PERFORMANCE EVALUATION

CS6140

Predrag Radivojac
KHOURY COLLEGE OF COMPUTER SCIENCES
NORTHEASTERN UNIVERSITY

Fall 2024

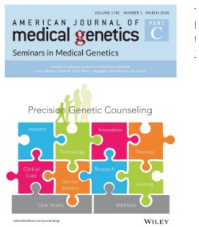
Need for (Deep) Evaluation in Machine Learning

A turtle—or a rifle? Hackers easily fool AIs into seeing the wrong thing

By Matthew Hutson | Jul. 19, 2018 , 2:15 PM

With the help of stickers, image recognition algorithms were tricked into thinking a stop sign was a speed limit sign. K. EYKHOLT *ET AL.*, ARXIV:1707.08945 (2017)

Pages: i | First Published: 19 April 2018



Abstract | PDF | Request permissions

Artificial Intelligence Is Rushing Into Patient Care—And Could Raise Risks

AI systems are not as rigorously tested as other medical devices, and have already made serious mistakes

Audio Adversarial Examples: Targeted Attacks on Speech-to-Text

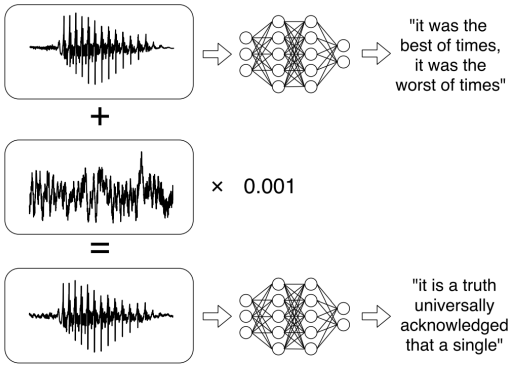
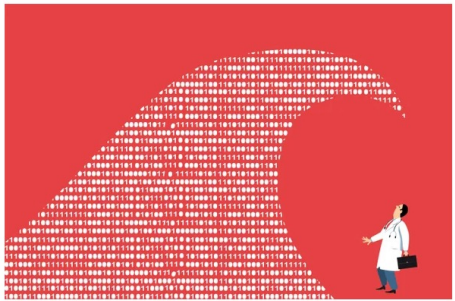


Figure 1. Illustration of our attack: given any waveform, adding a small perturbation makes the result transcribe as any desired target phrase.



Credit: Getty Images

Evaluation in Machine Learning

Machine learning:

- field with strong theory, rooted in probability, statistics, computer science
- algorithms and framorks must work on data from the real world
- algorithms and models must be evaluated on data from the real world
- need mechanisms to find the best model or algorithm out of many options

Empirical evaluation:

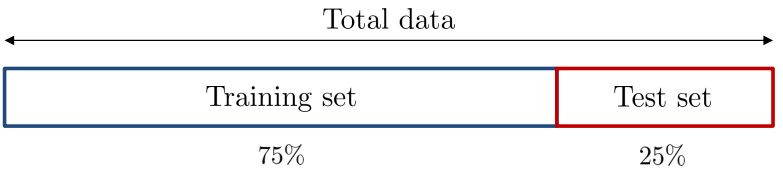
- core part of machine learning
- gives confidence to deploy
- our focus on accuracy estimation

Related important concepts:

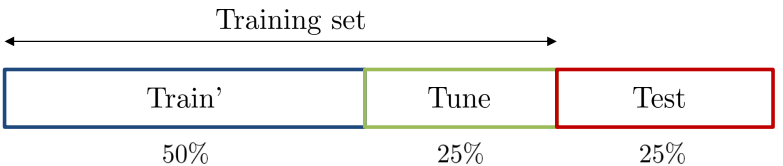
- reliability
- interpretability
- fairness

Basic Idea for the Protocol (All Data Labeled)

- randomly split data into training set and test set (i.i.d.)
- train a model on the training set; predict on the test set and check accuracy



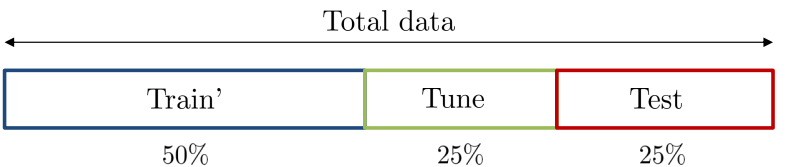
- works when we train a single model and have a lot of data
- what if we have to pick between a few models?



Tune set = Development set

How Does the Process Work?

- train many models on Train' (e.g., vary parameters)
- pick the best parameters using Tune based on some criterion ← model selection
- train a model with selected parameters on Train = Train' + Tune
- compute accuracy on Test ← estimated performance
- train the final model on all data (+ model selection) ← final model
- deploy the final model with the hope it works as well as we think

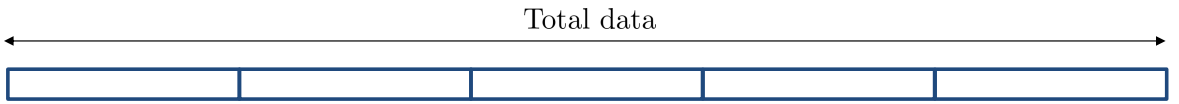


Test set can be used only once, at the very end.

Cross-Validation

- data less abundant relative to the complexity of model space
- partition data into N groups (uniformly randomly)

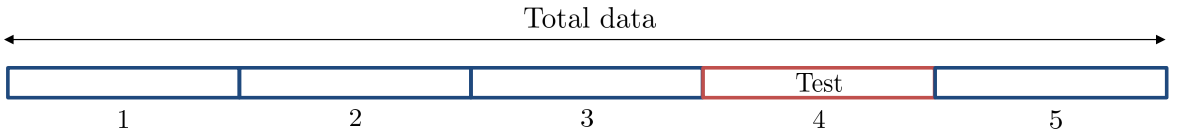
$N = 5$



Repeat N times
- (step i) Test = partition i; Training = the rest
- (step i) Compute accuracy of step i

Average N accuracy measures at the end

$i = 4$



N-Fold Cross-Validation

f^{-i} = model trained on partitions $\{1, 2, \ldots, N\} \setminus \{i\}$

perf^{-i} = performance of f^{-i} on test partition i

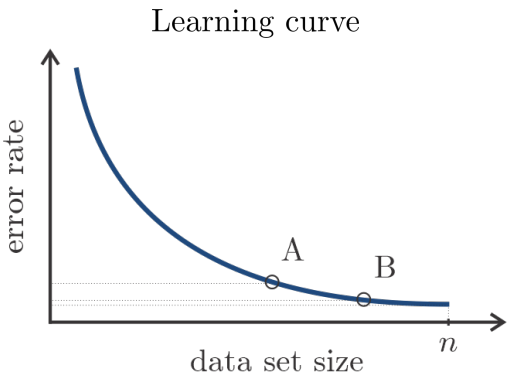
perf = final performance, averaged over N different perf^{-i}'s

Two questions:

- Is it a problem that each f^{-i} is different? No.
- How should we choose N?

