

Tying up some loose ends

Multi-class classification, Model Selection, and Evaluation

CS534

MULTICLASS CLASSIFICATION

Adapted from slides by Ben Taskar and Simon Lacoste-Julien

Basic Classification in ML

Input

$\mathbf{x} \in \mathcal{X}$

Output

$y \in \mathcal{Y}$

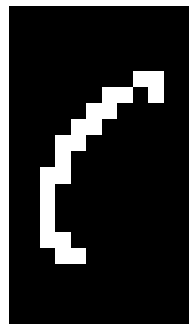
Spam
filtering



Binary



Character
recognition



Multi-Class

C

Multi-Class Classification

- Multi-class classification : direct approaches

Perceptron	Logistic regression	SVM	Naïve Bayes	Decision Tree	Neural Network
?	Softmax $P(y = i x) = \frac{\exp w_i^T x}{\sum_j \exp w_j^T x}$?	No change $P(y = i) \& P(x y = i)$ $i = 1, \dots, K$	No change	K output nodes with softmax activation

- Multi-class classification through binary classification
 - One-vs-All
 - All-vs-all
 - ...

Linear classification

- Each class has a parameter vector (w_k, b_k)
- x is assigned to class k iff $w_k^\top x + b_k \geq \max_j w_j^\top x + b_j$
- For simplicity set $b_k = 0$ (add the bias dimension in x and add b_k to w_k)
- So learning goal given separable data: choose w_k s.t.

$$\forall (x^i, y^i), \quad w_{y^i}^\top x^i \geq \max_j w_j^\top x^i$$

K-class Perceptron and SVM

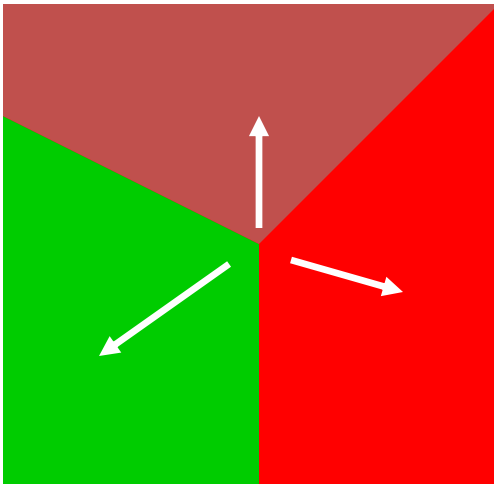
Perceptron: $\max_W \sum_i \left[w_{y^i}^\top x^i - \max_k w_k^\top x^i \right]$

K-class SVM: $\max_W \sum_i \left[w_{y^i}^\top x^i - \max_k (w_k^\top x^i + \mathbf{1}\{k \neq y^i\}) \right]$

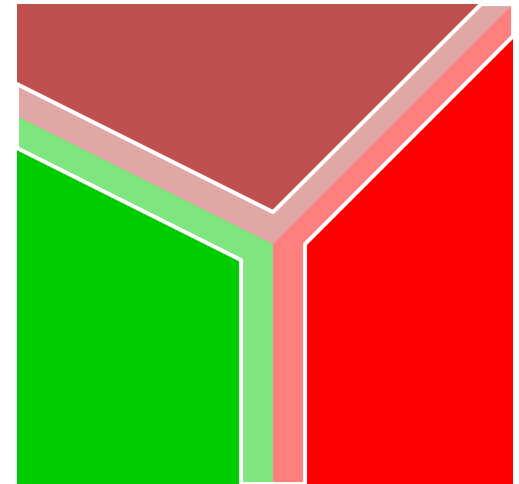
Without regularization

Geometry of Linear classification

Perceptron



K-class SVM



Multiclass Perceptron

Online: for each datapoint

$$\text{Predict: } \hat{y}_i = \arg \max_y w_y^\top x^i$$

$$\begin{aligned} \text{Update: if } \hat{y}_i \neq y^i \text{ then} \\ \left\{ \begin{aligned} w_{y^i, t+1} &= w_{y^i, t} + \alpha x^i \\ w_{\hat{y}_i, t+1} &= w_{\hat{y}_i, t} - \alpha x^i \end{aligned} \right. \end{aligned}$$

