CS 584 – Machine Learning

Topic: Classifier Evaluation

Mustafa Bilgic

http://www.cs.iit.edu/~mbilgic

https://twitter.com/bilgicm

TASK

- Given a labeled dataset $\mathcal{D} = \{\langle x_i, y_i \rangle\}$, where x_i is the input and y_i is the discrete output
- Train a classifier $f: \mathcal{X} \rightarrow \mathcal{Y}$ using $\mathcal{D}$
- The purpose of f is to perform "well" on unseen data
- How do we define "well"?

0/1 Error & Accuracy

- The simplest measure is "is the prediction correct?"
- Examples
  - Given an email, the model predicts it's spam. Is it correct?
  - Given a patient, the model predicts the patient is suffering from Heart disease. Is it correct?
  - Given a loan application, the model recommends reject. Is the recommendation correct?
- Given a dataset, accuracy is the percentage of objects the model's predictions are correct

Some Problems with Accuracy

- All mistakes are considered equal; for example
  - Misclassifying a ham email as spam, and misclassifying a spam email as ham are considered equally bad
  - Approving a loan application that should have been rejected, and rejecting a loan application that should have been approved are considered equally bad
- If a class is dominant, it's often easy to get high accuracy by simply predicting every object as the dominant class; for example
  - If 80% of the emails are ham, a classifier that classifies every email as ham will have 80% accuracy
- All cases are considered equal; for example, email from your family, boss, bank, social media updates, … are all considered equally important, which might or might not be true

Types of Errors – Classification

- Assume a target/positive class
  - Spam, HasHeartDisease, Approve, etc.
  - This step is important; positive does not mean "good"; positive mean the concept of interest and you decide which class is positive
    - For example, positive covid test does not mean "good" news
- *False positive*
  - Falsely classifying an object as positive
    - E.g., classifying a ham email as spam, diagnosing a healthy patient as having heart disease, approving a loan that should have been rejected, and so on
  - Also called *Type I* error
- *False negative*
  - Falsely classifying an object as negative
    - E.g., classifying a spam email as ham, claiming that a heart-disease patient is healthy, rejecting a loan that should have been approved, and so on
  - Also called *Type II* error

Confusion Matrix

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positive	False Negative
	Negative	False Positive	True Negative

Accuracy

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	TP True Positive	FN False Negative
	Negative	FP False Positive	TN True Negative

$$Accuracy = \frac{Num\ Correct}{Data\ Size} = \frac{TP + TN}{TP + TN + FP + FN}$$

PRECISION

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positive	False Negative
	Negative	False Positive	True Negative

$$Precision = \frac{True\ Positive}{Predicted\ Positive} = \frac{TP}{TP + FP}$$

True Positive Rate – Recall – Sensitivity

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positive	False Negative
	Negative	False Positive	True Negative

$$TPR = Recall = \frac{True\ Positive}{Actual\ Positive} = \frac{TP}{TP + FN}$$

True Negative Rate – Specificity

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positive	False Negative
	Negative	False Positive	True Negative

$$TNR = Specificity = \frac{True\ Negative}{Actual\ Negative} = \frac{TN}{TN + FP}$$

False Positive Rate – Fall-out

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positive	False Negative
	Negative	False Positive	True Negative

$$FPR = FallOut = \frac{False\ Positive}{Actual\ Negative} = \frac{FP}{TN + FP}$$

False Negative Rate – Miss Rate

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positive	False Negative
	Negative	False Positive	True Negative

$$FNR = Miss\ Rate = \frac{False\ Negative}{Actual\ Positive} = \frac{FN}{TP + FN}$$

F1

		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	True Positive	False Negative
	Negative	False Positive	True Negative

Harmonic mean of Prec. & Rec.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall}$$

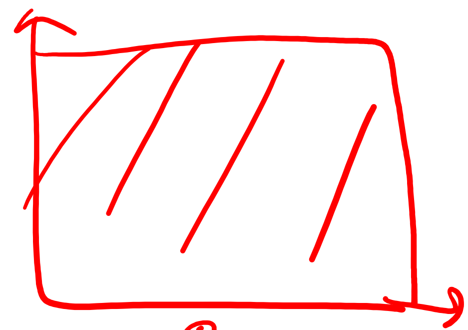
$$\frac{2}{\frac{1}{P} + \frac{1}{R}}$$

Other Measures based on Confusion Matrix

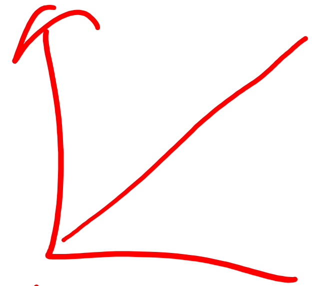
- False discovery rate = FP/PP
- False omission rate = FN/PN
- Negative predictive value = TN/PN
- Positive likelihood ratio = TPR/FPR
- Negative likelihood ratio = FNR/TNR
- Diagnostic odd ratio = PLR / NLR
- …