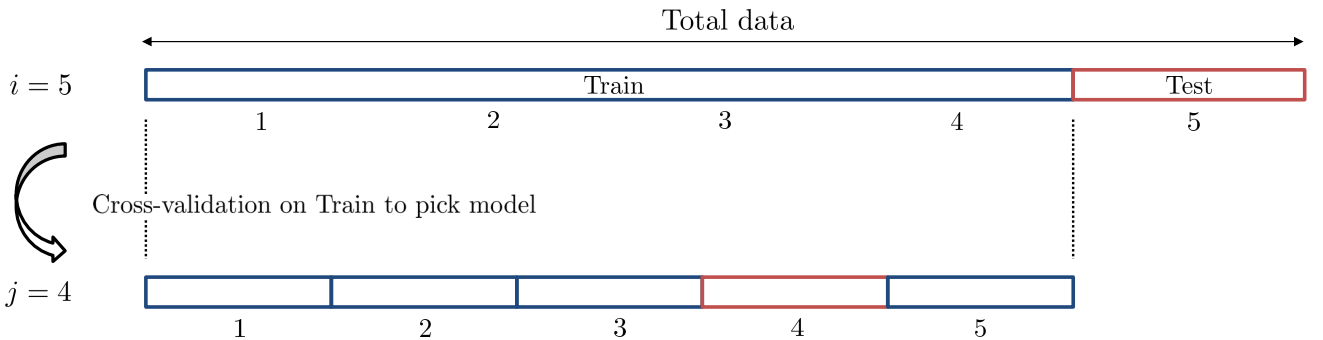
Choice of N:

- point A: high bias, low variance
- point B: low bias, high variance
- often $N = 10$, giving 10-fold cross-validation
- when $N = n$, it is called leave-one-out estimation



Nested Cross-Validation

- old: Train = Train' + Tune → best model
- new: N-fold cross-validation on Train → best model



Run time:

- each parameter combination is trained N^2 times; $\forall i, j \in \{1, 2, \ldots, N\}$
- best models are trained N times combined

The Roles of the Training Data

Estimate accuracy:

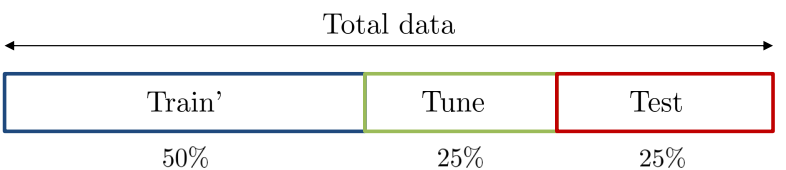
- nested cross-validation on the entire data set
- this accuracy is reported for the model trained at the very end

Choose parameters:

- cross-validation on the entire data set
- the accuracy of this model may not match the estimated accuracy

Train final model:

- retrain the best performing model on the entire data
- this model is never really evaluated (unless new data accumulates)



Other Evaluation Scenarios

Random sampling:

- randomly sample w/o replacement a fraction of the data set
- train on training set, test on test set
- repeat many times

Bootstrapping:

- sample w/ replacement n examples from the data set
- train on training set, test on examples not selected during sampling
- repeat many times

Out-of-bag evaluation:

- applies to ensembles of models (committee machines)
- performs boostrap-like estimation using multiple trained models

CLASSIFICATION

Given: Test data: $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$, $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, drawn i.i.d. from $p(x, y)$

Trained model: $f(x) \in \mathcal{Y}$

Error rate and accuracy:

$$\text{error} = P(f(X) \neq Y)$$

$$\text{accuracy} = 1 - \text{error}$$

Assumption: $\mathcal{D}$ sampled i.i.d. from $p(x, y)$

$e_T = P(f(X) \neq Y)$ ← true error

$e_S = \frac{1}{n} \sum_{i=1}^{n} I(f(x_i) \neq y_i)$ ← estimated (sample) error

Confidence intervals:

With 95% probability it holds that:

$$e_T = e_S \pm 1.96 \cdot \underbrace{\sqrt{\frac{e_S(1 - e_S)}{n}}}_{\text{standard error}}$$

Details later...

Binary Classification: Metrics

Given: Test data: $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, $x \in \mathcal{X}$ and $y \in \mathcal{Y} = \{0, 1\}$, drawn i.i.d. from $p(x, y)$

Trained model: $f(x) \in \mathcal{Y}$

True positive rate, sensitivity, recall:	$P(f(X) = 1 \mid Y = 1)$
False positive rate:	$P(f(X) = 1 \mid Y = 0)$
True negative rate, specificity:	$P(f(X) = 0 \mid Y = 0) = 1 - P(f(X) = 1 \mid Y = 0)$
False negative rate:	$P(f(X) = 0 \mid Y = 1) = 1 - P(f(X) = 1 \mid Y = 1)$
Precision, positive predictive value:	$P(Y = 1 \mid f(X) = 1)$
False discovery rate:	$P(Y = 0 \mid f(X) = 1) = 1 - P(Y = 1 \mid f(X) = 1)$
Negative predictive value:	$P(Y = 0 \mid f(X) = 0)$

Binary Classification: Derived Metrics

Given: Test data: $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, $x \in \mathcal{X}$ and $y \in \mathcal{Y} = \{0, 1\}$, drawn i.i.d. from $p(x, y)$

Trained model: $f(x) \in \mathcal{Y}$

Balanced accuracy: $$\frac{\text{sensitivity} + \text{specificity}}{2}$$

Matthews correlation: $\text{Corr}[f(X), Y]$

F-measure: $$\frac{1}{\alpha \cdot \frac{1}{\text{precision}} + (1-\alpha) \cdot \frac{1}{\text{recall}}} = \frac{(\beta^2 + 1) \cdot \text{precision} \cdot \text{recall}}{\beta^2 \cdot \text{precision} + \text{recall}} \qquad \beta^2 = \frac{1-\alpha}{\alpha}$$

$$F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \qquad \text{when } \alpha = \tfrac{1}{2},\ \beta = 1$$

ESTIMATION

Confusion matrix:

True class \ Predicted class	positive	negative
positive	tp	fn
negative	fp	tn

tp = true positives tn = true negatives
fp = false positives fn = false negatives

$$\mathrm{tp} + \mathrm{tn} + \mathrm{fp} + \mathrm{fn} = n$$

Estimated quantities:

$$\mathrm{tpr} = \frac{\mathrm{tp}}{\mathrm{tp}+\mathrm{fn}}$$

$$\mathrm{fpr} = \frac{\mathrm{fp}}{\mathrm{fp}+\mathrm{tn}}$$

$$\mathrm{fdr} = \frac{\mathrm{fp}}{\mathrm{fp}+\mathrm{tp}}$$

$$\mathrm{accuracy} = \frac{\mathrm{tp}+\mathrm{tn}}{n}$$

$$\mathrm{error} = \frac{\mathrm{fp}+\mathrm{fn}}{n}$$

$$\mathrm{mcc} = \frac{\mathrm{tp}\cdot\mathrm{tn}-\mathrm{fp}\cdot\mathrm{fn}}{\sqrt{(\mathrm{tp}+\mathrm{fp})(\mathrm{tp}+\mathrm{fn})(\mathrm{tn}+\mathrm{fp})(\mathrm{tn}+\mathrm{fn})}}$$

ESTIMATION

Confusion matrix:

True class \ Predicted class	positive	negative
positive	tp	fn
negative	fp	tn

tp = true positives tn = true negatives
fp = false positives fn = false negatives

$$\mathrm{tp} + \mathrm{tn} + \mathrm{fp} + \mathrm{fn} = n$$

Estimated quantities:

$$\mathrm{tpr} = \frac{\mathrm{tp}}{\mathrm{tp}+\mathrm{fn}}$$

$$\mathrm{fpr} = \frac{\mathrm{fp}}{\mathrm{fp}+\mathrm{tn}}$$

$$\mathrm{fdr} = \frac{\mathrm{fp}}{\mathrm{fp}+\mathrm{tp}}$$

$$\mathrm{accuracy} = \frac{\mathrm{tp}+\mathrm{tn}}{n}$$

$$\mathrm{error} = \frac{\mathrm{fp}+\mathrm{fn}}{n}$$

$$\mathrm{mcc} = \frac{\mathrm{tp}\cdot\mathrm{tn}-\mathrm{fp}\cdot\mathrm{fn}}{\sqrt{(\mathrm{tp}+\mathrm{fp})(\mathrm{tp}+\mathrm{fn})(\mathrm{tn}+\mathrm{fp})(\mathrm{tn}+\mathrm{fn})}}$$

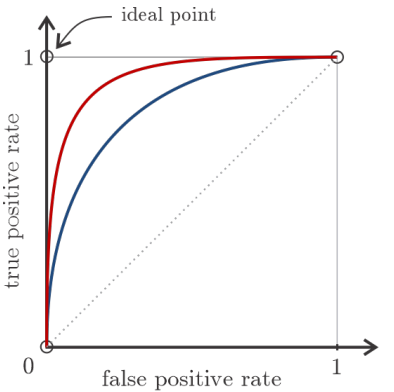
Two-Dimensional Plots

- a predictor is typically given as a score function $s(x) \in \mathbb{R}$
- a binarized prediction $f(x)$ is usually obtained by thresholding $s(x)$ as

$$f_\tau(x) = \begin{cases} 1 & s(x) \geq \tau \\ 0 & s(x) < \tau \end{cases}$$

Receiver Operating Chacteristic (ROC) Curve:

