COMP472: Artificial Intelligence
Machine Learning
Evaluation

video #5

- Russell & Norvig: Sections 19.4

Today

1. Introduction to ML
2. Naïve Bayes Classification Model #1
   a. Application to Spam Filtering
3. Decision Trees Model #2
4. Evaluation 

5. Unsupervised Learning
6. Neural Networks
   a. Perceptrons
   b. Multi Layered Neural Networks

Data Sets

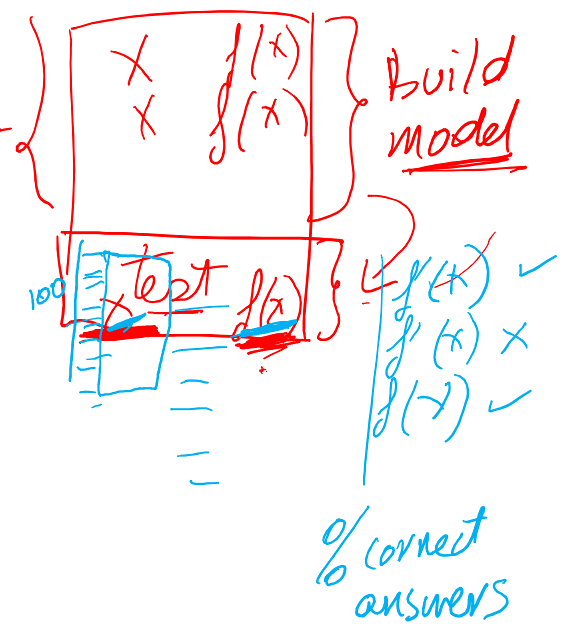
supervised learning

- How do you know if what you learned is correct?
- You run your classifier on a data set of unseen examples (that you did not use for training) for which you know the correct classification

- Split data set into 3 sub-sets
  1. Actual **training** set (~80%)
  2. **Validation** set (~20%)
  3. **Test** set

$\left.\begin{array}{l}\text{Actual training set (~80\%)}\\ \text{Validation set (~20\%)}\end{array}\right\}$ ~80%

Test set } ~20%



Standard Methodology

1. Collect a large set of examples (all with correct classifications)
2. Divide collection into training, validation and test set

Loop:

3. Apply learning algorithm to the training set to learn the parameters
4. Measure performance with the validation set, and adjust hyper-parameters* to improve performance

5. Measure performance with the test set

- DO NOT LOOK AT THE TEST SET until step 5.

Parameters: basic values learned by the ML model. eg.

- for NB: prior & conditional probabilities likelihood
- for DTs: features to split
- for ANNs: weights

Hyper-parameters: parameters used to set up the ML model. eg.

- for NB: value of delta for smoothing, ex. add-1 0.5 0.3
- for DTs: pruning level
- for ANNs: nb of hidden layers, nb of nodes per layer…

Metrics

- 1 Accuracy
  - % of instances of the test set the algorithm correctly classified
  - when all classes are equally important and represented

f(x)

balanced dataset

- 2 Recall, Precision & F-measure
  - when one class is more important and the others

80% dolph
10% dog
10% cat

not balanced

no-fracture 99%

fracture 1%

Accuracy

- % of instances of the test set the algorithm correctly classifies
- when all classes are equally important and represented
- problem:
  - when one class (eg. sick) is more important and the others
  - eg. when data set is unbalanced

correct classification

prediction

5 mistakes

	Target	system 1
X1	sick	ok ×
X2	sick	ok ×
X3	sick	ok ×
X4	sick	ok ×
X5	sick	ok ×
X6	ok	ok ✓
X7	ok	ok ✓
…	…	…
…	…	…
X500	ok	ok ✓
Accuracy		495/500 = 99% !

Recall, Precision

- Recall: What proportion of the instances in the class of interest (eg. sick) are labelled correctly?
- Precision: What proportion of instances labeled with the class of interest (eg. sick) are actually correct?