AREA UNDER THE CURVE (AUC)

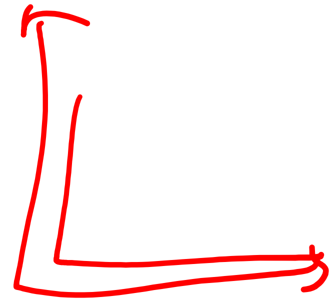
- **A**rea **U**nder the **C**urve
- What curve? ROC Curve
  - **R**eceiving **O**perating **C**haracteristic
  - The X axis is False Positive Rate
  - The Y axis is True Positive Rate
  - The curve is plotted by varying the “decision” threshold



Best



Random



Worst

AUC Example

- Assume 10 actual positives and 20 actual negatives
- Plot the ROC curve and compute the area under it for the following cases:
  - P, P, …, P, N, N, …, N
  - P, N, N, P, N, N, …, P, N, N

True Error

- Given $h(x)$, we are interested in
  - $\sum_{x \sim \mathcal{X}} P(x) 1[\![h(x) \neq c(x)]\!]$, where
  - $\mathcal{X}$ is the space of all possible instances
  - $P(x)$ is the probability of seeing instance x
  - $1[\![h(x) \neq c(x)]\!]$ is 1 if the prediction by h is incorrect; 0 otherwise
- Problems
  - $\mathcal{X}$ is super large; exponential in the size of the domains of the variables
  - We do not know $P(x)$

$\frac{1}{8} = 0.125$

	MP	I	W	P(x)
0 →	Y	L	L	0.05
1 →	Y	L	H	0.04
1	Y	H	L	0.045
0	Y	H	H	0.03
⋮	⋮	⋮	⋮	⋮
	L	H	H	0.20
+				+
				1

True Error

- Given $h(x)$, we are interested in
  - $\sum_{x \sim \boldsymbol{\mathcal{X}}} P(x) 1[\![h(x) \neq c(x)]\!]$, where
  - $\boldsymbol{\mathcal{X}}$ is the space of all possible instances
  - $P(x)$ is the probability of seeing instance x
  - $1[\![h(x) \neq c(x)]\!]$ is 1 if the prediction by h is incorrect; 0 otherwise
- Problems binary 100 variables 2^{100}
  - $\boldsymbol{\mathcal{X}}$ is super large; exponential in the size of the domains of the variables
  - We do not know $P(x)$

Sampling

- When space is large, sample from $P(x)$
- When $P(x)$ is not known, or sampling from $P(x)$ is not possible, collect a "representative" sample
- Let $\mathcal{D} \sim P(x)$ be a representative sample
- For example, true mean versus sample mean
  - $\sum_{x \sim \boldsymbol{x}} P(x) x$
  - $\frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} x$
- How do we know how close the true mean and the sample mean are?

$n = 40$ $18H$ $22T$ $\hat{p} = \frac{18}{40}$ p^*

SAMPLE ERROR

$\_ \leq \hat{p} \leq \_$ $p^* \leq 5\%$

n $\hat{p} = \frac{\#H}{n}$

$|D|$ $\frac{\#\text{errors}}{|D|}$

- Let $\mathcal{D} \sim P(x)$ be a representative dataset
- Sample error
  - $\frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} 1[\![h(x) \neq c(x)]\!]$
- Remember the binomial distribution
  - n experiments, each with p success probability
  - n data points, each with p true error
  - Sample error is based on a binomial distribution where exactly k objects are incorrectly labeled
  - What is the probability that k objects are incorrectly labeled? What is the expectation? What is the variance? What is the 95% confidence interval?