Multi-class SVM

Primal problem: QP

$$\begin{aligned} \min_{w_1, \dots, w_K} \quad & \frac{1}{2} \|(w_1, \dots, w_K)\|^2 + C \sum_{ik} \xi_{ik} \\ \text{s.t.} \quad & \forall(i, k), \quad w_{y^i}^\top x^i - w_k^\top x^i \geq 1\{k \neq y^i\} - \xi_{ik} \end{aligned}$$

Again can be solved in the dual formulation, also Quadratic Program

Combining binary classifiers

One-vs-all For each class build a classifier for that class vs the rest

- Often very imbalanced classifiers --- often challenging to learn a good classifier

All-vs-all build a classifier for each pair of classes

- A priori a large number of classifiers $\binom{n}{2}$ to build **but...**
 - The pairwise classification are way much faster
 - The classifications are balanced

... so that in many cases it is faster than one-vs-all

MODEL SELECTION

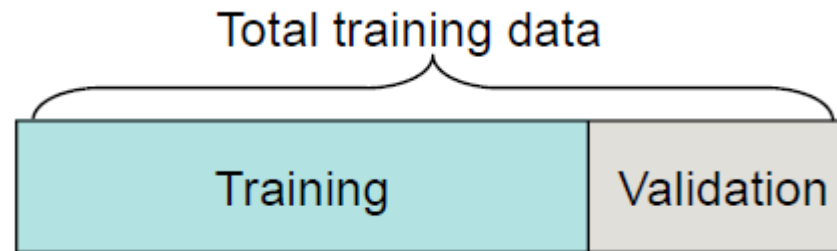
General Model Selection Problem

- Assume that we have a set of models $M = \{M_1, M_2, \dots, M_d\}$ that we are trying to select from. Some examples include:
 - **Feature Selection:** each M_i corresponds to using a different feature subset from a large set of potential features
 - **Algorithm Selection:** each M_i corresponds to an algorithm, e.g., Naïve Bayes, Logistic Regression, DT ...
 - **Parameter selection/tuning:** each M_i corresponds to a particular parameter choice, e.g., the choice of kernel and C for SVM

Approaches to Model Selection

- **Holdout and Cross-validation methods**
 - Experimentally determine when overfitting occurs and which model works well on the specific data in hand
- Penalty methods - regularization based, introducing penalty on complex models
 - MAP Penalty
 - Minimum Description Length – an alternative approach to MAP
- You can also avoid model selection and use ensembles
 - Instead of choosing, consider many possibilities and let them vote, or learn to combine their decisions

Simple Holdout Method

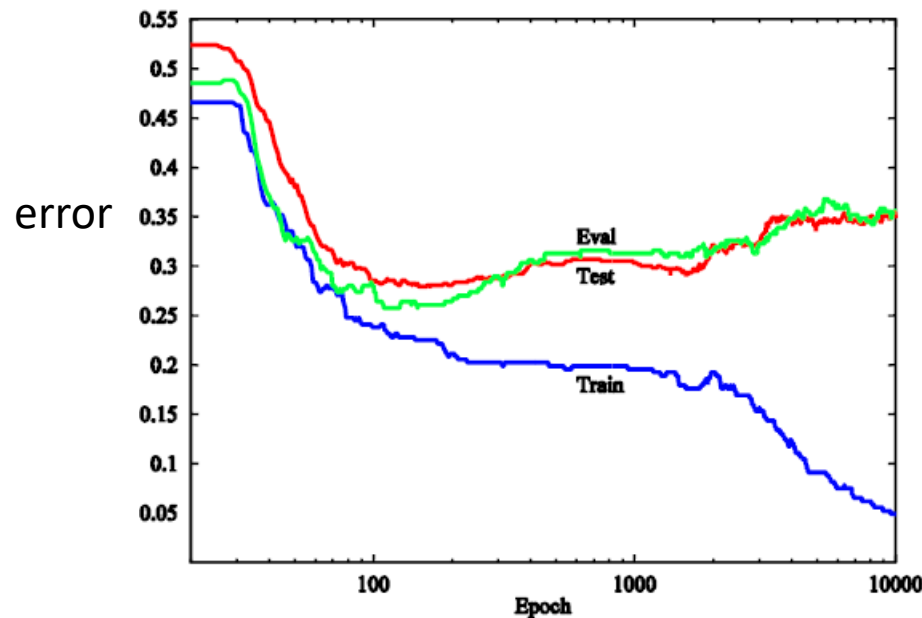


1. Divide training set S into S_{train} and S_{valid}
2. Train each model M_i on S_{train} to get a hypothesis h_i
3. Choose and output h_i with the smallest error rate on S_{valid}