		Correct class	
		instance should be in class C	instance should not be in class C
Model prediction	instance is put in class C	True Positive (TP)	False Positive (FP)
	instance is not put in class C	False Negative (FN)	True Negative (TN)

$$\text{Precision} = \frac{TP}{TP+FP}$$

TP: nb of instances that are in class C and that the model identified as class C

TP+FP: nb of instances that the model labelled as class C

$$\text{Recall} = \frac{TP}{TP+FN}$$

TP: nb of instances that are in class C and that the model identified as class C

TP+FN: All instances that are in class C

Example

correct classification

class of interest = sick

	Target	system 1	system 2	system 3
X1	sick	sick ✓	sick ✓	ok ×
X2	sick	ok ×	ok ×	sick ✓
X3	sick	ok ×	sick ✓	sick ✓
X4	sick	ok ×	sick ✓	sick ✓
X5	sick	ok ×	ok ×	sick ✓
X6	ok	ok ✓	ok ✓	sick ×
X7	ok	ok ✓	ok ✓	sick ×
..	ok	ok ✓	ok ✓	ok ✓
..	ok	ok ✓	ok ✓	ok ✓
X500	ok	ok ✓	ok ✓	ok ✓
Accuracy		496/500 = 99%	498/500 = 99.6%	498/500 = 99.6%
Precision		1/1 = 100%	3/3 = 100%	4/6 = 66.7%
Recall		1/5 = 20%	3/5 = 60%	4/5 = 80%

Which system is better?

A Single Measure

- cannot take mean of P&R
  - if R = 50% P = 50% M = 50%
  - if R = 100% P = 10% M = 55% (not fair)

1. take harmonic mean
   - which penalizes extreme values

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

HM is high only when both P&R are high

if R = 50% and P = 50% HM = 50%

if R = 100% and P = 10% HM = 18.2%

2. if P and R should not have the same importance in the problem domain, take weighted harmonic mean

$$WHM = \frac{1}{\frac{1}{2}\frac{1}{R} + \frac{1}{2}\frac{1}{P}}$$ // if weight R = weight P = $\frac{1}{2}$

$\frac{1}{2} + \frac{1}{2} = 1$

$$WHM = \frac{1}{\frac{1}{a}\frac{1}{R} + \frac{1}{b}\frac{1}{P}}$$ // if weight R = $\frac{1}{a}$ weight P = $\frac{1}{b}$ and $\boxed{\frac{1}{a} + \frac{1}{b} = 1}$

Weighted Harmonic Mean of P&R

$$WHM = \frac{1}{\frac{1}{a}\frac{1}{R} + \frac{1}{b}\frac{1}{P}}$$ // if weight R = $\frac{1}{a}$ weight P = $\frac{1}{b}$ and $\frac{1}{a} + \frac{1}{b} = 1$

1. let $w_R = \frac{\delta}{\delta+1}$ $w_P = \frac{1}{\delta+1}$ // so that $w_R + w_P = \frac{\delta+1}{\delta+1} = 1$

$$WHM = \frac{1}{\left(\frac{\delta}{\delta+1}\right)\frac{1}{R} + \left(\frac{1}{\delta+1}\right)\frac{1}{P}} = \frac{\delta+1}{\delta\frac{1}{R} + 1\frac{1}{P}} = \frac{(\delta+1)PR}{\delta P + 1R}$$

2. let $\delta = \beta^2$

$$WHM = \boxed{\frac{(\beta^2+1)PR}{\beta^2 P + 1R}}$$ // called the F-measure

F-measure

- A weighted harmonic mean of precision and recall

$$F_{\beta} = \frac{(\beta^2 + 1)PR}{(\beta^2 P + R)}$$

- β represents the relative importance of recall to precision
  - when β = 1
    - F1 measure
    - precision & recall have same importance
  - when β > 1
    - recall is given more weigth
    - e.g. F_2 measure, recall is considered 2x more important than precision
  - when β < 1
    - precision is given more weigth
    - e.g. $F_{0.5}$ measure, precision is considered 2x more important than recall

