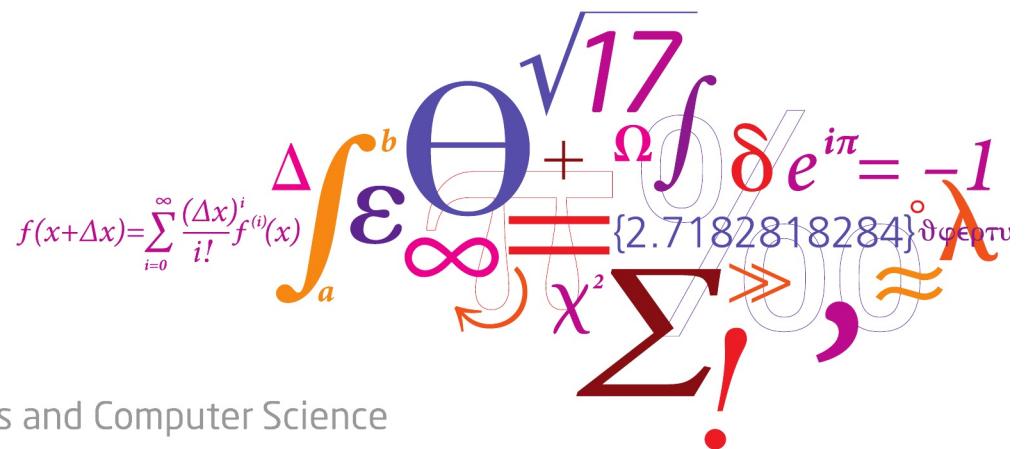


02450: Introduction to Machine Learning and Data Mining

AUC and ensemble methods

Tue Herlau

DTU Compute, Technical University of Denmark (DTU)

$$f(x+\Delta x) = \sum_{i=0}^{\infty} \frac{(\Delta x)^i}{i!} f^{(i)}(x)$$


DTU Compute

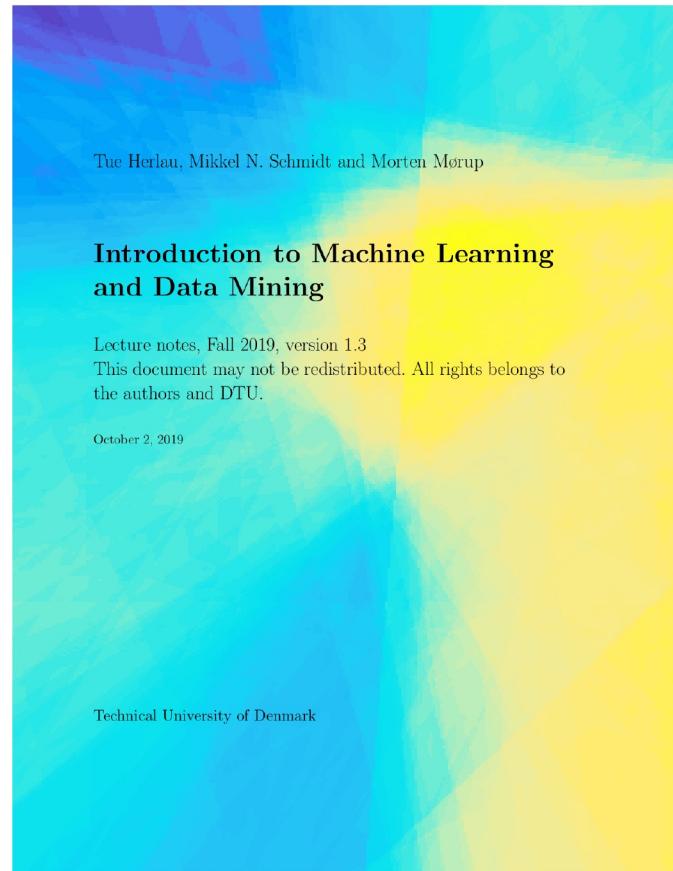
Department of Applied Mathematics and Computer Science

Today

Feedback Groups of the day:

Rögnvaldur Ingvi Örnolfsson, Yuwen Fan, Xin Lu Guo, Jakob Hartvig Stiesmark, Mathias Høxbro Juel Vendt, Oscar Laskowski, Kenneth Scheel, Zoey Lynn McKinley, Laia Meseguer Monfort, Tobias Kragholm, Paula Rodriguez Garcia, Johannes Boe Reiche, Simon Lucas Jonker, Malthe Andreas Lejbølle Jelstrup, Cara Ditmar, Christopher Lyke, Tyler Brody, Sükrü Han Sahin, Altug Tosun, Michael Minh Phan, William Schjelderup, Weston Jones, Michael Rupprecht, Jerome Pierre Alexandre Pinti, August Thomas Hjortshøj Schreyer, Lucas Koefod Schnügger, Oliver Orla Nielsen, Anders Vingaard Sehested, Mitul Vekariya, Hjalte Reuss Schmidt, Tobias Udbjørg Marcussen, Lukas Leindals, Gustav Gamst Larsen, Gabriella Angela Morote Kierulff, Asger Sturis Tang, Priya Sapra, Felix Pacheco Pastor, Laura Sans Comerma, Dagh Mikael Einer Nielsen, Paul Gerard R Seghers, Marius-Cristian Mic, Ali Mohebbi, Philip Frisch Møller, Morten Mørup, Tahany Nemér