- consider a series of thresholds $\mathcal{T}$ and predictors $f_\tau(x)$, $\tau \in \mathcal{T}$
- each $f_\tau(x)$ gives a true positive and a false positive rate
- ROC curve is a plot of tpr_τ as a function of fpr_τ
- the red predictor is better then the blue; see Figure



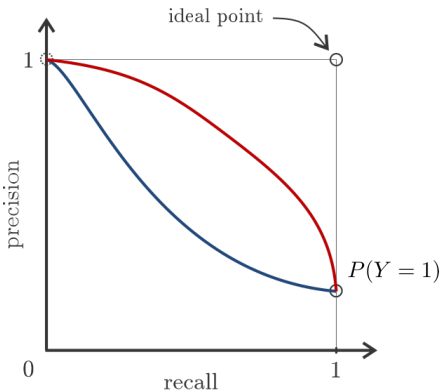
Two-Dimensional Plots

- a predictor is typically given as a score function $s(x) \in \mathbb{R}$
- a binarized prediction $f(x)$ is usually obtained by thresholding $s(x)$ as

$$f_\tau(x) = \begin{cases} 1 & s(x) \geq \tau \\ 0 & s(x) < \tau \end{cases}$$

Precision-Recall Curve:

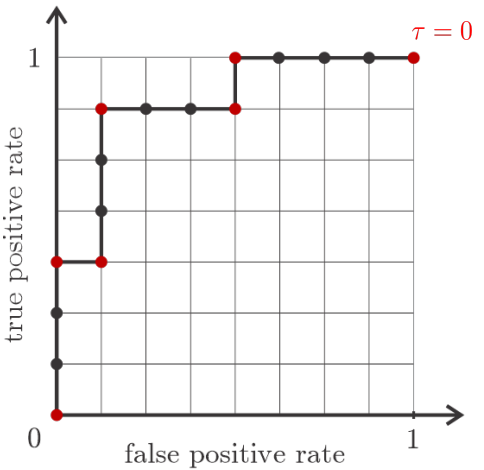
- consider a series of thresholds $\mathcal{T}$ and predictors $f_\tau(x)$, $\tau \in \mathcal{T}$
- each $f_\tau(x)$ gives a precision and a recall
- pr-rc curve is a plot of pr_τ as a function of rc_τ
- the red predictor is better then the blue; see Figure



Area Under the ROC Curve (AUC)

- we have a data set with $n = 15$ points and predictions $s(x) \in [0, 1]$
- there are $n_1 = 7$ positive and $n_0 = 8$ negative examples

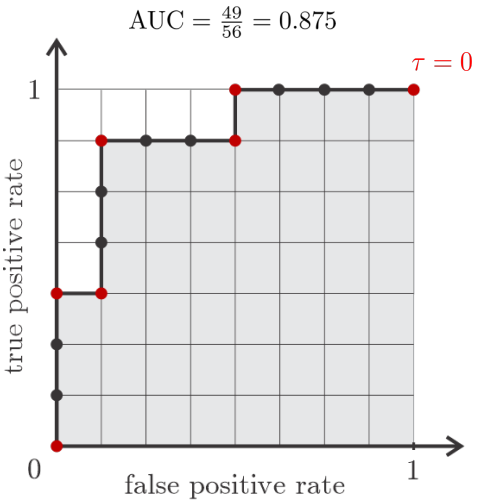
Prediction	True class	
0.953	1	$\tau = 1.001$
0.920	1	
0.799	1	$\tau = 0.794$
0.788	0	$\tau = 0.769$
0.750	1	
0.679	1	
0.612	1	$\tau = 0.598$
0.583	0	
0.477	0	
0.378	0	$\tau = 0.373$
0.367	1	$\tau = 0.306$
0.248	0	
0.214	0	
0.157	0	
0.112	0	$\tau = 0.000$



Area Under the ROC Curve (AUC)

- we have a data set with $n = 15$ points and predictions $s(x) \in [0, 1]$
- there are $n_1 = 7$ positive and $n_0 = 8$ negative examples

Prediction	True class	
0.953	1	$\tau = 1.001$
0.920	1	
0.799	1	$\tau = 0.794$
0.788	0	$\tau = 0.769$
0.750	1	
0.679	1	
0.612	1	$\tau = 0.598$
0.583	0	
0.477	0	
0.378	0	$\tau = 0.373$
0.367	1	$\tau = 0.306$
0.248	0	
0.214	0	
0.157	0	
0.112	0	$\tau = 0.000$



AUC IS A RANKING MEASURE

- two models that provide the same ranking of examples in a data set have identical AUCs

Model 1

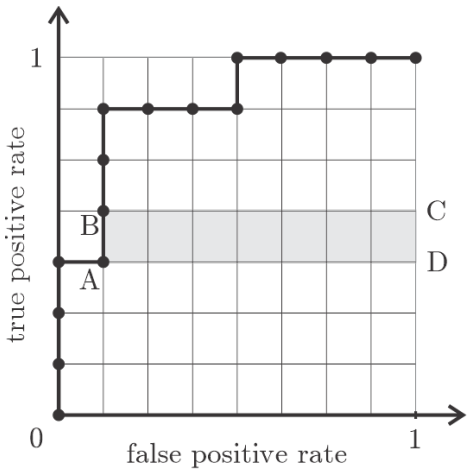
Example	Prediction	True class
6	0.953	1
9	0.920	1
5	0.799	1
13	0.788	0
14	0.750	1
12	0.679	1
7	0.612	1
10	0.583	0
8	0.477	0
11	0.378	0
2	0.367	1
4	0.248	0
1	0.214	0
3	0.157	0
15	0.112	0

Model 2

Example	Prediction	True class
6	1.000	1
9	0.999	1
5	0.998	1
13	0.909	0
14	0.895	1
12	0.888	1
7	0.845	1
10	0.844	0
8	0.830	0
11	0.828	0
2	0.818	1
4	0.817	0
1	0.816	0
3	0.814	0
15	0.800	0

INTERPRETATION OF AUC

AUC = probability that a randomly chosen positive example has a higher score than a randomly chosen negative example



* = no two data points have the same prediction score

Proof.* ABCD area $= \frac{1}{n_1} \cdot \frac{n_0 - 1}{n_0}$ ← 1 false positive ($f_4 = 1$; B is 4th positive)

f = number of false positives at present point

$$\text{AUC} \overset{\text{est}}{=} \frac{1}{n_1 n_0} \sum_{i=1}^{n_1} (n_0 - f_i)$$

x^+ = positive example; x^- = negative example; $s(x)$ = prediction on x

$$S(x^+, x^-) = \begin{cases} 1 & s(x^+) > s(x^-) \\ 0 & \text{otherwise} \end{cases}$$

$$P(s(x^+) > s(x^-)) \overset{\text{est}}{=} \frac{\sum_{i=1}^{n_1} \sum_{j=1}^{n_0} S(x_i^+, x_j^-)}{n_1 n_0} = \frac{1}{n_1 n_0} \sum_{i=1}^{n_1} (n_0 - f_i)$$

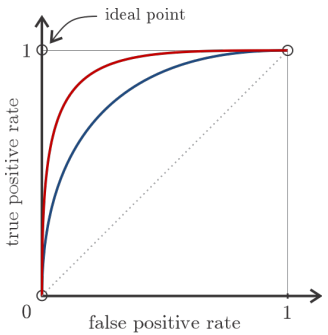
because $\sum_{j=1}^{n_0} S(x_i^+, x_j^-) = n_0 - f_i$

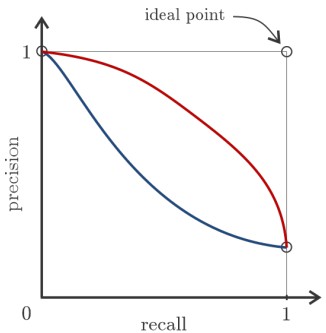
ROC Curve vs. Pr-Rc Curve

Given: Two models and a data set

Claim 1: If ROC curve 1 dominates ROC curve 2, then the corresponding pr-rc curve 1 dominates pr-rc curve 2.