$\beta = 2$

$\beta = 0.5$

Example

	Target	system 1	system 2	system 3
X1	sick	sick ✓	sick ✓	ok ×
X2	sick	ok ×	ok ×	sick ✓
X3	sick	ok ×	sick ✓	sick ✓
X4	sick	ok ×	sick ✓	sick ✓
X5	sick	ok ×	ok ×	sick ✓
X6	ok	ok ✓	ok ✓	sick ×
X7	ok	ok ✓	ok ✓	sick ×
..	ok	ok ✓	ok ✓	ok ✓
..	ok	ok ✓	ok ✓	ok ✓
X500	ok	ok ✓	ok ✓	ok ✓
Accuracy		496/500 = 99%	498/500 = 99.6%	498/500 = 99.6%
Precision		1/1 = 100%	3/3 = 100%	4/6 = 66.7%
Recall		1/5 = 20%	3/5 = 60%	4/5 = 80%
F1-measure 2PR/(P+R)		2*100*20/ (100+20) = 33%	75%	72.9%

$\beta = 1$ $F_1 = \frac{2PR}{P+R}$

P, R and F for Multiclass Classification

- previous P, R and F are ok when 1 particular class interests us (eg. sick)
- What if several classes interest us?
- then
  - compute *per-class* P, R, F
  - and to have a single measure for all classes: combine per-class F-measures via
    - macro F-measure, or
    - weighted-average F-measure

Per-class Precision & Per-class Recall

gold

model prediction

		Correct Class			
		Cat	Dog	Fish	Total
Predicted Class	Cat	4	6	3	Σ 13
	Dog	1	2	0	3
	Fish	1	2	6	9
	Total	6	10	9	25

images

- precision of class Cat: 4/(4+6+3) = 30.8%
- precision of class Dog: 2/(1+2+0) = 66.7%
- precision of class Fish: 6/(1+2+6) = 66.7%

- recall of class Cat: 4/(4+1+1) = 66.7%
- recall of class Dog: 2/(2+6+2) = 20%
- recall of class Fish: 6/(3+0+6) = 66.7%

Per-class F1-measure

Model #1

Model #2

F1

	Precision	Recall	F1
Cat	30.8%	66.7%	42.1%
Dog	66.7%	20.0%	30.8%
Fish	66.7%	66.7%	66.7%

F1 = 2PR/(P+R)

$F_{b=1}$

- F1 of class Cat: (2 x .308 x .667) / (.308 + .667) = 0.421
- F1 of class Dog: (2 x .667 x .200) / (.667 + .200) = 0.308
- F1 of class Fish: (2 x .667 x .667) / (.667 + .667) = 0.667

Macro and Weighted-Average Measures

macro precision,
macro recall,
macro F1

	Precision	Recall	F1
Cat	30.8%	66.7%	42.1%
Dog	66.7%	20.0%	30.8%
Fish	66.7%	66.7%	66.7%
average	(30.8+66.7+66.7) / 3 = 54.7%	(66.7 + 20.0 + 66.7) / 3 = 51.1%	(42.1+30.8+66.7) = 46.5%
weighted-average	(6x30.8 // 6 cat + 10x66.7 // 10 dog + 9x66.7) // 9 fish / (6+10+9) // 25 samples = 58.1%	(6x66.7 + 10x20.0 + 6x66.7) / 25 = 48.0%	(6x42.1 + 10x30.8 + 6x66.7) / 25 = 46.4%

