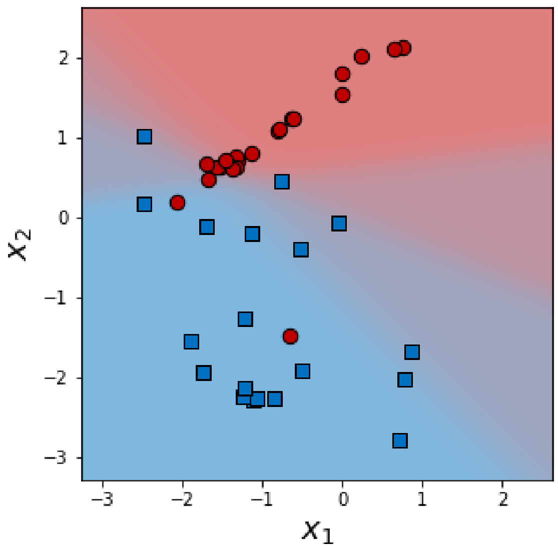
Photogrammetry & Robotics Lab

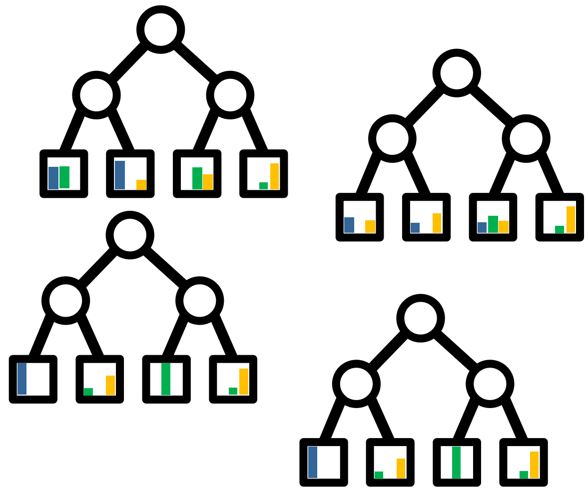
Machine Learning for Robotics and Computer Vision Tutorial