Claim 2: Algorithms that optimize for area under the ROC curve do not necessarily optimize for the area under the pr-rc curve.





Davis & Goadrich, ICML 2006.

Trivial and Random Classifiers

Let $p(y)$ be the distribution of class priors; $y \in \mathcal{Y}$.

Trivial classifier: predicts the most prevalent class from the training set.

$$\hat{y} = \arg\max_{y \in \mathcal{Y}} \{p(y)\} \qquad \leftarrow \text{maximum a priori (instead of "a posteriori")}$$

Random classifier: Draw a random number from options in $\mathcal{Y}$ according to $p(y)$.

Example: below is a confusion matrix for some classifier.

What is the accuracy of trivial and random classifier?

$$P(\text{corr}) = P(\text{corr}|+)P(+) + P(\text{corr}|-)P(-)$$

Predicted class

True class	positive	negative
positive	90	10
negative	90	810

$p(Y = 0) = \frac{900}{1000}$

$p(Y = 1) = \frac{100}{1000}$

Trivial: $\max(\frac{900}{1000}, \frac{100}{1000}) = 0.90$

Random: $\left(\frac{100}{1000}\right)^2 + \left(\frac{900}{1000}\right)^2 = 0.82$

Accuracy of Trivial and Random Classifiers

Trivial classifier:

$$\text{accuracy} = \max_{y \in \mathcal{Y}} \{p(y)\}$$

Random classifier:

$$\text{accuracy} = \sum_{y \in \mathcal{Y}} p(y)^2 \qquad 1 - \text{gini index (used for classification trees)}$$

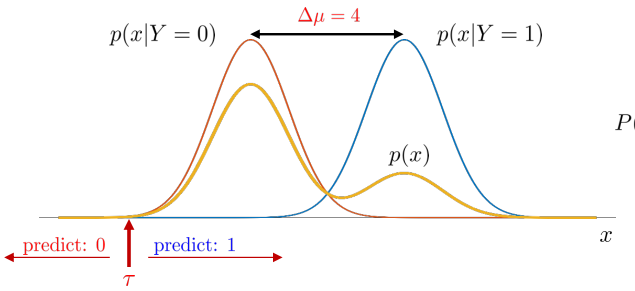
- these two are identical when $p(y)$ is uniform
- estimates are found by estimating class priors $p(y)$

Relationship to decision tree splitting criteria:

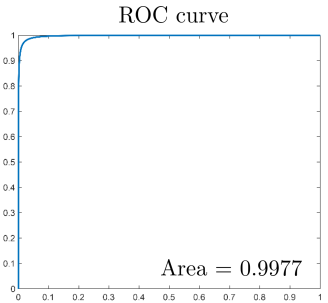
- minimum error: error of a trivial classifier
- gini index: error of a random classifier

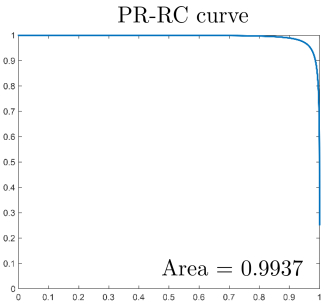
Imbalanced Data: Impact on 2D Measures

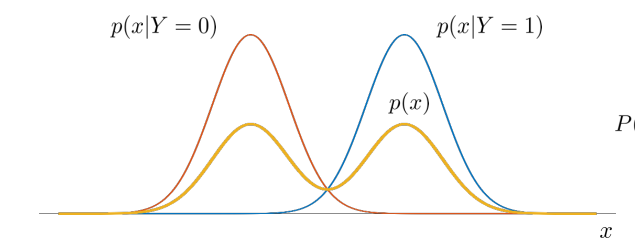
- class-conditional distributions are unit-variance Gaussians with $\Delta\mu = 4$; predictor based on a threshold.
- imbalanced labeled data vs. balanced labeled data



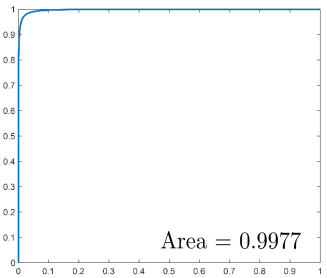
$P(Y=1) = \frac{1}{4}$

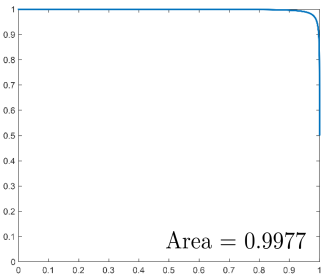






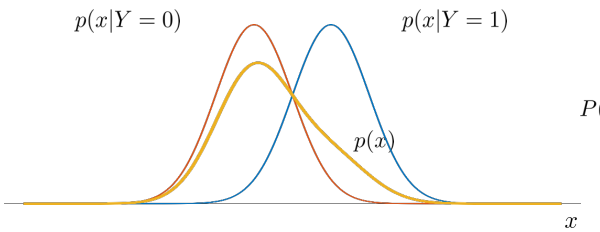
$P(Y=1) = \frac{1}{2}$



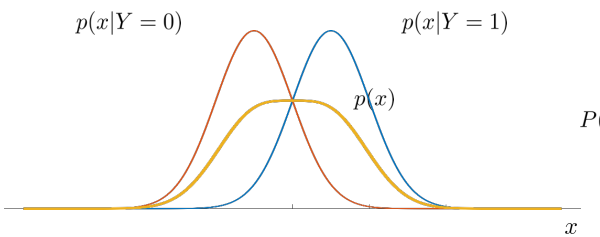


Imbalanced Data: Impact on 2D Measures

- class-conditional distributions are unit-variance Gaussians with $\Delta\mu = 2$; predictor based on a threshold.
- imbalanced labeled data vs. balanced labeled data

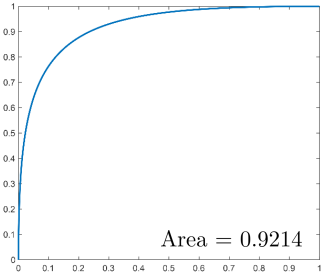


$P(Y=1) = \frac{1}{4}$

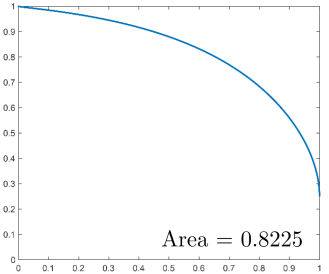


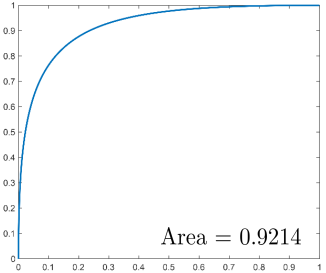
$P(Y=1) = \frac{1}{2}$

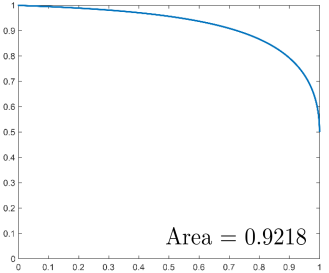
ROC curve



PR-RC curve

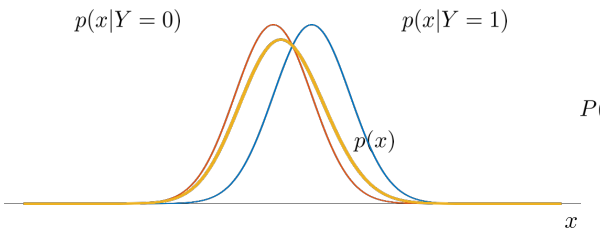




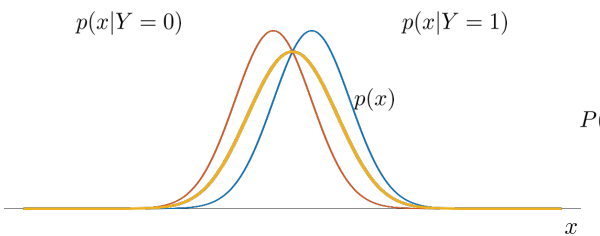


Imbalanced Data: Impact on 2D Measures

- class-conditional distributions are unit-variance Gaussians with $\Delta\mu = 1$; predictor based on a threshold.
- imbalanced labeled data vs. balanced labeled data