Reading material: Chapter 16, Chapter 17



Lecture Schedule

1 Introduction

3 September: C1

Data: Feature extraction, and visualization

2 Data, feature extraction and PCA

10 September: C2, C3

3 Measures of similarity, summary statistics and probabilities

17 September: C4, C5

4 Probability densities and data Visualization

24 September: C6, C7

Supervised learning: Classification and regression

5 Decision trees and linear regression

1 October: C8, C9 (Project 1 due before 13:00)

6 Overfitting, cross-validation and Nearest Neighbor

8 October: C10, C12

7 Performance evaluation, Bayes, and Naive Bayes

22 October: C11, C13

Piazza online help: <https://piazza.com/dtu.dk/fall2019/02450>

Videos/streaming of lectures: <https://video.dtu.dk>

8 Artificial Neural Networks and Bias/Variance

29 October: C14, C15

9 AUC and ensemble methods

5 November: C16, C17

Unsupervised learning: Clustering and density estimation

10 K-means and hierarchical clustering

12 November: C18 (Project 2 due before 13:00)

11 Mixture models and density estimation

19 November: C19, C20

12 Association mining

26 November: C21

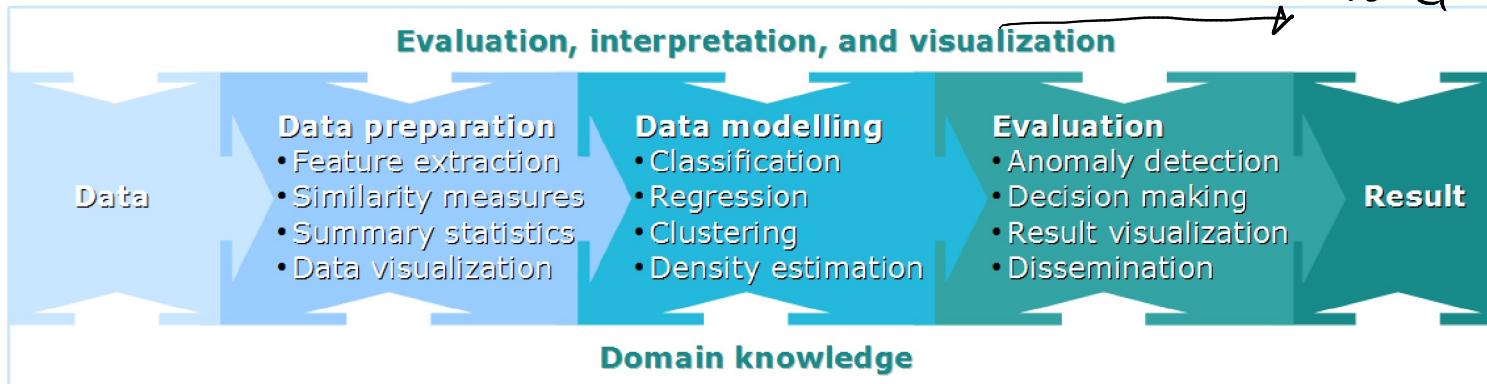
Recap

13 Recap and discussion of the exam

3 December: C1-C21 (Project 3 due before 13:00)

$$E(\lambda) = \sum_i \|y_i - \hat{y}_i\|^2 + \lambda \|w\|^2$$

↑ regularization.
linear reg. ANN (large)
↓ model complexity

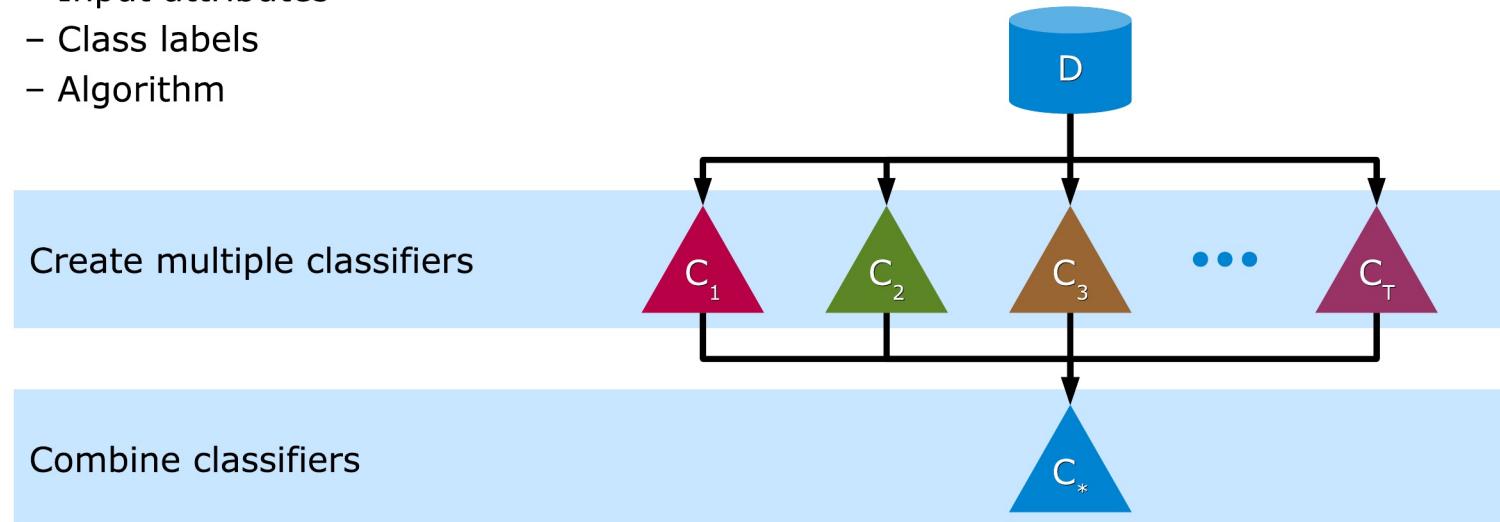


Learning Objectives

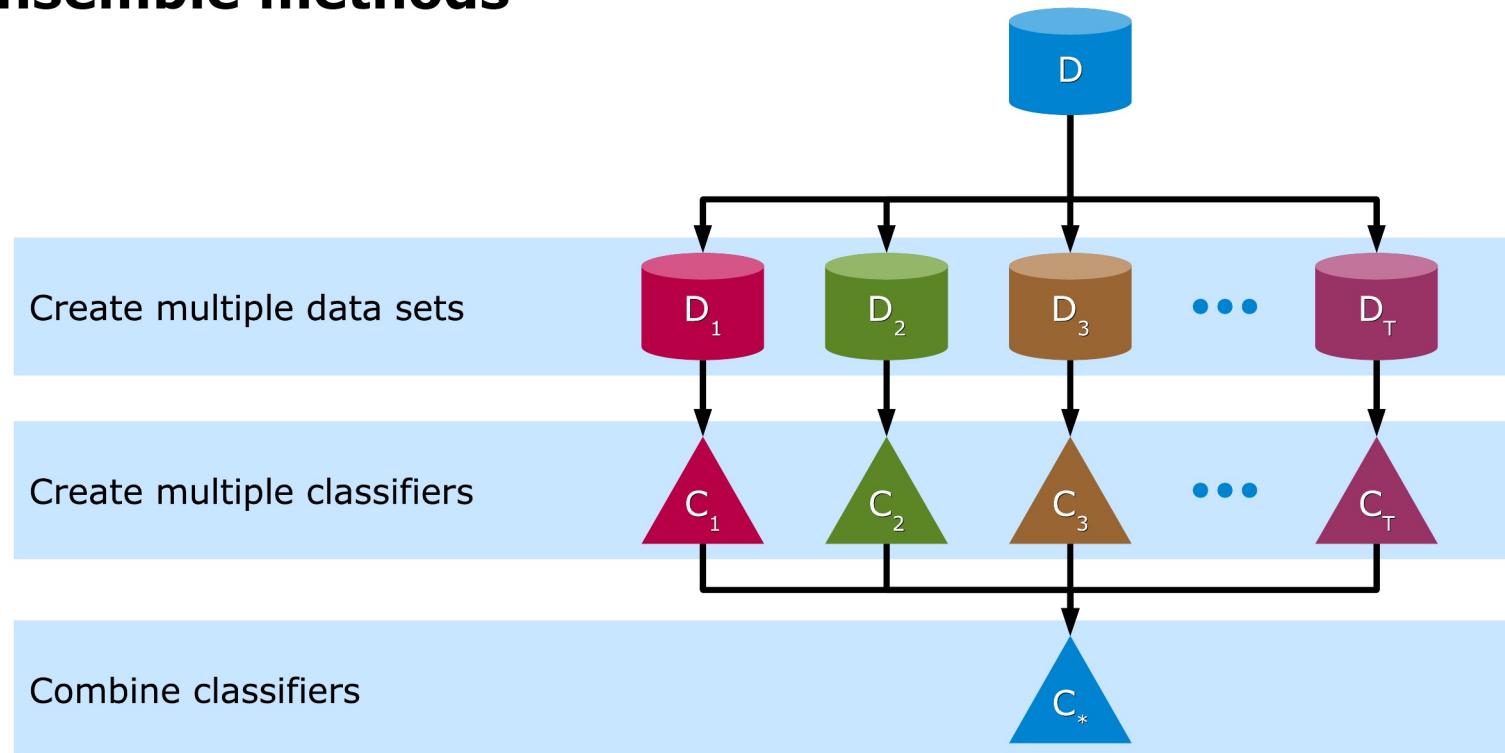
- Explain the principle behind boosting and bagging and apply it to improve classifiers
- Be able to address issues of class-imbalance and resampling
- Understand the definition of Precision, Recall, ROC, and AUC

Ensemble methods

- Combine multiple (weak) classifiers into one (strong) classifier
- Each classifier trained using different variations of
 - Data set
 - Input attributes
 - Class labels
 - Algorithm



