Introduction to Machine Learning

Evaluating Model Performance

Performance measures

predicted → / real ↓	*Class_pos*	*Class_neg*
Class_pos	TP	FN
Class_neg	FP	TN

Which classifier is better?

Algo	Acc	RMSE	TPR	FPR	Prec	Rec	F	AUC
NB	71.7	.4534	.44	.16	.53	.44	.48	.7
C4.5	75.5	.4324	.27	.04	.74	.27	.4	.59
3NN	72.4	.5101	.32	.1	.56	.32	.41	.63
Ripp	71	.4494	.37	.14	.52	.37	.43	.6
SVM	69.6	.5515	.33	.15	.48	.33	.39	.59
Bagg	67.8	.4518	.17	.1	.4	.17	.23	.63
Boost	70.3	.4329	.42	.18	.5	.42	.46	.7
RanF	69.23	.47	.33	.15	.48	.33	.39	.63

Accuracy

predicted → / real ↓	*Class_pos*	*Class_neg*
Class_pos	TP	FN
Class_neg	FP	TN

P = TP+FN

N = TN+FP

- Accuracy is % correct (fraction correct)
- Accuracy = (TP + TN)/(P+N)

Issues with Accuracy

Predict → / True ↓	Pos	Neg
Pos	200	300
Neg	100	400

Predict → / True ↓	Pos	Neg
Pos	400	100
Neg	300	200

Issues with Accuracy

Predict → / True ↓	Pos	Neg
Pos	200	300
Neg	100	400

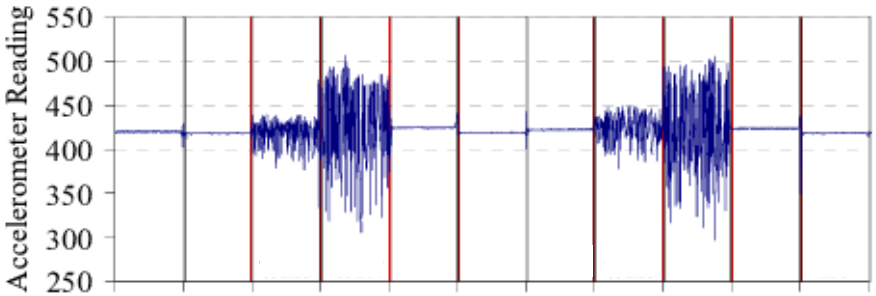
Predict → / True ↓	Pos	Neg
Pos	400	100
Neg	300	200

Issues with Accuracy

Predict → / True ↓	Pos	Neg
Pos	200	300
Neg	100	400

Predict → / True ↓	Pos	Neg
Pos	400	100
Neg	300	200

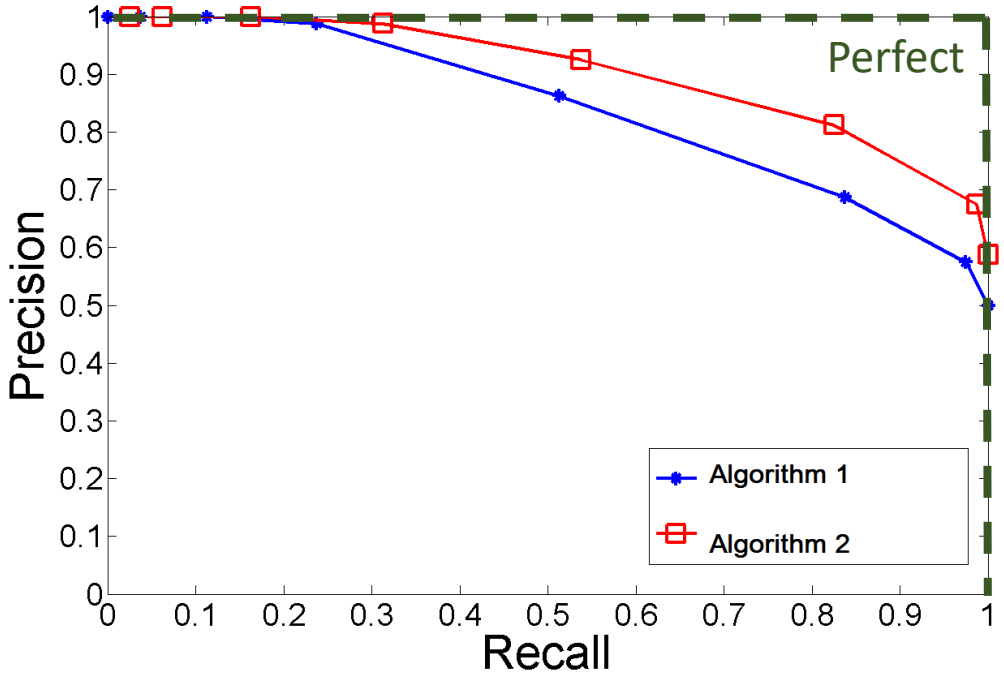
Discrimination vs detection (spotting)