Could retrain the selected model on the whole dataset to get the final hypothesis h - this will improve the original h_i because of more training data

Notes on hold-out methods

- Hold-out method often used for choosing among nested hypotheses:
 - Deciding # of training epochs for Neural net
 - Deciding when to stop growing or pruning a decision tree
 - Deciding when to stop growing an ensemble



Example:

Selecting # of epochs for neural net

Issues

- It wastes part of the data
 - The model selection choice is still made using only part of the data
 - Still possible to overfit the validation data since it is a relatively small set of data
- To address these problems, we can use **Cross-Validation**

k-fold Cross-validation

- Partition (randomly) S into k disjoint subsets S_1, \dots, S_k (preferably in a class-balanced way)
- To evaluate model M_j :

for $i=1:K$

1. Train M_j on $S \setminus S_i$ (S removing S_i) $\rightarrow h_{ji}$

2. Test h_{ji} on $S_i \rightarrow \epsilon_j(i)$

End for

$$\epsilon_j = \frac{1}{K} \sum_i \epsilon_j(i)$$

- Select model that minimizes the error:

$$M^* = \operatorname{argmin}_{M_j} \epsilon_j$$

- Train M^* on S and output resulting hypothesis

Cross-validation visualized

Available Labeled Data

Identify k partitions

Fold 1

Train

Train

Train

Train

Train

Val

ϵ_1

Cross-validation visualized

Available Labeled Data

Identify k partitions

Fold 2

Train

Train

Train

Train

Val

Train

ϵ_2

Cross-validation visualized

Available Labeled Data

Identify k partitions

Fold 3

Train

Train

Train

Val

Train

Train

ϵ_3

Cross-validation visualized

Available Labeled Data

Identify k partitions

Fold 4

Train

Train

Val

Train

Train

Train

ϵ_4

Cross-validation visualized

Available Labeled Data

Identify k partitions

Fold 5

Train

Val

Train

Train

Train

Train

ϵ_5

Cross-validation visualized

Available Labeled Data

Identify k partitions

Fold 6

Val

Train

Train

Train

Train

Train

ϵ_6

Comments on k-fold Cross-Validation

- Computationally more expensive than simple hold-out method but better use of data
 - Every data point in the training set is used in validating the model selection choices
- If the data is really scarce, we can use the extreme choice of $k = |S|$
 - Each validation set contains only one data point
 - Often referred to as **Leave-one-out (LOO) cross-validation**

EVALUATION OF CLASSIFIERS

How do we evaluate a classifier?

- What measure to use?
 - Accuracy
 - Precision, recall, F-Measure
 - ROC Curves
- What sampling method to use?
 - Train/test split
 - Cross-validation
 - Leave-one-out
- How to tell if the difference is significant?

Performance Measure: Accuracy

- Easily the most common and intuitive measure of classification performance.

$$Accuracy = \frac{\#correct}{N}$$

Confusion matrix

		True Values	
		Positive	Negative
Pred. Values	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Definition: Accuracy

		True Values	
		Positive	Negative
Pred. Values	Positive	TP	FP
	Negative	FN	TN

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

Precision, Recall, F-measure

		True Values	
		Positive	Negative
Pred. Values	Positive	TP	FP
	Negative	FN	TN

- Precision: how many predicted positives were true positives

$$P = \frac{TP}{TP + FP}$$

- Recall: how many of the true positives were identified $R = \frac{TP}{TP + FN}$

- F-Measure: Harmonic mean of precision and recall $F = \frac{2PR}{P + R}$

What's wrong with Accuracy?