Ensemble methods

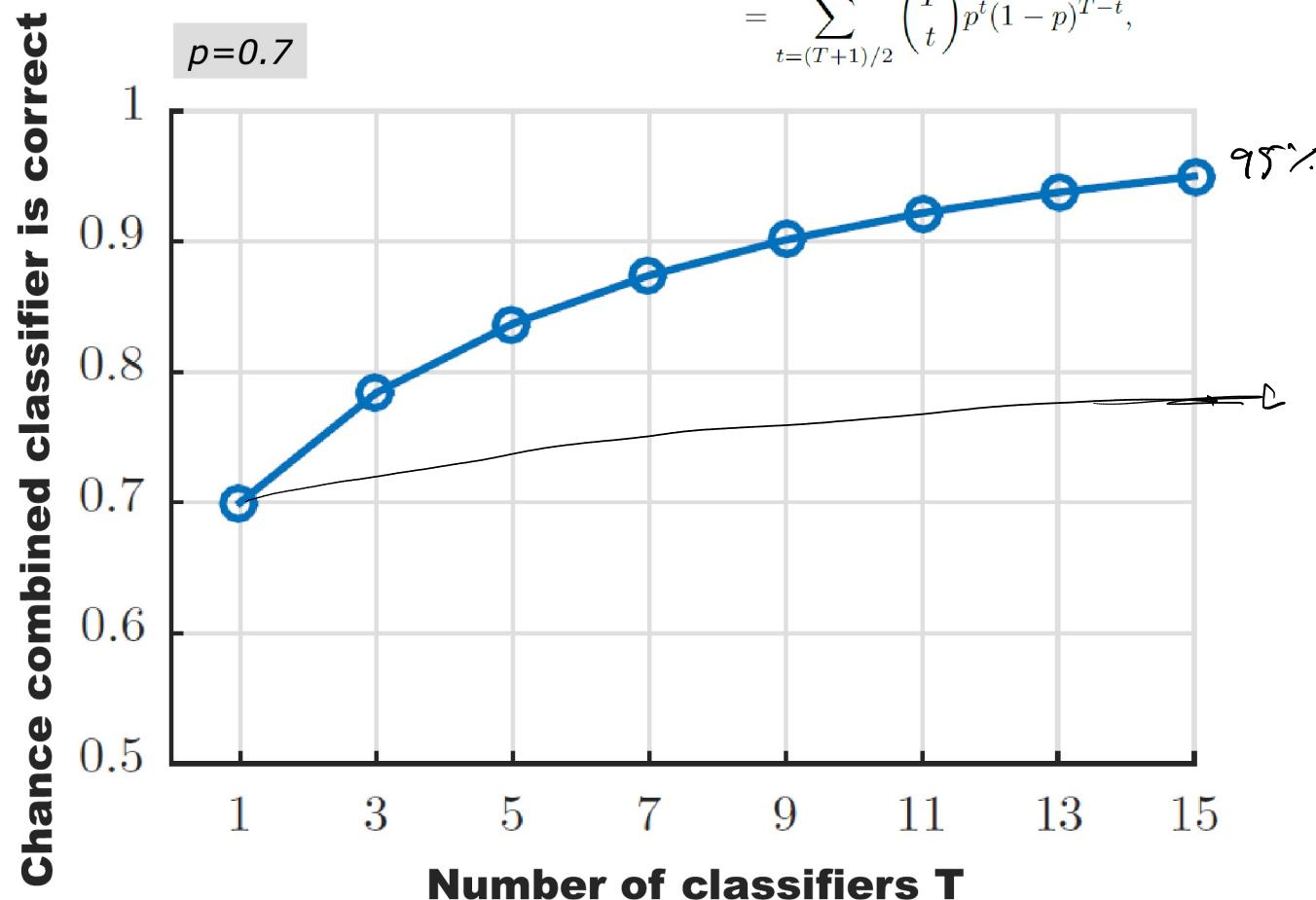


Why ensemble methods?

- Can improve classification algorithms in terms of
 - Better classification accuracy
 - Increased stability
 - Reduced variance
 - Less overfitting
- Consider T independent classifiers for binary classification, each with accuracy p .
The probability a classifier which use majority voting is correct is then given by:

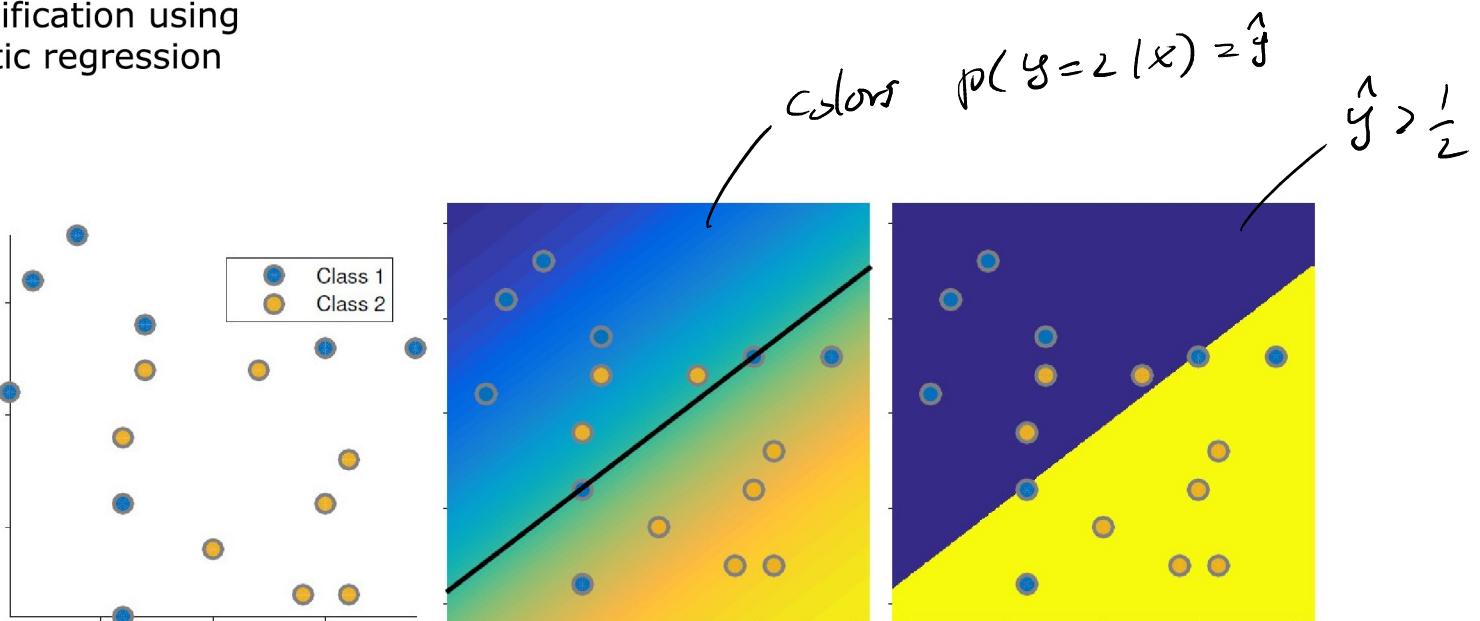
$$\begin{aligned}
 P(\text{Majority voting is correct}) &= \sum_{t=(T+1)/2}^T P\{t \text{ of the classifiers are correct}\} \\
 &= \sum_{t=(T+1)/2}^T \binom{T}{t} p^t (1-p)^{T-t},
 \end{aligned}$$

$$\begin{aligned}
 P(\text{Majority voting is correct}) &= \sum_{t=(T+1)/2}^T \{t \text{ of the classifiers are correct}\} \\
 &= \sum_{t=(T+1)/2}^T \binom{T}{t} p^t (1-p)^{T-t},
 \end{aligned}$$