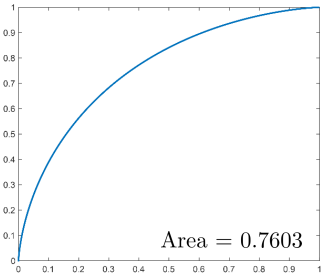


$P(Y=1) = \frac{1}{4}$

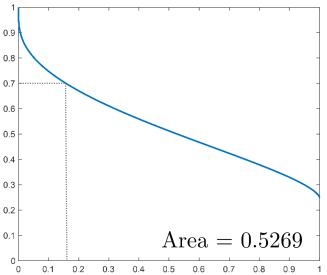


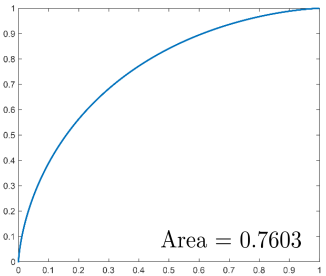
$P(Y=1) = \frac{1}{2}$

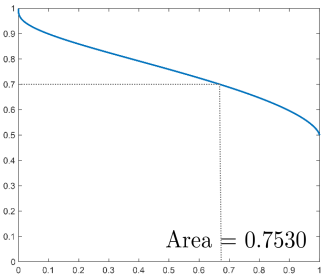
ROC curve



PR-RC curve

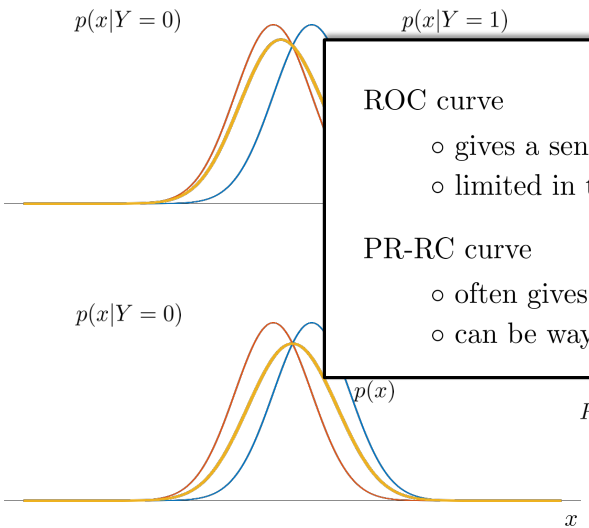






Imbalanced Data: Impact on 2D Measures

- class-conditional distributions are unit-variance Gaussians with $\Delta\mu = 1$; predictor based on a threshold.
- imbalanced labeled data vs. balanced labeled data

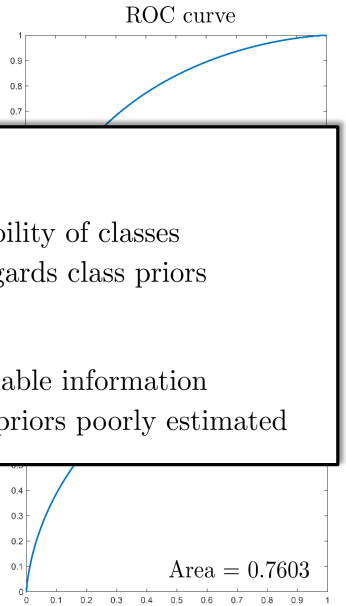


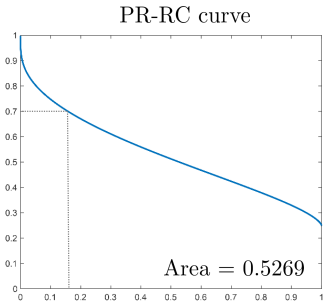
ROC curve

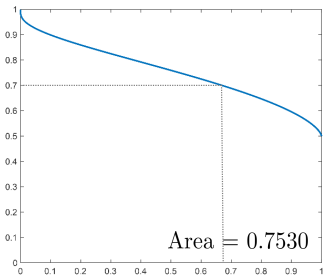
- gives a sense of separability of classes
- limited in that it disregards class priors

PR-RC curve

- often gives more actionable information
- can be way off if class priors poorly estimated

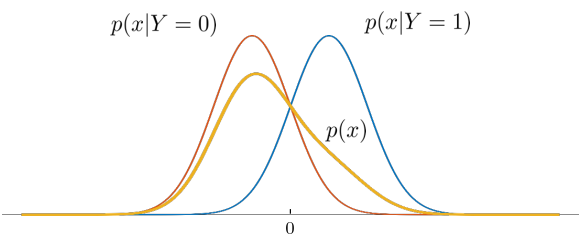


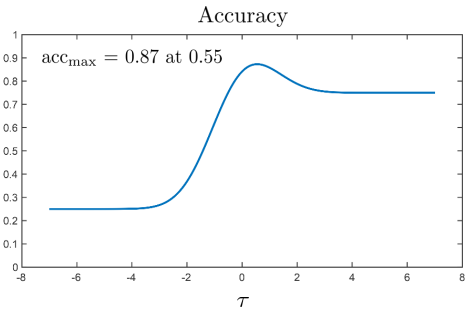


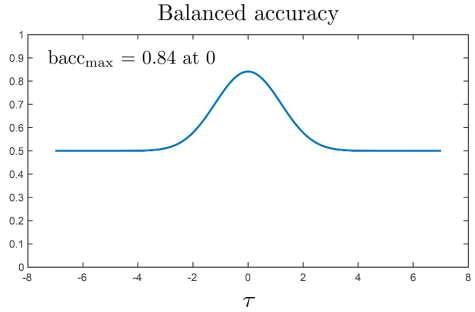


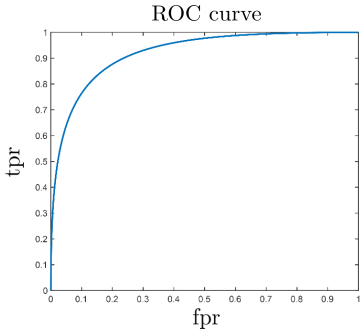
Imbalanced Data: Impact on 1D Measures

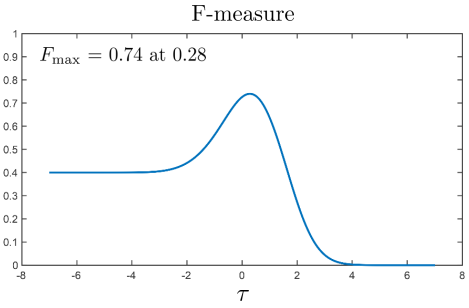
- consider $\Delta\mu = 2$, symmetric around 0, and $P(Y = 1) = \frac{1}{4}$

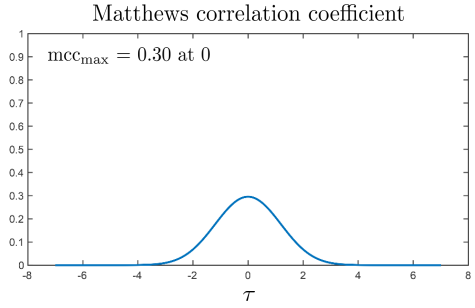












Noisy Class Labels: Model

Before we look at impact on performance estimation, let's look at model

- Y = true but unknown class label
- Z = noisy class label

- $p(x)$ = unchanged by class label noise
- $p(x|z) \neq p(x|y)$ and $p(z) \neq p(y)$

