Classification Metrics and Loss Functions

Old Dominion Vision Lab Seminar

Alex Glandon

2025

Overview

- Statistical Point Estimation [1]
- Statistical Hypothesis Testing [3]
- Classification Models
  - Relationship to Point Estimation
  - Relationship to Hypothesis Testing
- Classification Metrics for Example Applications
- Classification Metrics vs Loss Functions
  - Imbalanced Data [4]

Notation

- Lower case variable -> sample
- Upper case variable -> random variable
- Examples we will see later
  - $\hat{Y}_{train} \sim f_{\hat{y}_{train}}(\hat{y}_{train} | x_{train}, \hat{\theta})$, the output of a neural network is a random variable with discrete support (number of possible classes) and it is defined by a conditional distribution given the data sample x_{train} and neural network weights $\hat{\theta}$
  - y_{train}, the target of a training sample is a deterministic scalar, an observation of the random variable Y (a marginal of our dataset)
  - The idea is predictions are distributions over labels, targets are samples that identify a deterministic label
- An estimator is a function of a random variable (the generative distribution) and therefore is also a random variable
- An estimate is a function of a particular sample (which is deterministic) and therefore is also deterministic
- [1,2]

Statistical Point Estimation

- Given a generative distribution $f_x(x|\theta)$ (e.g. Normal, Poisson)
- Design an estimator for $\hat{\theta}$ given the sample data
- $D_{sample} = (X_1, X_2, \dots, X_n)$
- D_{sample} (the dataset) is a set of n observations of the random variable X
- Called "point estimation" because for a given measurement D_{sample}, $\hat{\theta}$ is a deterministic function of the dataset (not a distribution with a spread)

Statistical Point Estimation

- In the general formulation we may not be concerned with estimating θ, we may only want to know a function of the parameter $g(\theta)$ – we see this later for neural networks
- Suppose we know the loss that is incurred given an estimate $\hat{g}$ of g is given by $l(g, \hat{g})$
- The problem is to minimize the expected loss $l(g, \hat{g})$ given that the dataset D is a random vector (of random variables $X_1, X_2, \ldots, X_n$)
- We wish to minimize risk, meaning to find an estimator $\hat{g}(D)$ that minimizes the risk given as $\mathrm{E}[l(g, \hat{g})]$
- "As stated, this problem has no solution" [1]
  - For each ground truth θ, $argmin_{\hat{\theta}}\ \mathrm{E}[l(\theta, \hat{\theta}(D))]$ has a different solution

Constraining the Point Estimation Problem

- To solve the point estimation problem of designing an estimator to minimize risk, we must limit our problem to a constrained space of estimators [1]
- Examples include:
  - Unbiasedness $\mathrm{E}\left[\hat{\theta}\right] = \theta$
  - For example, sample mean is unbiased because $\mathrm{E}\left[\frac{1}{n}\sum_{i=1}^{n} x_i\right] = \frac{1}{n}\sum_{i=1}^{n} \mathrm{E}[x_i] = \frac{1}{n}\sum_{i=1}^{n} \mu_x = \frac{1}{n} \cdot n \cdot \mu_x = \mu_x$
  - Equivariance
    - $\mathrm{E}\left[l\left(\theta - x, \hat{\theta} - x\right)\right] = \mathrm{E}\left[l(\theta, \hat{\theta})\right]$ used in location families, for example estimating mean of a normal distribution
    - $\mathrm{E}\left[l\left(c\theta, c\hat{\theta}\right)\right] = \mathrm{E}\left[l(\theta, \hat{\theta})\right]$ used in scale families, for example estimating the variance of a normal distribution
  - Assume a prior over θ (Bayesian formulation)
    - Find the estimator $\hat{\theta}$ that minimize $\mathrm{E}\left[l(\theta, \hat{\theta})\right]$ where $\theta \sim f_\theta(\theta)$ is a prior distribution of parameters
  - Minimax
    - Find the estimator $\hat{\theta}$ that minimizes the worst-case risk for possible parameter values θ given our particular loss function
    - $inf_{\hat{\theta}}\, sup_{\theta}\, E[l(\theta, \hat{\theta})]$

Example of Constrained Point Estimation

- Consider drawing a ball from an urn, where there are n balls labeled with the numbers $1,2, \dots, n$
- Suppose we take a ball from the urn, read the number on the ball, and then want to estimate the number of balls n in the urn
- If k is the number on the ball we draw, then the random variable K is represented by a distribution parameterized by n
- K is a categorical distribution (maximum entropy)
- See previous seminar for categorical distribution as neural network classifier
- If we take m samples, then we have a multinomial distribution

Bayesian solution for urn size

- Consider a prior distribution of the number balls in the urn of the form
- $P(N = n) = (1 - p)^{n-1}p, \mathrm{n} = 1,2,3, \ldots \quad (0 < p \leq 1)$
- This is a geometric distribution and is useful because we want a prior where the likelihood of *N*=*n* decays as n becomes larger (in other words we believe our urn becomes less likely as the number of balls approaches extremely large numbers)

Bayesian solution for urn size (see related MAP notes from previous lab seminar)

- Bayesian MAP solution
- $P(N = n|k) = P(K = k|n)P(N = n)/P(K = k)$

- $P(N = n|k) \propto P(K = k|n)P(N = n)$
- $P(K = k|n) = \begin{cases} \frac{1}{n}, & k \in 1,2,\dots,n \\ 0, & otherwise \end{cases}$
- $P(N = n) = (1-p)^{n-1}p,\ n \geq 1$
- MAP solution, find $argmax_n \frac{1}{n}(1-p)^{n-1}p$
- nonzero where $k \leq n$

Bayesian solution for urn size (see related notes from my previous lab seminar)

- $argmax_n \frac{1}{n}(1-p)^{n-1}p =$
- $argmax_n \log\left(\frac{1}{n}(1-p)^{n-1}p\right) =$
- $argmax_n log\left(\frac{1}{n}\right) + (n-1)\log(1-p) + \log(p) =$
- $argmax_n log\left(\frac{1}{n}\right) + (n-1)\log(1-p)$
- Consider that $log\left(\frac{1}{n}\right) + (n-1)\log(1-p)$ decreases as n increases
- So we are looking for the smallest possible n to maximize $\mathrm{P}(N = n|k)$
- The domain of n given by the prior geometric distribution is $n \in 1,2,3, \ldots$, but $\mathrm{P}(N = n|k)$ is only nonzero where $\mathrm{n} \geq \mathrm{k}$, therefore:
- $argmax_n log\left(\frac{1}{n}\right) + (n-1)\log(1-p) = k$

Bayesian solution is biased

- Our estimator is $\widehat{N}(K) = K$
- $P(K = k|n) = \begin{cases} \frac{1}{n}, & k \in 1,2, \dots, n \\ 0, & otherwise \end{cases}$
- $\mathrm{E}\left[\widehat{N}\right] = \mathrm{E}[K] = \sum_{k=1}^{N} k \cdot P(K = k|N) =$
- $\mathrm{E}\left[\widehat{N}\right] = \sum_{k=1}^{N} \frac{k}{N} = \frac{1}{\mathrm{N}} \sum_{k=1}^{N} k = \frac{1}{\mathrm{N}} \frac{N(N+1)}{2} = \frac{N+1}{2}$
- $\mathrm{E}\left[\widehat{N}\right] = \frac{\mathrm{N}+1}{2}$
- Biased: our Bayesian estimate of the number of balls in the urn is expected to be an underestimate. If there are 11 balls in the urn, our estimator is expected to on average estimate that there are 6 balls.