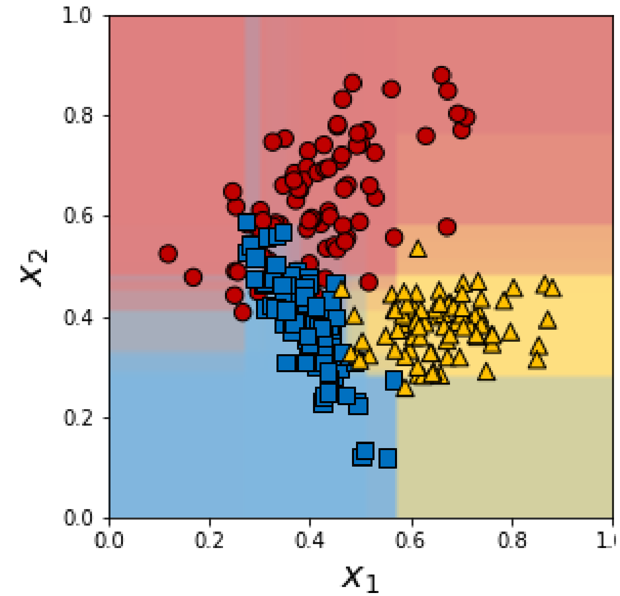
More on Ensembles and Metrics

Jens Behley

Recap: Ensembles

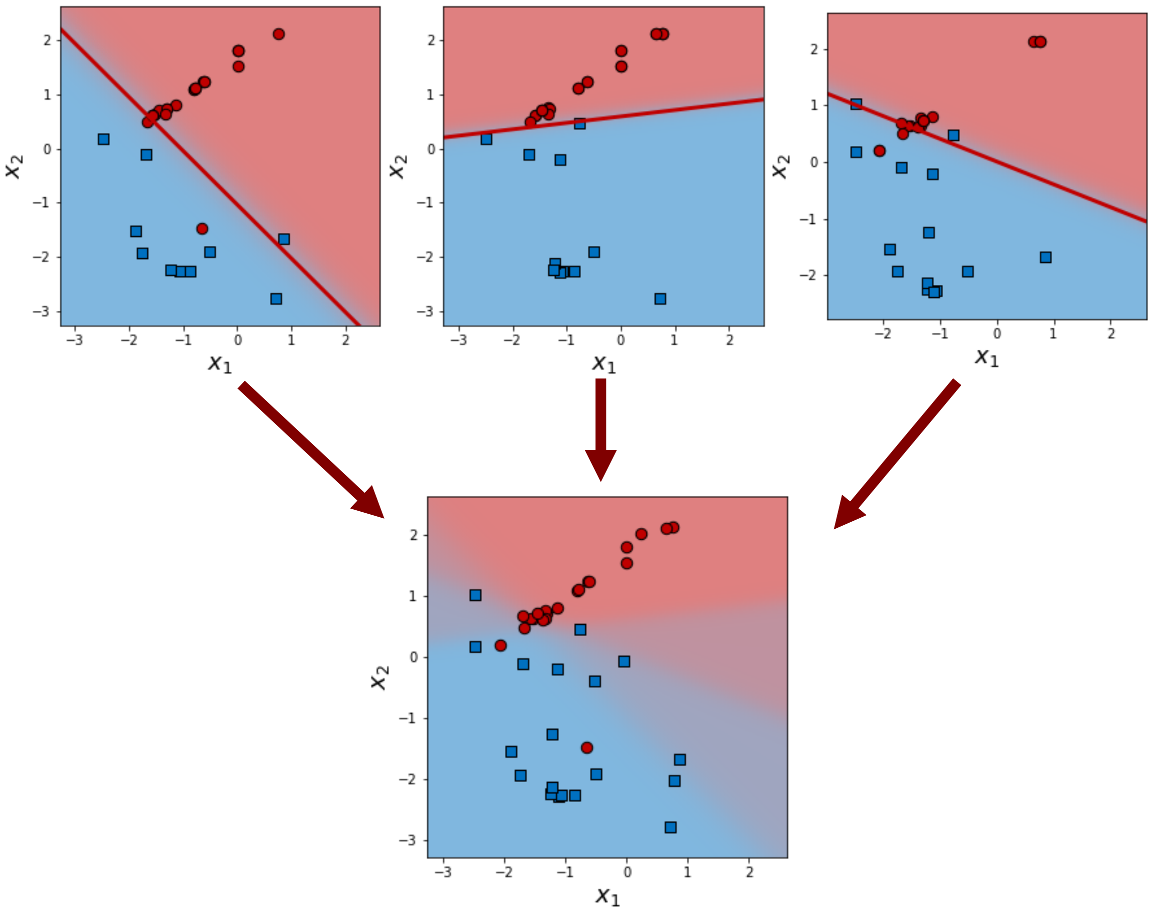






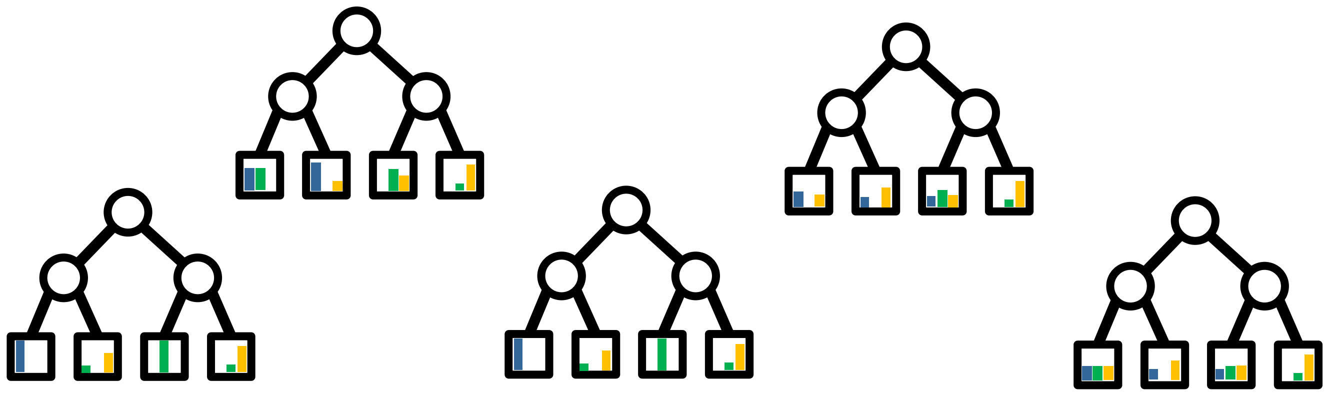
- Discussed ensemble methods:
  - Bagging
  - Random Forests
  - Boosting: AdaBoost and Gradient Tree Boosting
- Powerful off-the-shelf methods

Recap: Bagging



- Ensemble Prediction: $P(y|\mathbf{x}) = T^{-1}\sum_{j=1}^{T} P_j(y|\mathbf{x})$
- Combined predictions can be more accurate

Recap: Random Forest



- Use T Decision Trees as weak learners
- Each Decision Tree is randomized by:
  1. Selecting subset of features and split functions
  2. Bagging for each Decision Tree