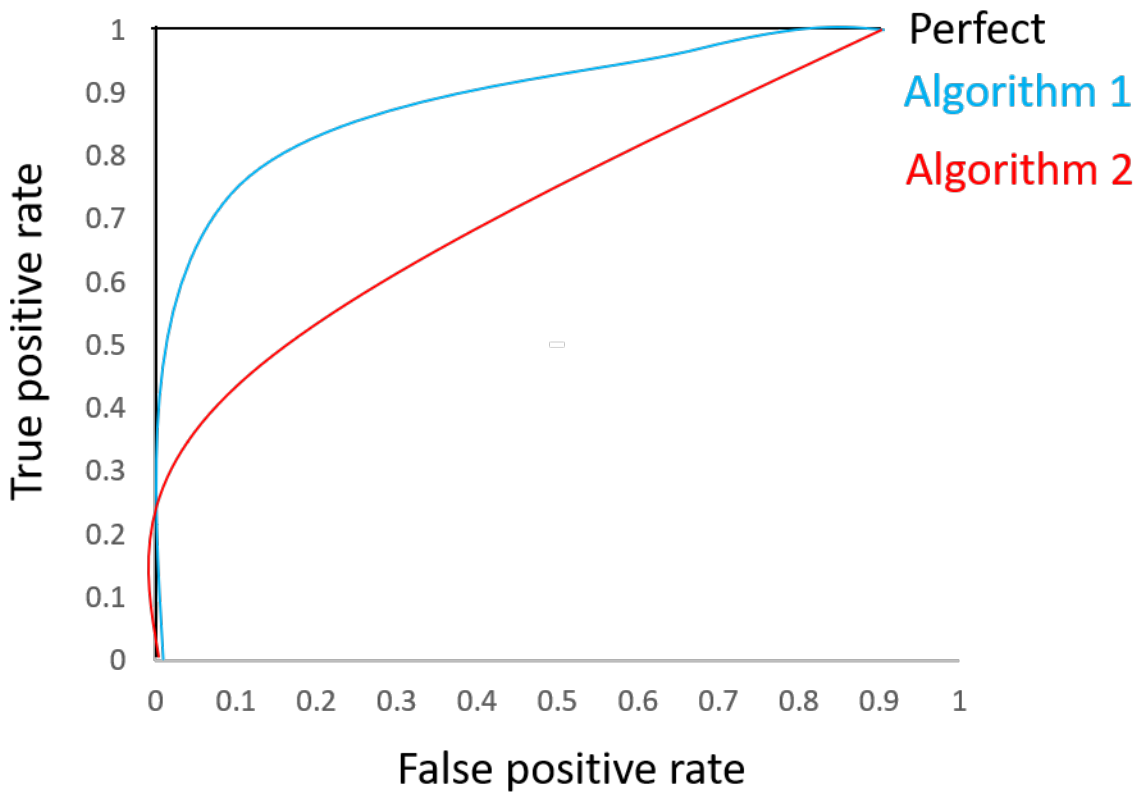
- Precision: of all the Run activities you found, how many were actually Run?
- Precision = TP/(TP+FP)

- Recall: of all of the Runs that exist, how many did you find?
- Recall = TP/(TP+FN) = TP/P

Precision / Recall curve



ROC curve


Perfect
Algorithm 1
Algorithm 2
True positive rate
False positive rate
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

F-Measure

- $F = \frac{2 \times P \times R}{P + R}$
- Combines Precision and Recall

	0.0	0.2	0.4	0.6	0.8	1.0
0.0	0.00	0.00	0.00	0.00	0.00	0.00
0.2	0.00	0.20	0.26	0.30	0.32	0.33
0.4	0.00	0.26	0.40	0.48	0.53	0.57
0.6	0.00	0.30	0.48	0.60	0.68	0.74
0.8	0.00	0.32	0.53	0.68	0.80	0.88
1.0	0.00	0.33	0.57	0.74	0.88	1.00