Example Unbiased Solution

- Let $\widehat{N} = \alpha K + \beta$
- We want $\mathrm{E}\left[\widehat{N}(K)\right] = N$
- $\mathrm{E}\left[\widehat{N}(K)\right] = \mathrm{E}[\alpha K + \beta] = \alpha \mathrm{E}[K] + \beta = \alpha \frac{N+1}{2} + \beta$
- For our estimator to be unbiased, let $\alpha = 2, \beta = -1$
- Then $\mathrm{E}\left[\widehat{N}(K)\right] = \alpha \frac{N+1}{2} + \beta = N$
- We have an unbiased estimator $\widehat{N}(K) = 2K - 1$
- If there are 11 balls in the urn, our estimator is expected to on average estimate that there are 11 balls
- The tradeoff is that our estimator can overestimate for a given sample. Bayesian $\widehat{N}(K) = K$ cannot overestimate since we know *N* is at least *K*

UMVU Estimators

- Uniform Minimum Variance Unbiased Estimators
- A family of unbiased solutions exists [1]
- $\widehat{N}(K) = 2K - 1 - U(K)$, where $\mathrm{E}[U(K)] = 0$
- In our example, minimum variance estimator depends on n
- Goal is to minimize risk
- $argmin_U \mathrm{E}\left[l\left(\widehat{N}(K) - U(K), n\right)\right] = argmin_U \mathrm{E}\left[\left(\left(\widehat{N}(K) - U(K)\right) - n\right)^2\right] =$
- $argmin_U \mathrm{bias}\left[\widehat{N}(K) - U(K)\right] + \mathrm{var}\left[\widehat{N}(K) - U(K)\right]$ (see bias-variance decomposition of mean squared error loss)
- $argmin_U \mathrm{var}\left[\widehat{N}(K) - U(K)\right]$ (because bias is 0)
- $argmin_U \mathrm{E}\left[\left(\widehat{N}(K) - U(K)\right)^2\right] - \mathrm{E}\left[\widehat{N}(K) - U(K)\right]^2 =$
- $argmin_U \mathrm{E}\left[\left(\widehat{N}(K) - U(K)\right)^2\right] - \mathrm{E}[\widehat{N}(K)]$ (because $\mathrm{E}[U(K)] = 0$)
- $argmin_U \mathrm{E}\left[\left(\widehat{N}(K) - U(K)\right)^2\right] - n$ (because $\widehat{N}(K)$ is unbiased)
- $argmin_U \mathrm{E}\left[\left(\widehat{N}(K) - U(K)\right)^2\right]$ (because $argmin_U$ is independent of isolated n term)

UMVU Estimators

- For example n=1
- $argmin_U \, \mathrm{E}[\left(2K - 1 - U(K)\right)^2] =$
- $argmin_U \sum_{k=1}^{n} P(K = k)(2K - 1 - U(k))^2 = argmin_U \, 1 \cdot \left(1 - U(1)\right)^2$
- The condition $\mathrm{E}[U(K)] = 0$ means $\sum_{k=1}^{n} P(K = k)U(k) = 1 \cdot U(1) = 0$, implies U(1)=0
- The minimum variance estimator is $2K - 1 - \mathrm{U(K)} = 2\mathrm{K} - 1$
- If *n*=1, $\widehat{N}(K) = 2K - 1$ is minimum variance

UMVU Estimators

- For example n=2
- $argmin_U \, \mathrm{E}[\big(2K-1-U(K)\big)^2] =$
- $argmin_U \sum_{k=1}^{n} P(K=k)(2K-1-U(k))^2 \; =$
- $argmin_U \frac{1}{2}\big(1-U(1)\big)^2 + \frac{1}{2}\big(3-U(2)\big)^2 =$
- The condition $\mathrm{E}[U(K)] = 0$ means $\sum_{k=1}^{n} P(K=k)U(k) = \frac{1}{2}U(1) + \frac{1}{2}U(2) = 0$
- Using the condition $U(1) = -U(2)$
- $argmin_U \frac{1}{2}\big(1+U(2)\big)^2 + \frac{1}{2}\big(3-U(2)\big)^2 =$
- Using derivative $(1+\mathrm{U}(2)) - (3-U(2)) = 0$
- Meaning $U(2) = 1, U(1) = -1$
- If *n*=2, $\widehat{N} = 2K - 1 - U(K)$ is minimum variance if $U(1) = -1, U(2) = 1$

UMVU Estimators

- Because the minimum variance estimator depends on the ground truth parameter n
- *n*=1 -> $\widehat{N}(K) = 2K - 1$ is minimum variance
- *n*=2 -> $\widehat{N}(K) = 2K - 1 - U(K)$ is minimum variance
  - $U(1) = 1, U(2) = -1$
- Therefore, no uniform minimum variance unbiased estimator exists

Summary

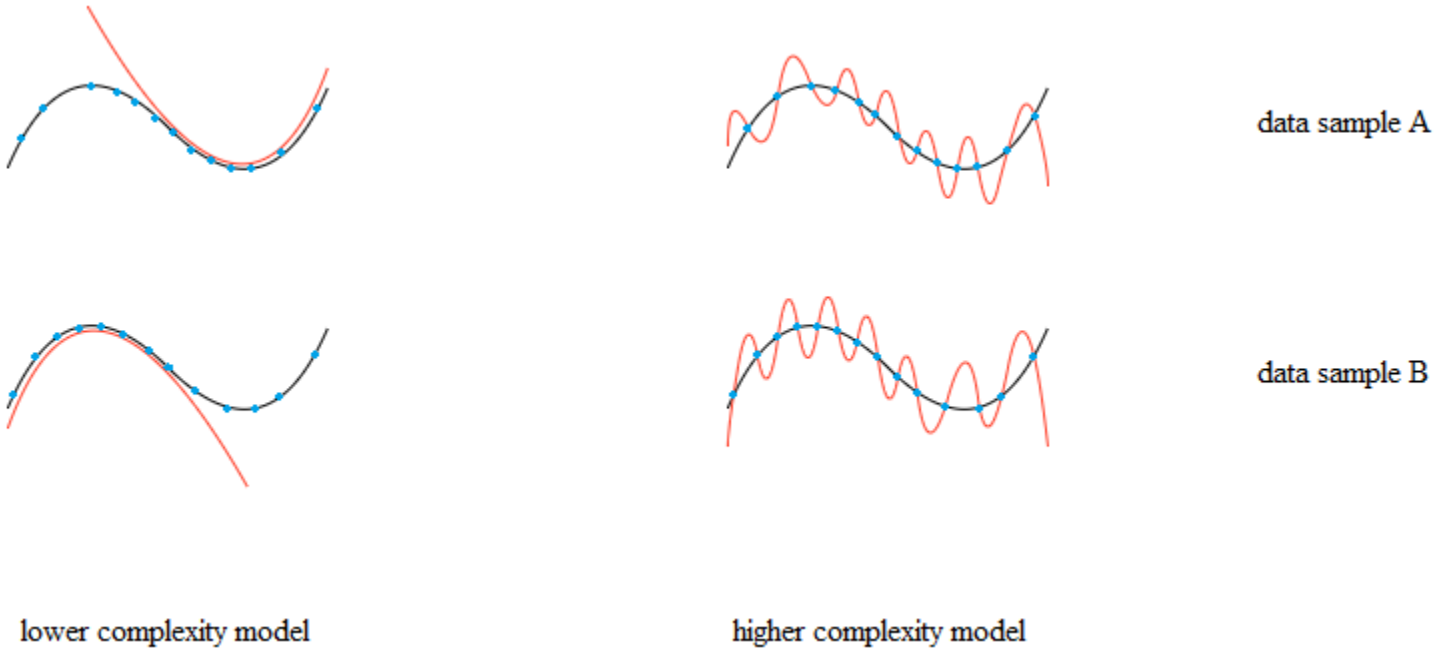
- Picking a ball out of an urn with numbered balls
- Estimating the number of balls in the urn has no best solution and we must consider tradeoffs
- Bayes is biased
- If we find any unbiased estimator, it will have high variance for certain values of n (because no UMVU exists)

Prediction Models
-Bias and Variance

- Point Estimation bias and variance
  - Makes sense for parameter estimation where loss is squared error $\left(\theta - \hat{\theta}\right)^2$
  - Square error loss can be decomposed into bias and variance
- Prediction models bias and variance
  - Good for regression problems again where squared error loss can be decomposed into bias and variance $(y - \hat{y})^2$