- Randomization of split functions reduces correlation of individual trees

[Criminisi et al., 2011]

Recap: Gradient Boosting

1. $H_0(\mathbf{x}) = 0$
2. For $m = 1$ to M:

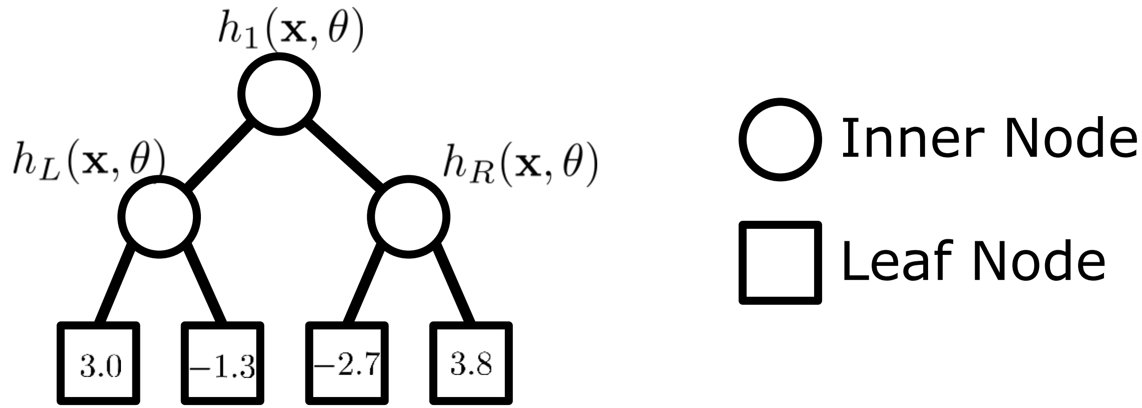
   (a) $\tilde{y}_i = -\left[\frac{\partial \ell(y_i, H(\mathbf{x}_i))}{\partial H(\mathbf{x}_i)}\right]_{H(\mathbf{x})=H_{m-1}(\mathbf{x})}, i = 1, \ldots, N$

   (b) $h_m = \arg\min_h \sum_i \left[\tilde{y}_i - h(\mathbf{x}_i)\right]^2$

   (c) $H_m(\mathbf{x}) = H_{m-1}(\mathbf{x}) + \nu h_m(\mathbf{x})$

- As long as we perform a step in the right direction (given by the gradient), we are fine with doing a small step by $\nu \in \mathbb{R}$
- $\nu \in \mathbb{R}$ is the learning rate