$\binom{n}{k} p^k (1-p)^{n-k}$

- Important note: h and $\mathcal{D}$ must be independent; h cannot depend on $\mathcal{D}$

SPLITTING THE DATASET

1. Train-test splits
2. Train-validation-test splits
3. Cross-validation

Train-Test Split

- Randomly split the data into two disjoint sets
- A typical approach: 2/3 for train and 1/3 for test
- Train your model on training data and evaluate it on the test data
  - Use your favorite performance metric
- Report your performance as the expected performance on unseen data
- Caveats:
  - You need a large dataset for this to work
  - You cannot tune your parameters on the test data

Train-Validation-Test Split

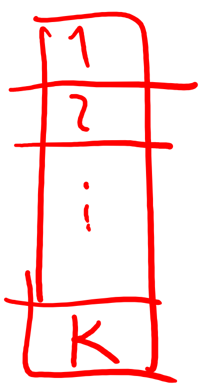
- Split your data into three disjoint sets
  - Train, validation, test
- Train your model(s) on the training data
- Evaluate your model(s) on the validation data
- Pick the model that performs best on the validation data
- Test the model on the test data, and report its performance

only once

- Caveat:
  - You need a really big dataset for this to work

Cross-Validation

- Split your data into k disjoint sets
- Each time, one set is the test set and the rest is the training set



f_1 test, $f_2\, f_3 \cdots f_K$ train
f_2 test, $f_1\, f_3 \cdots f_k$ tn
f_3 test, ... —
⋮
f_K test, ... —

Cost-Sensitive Classification

- Given a probabilistic output for an object, say $\langle p, 1-p \rangle$, how do we decide which class to assign to this object?
- The simplest approach is check whether $p > 0.5$ and make a decision accordingly
- This assumes each mistakes (False Positives and False Negatives) are equally costly

Equal Misclassification Costs?

- Which one is worse for you:
  - Delivering a spam email into your Inbox (False Negative), or
  - Delivering a legitimate email into your Spam folder (False Positive)?
- If one is worse than the other, then, should we use 0.5 as the decision threshold or should we adjust it to your preference?

Cost Matrix

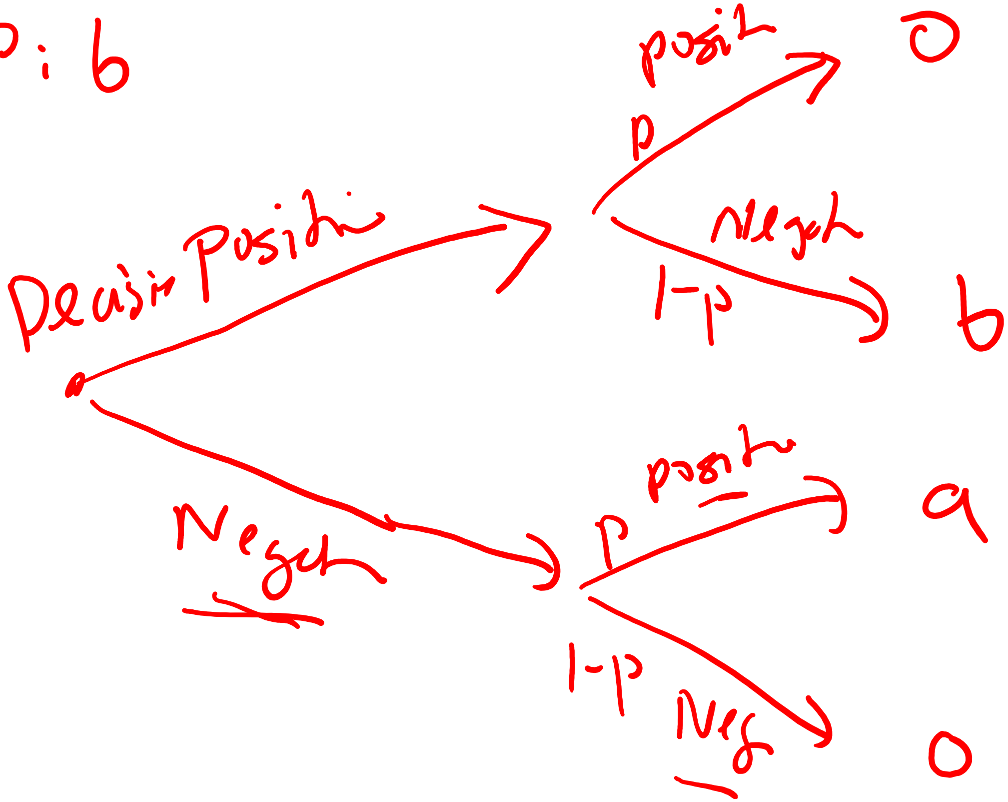
		Predicted Class	
		Positive	**Negative**
Actual Class	**Positive**	0	a
	Negative	b	0

Given a probability distribution of $\langle p, 1-p \rangle$ for $\langle Positive, Negative \rangle$ respectively, and given the above cost matrix, under what conditions (in terms of a, b, and p) would you classify an object as $Positive$?

Positive: p

FN: a

FP: b



Cost

Label positive

$(1-p)\, b$

Label negative

$p \cdot a$

decide Positive

$(1-p)b < pa$