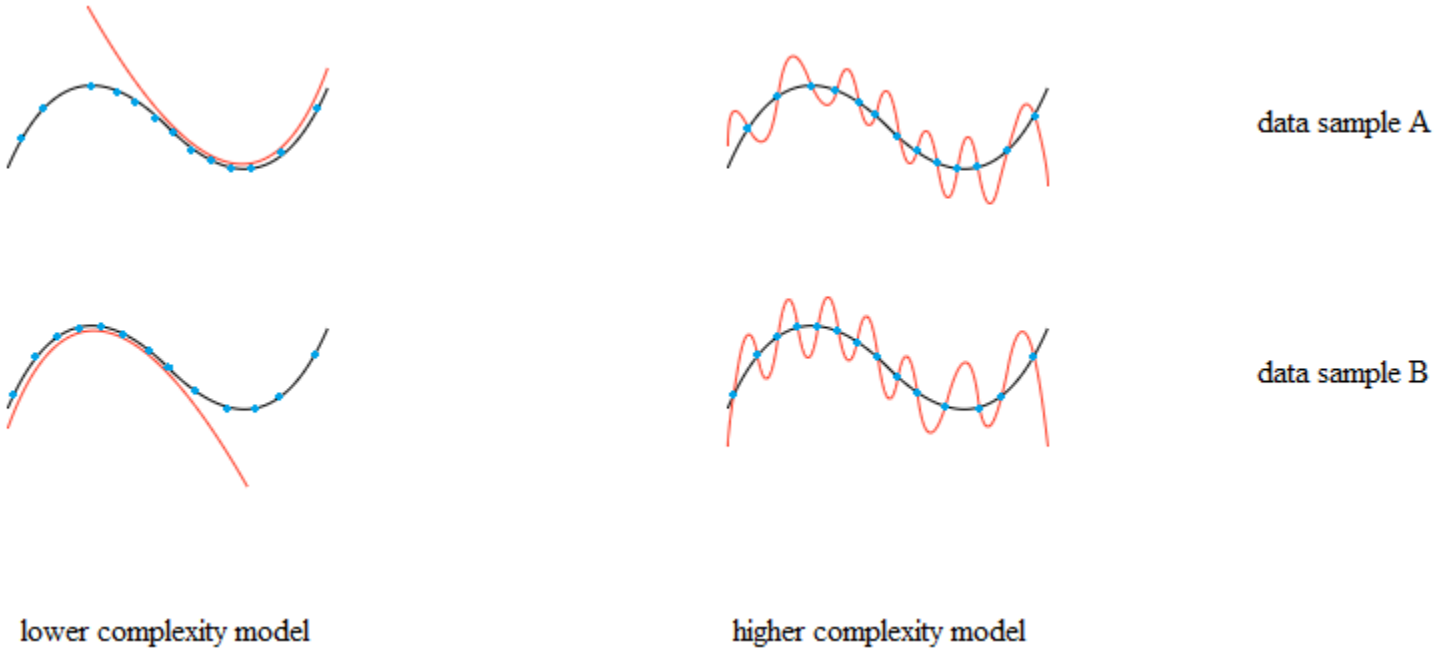
Prediction Models
-Bias and Variance

- Overfitting and Underfitting
- If loss is squared error, a model that is overfit or underfit will have loss with both bias and variance



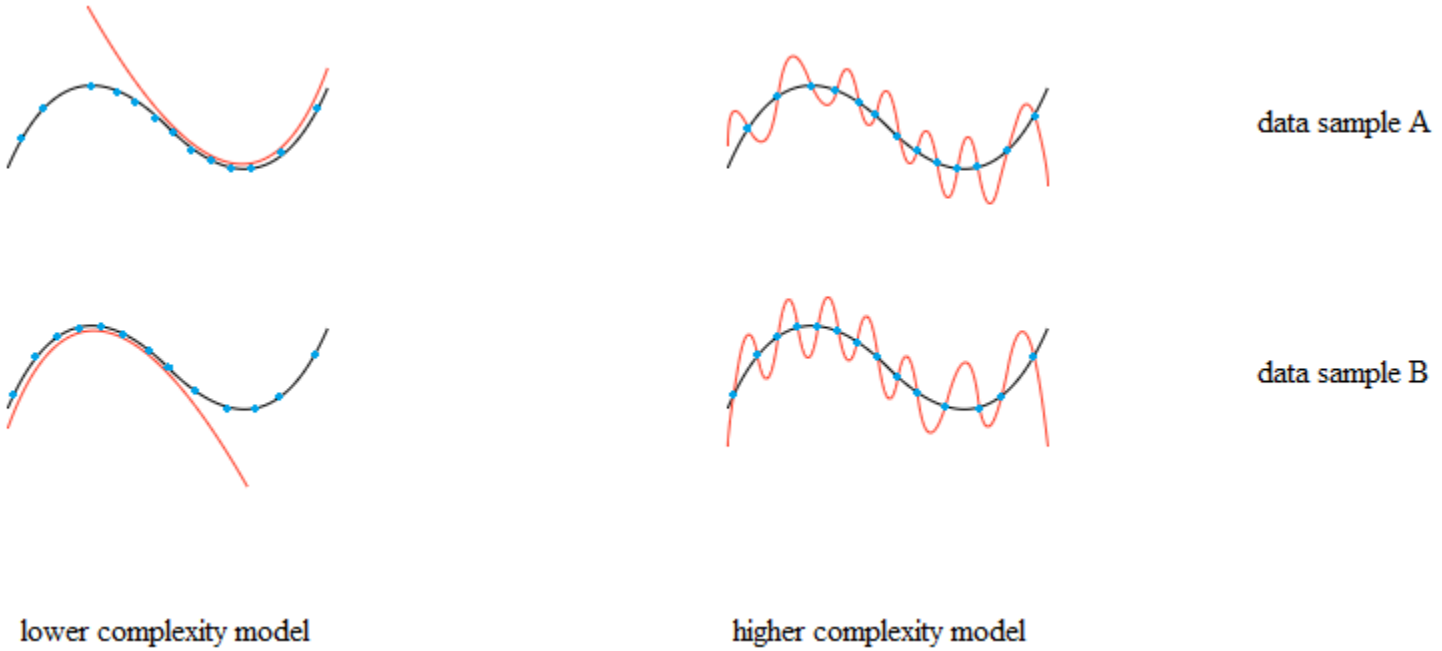
Prediction Models
-Bias and Variance

- Underfitting example on left
  - High bias because half of the curve is wrong half of the time
  - High variance because there are two modes



Prediction Models
-Bias and Variance

- Overfitting example on left
  - High bias because most of the curve is high error between the points
  - High variance because the curve changes based on the sample



Statistical Hypothesis Testing [3]

- Again, assume a generative distribution $f_x(x|\mu)$ parameterized by μ (we called this θ in point estimation)
- Again, the data is a collection of n samples
- $D_{sample} = (X_1, X_2, \ldots, X_n)$
- The problem is to determine if a hypothesis H_0 is accepted (the null hypothesis)
- When we reject H_0, we say that the alternative hypothesis H_1 is accepted

Statistical Hypothesis Testing

- Suppose we want to test null $H_0: \mu = \mu_0$ vs $H_1: \mu < \mu_0$
- An example solution is as follows
- Define a significance α, for example 5%
- If the null hypothesis is true, the probability of finding a sample mean at $\bar{x}$ decreases as we consider values smaller than μ
- Find a constant (called a critical value) such that $P(\bar{X} < \mu_0 - c | H_0) = \alpha$
- This means, that if the null hypothesis is true, the probability of getting a sample mean c or more to the left of μ_0 is α

Statistical Hypothesis Testing

- P($\bar{X} < \mu_0 - c | H_0) = \alpha$
- For Z test, the above is rewritten as $P(Z < z_{test} | H_0) = \alpha$, or $P\left(\frac{\bar{X} - \mu_0}{\sigma/\sqrt{n}} < z_{test} \middle| H_0\right) = \alpha$
- To perform the test, we observe a sample, and calculate the sample mean $\bar{x}$ (or relevant test statistic z).
- If $\bar{x} < \mu_0 - c$ or equivalently $z < z_{test}$, we conclude that H_0 is not likely (we reject H_0)
- This is because there is only a 5% chance to find a sample $z < z_{test}$ when H_0 is true

Statistical Hypothesis Testing

- Suppose $H_0: \mu = \mu_0$ vs $H_1: \mu = \mu_0 - 0.0000001$
- Then even if our test says H_0 is likely, we may also believe H_1 is likely
- How do we solve this?