Recap: Regression Tree



- Same algorithm, but now targets $y_i \in \mathbb{R}$
- Leaves return estimate for region
- Training objective: $\sum_{(\mathbf{x}_i, y_i) \in \mathcal{X}_{\text{train}}} (y_i - H(\mathbf{x}_i))^2$
- Split functions still threshold function:

$$h(\mathbf{x}|d, \tau) = \begin{cases} x_d < \tau, & 0 \\ x_d \geq \tau, & 1 \end{cases}$$

Recap: Building Regression Tree

- Same as before: Recursively split examples via split functions at inner nodes.
- Estimated value of examples reaching leaf:

$$\bar{y} = \frac{1}{|\mathcal{S}|} \sum_{(\mathbf{x}_i, y_i) \in \mathcal{S}} y_i$$

- To determine quality of split $Q(\mathcal{S}_L, \mathcal{S}_R)$, we can use the square loss with respect to the average $\bar{y}$:

$$Q(\mathcal{S}_L, \mathcal{S}_R) = \sum_{(\mathbf{x}, y) \in \mathcal{S}_L} (y - \bar{y})^2 + \sum_{(\mathbf{x}, y) \in \mathcal{S}_R} (y - \bar{y})^2$$

Evaluation Metrics

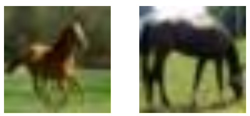
Evaluation Metrics

- Evaluation metrics measure performance of algorithms
- Enable comparison of different approaches

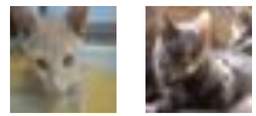
- Here, we concentrate on commonly used metrics for classification. But different tasks have also more task specific metrics…

Confusion Matrix

- Example: Let’s say we examples, K = 3 (horse, cat, dog) and 6 test images.



horse

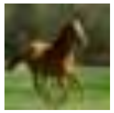
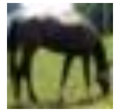
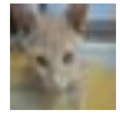
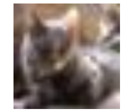
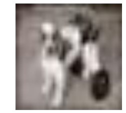
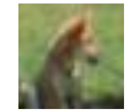


cat

dog

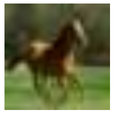
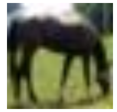
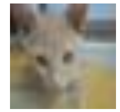
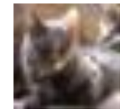
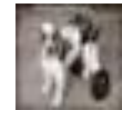
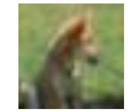
- Sidenote: CIFAR-10 images, 32x32 color images with 10 classes
- We train our classifier, we apply it and get the following predictions:

Our classifier results

						
horse	0.98	0.20	0.01	0.20	0.30	0.02
cat	0.00	0.02	0.83	0.73	0.67	0.45
dog	0.20	0.78	0.16	0.07	0.03	0.53

$$\hat{y} = \arg\max_y P(y|\mathbf{x})$$

Our classifier results

						
horse	$\boxed{0.98}$	0.20	0.01	0.20	0.30	0.02
cat	0.00	0.02	$\boxed{0.83}$	$\boxed{0.73}$	$\boxed{0.67}$	0.45
dog	0.20	$\boxed{0.78}$	0.16	0.07	0.03	$\boxed{0.53}$
$\hat{y}$	horse	dog	cat	cat	cat	dog
	✓	✗	✓	✓	✗	✓

Confusion Matrix

$\hat{y}$	horse	dog	cat	cat	cat	dog
	✓	✗	✓	✓	✗	✓

Prediction \ GT	horse	cat	dog
horse			
cat			
dog			

- Confusion matrix counts number of ground truth (gt) vs. predicted.