Data example

- Classification using logistic regression

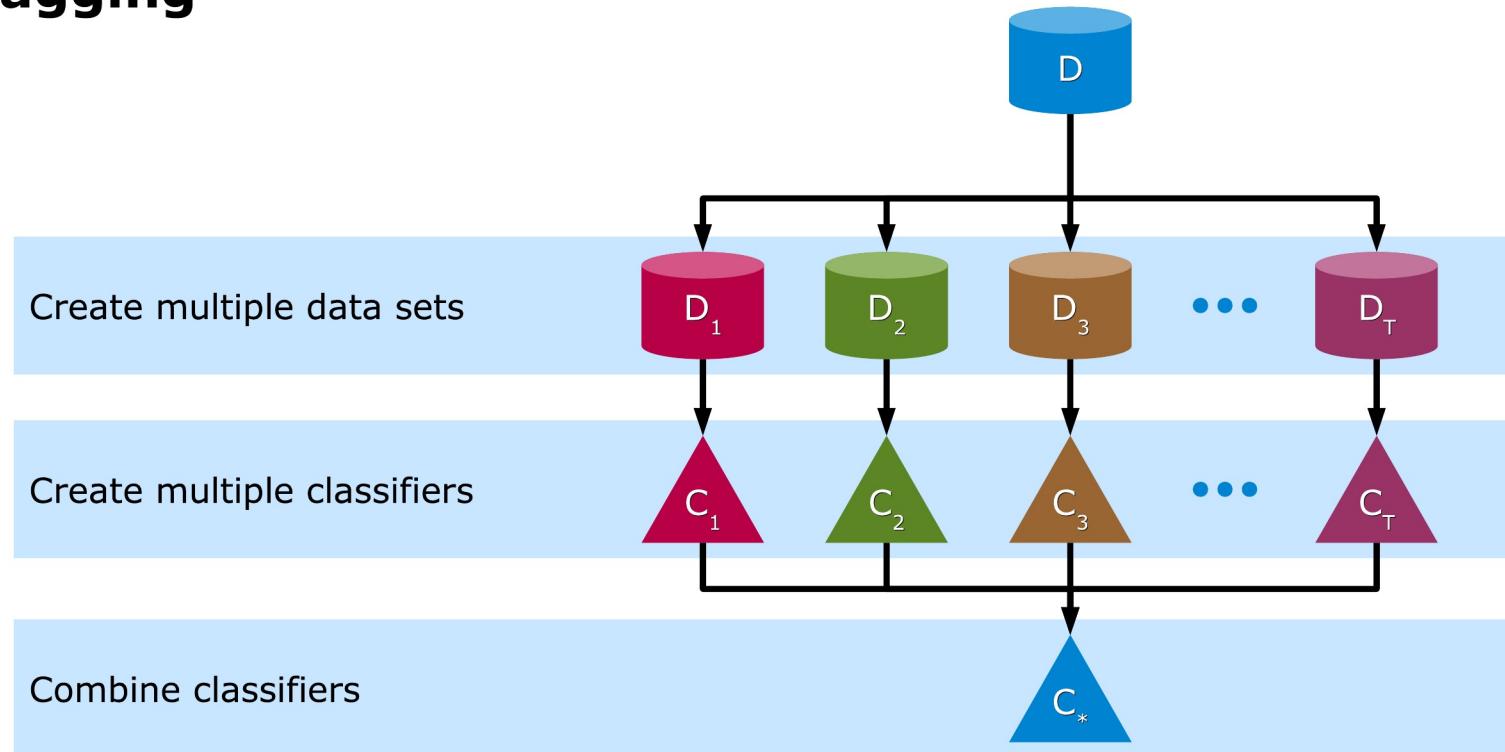


Bagging

- New training data sets drawn randomly from pool with replacement

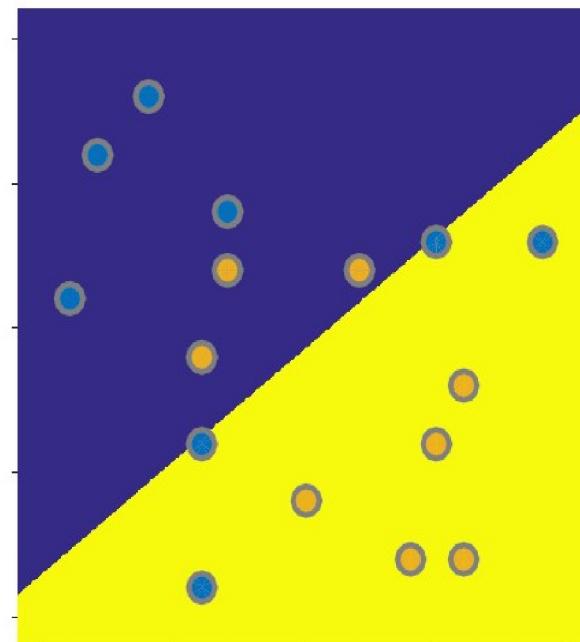
Pool of training data	1	2	3	4	5	6	7	8	9	10
	3	5	4	3	9	7	9	5	1	1
	5	8	2	6	2	3	8	3	5	1
New training data sets	1	7	4	1	10	6	10	8	8	7
	4	3	8	5	2	4	7	10	10	8
	⋮									

Bagging

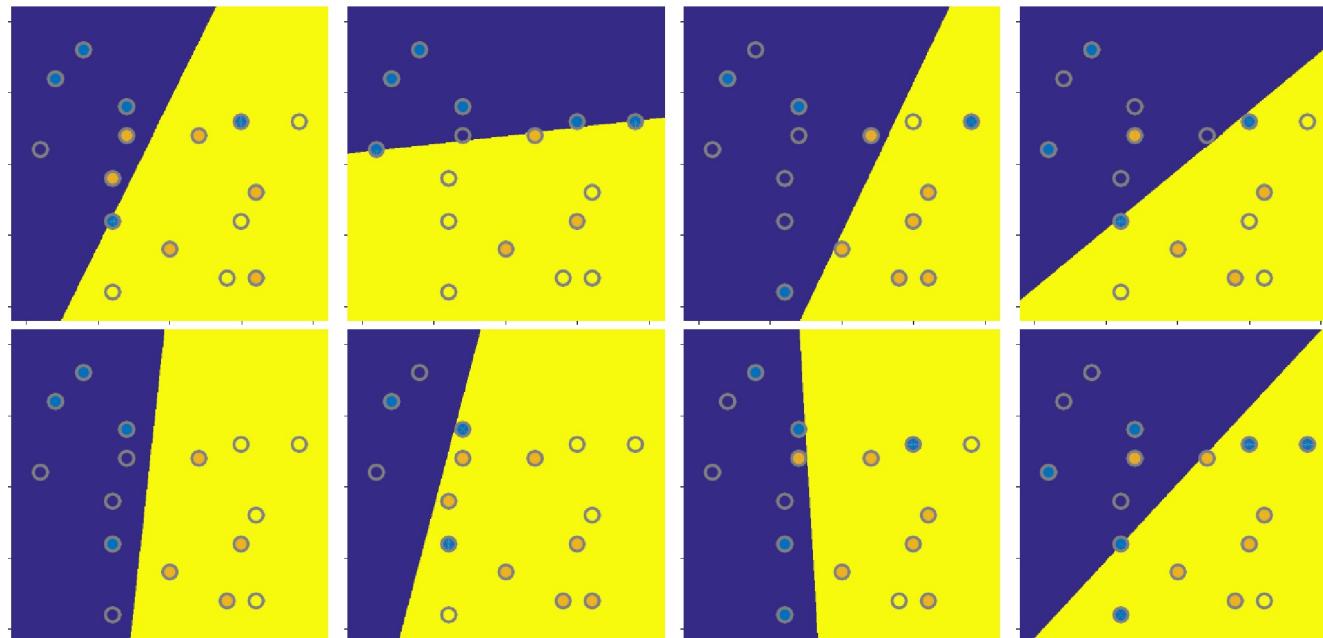


Bagging

- **Single classifier**
 - Logistic regression
 - Two features, (x, y)