Statistical Hypothesis Testing

- Suppose $H_0: \mu = \mu_0$ vs $H_1: \mu_1 = \mu_0 - 0.0000001$
- We need to create a test of sufficient power
- Power is $P(\text{accepting } H_1 | H_1 \text{is true})$
- To increase power, we need:
  - Larger sample size
  - Relatively smaller variance
  - Relatively larger difference between μ_0 and μ_1
- With large enough sample size, $P(\text{accepting } H_1 | H_1 \text{is true})$ becomes reasonable (even in our example of $\mu_1 = \mu_0 - 0.0000001$)

Classification Models
-Relationship to Point Estimation

- Point Estimation has no general solution
  - Risk Minimization has no general solution
  - For each ground truth θ, $argmin_{\hat{\theta}}\ \mathrm{E}[l(\theta, \hat{\theta}(D))]$ has a different solution
- Classification Risk Minimization has no general solution
  - Classification can be constrained to find the minimum risk given the training data
  - Our loss function no longer depends on θ, but on a sample d from the distribution $D \sim f_{X,Y}(x, y|\theta)$
  - We no longer minimize $argmin_{\hat{\theta}}\ \mathrm{E}\left[l\left(\theta, \hat{\theta}(D)\right)\right]$, but only minimize the loss for a given sample d of the distribution of the data D, where $d = x_{train}, y_{train}$
  - $l\left(\theta, \hat{\theta}(d)\right) = l^*(Y_{train}(\theta), \hat{Y}_{train}(\hat{\theta})) =$
  - We are modeling the conditional distribution $\hat{Y}_{train} \sim f_{\hat{y}_{train}}(\hat{y}_{train}|x_{train}, \hat{\theta})$
  - $\hat{Y}_{train}$ is a random variable, which is a vector of probabilities (a predicted probability for each label)
  - There is no algorithm to find the global minimum of the network parameter $\hat{\theta}$ (weights), no best solution can be found for $argmin_{\hat{\theta}}\ l^*(Y_{train}(\theta), \hat{Y}_{train}(\hat{\theta}))$
  - Because we constrained the Risk to a given sample, a solution may exist, called a global minimum
  - Optimization problem complexity prohibits finding a global minimum solution

Classification Models
-Relationship to Hypothesis Testing

- Confusion Matrix interpretation for Hypothesis Testing
- α is defined as the probability of accepting H_1 given that H_0 is true (Type I error)
- Power is defined as probability of accepting H_1 given that H_1 is true
- β is defined as probability of accepting H_0 given that H_1 is true (Type II error)
- Therefore Power = $1 - \beta$

	H_0	H_1
$\widehat{H}_0$	$1 - \alpha$	β
$\widehat{H}_1$	α	$1 - \beta$

Classification Models
-Relationship to Hypothesis Testing

- Confusion Matrix interpretation for Hypothesis Testing and Classification Models
- Confusion matrix in both paradigms represents Prediction on the Left and Ground Truth on the Top
- Can be extended to multiple classes

	p	n
$\hat{p}$	$p\hat{p}$	$p\hat{n}$
$\hat{n}$	$p\hat{n}$	$n\hat{n}$

	H_0	H_1
$\widehat{H}_0$	$1-\alpha$	β
$\widehat{H}_1$	α	$1-\beta$

Classification Metrics for Example Applications - Sensitivity and Specificity

- Like DSP filtering
- When we design a filter, we are trying to be specific, keeping only information in frequencies that we are interested in
- If a signal has information in abnormal regions, our filter may miss that signal (it is not sensitive to signals that appear to be noise)
- Specificity $\mathrm{n} \rightarrow \hat{n}$, if a signal is noise, filter it
- Sensitivity $p \rightarrow \hat{p}$, if a signal has information, detect it

Classification Metrics for Example Applications - ROC

- If we have a set of prediction probabilities, we can focus on specificity or sensitivity by setting a prediction threshold
- If this is DSP, to make a filter sensitive, we make the filter wider band
- Sensitive means we have a low threshold for predicting positive
- $p \to \hat{p}$ more often and $n \to \hat{p}$ more often
- $p \to \hat{p}$ more often means $\frac{p\hat{p}}{p} = \frac{p\hat{p}}{p\hat{p}+p\hat{n}}$, the sensitivity or True Positive Rate is higher
- If this is DSP, to make a filter specific, we make the filter narrower band
- Specific means we have a high threshold for predicting positive
- $p \to \hat{p}$ less often and $n \to \hat{p}$ less often
- $n \to \hat{p}$ less often means $n \to \hat{n}$ more often
- $n \to \hat{n}$ more often means $\frac{n\hat{n}}{n} = \frac{n\hat{n}}{n\hat{p}+n\hat{n}}$, the specificity or True Negative Rate is higher

Classification Metrics for Example Applications - ROC

- ROC Plots a parametric curve TPR(threshold), FPR(threshold) as threshold (the parameter) is varied between a low detection threshold of 0 and a high detection threshold of 1
- $TPR = \frac{p\hat{p}}{p} = \frac{p\hat{p}}{p\hat{p}+p\hat{n}}$ (sensitivity)
- $FPR = \frac{n\hat{p}}{n} = \frac{n\hat{p}}{n\hat{p}+n\hat{n}} = \frac{(n\hat{p}+n\hat{n})-(n\hat{p}+n\hat{n})+n\hat{p}}{n\hat{p}+n\hat{n}} = \frac{(n\hat{p}+n\hat{n})-n\hat{n}}{n\hat{p}+n\hat{n}} =$

$$1 - \frac{n\hat{n}}{n\hat{p}+n\hat{n}} = 1 - TNR$$ (1-specificity)

Classification Metrics for Example Applications - ROC Multiclass

- Applied Optics Paper
  - Selected a single subject vs all other subjects to present TPR and FPR
  - $TPR = \frac{\#(y_{pred}=subject)}{\#(y_{target}=subject)}$
  - $FPR = 1 - \frac{\#(y_{pred}\neq subject)}{\#(y_{target}\neq subject)}$

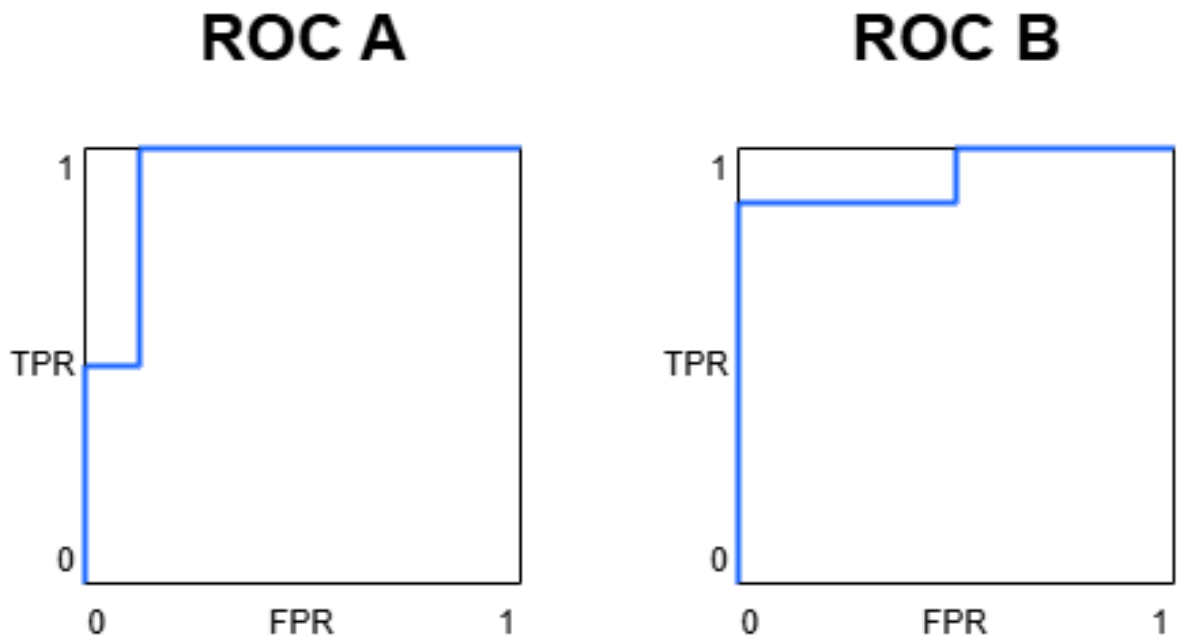
- Similar to Cumulative Match Characteristic Curve (CMC Curve)
- In CMC, select a single subject, and find top-1,top-2,… accuracies