True class →	Pos	Neg
Pos	0	0
Neg	100	900
	P=100	N=900

True class →	Pos	Neg
Pos	90	90
Neg	10	810
	P=100	N=900

- Both classifiers obtain 90% accuracy
- They exhibit very different behaviors:
 - On the left: classifier completely give up on the positive class
 - On the right: classifier is able to catch some positive examples

What's wrong with Precision/Recall?

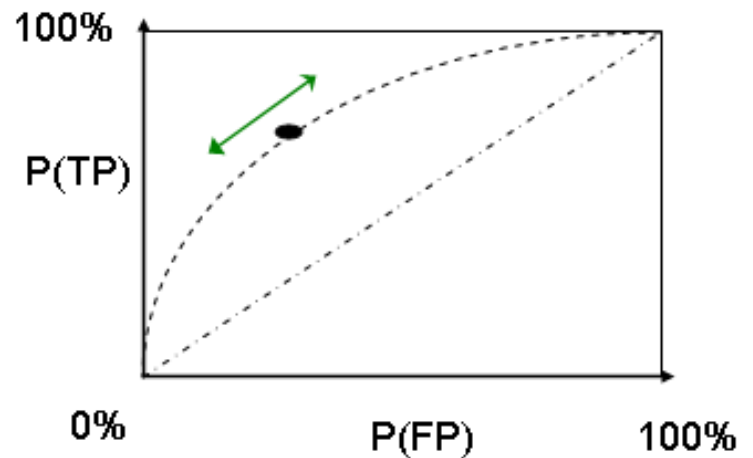
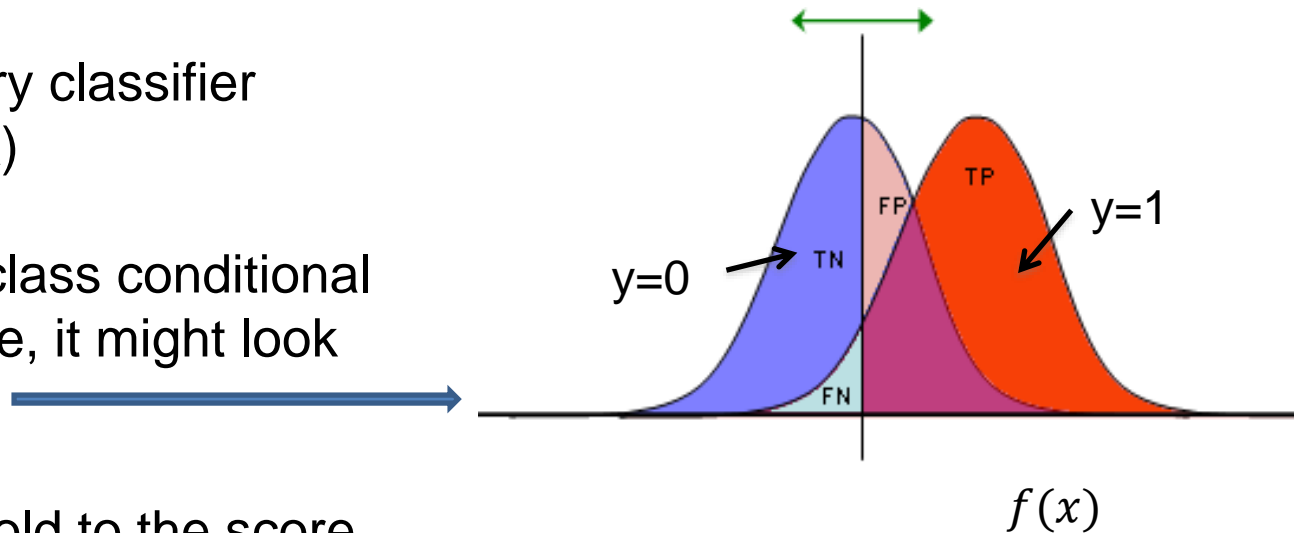
True class →	Pos	Neg
Pos	200	100
Neg	300	400
	P=500	N=500

True class →	Pos	Neg
Pos	200	100
Neg	300	0
	P=500	N=100

- Both classifiers obtain the same precision and recall values of 66.7% and 40%
- They exhibit very different behaviors:
 - Same positive recognition rate
 - Extremely different negative recognition rate: **strong** on the left / **nil** on the right
- Note: Accuracy has no problem catching this!