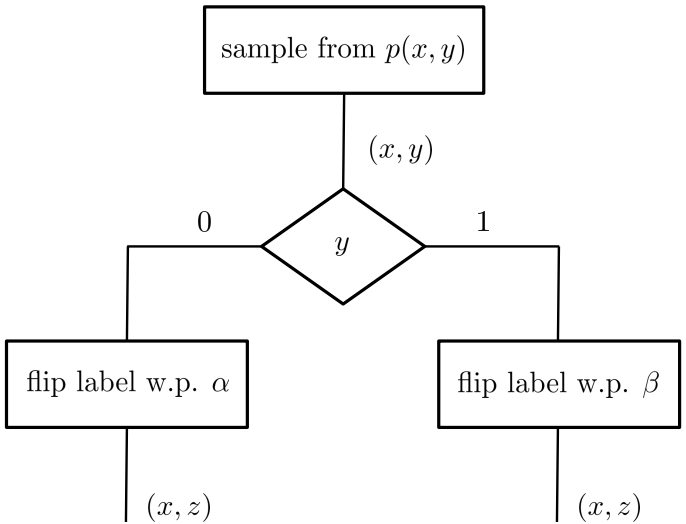
Expressing $p(z)$, $p(z|x)$, $p(x|z)$

$$p_Z(1) = p_Y(1) \cdot (1 - \beta) + p_Y(0) \cdot \alpha$$

$$p(Z = 1|x) = p(Y = 1|x) \cdot (1 - \beta) + p(Y = 0|x) \cdot \alpha$$

↑

flipping labels independent of x



Noisy Class Labels: Model

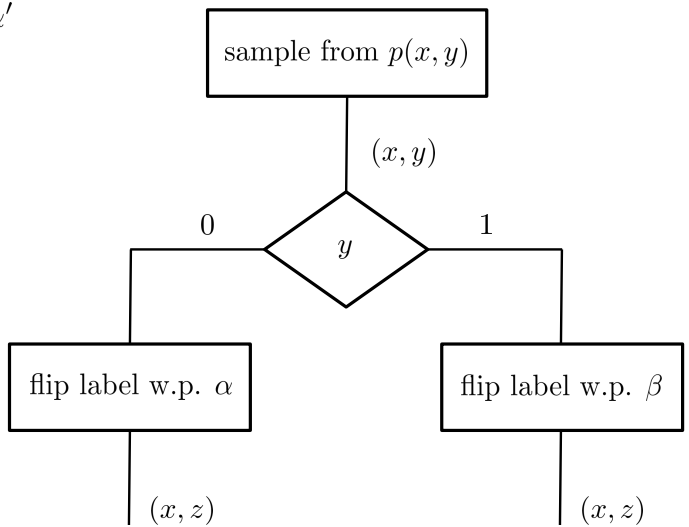
Class-conditional distributions:

$$p(x|Z=1) = p(x|Y=1)\cdot(1-\beta') + p(x|Y=0)\cdot\beta'$$
$$p(x|Z=0) = p(x|Y=0)\cdot(1-\alpha') + p(x|Y=1)\cdot\alpha'$$

where

$$1-\beta' = \frac{p_Y(1)}{p_Z(1)}(1-\beta)$$

$$1-\alpha' = \frac{p_Y(0)}{p_Z(0)}(1-\alpha)$$



Example: $\alpha = 0.05$, $\beta = 0.15$, $p_Y(1) = \frac{1}{4}$

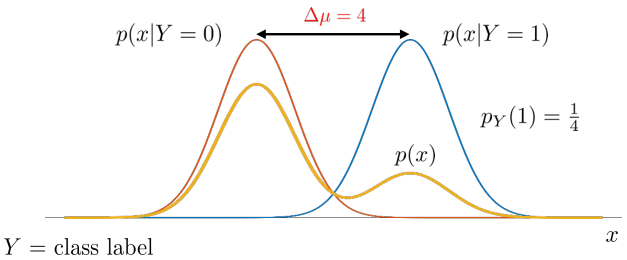
$$p_Z(1) = \tfrac{1}{4}(1-\beta) + \tfrac{3}{4}\alpha = \tfrac{1}{4}$$

$$\alpha' = \alpha,\ \beta' = \beta$$

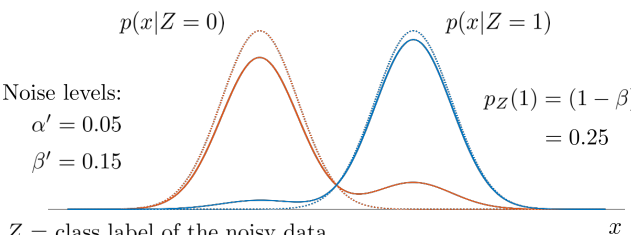
This was a coincidence, because of choices for α, β, $p_Y(1)$

Noisy Data: Impact on 2D Measures

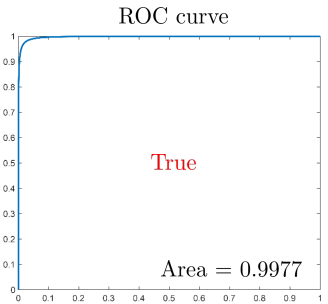
- class-conditional distributions are unit-variance Gaussians with $\Delta\mu = 4$; predictor based on a threshold.
- clean labeled data vs. noisy labeled data

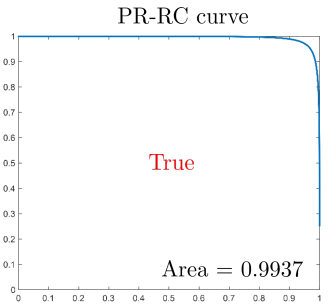


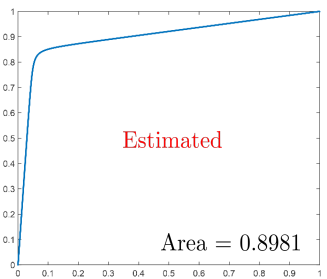
Y = class label

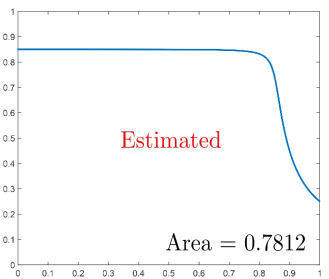


Z = class label of the noisy data









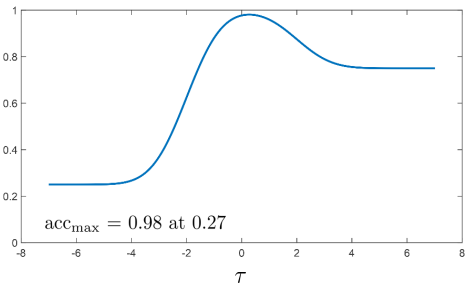
Noisy Data: Impact on 1D Measures

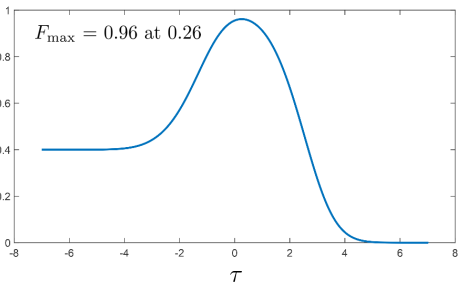
- class-conditional Gaussians, $\Delta\mu = 4$, $p_Y(1) = \frac{1}{4}$, $\alpha = 0.05$, $\beta = 0.15$

Accuracy

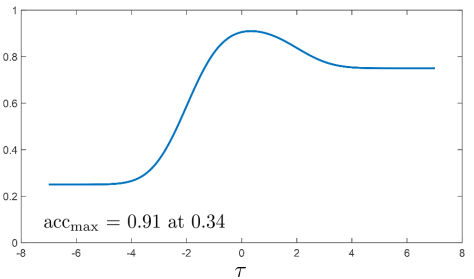
F-measure

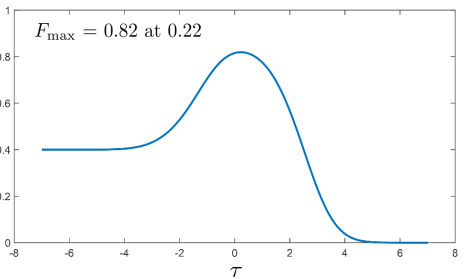
no noise
(true perf.)