Classification Metrics for Example Applications - ROC Security Application

- Security: Sensitivity is a Must
- If a subject is a bad actor, we must detect the subject (high sensitivity)

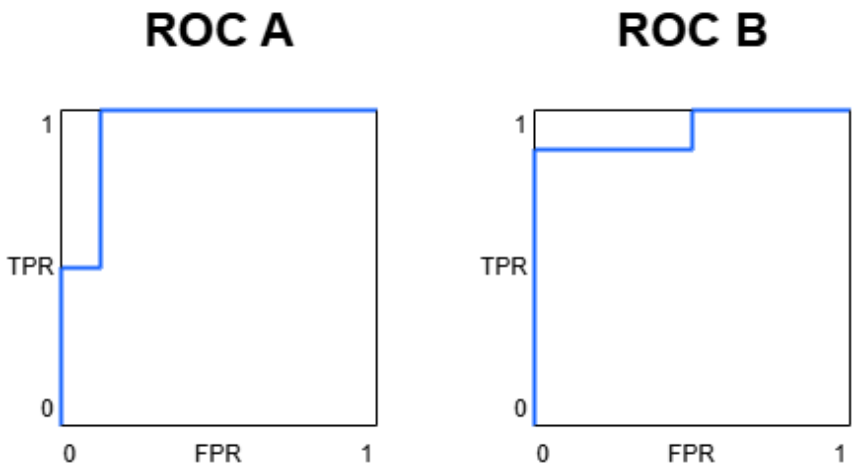
Classification Metrics for Example Applications - ROC Multiclass

- These example ROC plots have the same area under curve (AUC)
- Which ROC is preferred for security or high sensitivity demand?



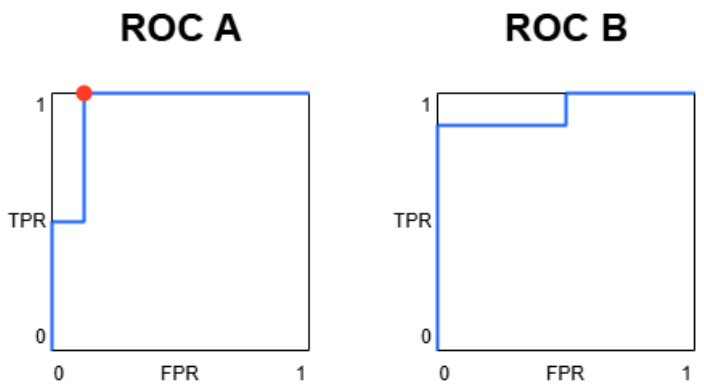
Classification Metrics for Example Applications - ROC Multiclass

- ROC A:
  - TPR 50%, FPR 0% -> sensitivity 50%, specificity 100%
  - TPR 100%, FPR 12.5% -> sensitivity 100%, specificity 87.5%
- ROC B:
  - TPR 87.5%, FPR 0% -> sensitivity 87.5%, specificity 100%
  - TPR 100%, FPR 50% -> sensitivity 100%, specificity 50%