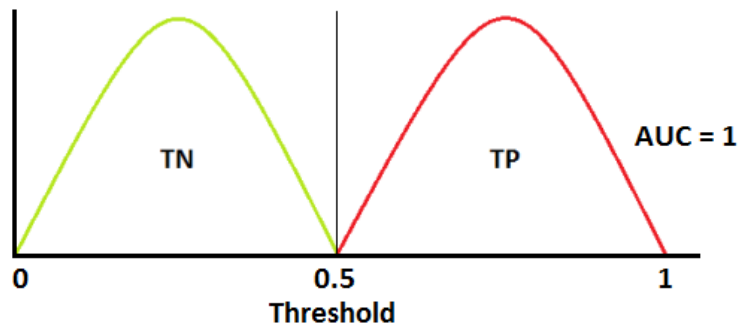
ROC Curve

- Assume your binary classifier outputs a score $f(x)$
- If we can plot the class conditional density of the score, it might look like this
- We apply a threshold to the score $f(x)$ for prediction: $y = I(f(x) > \theta)$
- As you change the value θ the true positive rate (TP) and the false positive rate (FP) will change.
- By plotting the entire curve you can see the tradeoffs.

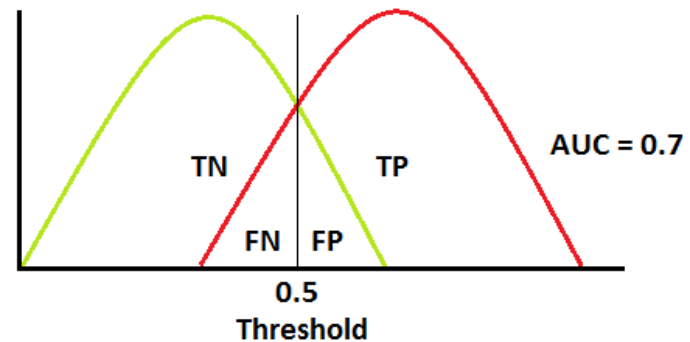


ROC measures how the learned score $f(x)$ separates positive from negative

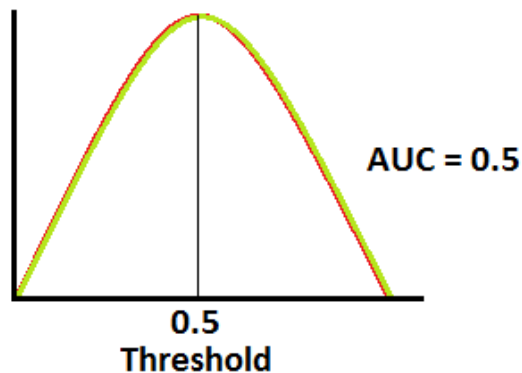
A. Complete separation



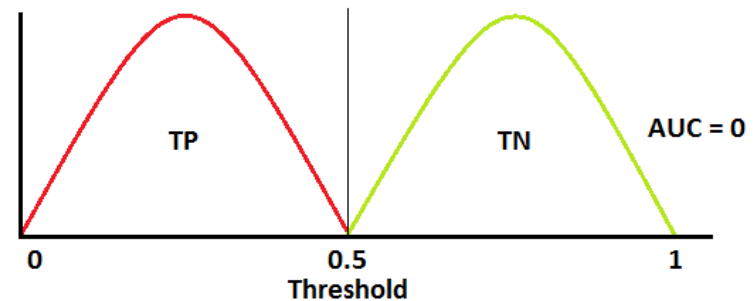
B. Some overlap



C. Complete overlap

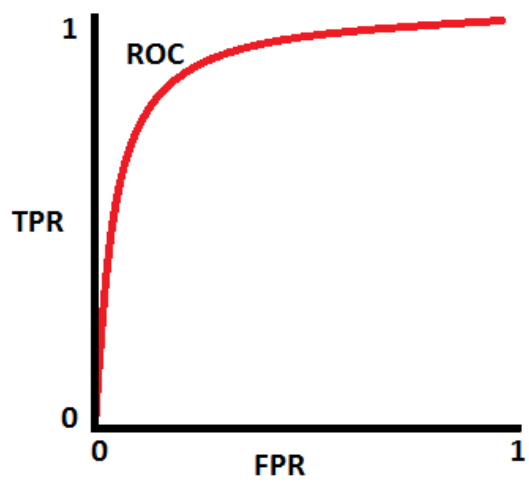


D. Separate but reversed

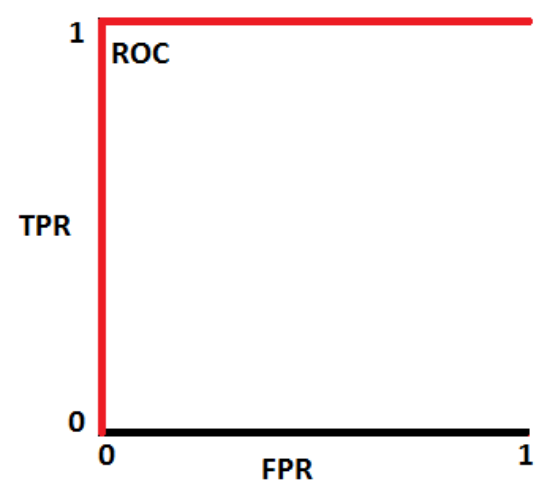


