- So far, we've looked at a single error rate to compare hypotheses/learning algorithms/etc.
- This may not tell the whole story:
 - 1000 test examples: 20 positive, 980 negative
 - h^1 gets 2/20 pos correct, 965/980 neg correct, for accuracy of $(2 + 965)/(20 + 980) = 0.967$
 - Pretty impressive, except that always predicting negative yields accuracy = 0.980
 - Would we rather have h^2 , which gets 19/20 pos correct and 930/980 neg, for accuracy = 0.949?
 - Depends on how important the positives are, i.e., frequency in practice and/or cost (e.g., cancer diagnosis)

Confusion Matrices

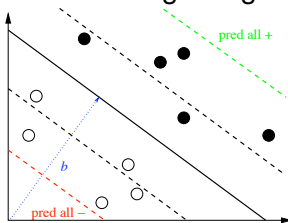
Break down error into type: true positive, etc.

True Class	Predicted Class		
	Positive	Negative	Total
Positive	tp : true positive	fn : false negative	p
Negative	fp : false positive	tn : true negative	n
Total	p'	n'	N

Generalizes to multiple classes

Allows one to quickly assess which classes are missed the most, and into what other class

- Consider classification via ANN + linear threshold unit
- Normally threshold $f(\mathbf{x}; \mathbf{w}, b)$ at 0, but what if we changed it?
- Keeping \mathbf{w} fixed while changing threshold = fixing hyperplane's slope while moving along its normal vector



- Get a **set** of classifiers, one per labeling of test set
- Similar situation with any classifier with confidence value, e.g., probability-based

ROC Curves

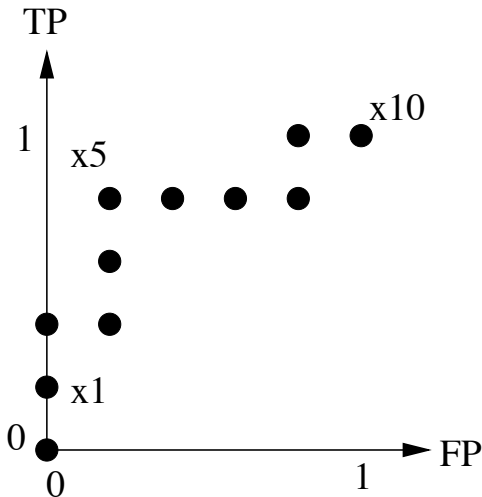
Plotting tp versus fp

- Consider the “always –” hyp. What is fp ? What is tp ?
What about the “always +” hyp?
- In between the extremes, we plot TP versus FP by sorting the test examples by the confidence values

Ex	Confidence	label	Ex	Confidence	label
x_1	169.752	+	x_6	-12.640	-
x_2	109.200	+	x_7	-29.124	-
x_3	19.210	-	x_8	-83.222	-
x_4	1.905	+	x_9	-91.554	+
x_5	-2.75	+	x_{10}	-128.212	-

ROC Curves

Plotting tp versus fp (cont'd)



CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

Measuring
Performance

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

Other
Performance
Metrics

ROC Curves

Convex Hull

CSCE 970
Lecture 3:
Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

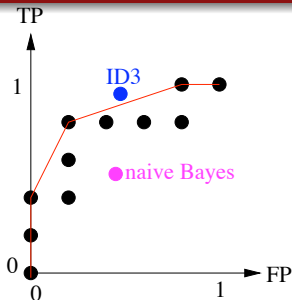
Measuring
Performance

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

Other
Performance
Measures



- The **convex hull** of the ROC curve yields a collection of classifiers, each optimal under different conditions
 - If FP cost = FN cost, then draw a line with slope $|N|/|P|$ at $(0, 1)$ and drag it towards convex hull until you touch it; that's your **operating point**
 - Can use as a classifier any part of the hull since can randomly select between two classifiers

ROC Curves

Convex Hull

CSCE 970

Lecture 3:

Regularization

Stephen Scott
and Vinod
Variyam

Introduction

Outline

Machine
Learning
Problems

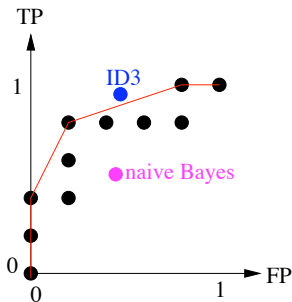
Measuring
Performance

Regularization

Estimating
Generalization
Performance

Comparing
Learning
Algorithms

Other
Performance
Measures



- Can also compare curves against “single-point” classifiers when no curves
 - In plot, **ID3** better than our SVM iff negatives scarce; **nB** never better

- What is the worst possible ROC curve?
- One metric for measuring a curve's goodness: **area under curve** (AUC):

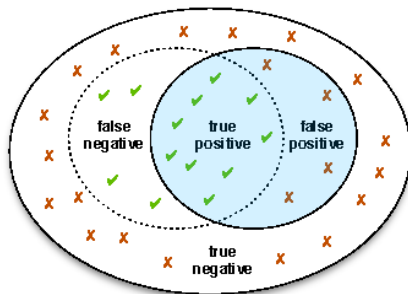
$$\frac{\sum_{x_+ \in P} \sum_{x_- \in N} I(h(x_+) > h(x_-))}{|P| |N|}$$

i.e., rank all examples by confidence in “+” prediction, count the number of times a positively-labeled example (from P) is ranked above a negatively-labeled one (from N), then normalize

- What is the best value?
 - Distribution approximately normal if $|P|, |N| > 10$, so can find confidence intervals
 - Catching on as a better scalar measure of performance than error rate
- ROC analysis possible (though tricky) with multi-class problems

Precision-Recall Curves

Consider information retrieval task, e.g., web search



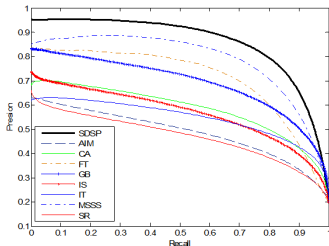
○ All documents ✓ relevant ✗ not relevant ● retrieved

precision = tp/p' = fraction of retrieved that are positive

recall = tp/p = fraction of positives retrieved

Precision-Recall Curves (cont'd)

As with ROC, can vary threshold to trade off precision against recall



Can compare curves based on containment

Use F_β -measure to combine at a specific point, where β weights precision vs recall:

$$F_\beta \equiv (1 + \beta^2) \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$