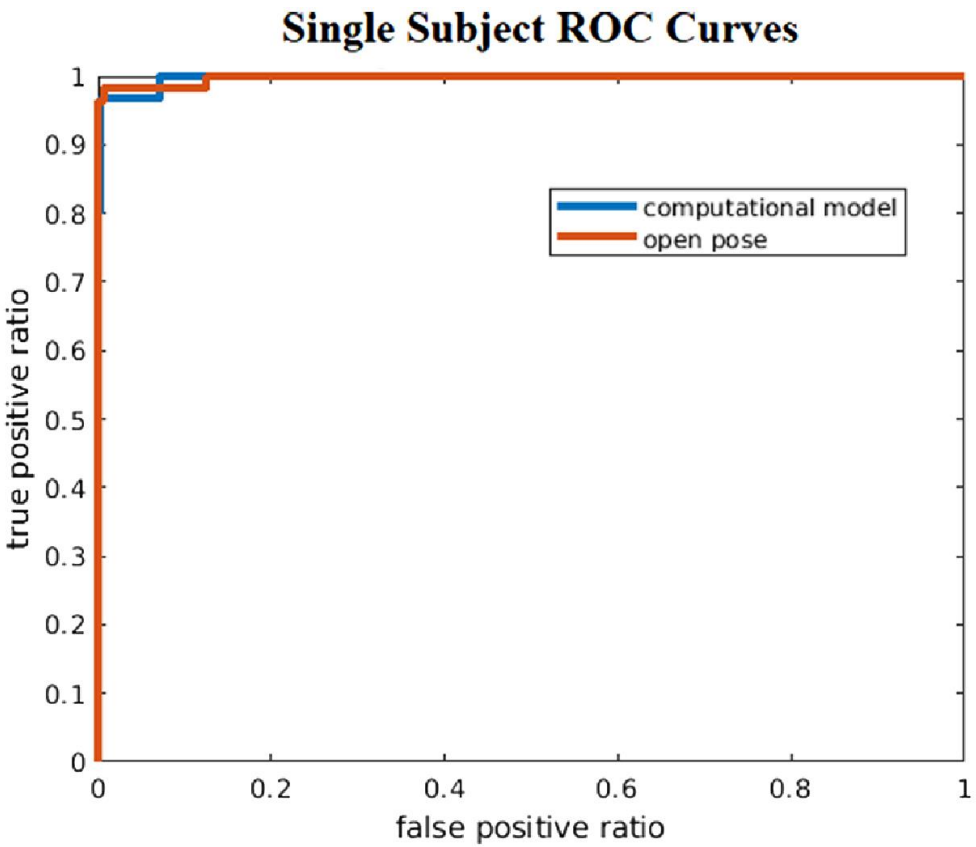


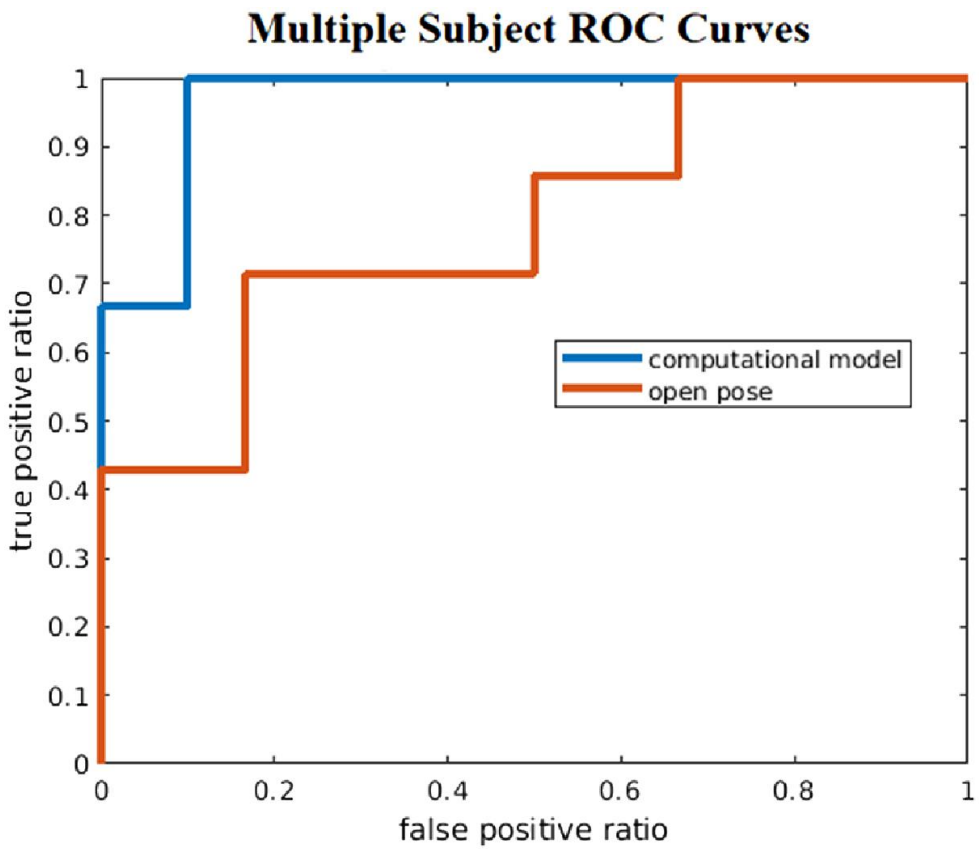
Classification Metrics for Example Applications - ROC Multiclass

- ROC A:
  - TPR 50%, FPR 0% -> sensitivity 50%, specificity 100%
  - TPR 100%, FPR 12.5% -> sensitivity 100%, specificity 87.5%
- ROC B:
  - TPR 87.5%, FPR 0% -> sensitivity 87.5%, specificity 100%
  - TPR 100%, FPR 50% -> sensitivity 100%, specificity 50%
- Best option is to maximize sensitivity at 100%, best specificity there is 87.5% (ROC A)



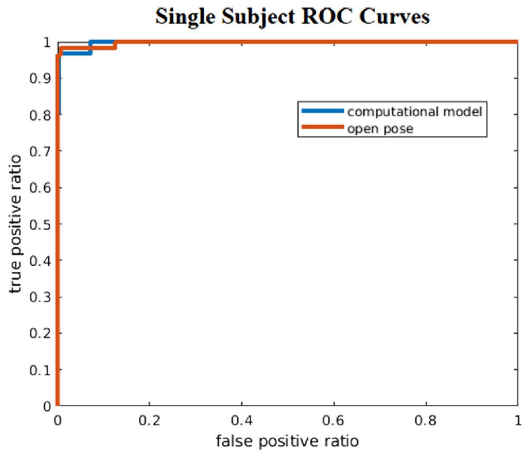
Classification Metrics for Example Applications - ROC Security Application

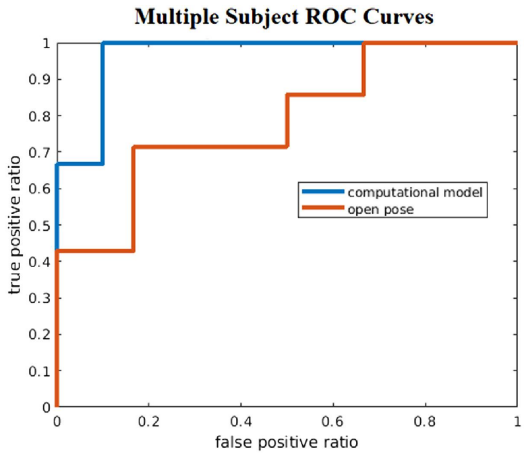




Classification Metrics for Example Applications - ROC Security Application

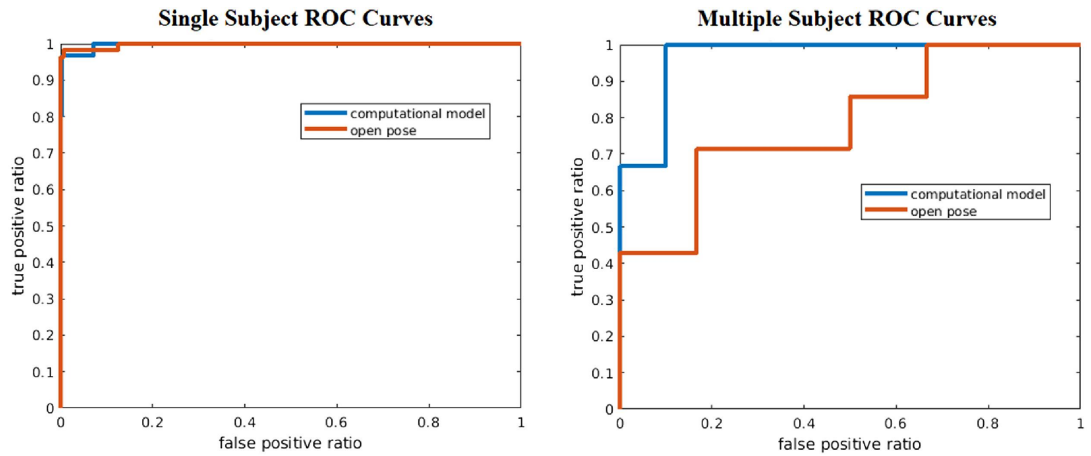
- Single Subject
  - Computational Model has 100% sensitivity at lower specificity than OpenPose –> meaning our model is better for security
  - I was incorrect about my interpretation during the previous presentation:
    - If the single subject (left ROC below) looked like ROC A that would be more desirable
    - Because we would have 100% sensitivity at an even higher specificity





Classification Metrics for Example Applications - ROC Security Application

- Multiple Subject
  - Computational Model better for whole curve



Classification Metrics for Example Applications -Precision and Recall

- Precision (positive predicted value)
- $\hat{p} \rightarrow p,$ predicted positive -> positive
- $\frac{p\hat{p}}{\hat{p}} = \frac{p\hat{p}}{p\hat{p}+n\hat{p}}$

- Recall (sensitivity)
- $p \rightarrow \hat{p}$, positive -> predicted positive
- $\frac{p\hat{p}}{p} = \frac{p\hat{p}}{p\hat{p}+p\hat{n}}$

Classification Metrics for Example Applications -Sensitivity and Specificity vs Precision and Recall

- Sensitivity and Specificity measure a classifier's expected performance on any sample (or more generally on any dataset)
- Precision and Recall measure a classifier’s performance on a particular dataset

Classification Metrics for Example Applications -Sensitivity and Specificity vs Precision and Recall

- Example:
- Validation Data Confusion Matrix:

	$\hat{p}$	$\hat{n}$
p	80	20
n	20	180

- For positive examples p, the classifier is expected to give 80% correct prediction of $\hat{p}$ (sensitivity)
- For negative examples n, the classifier is expected to give 90% correct prediction of $\hat{n}$ (specificity)
- Suppose we have a test dataset with a different balance, such as $p = 100, n = 1000$
- Then expected sensitivity and specificity are the same (assuming a strong validation dataset, called a standard test)
- Expected precision is different for this test dataset (next slide)

Classification Metrics for Example Applications -Sensitivity and Specificity vs Precision and Recall

- Example:
- Validation Data Confusion Matrix:

	$\hat{p}$	$\hat{n}$
p	80	20
n	20	180

- For positive examples p, the classifier is expected to give 80% correct prediction of $\hat{p}$ (sensitivity)
- For negative examples n, the classifier is expected to give 90% correct prediction of $\hat{n}$ (specificity)
- Suppose we have a test dataset with a different balance, such as $p = 100, n = 1000$
- Because the classifier gives 80% correct prediction of p, we expect $p\hat{p}$ = 80, $\mathrm{p}\hat{n} = 20$
- Because the classifier gives 90% correct prediction of n, we expect $n\hat{p}$ = 100, $\mathrm{n}\hat{n} = 900$
- The expected sensitivity and specificity are independent of the test data (assuming the validation dataset is large and representative)
- Expected precision is $\frac{p\hat{p}}{\hat{p}} = \frac{80}{180} = 44\%,$ which is different that the precision on the validation data
- Precision is affected by data balance (even if validation and test are both close to balanced, differences changes the expected precision)
- Expected recall is $\frac{p\hat{p}}{p} = \frac{80}{100} = 80\%$, which is the same as the recall (sensitivity) on the validation data