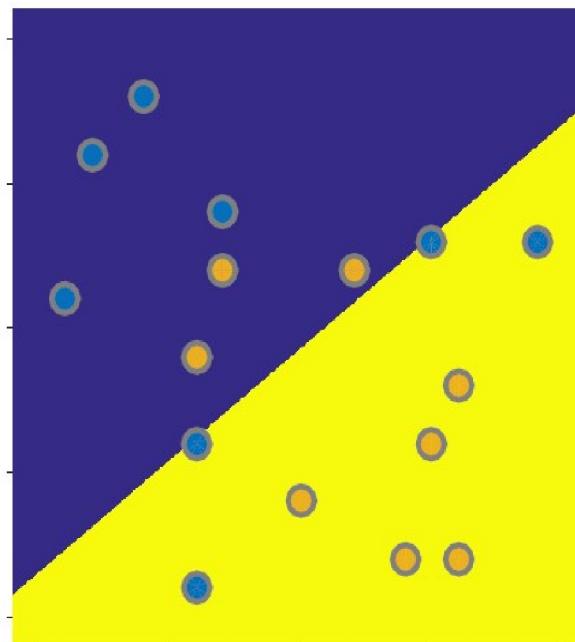
Bagging



Notice, hollow dots are observations not included in bagging round

Bagging

- Single classifier



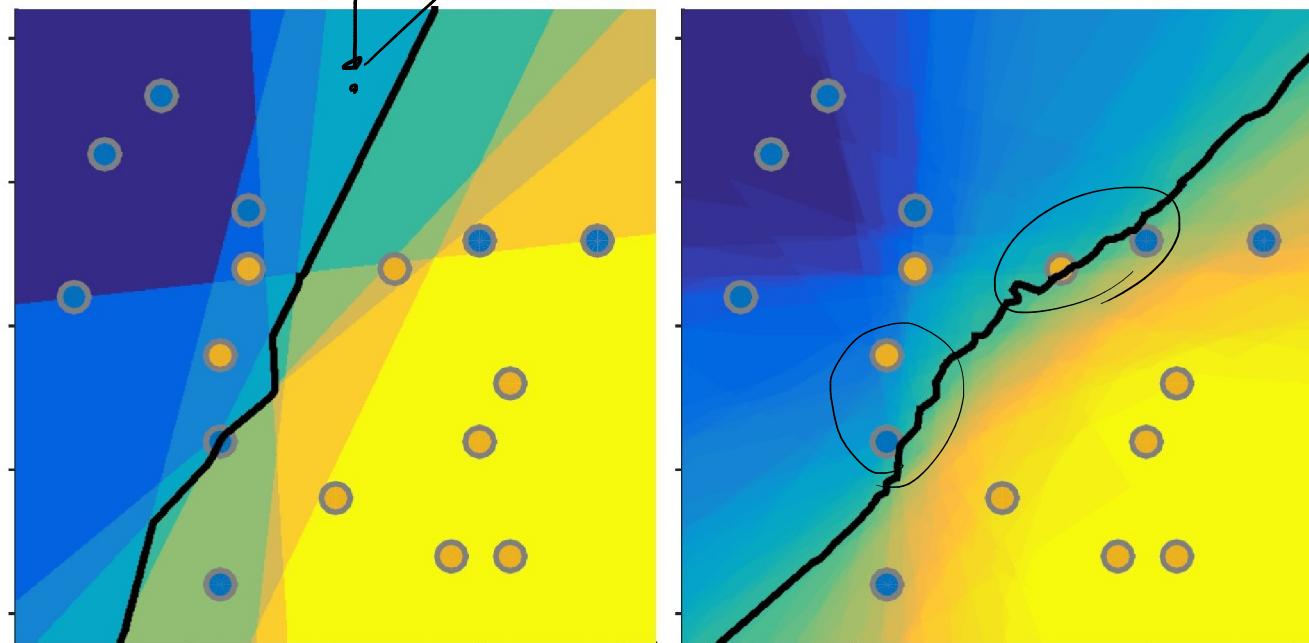
Bagging

$$z_k = P(y|x, D_k) > \frac{1}{2}$$

$$\frac{1}{T} \sum_{k=1}^T z_k \geq z$$

T=8

T=100



Boosting

Pool of training data	1	2	3	4	5	6	7	8	9	10
Weights	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1
New training data set	3	5	4	3	9	7	9	5	1	1
Train classifier										

$$w_i = \frac{1}{N}$$

Boosting

Pool of training data	1	2	3	4	5	6	7	8	9	10
Weights	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1
New training data set	3	5	4	3	9	7	9	5	1	1
Train classifier	 C_1									
Classify all data objects	1✓	2✗	3✓	4✗	5✓	6✗	7✓	8✓	9✓	10✓

Boosting

Pool of training data	1	2	3	4	5	6	7	8	9	10
Weights	.1	.1	.1	.1	.1	.1	.1	.1	.1	.1
New training data set	3	5	4	3	9	7	9	5	1	1
Train classifier										
Classify all data objects	1✓	2✗	3✓	4✗	5✓	6✗	7✓	8✓	9✓	10✓
Update weights	.07	.17	.07	.17	.07	.17	.07	.07	.07	.07

Boosting

Pool of training data	1 2 3 4 5 6 7 8 9 10
Weights	.1 .1 .1 .1 .1 .1 .1 .1 .1 .1
New training data set	3 5 4 3 9 7 9 5 1 1
Train classifier	
Classify all data objects	1✓ 2✗ 3✓ 4✗ 5✓ 6✗ 7✓ 8✓ 9✓ 10✓
Update weights	.07 .17 .07 .17 .07 .17 .07 .07 .07 .07
New training data set	6 4 7 3 2 4 10 2 5 6
Train classifier	
	⋮

AdaBoost

Algorithm 6: AdaBoost algorithm

- 1: Initialize $w_i(1) = \frac{1}{N}$ for $i = 1, \dots, N$ $t=1$ (first round).
- 2: **for** $t = 1, \dots, T$ **do**
- 3: Create \mathcal{D}_t by sampling (with replacement) from \mathcal{D} according to $w(t)$
- 4: Let f_t be the classifier trained on \mathcal{D}_t
- 5: $\epsilon_t = \sum_{i=1}^N w_i(t) (1 - \delta_{f_t(\mathbf{x}_i), y_i})$ (weighted error of f_t on all data)
- 6: $\alpha_t = \frac{1}{2} \log \frac{1-\epsilon_t}{\epsilon_t}$ *β if class. correct, γ if wrong.*
- 7: For each i update weights using eq. (15.7):

$$w_i(t+1) = \frac{\tilde{w}_i(t+1)}{\sum_{j=1}^N \tilde{w}_j(t+1)}, \quad \tilde{w}_i(t+1) = \begin{cases} w_i(t)e^{-\alpha_t} & \text{if } f_t(\mathbf{x}_i) = y_i \\ w_i(t)e^{\alpha_t} & \text{if } f_t(\mathbf{x}_i) \neq y_i. \end{cases}$$

8: **end for**

9: $f^*(\mathbf{x}) = \arg \max_{y=1,2} \sum_{t=1}^T \alpha_t \delta_{f_t(\mathbf{x}), y}$ (Majority voting classifier)

$\boxed{1/1/1/1}$

$$\frac{1}{2} \log \frac{1-\frac{1}{2}}{\frac{1}{2}} = 0$$

evaluate $f_1 = \sum_{t=1}^T \alpha_t \delta_{f_t(\mathbf{x}), 1}$

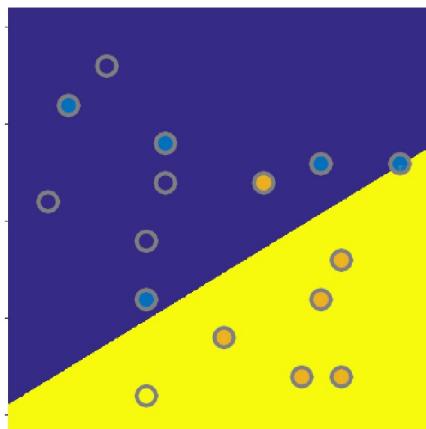
$f_2 = \sum_{t=1}^T \alpha_t \delta_{f_t(\mathbf{x}), 2}$

} if $f_1 > f_2 \rightarrow f^*(\mathbf{x}) = 1$
else $f^*(\mathbf{x}) = 2$.

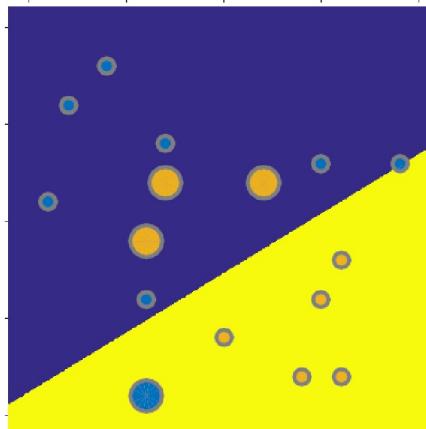
Boosting

$$w \in \mathbb{R}$$

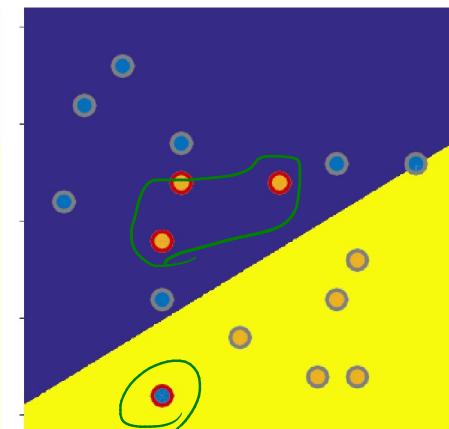
A:
A dataset is sampled with replacement and a classifier trained.



C:
Weights are updated such that more emphasis is given to these misclassified observations.