What do their corresponding ROC curves look like?

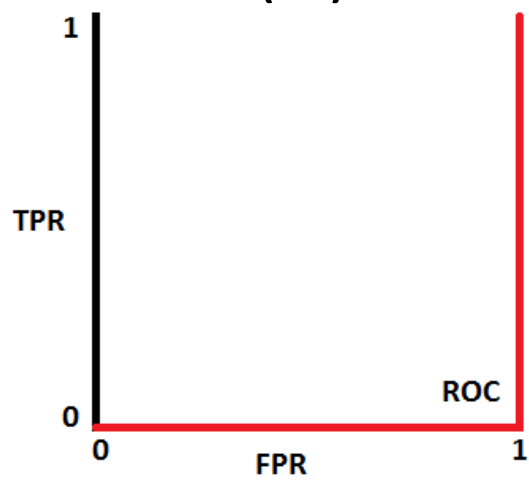
(i)



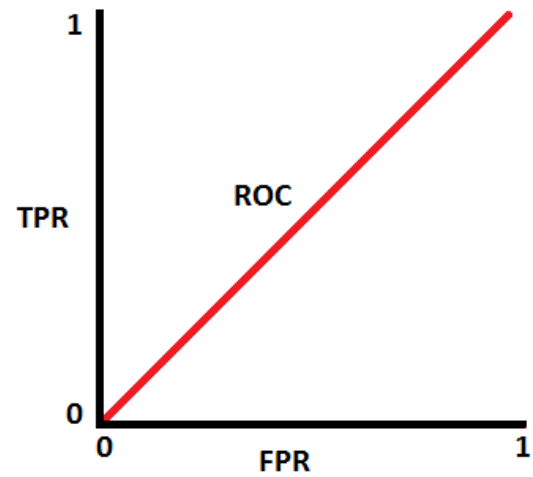
(ii)



(iii)



(iv)



Area under the ROC curves

- Area Under the Curve (AUC) measures the area under the ROC curve
- AUC = the probability that a randomly selected positive example will score higher than a randomly selected negative example
- AUC = 1 model achieves perfect separation
- AUC = 0.5 complete overlap, equivalent to random guessing
- AUC = 0 completely reversed

Comparing classifiers statistically

- Say I have two classifiers.
- A = 50% accuracy
- B = 75% accuracy
- B is better, right?

Comparing classifiers statistically

- Say I have another two classifiers
- A = 50% accuracy
- B = 50.5% accuracy
- Is B better?