F-Measure

- Note that f-measure is designed to evaluate binary classifiers

	0.0	0.2	0.4	0.6	0.8	1.0
0.0	0.00	0.00	0.00	0.00	0.00	0.00
0.2	0.00	0.20	0.26	0.30	0.32	0.33
0.4	0.00	0.26	0.40	0.48	0.53	0.57
0.6	0.00	0.30	0.48	0.60	0.68	0.74
0.8	0.00	0.32	0.53	0.68	0.80	0.88
1.0	0.00	0.33	0.57	0.74	0.88	1.00

Example

Class 1

	Correct Yes	Correct No
Predicted Yes	10	10
Predicted No	10	970

Class 2

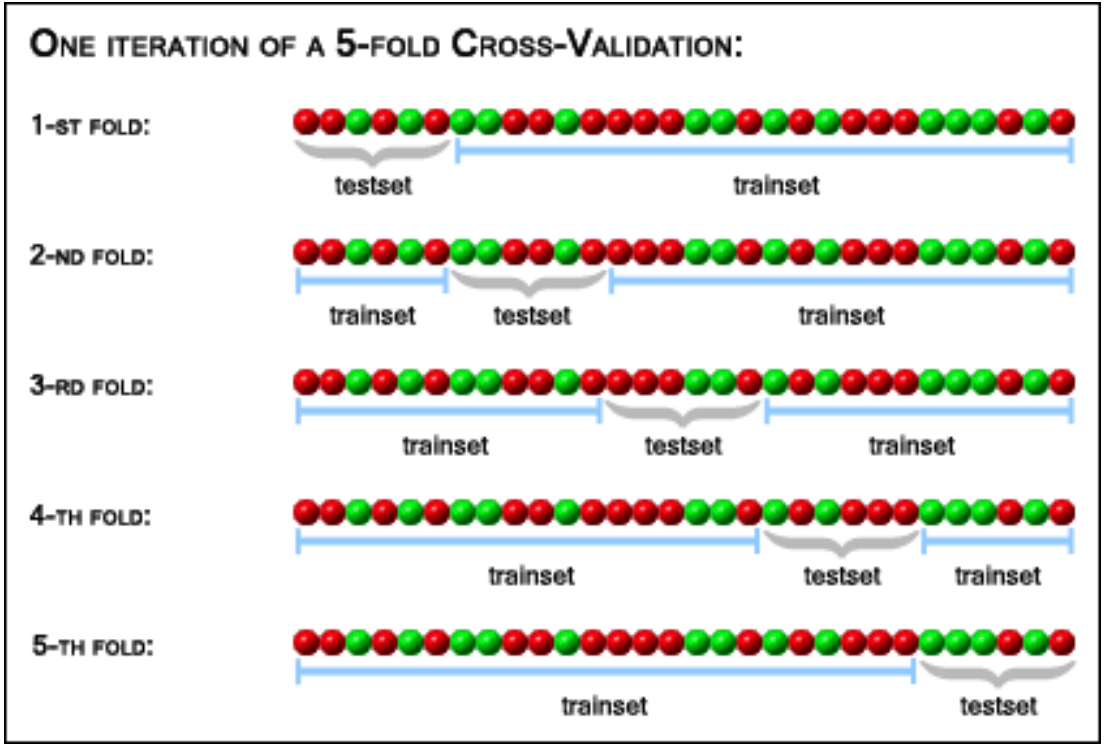
	Correct Yes	Correct No
Predicted Yes	90	10
Predicted No	10	890

Micro Average

	Correct Yes	Correct No
Predicted Yes	100	20
Predicted No	20	1860

"But before we move on, allow me to belabor the point even further..."