Classification Metrics vs Loss Functions

- Accuracy, Sensitivity, Specificity, F1, ROC – AUC are not appropriate loss functions
- All 5 metrics are based on $p\hat{p}, p\hat{n}, n\hat{p}, n\hat{n}$ (the confusion matrix)
- These values reduce the prediction probabilities to binary matching
- We don't train with these metrics for the same reason we use softmax instead of hardmax
- $\mathrm{softmax}(z_1, \ldots, z_I) = \left(\frac{e^{\alpha z_1}}{\sum_{i=1}^{I} e^{\alpha z_i}}, \ldots, \frac{e^{\alpha z_I}}{\sum_{i=1}^{I} e^{\alpha z_i}}\right)$
- $\mathrm{hardmax}(z_1, \ldots, z_I) = (\delta_{1=i}, \ldots, \delta_{I=i}),\ i = argmax_i\ (z_i)$
- hardmax is the limit of softmax as $\alpha \rightarrow\ \infty$, (minimum entropy)

Classification Metrics vs Loss Functions

- If we want to train a classifier, we need a loss that considers the predicted probability, not just the argmax or label prediction
- For multiclass classification we use cross entropy, which we can derive from maximum log likelihood estimation (MLE)
- We also see cross entropy in Maximum A Posteriori estimation (MAP) - see my last presentation

Classification Metrics vs Loss Functions -Imbalanced Data

- Focal Loss [4]
- Imbalanced data
- For example, consider a problem with only a few positive ground truth and mostly negative ground truth

Classification Metrics vs Loss Functions -Imbalanced Data

- For example, consider a problem with only a few positive ground truth and mostly negative ground truth
- $FL = -(1-\hat{y}_i)^{\gamma}\log(\hat{y}_i)\,, \gamma \geq 0$, called Focal Loss [4]
- Ground truth $y = 0$ (easy class)
  - $\hat{y}_0$ small -> $FL = -(1-\hat{y}_0)^{\gamma}\log(\hat{y}_0) \approx \log(\hat{y}_0)$, full cross entropy loss for easy class incorrect prediction
  - $\hat{y}_0$ large -> $FL = -(1-\hat{y}_0)^{\gamma}\log(\hat{y}_0) < \log(\hat{y}_0)$, down-weighted cross entropy loss for easy class correct prediction
- Ground truth $y = 1$ (hard class)
  - $\hat{y}_1$ small -> $FL = -(1-\hat{y}_1)^{\gamma}\log(\hat{y}_1) \approx \log(\hat{y}_1)$, full cross entropy loss for hard class incorrect prediction
  - $\hat{y}_1$ large -> $FL = -(1-\hat{y}_1)^{\gamma}\log(\hat{y}_1) < \log(\hat{y}_1)$, down-weighted cross entropy loss for hard class correct prediction (which is accepted because $\log(\hat{y}_1) \approx 0$ for high probability $\hat{y}_1$)

Classification Metrics vs Loss Functions -Imbalanced Data

- Balanced Focal Loss
  - $-a_i(1-\hat{y}_i)^{\gamma}\log(\hat{y}_i)\,,\gamma \geq 0$ (refined Focal Loss from paper) [4]
- Symmetric Focal Loss (example of loss design)
  - $-\left(1-\frac{1}{2}\delta_{i=1}\right)^{\gamma}\log(\hat{y}_i)\,,\gamma \geq 0$
  - Has a symmetric (more intuitive) interpretation
    - Down-weight the cross-entropy loss for the easy examples
    - Use full cross-entropy loss for the hard examples
  - If there are a large number of easy examples, original Focal Loss does not down-weight easy misclassified examples, which may overwhelm the few number of hard examples
  - Extends nicely to multiclass without modification (not explicitly done in Focal Loss paper)
  - $-(1-c_i)^{\gamma}\log(\hat{y}_i)\,,\gamma \geq 0$, this multiclass form down-weights each class based on the value of c_i (which can be interpreted as the relative balance of data per class *i*)
  - For all forms original focal loss, balanced focal loss, and symmetric focal loss the γ hyperparameter retains the same interpretation as the amount of down-weighting (γ needs to be larger that 1 to down-weight, which the paper doesn't mention)

Classification Metrics vs Loss Functions -Imbalanced Data

- Future consideration
- Can use decision theory to find loss that fixes specificity at a desired level and tries to maximize sensitivity

Loss vs Accuracy

- What is a better predictor of testing accuracy?
  - Validation Loss or
  - Validation Accuracy?
- I believe validation accuracy is a better indicator, because the goal of a predictor is to predict which is measured by accuracy. Just because a model is confident, doesn’t mean that it is correct. It can also be confident and incorrect. It is hard to find a satisfying interpretation of prediction confidence. I couldn’t find an answer, so I did the following analysis.

First Consideration

- For any model M^0 with validation loss L_V^0 and testing accuracy A_T^0, there exists a family of models M^* with validation loss $L^* \neq L_V$ and testing accuracy $A_T^* = A_T^0$.

First Consideration

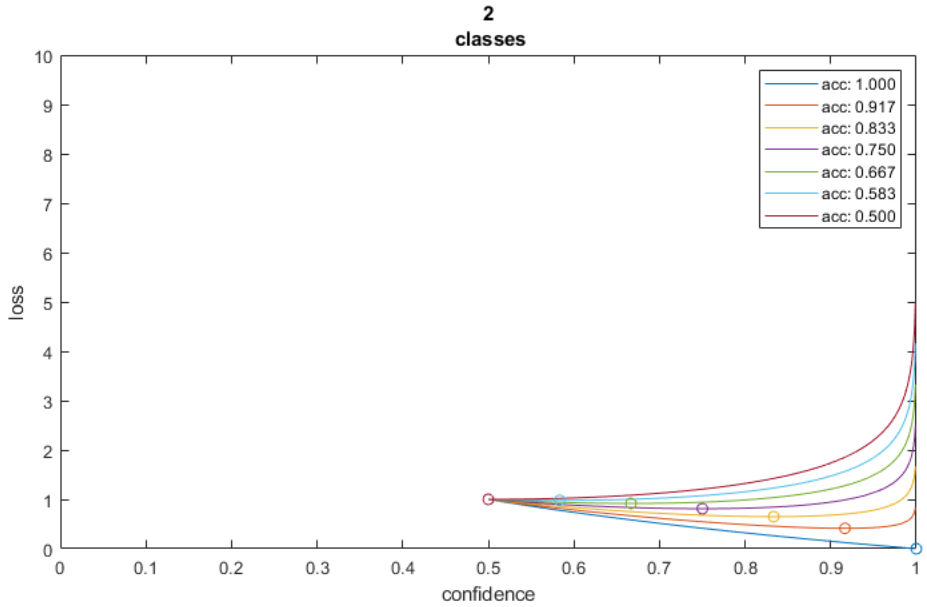
- Proof by construction:
  - Given original model M^0
  - M^* = {M^0 with new softmax temperature α^*}
- $\forall M^i \in M^*$, M^i has validation loss $L^i \neq L_V$ and testing accuracy $A_T^i = A_T^0$