Basic Evaluation

- Training data – used to identify model parameters
- Testing data – used for evaluation
- Optionally: Development / tuning data – used to identify model hyperparameters.
- Difficult to get significance or confidence values

k -fold Cross validation

- Partition the available data into k folds
- Train on $k - 1$ folds
- Test on the remaining fold.
- In the extreme $k = N$, this is known as “**leave-one-out**” (LOO) cross validation
- k -fold cross validation gives k samples of the performance of the classifier.

Cross-validation visualized

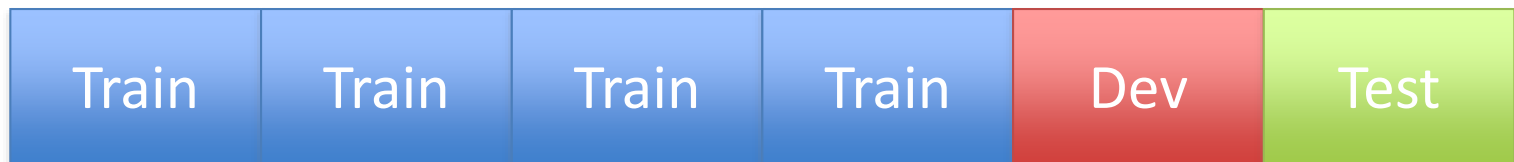
Available Labeled Data



Identify k partitions



Fold 1



Cross-validation visualized

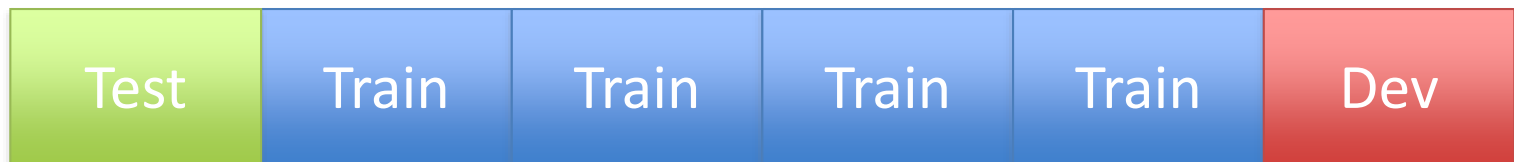
Available Labeled Data



Identify k partitions



Fold 2



Cross-validation visualized

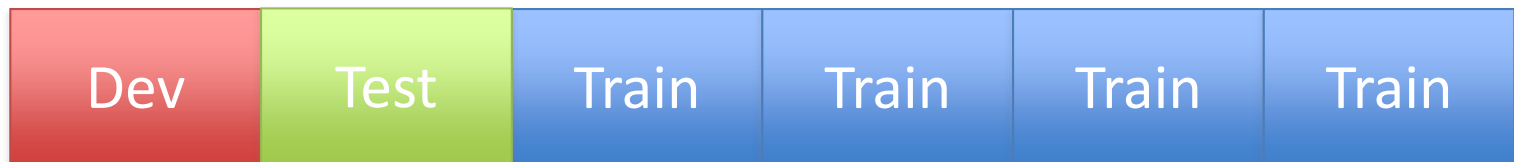
Available Labeled Data



Identify k partitions



Fold 3



Cross-validation visualized

Available Labeled Data



Identify k partitions



Fold 4



Cross-validation visualized

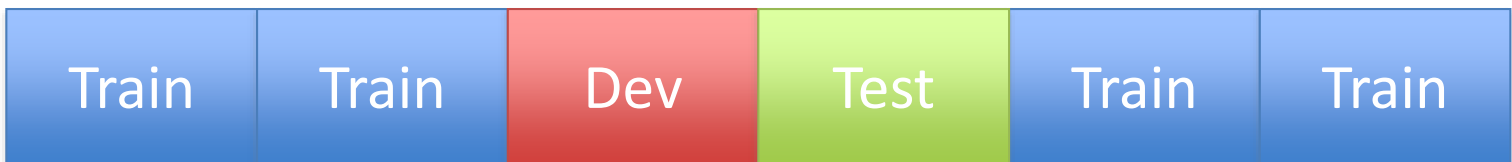
Available Labeled Data



Identify k partitions



Fold 5



Cross-validation visualized

Available Labeled Data



Identify k partitions



Fold 6

Train

Train

Train

Dev

Test

Train



Calculate Average Performance

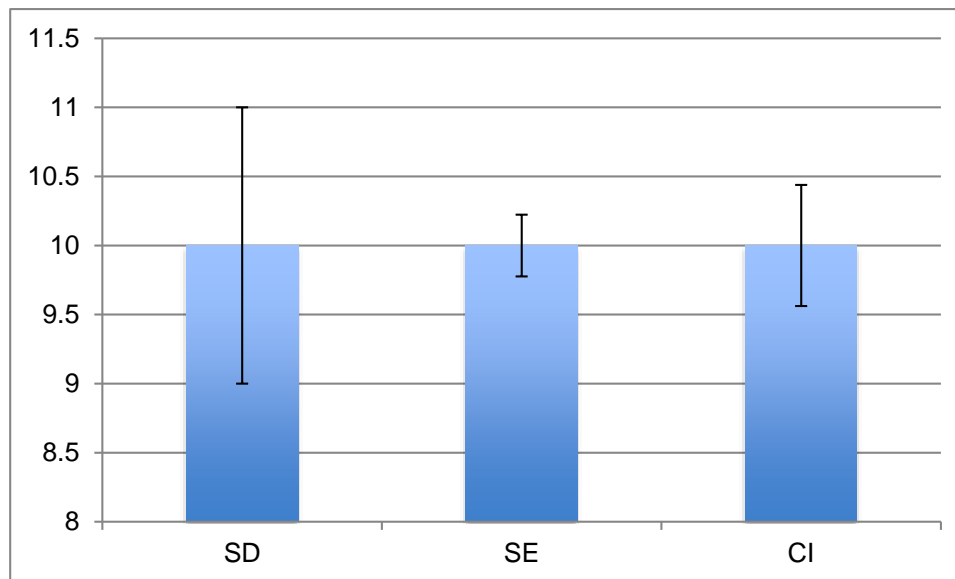
Confidence interval

- Given k samples of performance measures, we can compute the mean of the performance, μ
- While mean is important, the variance information should also be included
 - Standard deviation (SD): σ
 - Standard error (SE): $\frac{\sigma}{\sqrt{k}}$
 - Confidence interval ($CI_{\alpha\%}$)

$$CI_{95\%} = \mu \pm 1.96 * \frac{\sigma}{\sqrt{k}}$$

Confidence Bars

- Variance information is often represented as an error bar or confidence bar around the mean performance
- 95% confidence interval has the clearest interpretation.



Criticisms of cross-validation