weighted-averaged precision,
weighted-averaged recall,
weighted-averaged F1

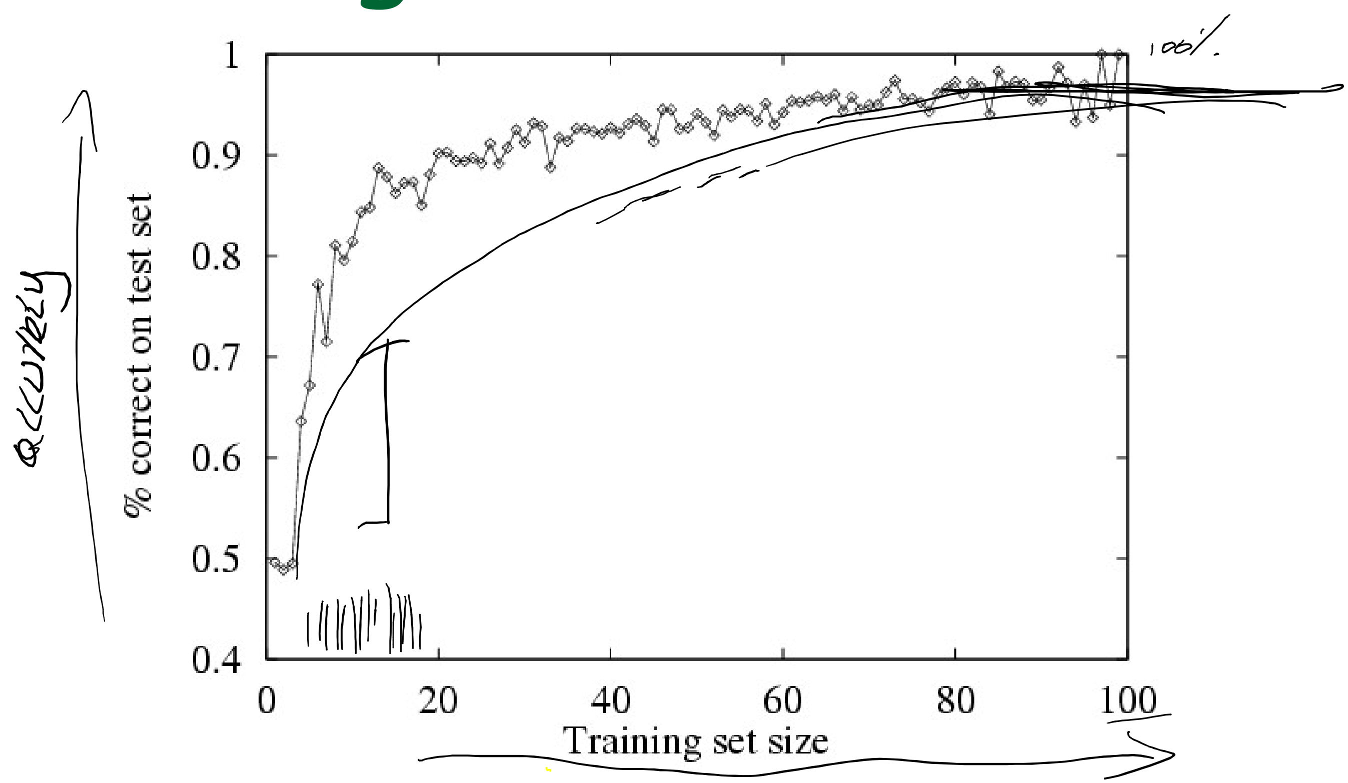
- To combine measures into a single one, we can:
  - take simple average
    - --> macro precision, macro recall, macro F1
  - take weighted average
    - ie. weight the average based on the nb of samples from each class
    - --> weighted averaged precision, weighted averaged recall, weighted averaged F1

Confusion Matrix

- to do an error analysis and find out where the model went wrong ?
- aka contingency table
- eg. 6 classes, 100 test instances

		Correct Class						
		C1	C2	C3	C4	C5	C6	Total
Predicted Class	C1	10✓	3×	0	0	3×	0	16
	C2	0	12✓	3×	4×	0	0	19
	C3	0	1×	9✓	2×	1×	2×	15
	C4	0	1×	3×	5✓	2×	0	11
	C5	0	0	3×	2×	10✓	3×	18
	C6	0	0	5×	0	5×	11✓	21
	Total	10	17	23	13	21	16	100