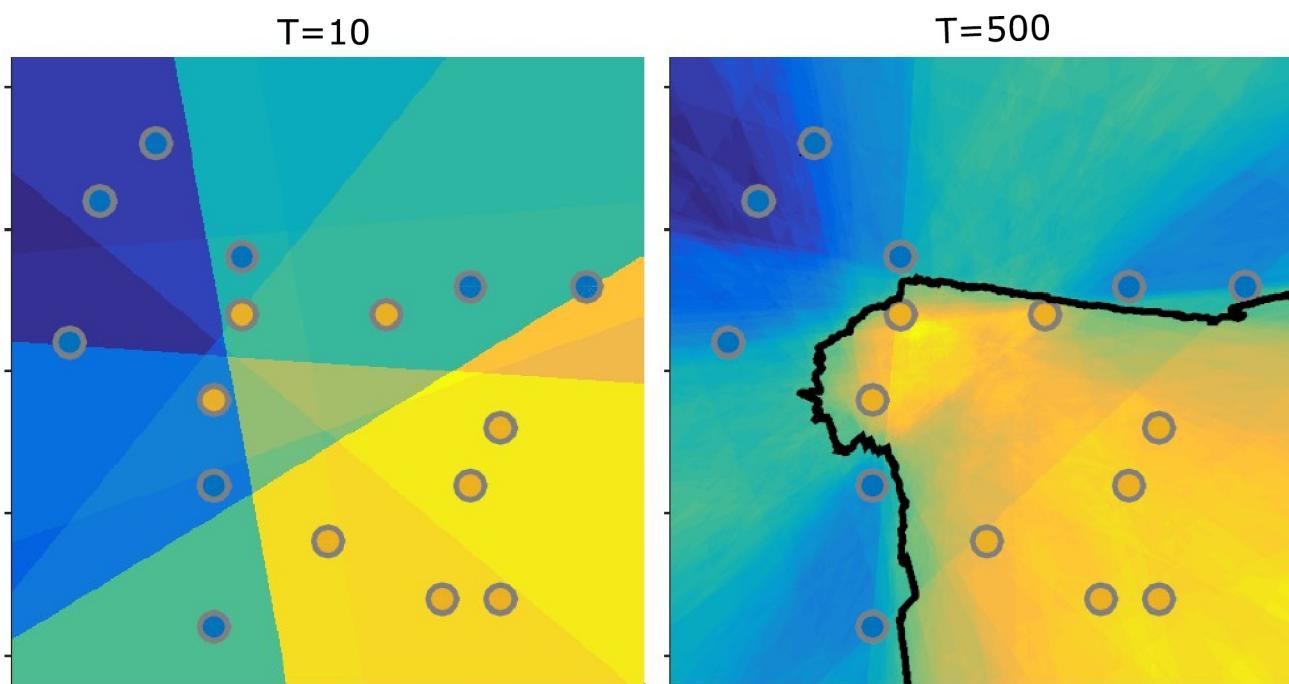


B:
Mis-classified observations are identified.

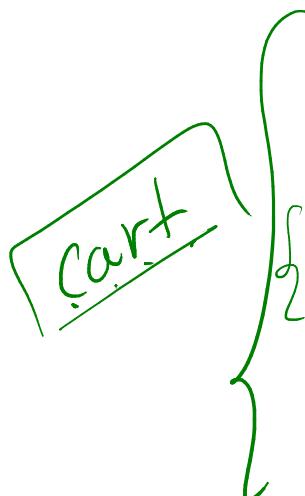


New round:
Based on the updated weights a new dataset is sampled and a classifier trained (shown), misclassified observations identified and given more emphasis...

Boosting



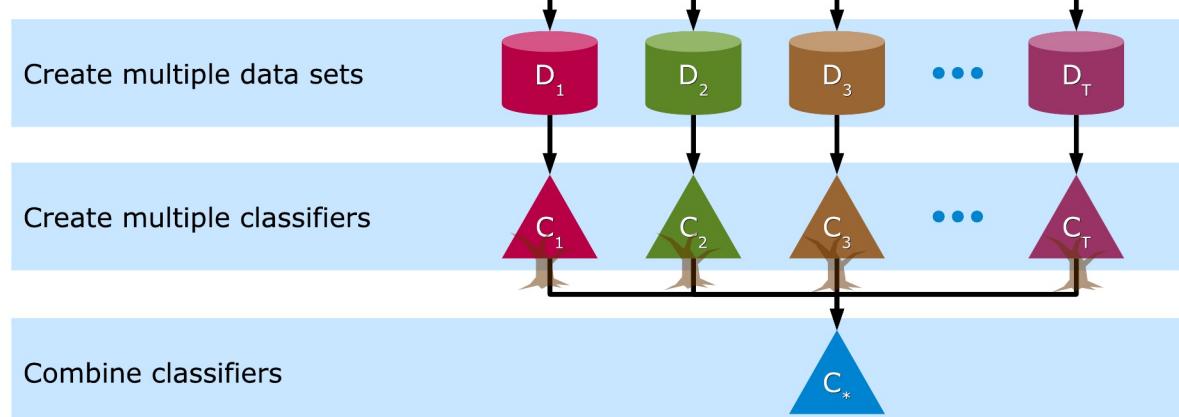
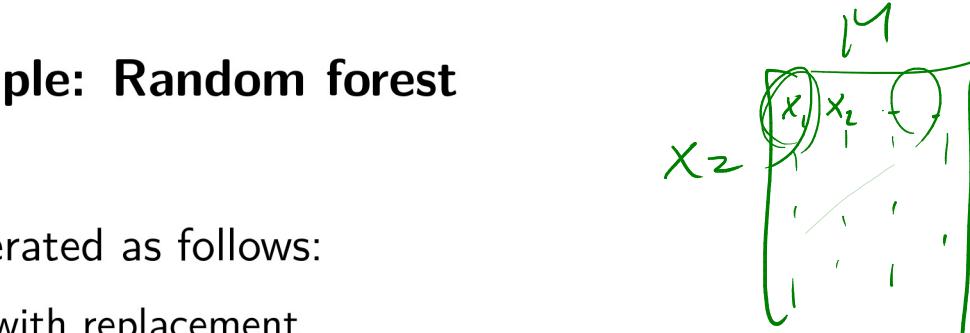
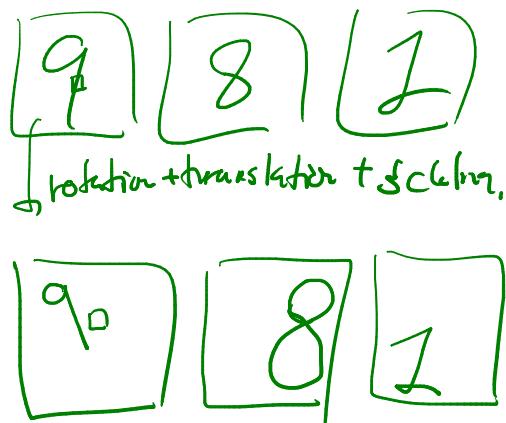
Bagging example: Random forest



Each tree is generated as follows:

- Sample dataset with replacement
- When generating each node in the tree, randomly select a subset of the features and only consider splits using these features

A large number of trees are generated and the trees are combined using majority voting (bagging)





Quiz 01 (please answer on Piazza): Adaboost (Spring 2016)

$$N=6, w_i = \frac{1}{N} = \frac{1}{6}$$

$$\varepsilon_1 = \sum_{i=1}^n w_i (1 - \delta_{f_1(x_i)}, s_i)$$

$$= \frac{1}{6} + \frac{1}{6} = \frac{1}{3}$$

$$\alpha_1 = \frac{1}{2} \log \frac{1 - \varepsilon_1}{\varepsilon_1}$$

$$= \frac{1}{2} \log \frac{\frac{2}{3}}{\frac{1}{3}} = \frac{1}{2} \log 2$$

*2 log √2
correctly
Vienna.*

$$\tilde{w} = [w_1 e^{\alpha_1}, w_2 e^{\alpha_1}, \dots]$$

$$= \frac{1}{6} \left[\frac{1}{\sqrt{2}}, \sqrt{2}, \dots \right]$$

$$\begin{aligned} & \frac{1}{\sqrt{2}} [1, 2, 1, 1, 1, 2] \\ & w = \underline{[1, 2, 1, 1, 1, 2]} \\ & z \end{aligned}$$

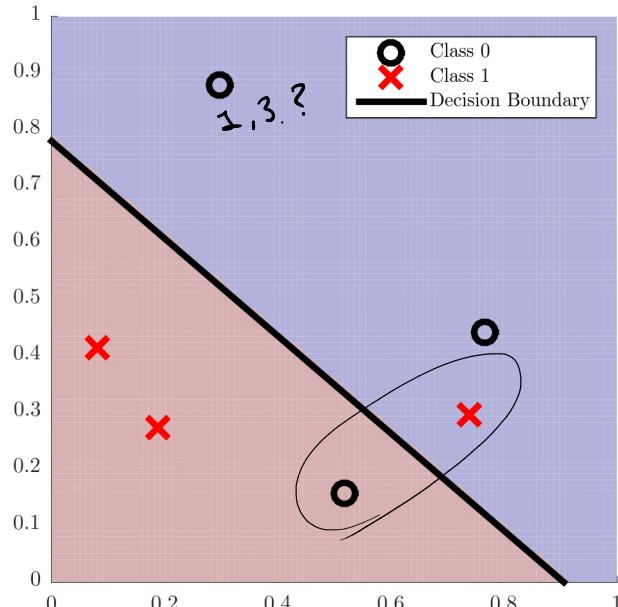


Figure 1: A binary classification problem and the decision boundary obtained by logistic regression. Observations left of the boundary are classified as belonging to the positive class 1 (red crosses) and observations right of the boundary to the negative class 0 (black circles)

- A. $w = [0.1 \quad 0.250 \quad 0.125 \quad 0.125 \quad 0.125 \quad 0.250]$
- B. $w = [0.026 \quad 0.447 \quad 0.026 \quad 0.026 \quad 0.026 \quad 0.447]$
- C. $w = [0.235 \quad 0.029 \quad 0.235 \quad 0.235 \quad 0.235 \quad 0.029]$
- D. $w = [0.1 \quad 0.3 \quad 0.1 \quad 0.1 \quad 0.1 \quad 0.3]$
- E. Don't know.