Cross-Validation



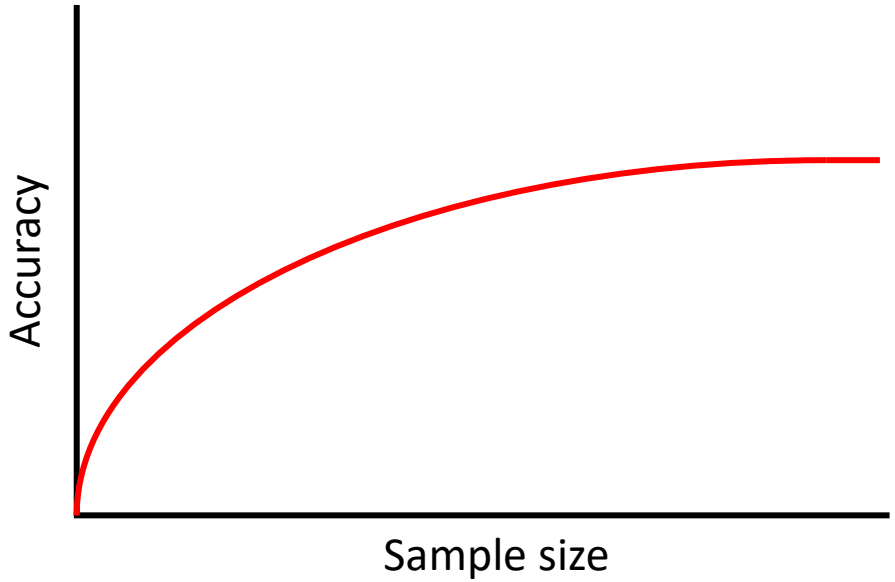
Algorithm 8 CROSSVALIDATE(*LearningAlgorithm*, *Data*, K)

1: $\hat{\epsilon} \leftarrow \infty$ // store lowest error encountered so far
2: $\hat{\alpha} \leftarrow$ unknown // store the hyperparameter setting that yielded it
3: **for all** hyperparameter settings α **do**
4: *err* $\leftarrow$ [ ] // keep track of the K-many error estimates
5: **for** $k = 1$ **to** K **do**
6: $train \leftarrow \{(x_n, y_n) \in Data : n \mod K \neq k-1\}$
7: $test \leftarrow \{(x_n, y_n) \in Data : n \mod K = k-1\}$ // test every Kth example
8: *model* $\leftarrow$ Run *LearningAlgorithm* on *train*
9: *err* $\leftarrow$ *err* $\oplus$ error of *model* on *test* // add current error to list of errors
10: **end for**
11: *avgErr* $\leftarrow$ mean of set *err*
12: **if** *avgErr* $< \hat{\epsilon}$ **then**
13: $\hat{\epsilon} \leftarrow$ *avgErr* // remember these settings
14: $\hat{\alpha} \leftarrow \alpha$ // because they're the best so far
15: **end if**
16: **end for**

Algorithm 9 KNN-TRAIN-LOO(**D**)

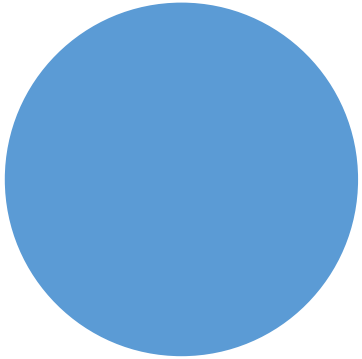
$err_k \leftarrow o, \forall 1 \leq k \leq N-1$ // err_k stores how well you do with kNN
for $n = 1$ **to** N **do**
  $S_m \leftarrow \langle ||\boldsymbol{x}_n - \boldsymbol{x}_m||, m \rangle, \forall m \neq n$ // compute distances to other points
  $S \leftarrow \text{SORT}(S)$ // put lowest-distance objects first
  $\hat{y} \leftarrow 0$ // current label prediction
  for $k = 1$ **to** $N-1$ **do**
    $\langle dist, m \rangle \leftarrow S_k$
    $\hat{y} \leftarrow \hat{y} + y_m$ // let kth closest point vote
    if $\hat{y} \neq y_m$ **then**
      $err_k \leftarrow err_k + 1$ // one more error for kNN
    end if
  end for
end for
return $\text{argmin}_k \, err_k$ // return the K that achieved lowest error

Learning curve



Other considerations

Lets try these out




Practical ML

Beginning to work on a machine learning project

Project

- Team
- Every project
  - Experimental results
  - Validate hypothesis/method

- Design of new variation on method
- Implementation of method discussed in class from ground-up
- Application of class techniques to a new problem
- Real working system (app) to solve ML task
- Experimental comparison of alternative techniques

Techniques Not Discussed in Class

- Semi-supervised learning
- Incremental decision tree
- Forecasting (stock market, weather, HAR-based activities)
- Collaborative filtering
- Daume chapters 16-18

Grading criteria (100 points)

- Proposal
  - 10 points
  - Submitted as part of HW #4
- Poster
  - 30 points
  - Presented last week of class
  - Will have google docs signup sheet
- Project
  - 40 points
  - Working code with video demo
  - Due day of final exam
- Extras
  - 20 points
  - Project scope and completeness
  - Creativity
  - Teamwork