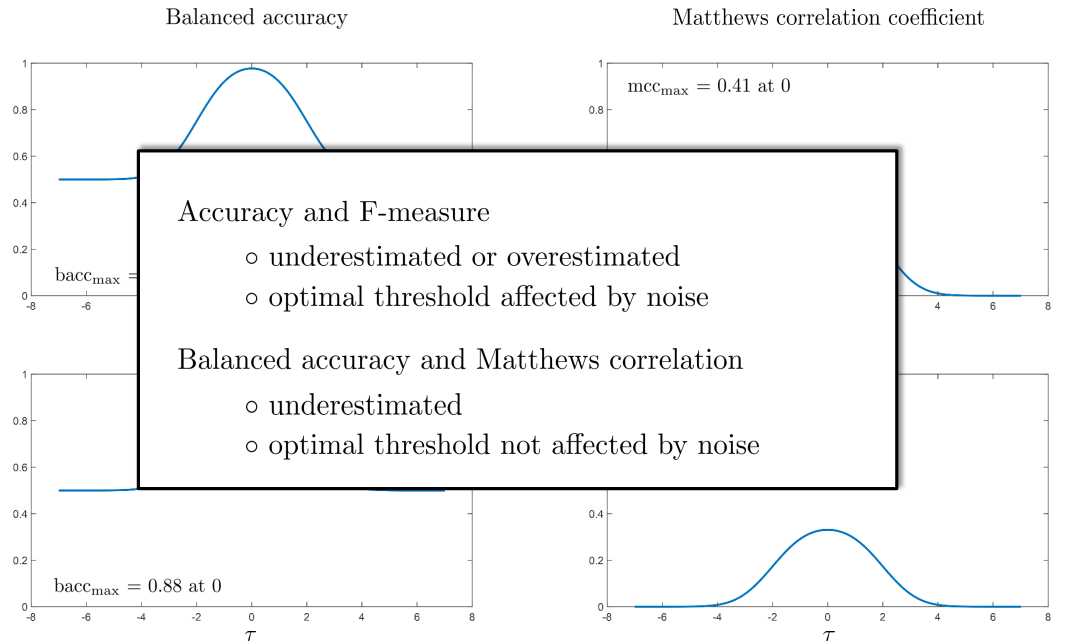
noise
(measured perf.)





Noisy Data: Impact on 1D Measures

- class-conditional Gaussians, $\Delta\mu = 4$, $p_Y(1) = \frac{1}{4}$, $\alpha = 0.05$, $\beta = 0.15$



Accuracy and F-measure

- underestimated or overestimated
- optimal threshold affected by noise

Balanced accuracy and Matthews correlation

- underestimated
- optimal threshold not affected by noise

Multi-Class Classification

Multidimensional output:

- $(\boldsymbol{f}(\boldsymbol{X}), \boldsymbol{Y})$ = pair of m-dimensional predictions and outputs
- $\forall\ j \in \{1, 2, ..., m\}$ and $\tau \in \mathbb{R}$ there is a confusion matrix $C_{j,\tau}$

True class \ Predicted class	positive	negative
positive	tp	fn
negative	fp	tn

Binary classification:

$$\text{accuracy} = \frac{\text{tp}+\text{tn}}{n}$$

Macro averaging:

$$\text{accuracy}_{\text{M}} = \frac{1}{m}\sum_{j=1}^{m} \text{accuracy}_j$$

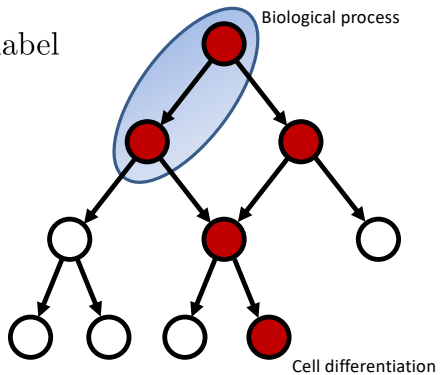
Micro averaging:

$$\text{accuracy}_{\mu} = \frac{1}{mn}\left(\sum_{j=1}^{m} \text{tp}_j + \sum_{j=1}^{m} \text{tn}_j\right)$$

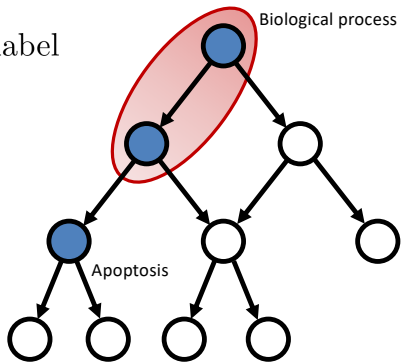
Evaluation of Structured-Output Prediction

Consistent sub-graph prediction:

Predicted label



True label



Precision:

$$\text{pr} = \frac{\#\text{true positives}}{\#\text{positive predictions}} = \frac{2}{5}$$

Recall:

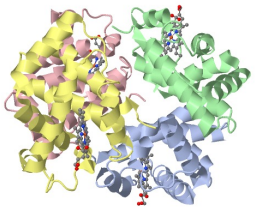
$$\text{rc} = \frac{\#\text{true positives}}{\#\text{positive terms}} = \frac{2}{3}$$

Predicting Impact of Disease Variants

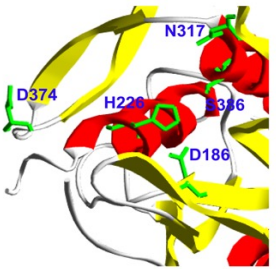
Problem: develop/evaluate a pathogenicity predictor of human loss-of-function (LOF) protein variants

Background: Proteins are important molecules that do things in our bodies

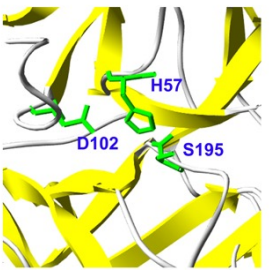
Hemoglobin →



Known disease variants →



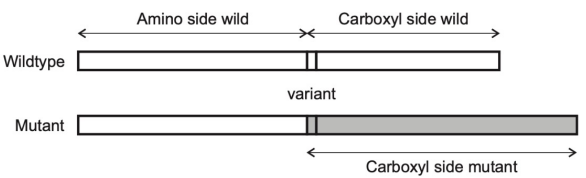
D374Y in PCSK9



H57R in F9

LOF variants: disrupt chunks of proteins

Average healthy human has 50+



Pagel *et al.*, Bioinformatics, 2017.

Predicting Impact of Disease Variants

Data:

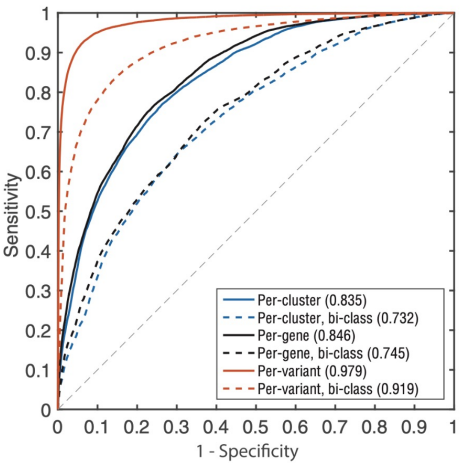
Table 1. Number of variants (proteins) present in each data set

	Disease	Neutral	Total
Frameshift	18 116 (1545)	90 135 (13 427)	108 251 (13 713)
Stop gain	14 318 (1681)	7960 (4990)	22 278 (6137)
Total	32 434 (1995)	98 095 (13 605)	

Issues:

- there are 20,000 genes in the human genome
- these genes are related (gene families)
- some genes do not have known disease mutations
- genes have different lengths and numbers of variants

How should we evaluate the performance?



Pagel *et al.*, Bioinformatics, 2017.