(Hint: First compute ε_1 , then α_1 , then the weights)

Solution:

The classifier classifies two out of $N = 6$ observations incorrectly. We therefore have:

$$\begin{aligned}\varepsilon_i &= \left[\sum_{j=1}^N w_j I(\hat{y}_j \neq y_j) \right] \\ \alpha_i &= \frac{1}{2} \log \left(\frac{1 - \varepsilon_i}{\varepsilon_i} \right)\end{aligned}$$

and accordingly $\varepsilon_1 = \frac{1}{N} \times 2 = \frac{1}{3}$. This gives

$$\alpha_1 = \frac{1}{2} \log \frac{1 - \frac{1}{3}}{\frac{1}{3}} = \frac{1}{2} \log 2$$

and so for \mathbf{w} we get

$$\mathbf{w} \propto [e^{-\alpha_1} \quad e^{\alpha_1} \quad e^{-\alpha_1} \quad e^{-\alpha_1} \quad e^{-\alpha_1} \quad e^{\alpha_1}]$$

Simplifying by moving $\frac{1}{\sqrt{2}}$ outside the vector:

$$\mathbf{w} \propto [1 \quad 2 \quad 1 \quad 1 \quad 1 \quad 2]$$

and normalizing:

$$\mathbf{w} = \frac{1}{8} [1 \quad 2 \quad 1 \quad 1 \quad 1 \quad 2]$$

accordingly option A is correct.

14:16

Class imbalance problem

- Many data sets have **imbalanced class distributions**
 - Example: Detection of defects that only occur rarely (e.g. 1/1,000,000)
 - Danger: Algorithm that says nothing is defect will be 99.999% correct
- **Solution approaches**
 - Resample to balance data sets
 - ↳ Modify existing classification algorithms)
 - Measure performance in a way that takes balance into account

Resampling balanced data

- New sample has equal number of data objects from each class
- **Approaches**
 - **Undersampling** majority class: Throws out potentially useful data
 - **Oversampling** minority class: Increase data size and computational burden
 - **Somewhere in between...**

Imbalanced training data	1	2	3	4	5	6	7	8	9	10
Oversampling	1	2	3	4	5	7	9	10	6	6
	6	6	8	8	8	8				
Undersampling	3	5	6	8						
Somewhere in between	3	5	4	3	9	6	6	8	8	8

Confusion matrix

$$\hat{y}_i = f(x_i)$$

		<i>Predicted</i>	
		<i>positive</i>	<i>negative</i>
<i>Actual</i>	<i>positive</i>	TP True Positive	FN False Negative
	<i>negative</i>	FP False Positive	TN True Negative

$$\text{acc.} = \frac{TP + TN}{N}$$

$$\text{err} = 1 - \text{acc.} = \frac{FN + FP}{N}$$

Precision and recall

- **Precision**

- Fraction of true positive among objects predicted to be positive

$$p = \frac{TP}{\underbrace{TP + FP}}$$

- **Recall**

- Fraction of objects predicted to be positive among all positive objects

$$r = \frac{TP}{TP + FN}$$

		<i>Predicted</i>	
		<i>positive</i>	<i>negative</i>
<i>Actual</i>	<i>positive</i>	TP	FN
	<i>negative</i>	True Positive	False Negative
		FP	TN
		False Positive	True Negative



Quiz 02 (please answer on Piazza): Precision/Recall

Consider two different classifiers, and suppose on a test set with 20 positive observations:

- Classifier 1 detects 54 positive of which 18 are actually positive
- Classifier 2 detects 16 positive of which 14 are actually positive

		Predicted	
		positive	negative
Actual	positive	TP	FN
	negative	FP	TN
positive	True Positive	False Negative	
	False Positive	True Negative	

What is the precision and recall of the two classifiers?

- A. Classifier 1: $p_1 = \frac{2}{3}, r_1 = \frac{7}{10}$
Classifier 2: $p_2 = \frac{1}{3}, r_2 = \frac{1}{3}$
 - B. Classifier 1: $p_1 = \frac{1}{3}, r_1 = \frac{9}{10}$
Classifier 2: $p_2 = \frac{2}{3}, r_2 = \frac{9}{10}$
 - C. Classifier 1: $p_1 = \frac{2}{3}, r_1 = \frac{7}{10}$
Classifier 2: $p_2 = \frac{1}{3}, r_2 = \frac{9}{10}$
 - D. Classifier 1: $p_1 = \frac{1}{2}, r_1 = \frac{9}{10}$
Classifier 2: $p_2 = \frac{7}{8}, r_2 = \frac{7}{10}$
- E : dont know.



Which classifier would you use if the objective was to detect credit-card fraud (the positive class corresponds to fraud)

• Precision

- Fraction of true positive among objects predicted to be positive

$$\text{Class. 1: } p = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{18}{54} = \frac{1}{3}$$

• Recall

- Fraction of objects predicted to be positive among all positive objects

$$r = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{18}{20} = \frac{9}{10}$$

Solution:

The precision is the number of true positives divided by the number *predicted* to be positive. I.e. for the two classifiers

$$p_1 = \frac{18}{54} = \frac{1}{3}, \quad p_2 = \frac{14}{16} = \frac{7}{8}.$$

The recall is the number of true positives divided by the number of *actual* positives:

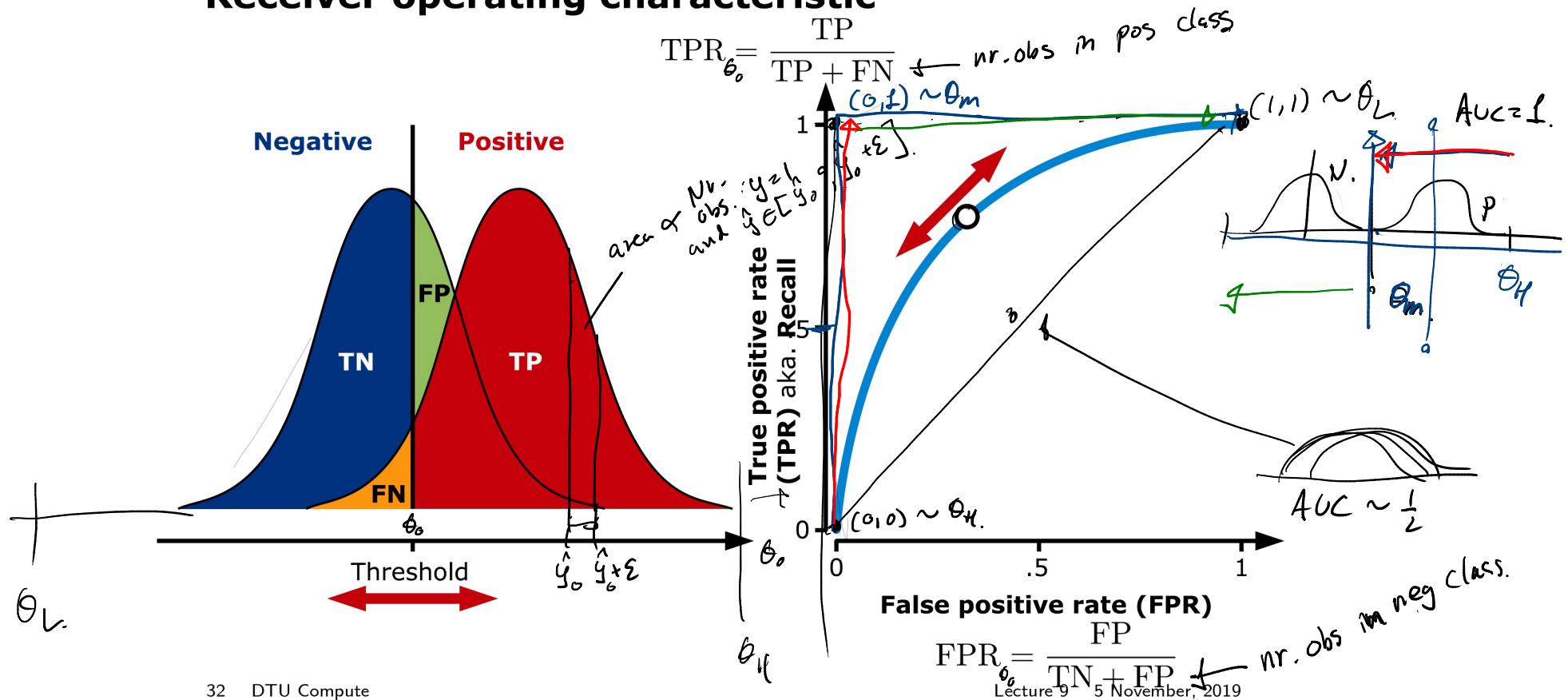
$$r_1 = \frac{18}{20} = \frac{9}{10}, \quad r_2 = \frac{14}{20} = \frac{7}{10}.$$

If the classifiers are supposed to detect credit-card fraud, we probably want a classifier which does not incorrectly *miss* many fraudulent transactions, i.e. it should have a low value of FN. In that case a high recall is desirable, and we should go with classifier 1.