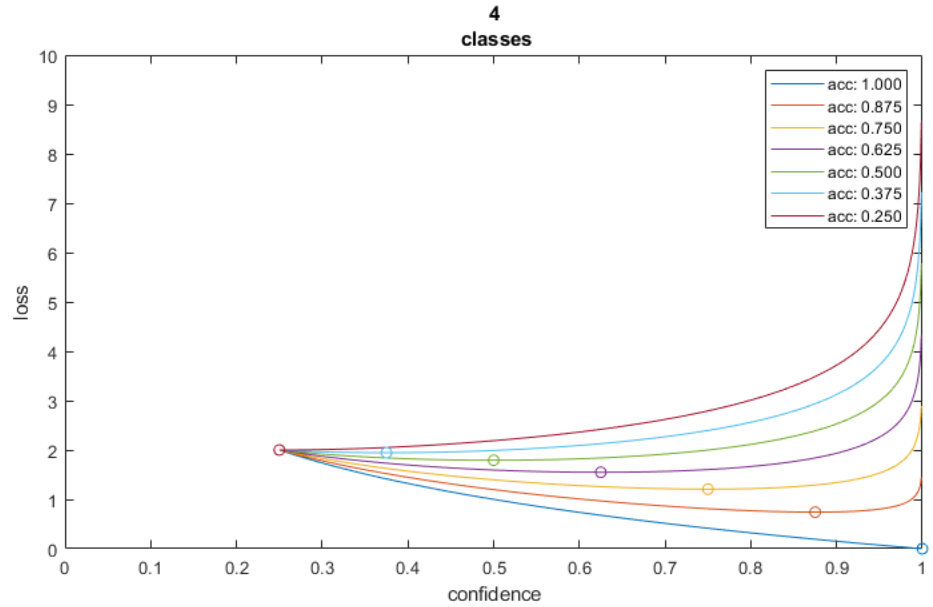
First Consideration

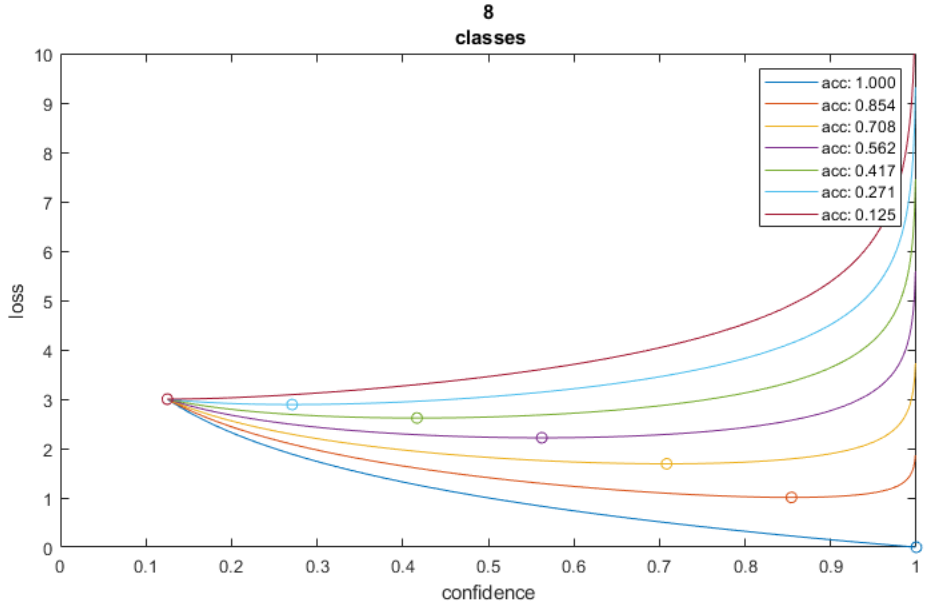
- We proved that there are models with different validations losses that have the same expected testing accuracy.
- This doesn’t mean that we proved expected testing accuracy is better predicted by validation accuracy.
- See next argument

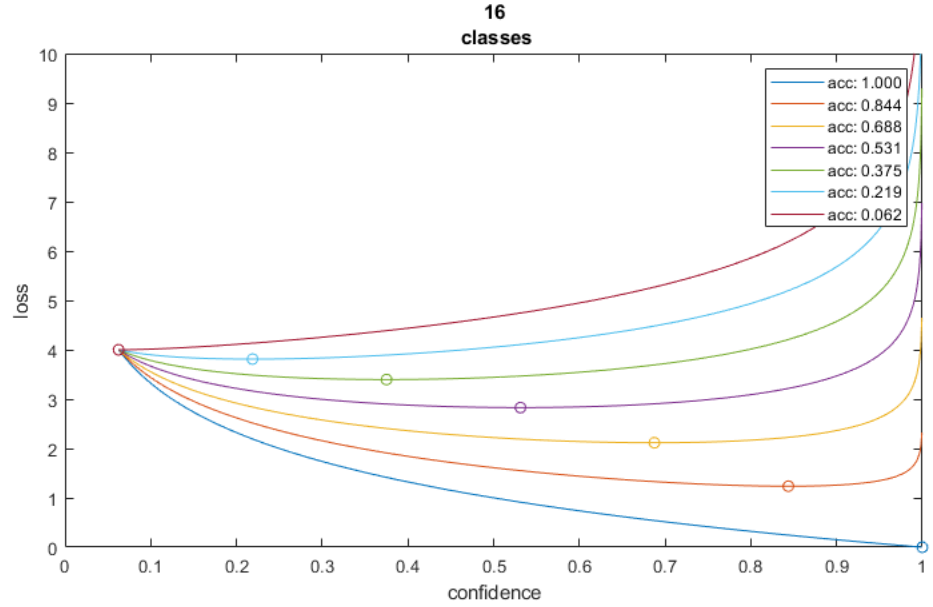
Second Consideration

- If all predictions had the same confidence, loss is minimized when confidence equals accuracy


2
classes
loss
confidence
acc: 1.000
acc: 0.917
acc: 0.833
acc: 0.750
acc: 0.667
acc: 0.583
acc: 0.500


4
classes
loss
confidence
acc: 1.000
acc: 0.875
acc: 0.750
acc: 0.625
acc: 0.500
acc: 0.375
acc: 0.250


8
classes
loss
confidence
acc: 1.000
acc: 0.854
acc: 0.708
acc: 0.562
acc: 0.417
acc: 0.271
acc: 0.125


16
classes
loss
confidence
acc: 1.000
acc: 0.844
acc: 0.688
acc: 0.531
acc: 0.375
acc: 0.219
acc: 0.062

Second Consideration

- Proof in general case of M classes
  - A is accuracy, N is the number of samples, M is the number of classes, P^* is the confidence of the predicted class which we fix for all examples and $(1 - P^*)/(M - 1)$ is the confidence of the non predicted classes
  - In other words, in we know nothing else, what fixed confidence will minimize loss?
  - $loss = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} P(y_i = j)(-1)\log(P(\hat{y} = \mathrm{j})) =$
  - $P(y_i = j_{correct})(-1)\log\big(P(\hat{y}_i = j)\big) + P(y_i = j_{incorrect})(-1)\log\big(P(\hat{y}_i = j)\big)$
  - $loss = A(-1)\log(P^*) + (1 - A)(-1)\log\left(\frac{1-P^*}{M-1}\right)$
  - $loss = -A\log(P^*) + (A - 1)\log\left(\frac{1-P^*}{M-1}\right)$

Second Consideration

- Proof continued
- $loss = -A\log(P^*) + (A-1)\log\left(\frac{1-P^*}{M-1}\right)$
- $argmin_{P^*}(-A)\log(P^*) + (A-1)\log\left(\frac{1-P^*}{M-1}\right) =$
- $argmin_{P^*}(-A)\log(P^*) + (A-1)\log(1-P^*) - (A-1)\log(M-1) =$
- $argmin_{P^*}(-A)\log(P^*) + (A-1)\log(1-P^*)$
- setting derivative to zero
- $(-A)\frac{1}{\ln(2)P^*} + (A-1)\frac{-1}{\ln(2)(1-P^*)} = 0$
- $(-A)\frac{1}{\ln(2)P^*} = (A-1)\frac{1}{\ln(2)(1-P^*)}$
- $(-A)(1-P^*) = (A-1)P^*$
- $-A + AP^* = AP^* - P^*$
- $P^* = A$

Second Consideration

- $P^* = A$
- Again, this doesn't prove that expected testing accuracy is better predicted by validation accuracy.

References

[1] Theory of Point Estimation - Lehmann and Casella

[2] Statistical Inference – Azzalini

[3] Fundamentals of Biostatistics – Rosner

[4] Ross et al. – Focal Loss for Dense Object Detection