STATISTICAL APPROACH

Given: Test data: $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, drawn i.i.d. from $p(x, y)$

Trained model: $f(x) \in \mathcal{Y}$

e_T = true error of $f(x)$
e_S = sample error of $f(x)$ on $\mathcal{D}$
X = number of errors in a data set of size n
X_i = number of errors on the i-th example

$$P(X = r) = \binom{n}{r} e_T^r (1 - e_T)^{n-r}$$

$$\mathbb{E}[X] = n\, e_T$$
$$\mathbb{V}[X] = n\, e_T\, (1 - e_T)$$

e_S can be thought of as random variable

$$\mathbb{E}[e_S] = \mathbb{E}\left[\tfrac{1}{n} \textstyle\sum_{i=1}^n X_i\right] = \tfrac{1}{n} \mathbb{E}\left[\textstyle\sum_{i=1}^n X_i\right] = \tfrac{n\, e_T}{n} = e_T$$

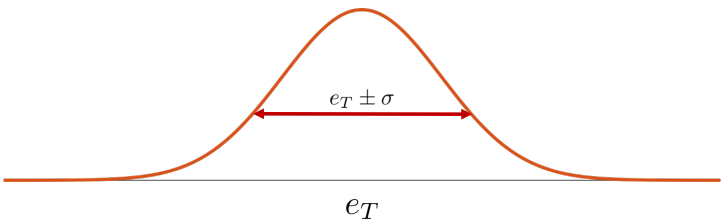
$$\mathbb{V}[e_S] = \mathbb{V}\left[\tfrac{1}{n} \textstyle\sum_{i=1}^n X_i\right] = \tfrac{1}{n^2} n\, e_T\, (1 - e_T) = \tfrac{e_T(1-e_T)}{n}$$

$$\tfrac{e_T(1-e_T)}{n} \approx \tfrac{e_S(1-e_S)}{n}$$

Statistical Approach

Let's approximate binomial distribution by a Gaussian

When $n\,e_T\,(1-e_T) \geq 5$ we have $X \sim \mathcal{N}(n\,e_T, n\,e_T\,(1-e_T))$



If e_T is known, e_S is within $e_T \pm z\sigma$. This also means, that

$$e_T = e_S \pm z \cdot \sqrt{\frac{e_S(1-e_S)}{n}}$$

$z = 1$: 68% confidence interval; $z = 1.96$: 95% confidence interval

Comparing Two Models

Given:

- two models, f_1 and f_2, evaluated on two different samples, $\mathcal{S}_1$ and $\mathcal{S}_2$
- available: e_{S_1} and e_{S_2}, say $e_{S_1} < e_{S_2}$
- not available: e_{T_1} and e_{T_2}

$$d = e_{T_2} - e_{T_1}$$

$$\hat{d} = e_{S_2} - e_{S_1}$$

$$\sigma^2_{\hat{d}} \approx \frac{e_{S_1}(1-e_{S_1})}{n_1} + \frac{e_{S_2}(1-e_{S_2})}{n_2}$$

← independent models

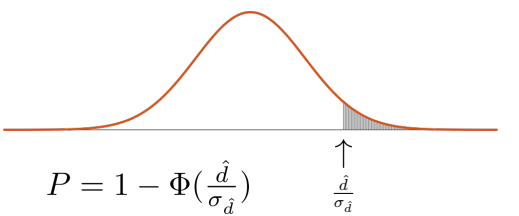
Hypothesis testing (one sided):

H_0: $d = 0$ (the algorithms are equal)

H_1: $d > 0$ (f_1 is better than f_2)

$$\frac{\hat{d}-d}{\sigma_d} : \mathcal{N}(0,1) \quad \rightarrow$$

P-value: area under the tail



$$P = 1 - \Phi(\frac{\hat{d}}{\sigma_{\hat{d}}})$$

BOOTSTRAPPING

- we have an estimator $\hat{\theta}$ of parameter θ
- a general nonparametric procedure for evaluating dispersion of $\hat{\theta}$

Given: data set $\mathcal{D} = \{x_i\}_{i=1}^n$, $x \in \mathcal{X}$ and some estimator $\hat{\theta} = g(\mathcal{D})$

Procedure:

For $b = 1$ to B

- draw sample $\mathcal{D}_b^* = \{x_i^*\}_{i=1}^n$ w/ replacement from $\mathcal{D}$
- compute $g(\mathcal{D}_b^*)$

end

$g^*(\mathcal{D}) = \frac{1}{B} \sum_{b=1}^B g(\mathcal{D}_b^*)$

$\hat{\sigma}^2 = \frac{1}{B-1} \sum_{b=1}^B (g(\mathcal{D}_b^*) - g^*(\mathcal{D}))^2$ ← standard error (SE) estimated as $\sqrt{\hat{\sigma}^2}$

Examples:

$g(\mathcal{D})$ = sample mean

$g(\mathcal{D})$ = AUC when $\mathcal{D}$ are prediction scores and true classes

Efron & Tibshirani. *Stat. Sci.* 1986.

IDEA BEHIND BOOTSTRAPPING AND WHY IT WORKS (WHEN IT DOES)

- consider n realizations $x_1, x_2, \ldots, x_n$ drawn from $p(x)$
- we don't know $p(x)$ but we have its empirical version $\hat{p}(x)$ with mass $\frac{1}{n}$ on every x_i
- we will sample m points $x_1^*, x_2^*, \ldots, x_m^*$ from $\hat{p}(x)$ to estimate μ_n

  each draw $b \in \{1, 2, ..., B\}$ of m points has mean $\mu_m^*(b)$

Sample mean: $\mu_n = \frac{1}{n} \sum_{i=1}^{n} x_i$ **Population mean:** $\mu = \mathbb{E}[X]$

Sample variance: $s_n = \frac{1}{n^2} \sum_{i=1}^{n} (x_i - \mu_n)^2$ **Population variance:** $\sigma^2 = \mathbb{V}[X]$

Sample mean μ_n is an estimate of μ, but μ_m^* is an estimate of μ_n

$$Q_n = \sqrt{n}(\mu_n - \mu)/s_n \quad \longrightarrow \quad \mathcal{N}(0, 1) \qquad \leftarrow \text{central limit theorem}$$

Theorem:

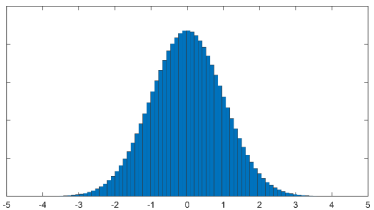
$$Q_m^* = \sqrt{m}(\mu_m^* - \mu_n) \quad \longrightarrow \quad \mathcal{N}(0, \sigma^2)$$
$$s_m^* \quad \longrightarrow \quad \sigma$$

Bickel & Freedman. *Ann. Stat.* 1981.

BOOTSTRAPPING

- standard error: true θ is within $\hat{\theta} \pm \text{SE}$ w.p. 68%

$X \sim \mathcal{N}(0,1)$



← normal distribution

- standard error $= \frac{\hat{\sigma}}{\sqrt{n}}$

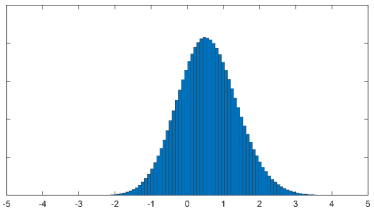
Example:

$n = 100$

$\mu = \hat{\mu} \pm 1.96 \cdot \text{SE}$

↑ 95% c.i.

$X \sim \max\{\mathcal{N}(0,1), \mathcal{N}(0,1)\}$



← skew normal distribution

- asymetric $100 \cdot \alpha$ confidence intervals determined by thresholds separating $\frac{1-\alpha}{2}$ smallest and $\frac{1-\alpha}{2}$ largest scores

Example:

$n = 100$

$\mu \in (-1.30, 2.00)$

↑ 95% c.i.

Comparing Learning Algorithms

Model: $f : \mathcal{X} \to \mathcal{Y}$

Algorithm: $a : (\mathcal{X} \times \mathcal{Y})^n \to \mathcal{F}$

	a_1	a_2
$\mathcal{D}_1$	1	0
$\mathcal{D}_2$	1	0
$\vdots$		
$\mathcal{D}_K$	0	1

- K datasets
- score wins (1s) vs. losses (0s)
- a_1 scored k wins

Hypothesis testing (one sided):

H_0: a_1 and a_2 are equal

H_1: a_1 is better than a_2

$$\rightarrow \qquad P = \sum_{j=k}^{K} \binom{K}{j} \alpha^j (1-\alpha)^{K-j} \qquad \alpha = \tfrac{1}{2}$$

- probability that k or more wins would be observed by chance (under H_0)

Concluding Remarks

Empirical evaluation:

- Core area of machine learning
- Science and art

Addtional issues:

- Reliability: extensive auditing to provide uncertainty guarantees that lead to actionable evidence
- Interpretability: models must be able to point out the factors that influence their decisions
- Fairness: models must be fair and must not cause unintended harm to users upon deployment