A Learning Curve



- Size of training set
  - the more, the better
  - but after a while, not much improvement…

source: Mitchell (1997)

Some Words on Training

- In all types of learning… watch out for:
  - Noisy input
  - Overfitting/underfitting the training data

Noisy Input

- In all types of learning… watch out for:
  - Noisy Input:
    - Two examples have the same feature-value pairs, but different outputs

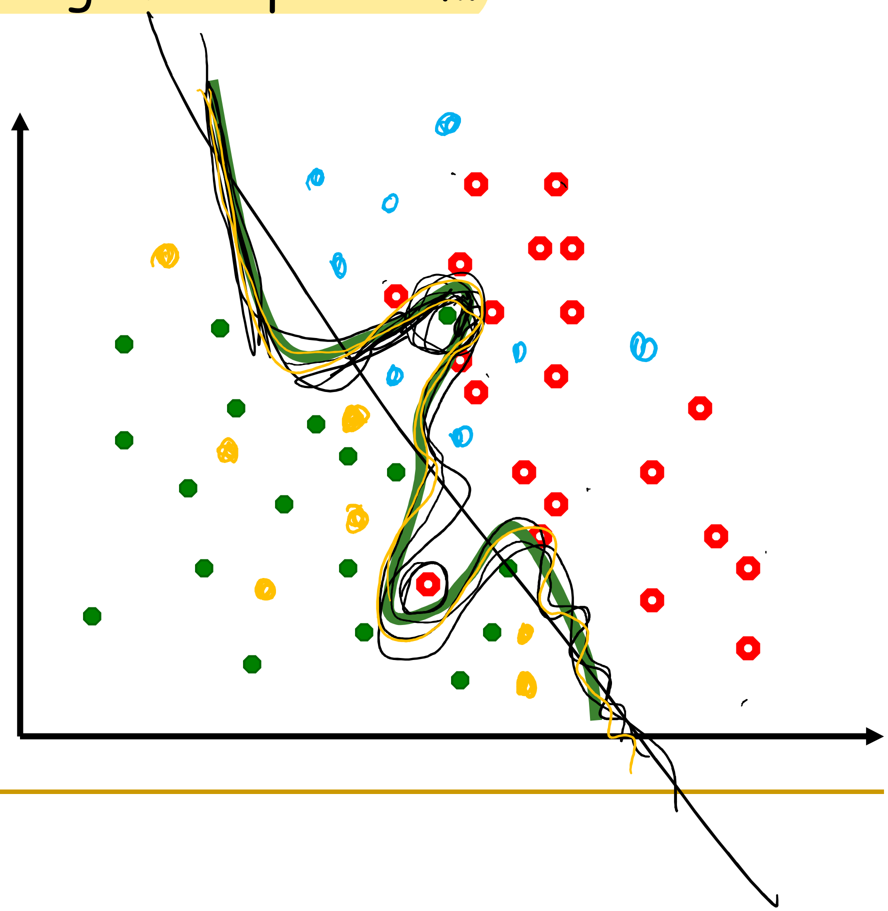
4th feature

Size	Color	NF	Shape	Output
Big	Red	a	Circle	+
Big	Red	b	Circle square	+

    - Some values of features are incorrect or missing (ex. errors in the data acquisition)
    - Some relevant attributes are not taken into account in the data set