- While the test data is independently sampled, there is a lot of overlap in training data.
 - The model performance may be correlated.
 - **Underestimation** of variance.
 - **Overestimation** of significant differences.
- One proposed solution* is to repeat **2-fold** cross-validation 5 times rather than **10-fold** cross validation
 - It more directly measures variation due to the choice of training set

*Dietterich, Thomas G. "Approximate statistical tests for comparing supervised classification learning algorithms." *Neural computation* 10.7 (1998): 1895-1923.

Significance Testing

- Is the performance of two classifiers different with statistical significance?
- Testing the means
 - If we have two samples of classifier performance (accuracy), we want to determine if they are drawn from the same distribution (Null hypothesis: no difference) or two different distributions.
- Statistical testing is extensively studied in the stat 561, 562, 563 sequence

Baseline Classifiers

- Majority Class baseline
 - Every data point is classified as the class that is most frequently represented in the training data
- Random baseline
 - Randomly assign one of the classes to each data point.
 - with an even distribution
 - with the training class distribution

Proper Evaluation: summary

- How do you measure performance?
 - Accuracy, precision/recall/F1, ROC/AUC --- often it is good to look at multiple measures and carefully think about which measure makes sense for your application
- How certain are you about the performance you claim?
 - Include variance information using SD, SE or Confidence interval
- How do you compare two algorithms and decide if one performs better than another?
 - Consider multiple runs of the algorithm on different samples, e.g., Cross-validation, or bootstrapping, and consider statistical testing when appropriate
- What algorithm you should compare to?
 - In addition to state-of-the-art, it is also important to compare against some dummy baselines