Pick a dataset

Pick a dataset

Feature engineering

- Transform raw data into features

> *The algorithms we used are very standard for Kagglers. […] We spent most of our efforts in feature engineering.*

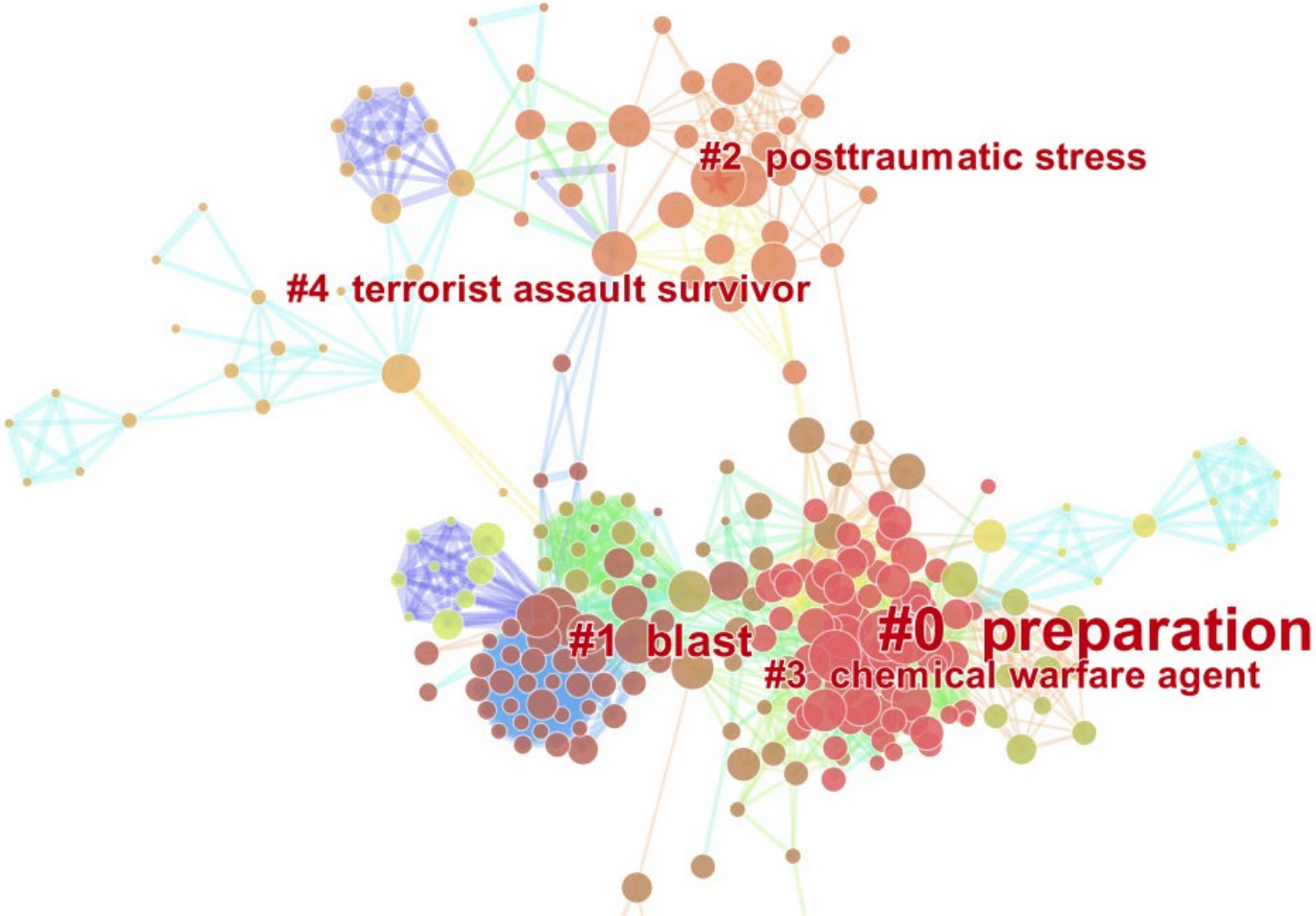
— Xavier Conort, on "Q&A with Xavier Conort" on winning the Flight Quest challenge on Kaggle

Decide on performance measure

Test a few diverse machine learning algorithms

Get creative!

Tell a story


#2 posttraumatic stress
#4 terrorist assault survivor
#1 blast
#0 preparation
#3 chemical warfare agent

Practical applications