Overfitting

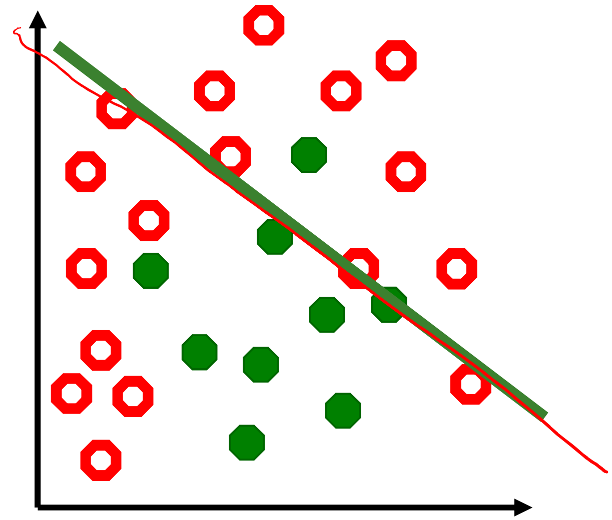
- If a large number of irrelevant features are there, we may find meaningless regularities in the data that are particular to the training data but irrelevant to the general problem.
- Complicated boundaries *overfit* the data
- they are too tuned to the particular training data at hand
- They do not *generalize* well to the new data
- Extreme case: "rote learning"



- Training error is low
- Testing error is high

Underfitting

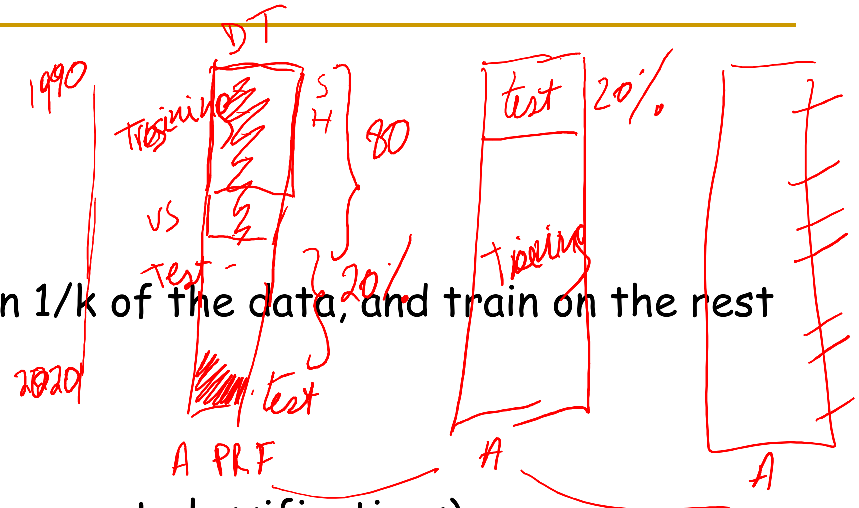
- We can also underfit data, i.e. find a decision boundary that is too simple
- Model is not expressive enough (not enough features, or not enough capacity)
- eg. There is no way to fit a linear decision boundary so that the training examples are well separated



- Training error is high
- Testing error is high

Cross-validation

- K-fold cross-validation
  - run k experiments, each time you test on 1/k of the data, and train on the rest
  - than you average the results



- ex: 10-fold cross validation
  1. Collect a large set of examples (all with correct classifications)
  2. Divide collection into two disjoint sets: training (90%) and test (10% = 1/k)
  3. Apply learning algorithm to training set
  4. Measure performance with the test set
  5. Repeat steps 2-4, with the 10 different portions
  6. Average the results of the 10 experiments

entire DS

1990 … 2020

exp1:	train									test 1/k
exp2:	train								test 1/k	train
exp3:	train							test	train	
…	…									

exp 10

F-measure = 80
F-measure 8/60
F - 83 99
F - 82 50
F

stable unstable

average

Today

1. Introduction to ML
2. Naïve Bayes Classification
   a. Application to Spam Filtering
3. Decision Trees
4. Evaluation
5. Unsupervised Learning
6. Neural Networks
   a. Perceptrons
   b. Multi Layered Neural Networks

Up Next

1. Introduction to ML
2. Naïve Bayes Classification
   a. Application to Spam Filtering
3. Decision Trees
4. (Evaluation
5. Unsupervised Learning)
6. Neural Networks
   a. Perceptrons
   b. Multi Layered Neural Networks