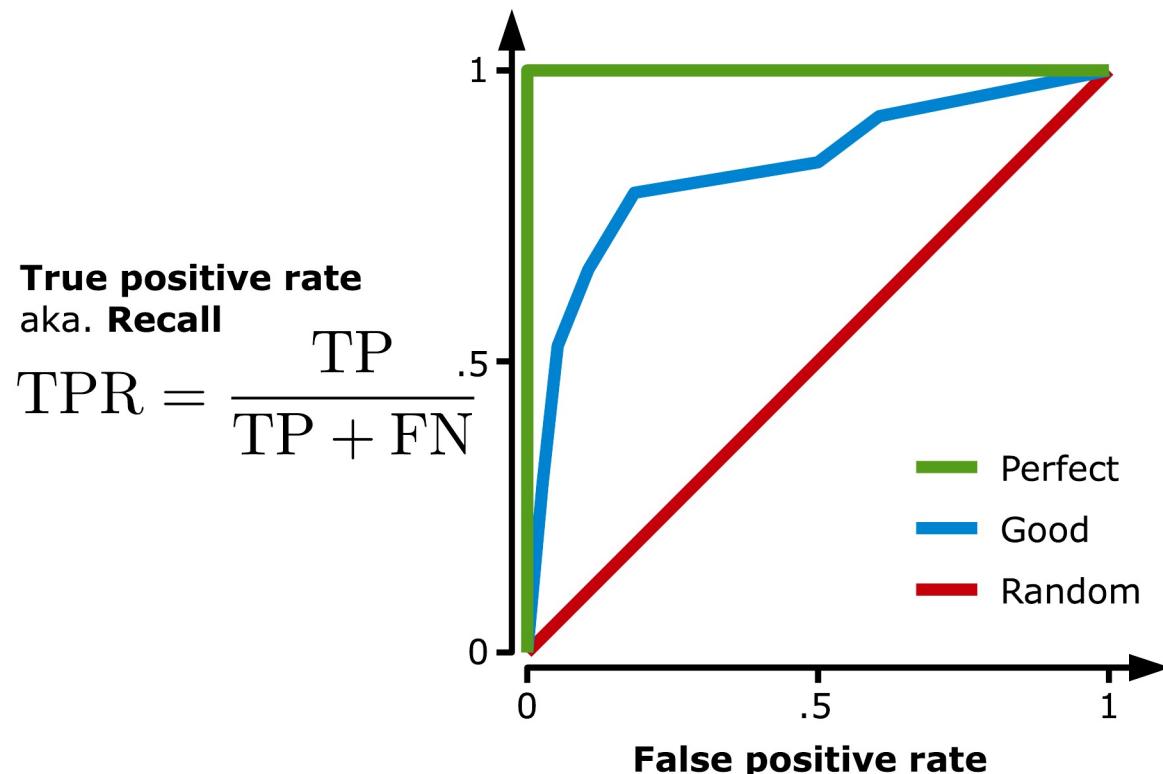
$$z = p(y \in I | x) > \theta_0.$$



Receiver operating characteristic



Receiver operating characteristic



True positive rate
aka. **Recall**

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

False positive rate

$$\text{FPR} = \frac{\text{FP}}{\text{TN} + \text{FP}}$$

Quiz 03 (please answer on Piazza): AUC (Spring 2017)

$$x_5 > 3.5 \Rightarrow \theta_0$$

	3 gears ($x_5 = 3$)	4 gears ($x_5 = 4$)	5 gears ($x_5 = 5$)
Low mpg ($y = 0$)	13	2	2
High mpg ($y = 1$)	2	10	3

Table 1: Number of low mpg and high mpg cars (i.e. $y = 0$ and $y = 1$) according to the number of gears, i.e. $x_5 = 3$, $x_5 = 4$, or $x_5 = 5$.

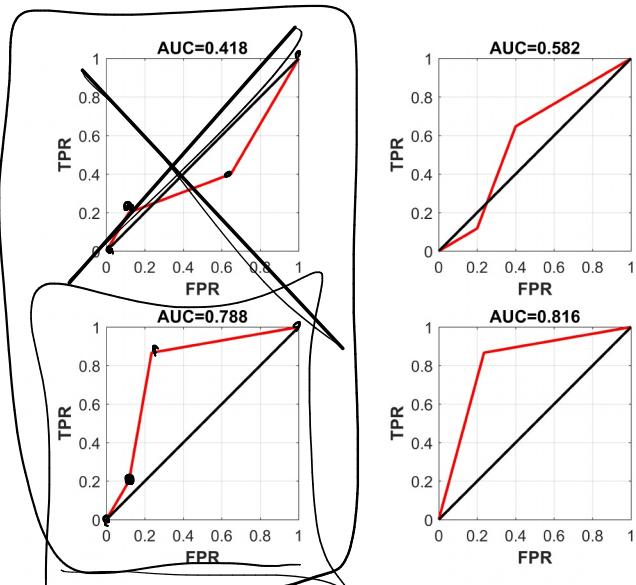


Figure 1: Four different receiver operator characteristic (ROC) curves and their area under curve (AUC) value.

A dataset representing cars contain an attribute x_5 corresponding to the number of gears. We wish to evaluate how well the number of gears predict low mpg, ($y = 0$, considered the negative class) from high mpg, ($y = 1$, considered the positive class) based on the data given in Table 1. For this purpose, we will evaluate the area under curve (AUC) of the receiver operator characteristic (ROC) using the feature x_5 . Which one of the ROC curves given in Figure 1 corresponds to using x_5 to discriminate between low mpg ($y = 0$) and high mpg ($y = 1$)?

- A. The curve having AUC=0.418
- B. The curve having AUC=0.582
- C. The curve having AUC=0.788
- D. The curve having AUC=0.816
- E. Don't know.

(Hint: Select a value e.g. $x_5 = 4$. We then predict cars with 4 or more gears as being in the positive class and otherwise negative. Compute the FPR and TPR using this prediction and use the (FPR, TPR) values to discriminate between the curves)

$$\begin{aligned} TPR &= \frac{TP}{acc. pos.} = \\ &\text{actually } \frac{TP}{FP + neg. pred. positive} \\ FPR &= \frac{FP}{acc. neg.} \end{aligned}$$

$$FPR = \frac{FP}{17}$$

$$TPR = \frac{TP}{15}$$

$$4.5 \text{ gears: } \left(\frac{2}{17}, \frac{3}{15} \right)$$

Solution:

The ROC curve can be calculated by lowering the threshold, as no cars have more than 5 forward gears a threshold above 5 will result in the point (0,0). Lowering the threshold we find at the value 5 that $2/17$ of the low mpg cars (FPR) are at 5 and $3/15$ of the high mpg cars (TPR) are at 5 corresponding to

the point $(2/17, 3/15)$. When lowering to a threshold of 4 gears or more we are at the point $(4/17, 13/15)$ and at a threshold at 3 gears or more we have $(1,1)$. Thus, this curve corresponds to the curve having $AUC=0.788$.

Resources

<https://www.youtube.com> Video tutorial on ROC curve and AUC

(<https://www.youtube.com/watch?v=0A16eAyP-yo>)

<https://towardsdatascience.com> More in-depth discussion of the Random Forrest algorithm and parameter choices

(<https://towardsdatascience.com/the-random-forest-algorithm-d457d499ffcd>)

<https://www.datacamp.com> Practical use of the random forrest algorithm in python

(<https://www.datacamp.com/community/tutorials/random-forests-classifier-python>)

<https://citeseerx.ist.psu.edu> Justification for the AdaBoost algorithm (technical) (<https://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.51.9525>)