Confusion Matrix

$\hat{y}$	horse	dog	cat	cat	cat	dog
	✓	✗	✓	✓	✗	✓

Prediction \ GT	horse	cat	dog
horse	1	0	0
cat	0	2	1
dog	1	0	1

True/False Positive/Negative

- For a specific class K, we can now defined TP, FP, TN & FN

- **True positive** (TP): Predicted K and is K
- **False positive** (FP): Classified as K, but actually a different class (not K)
- **True negative** (TN): Predicted not K and is not K
- **False negative** (FN): Predicted not K and is not K

Confusion Matrix

Prediction \ GT	**horse**	cat	dog
horse	1	0	0
cat	0	2	1
dog	1	0	1

- Which entries correspond to TP, FP, TN, FN?

Confusion Matrix

Prediction \ GT	**horse**	cat	dog
horse	1	0	0
cat	0	2	1
dog	1	0	1

True positive

False positive

True negative

False negative

Accuracy

Prediction	horse	cat	dog
horse	$\boxed{1}$	0	0
cat	0	$\boxed{2}$	1
dog	1	0	$\boxed{1}$

- **Accuracy**: Sum over true positives / number of all examples → 3/6 = 0.5 or 50%
- Accuracy affected by class distribution!

Precision and Recall

$$\text{Precision} = \frac{\#\text{Correct}}{\#\text{Predictions}} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\#\text{Correct}}{\#\text{Ground Truth}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **Precision**: How many of the predictions are correct?
- **Recall**: How many of ground truth have been found?

Example: Precision and Recall

Prediction	horse	cat	dog
horse	1	0	0
cat	0	2	1
dog	1	0	1

- What is the precision, recall for each class?

Confusion Matrix

GT / Prediction	horse	cat	dog	Precision
horse	1	0	0	1
cat	0	2	1	2/3
dog	1	0	1	1/2
Recall	1/2	2/2	1/2	

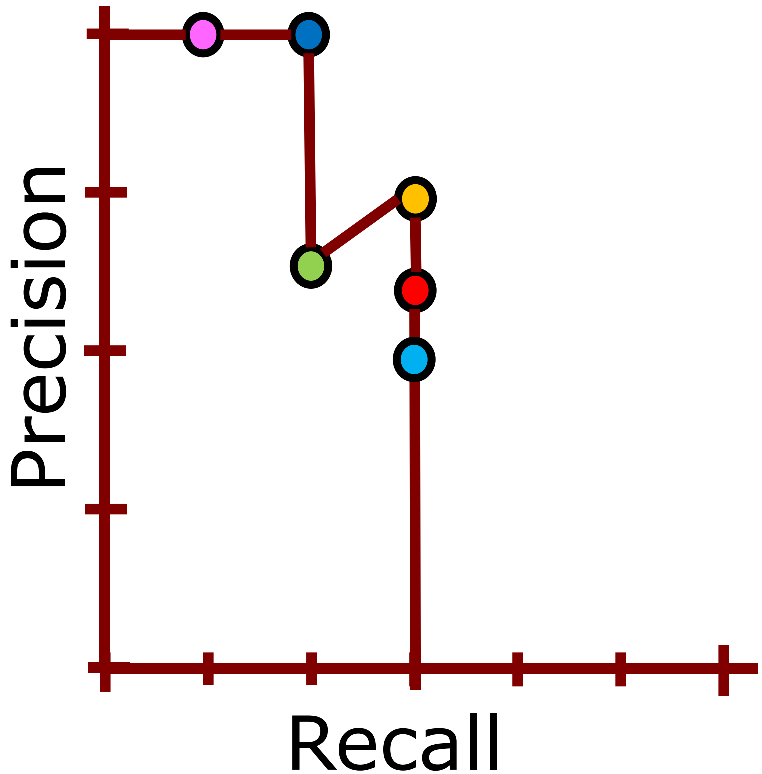
F1 score

- Usually one wants to have single metric
- F_1 score combines precision recall

$$F_1 = \frac{2\,\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

- For multiple classes: Average over class F1 scores.

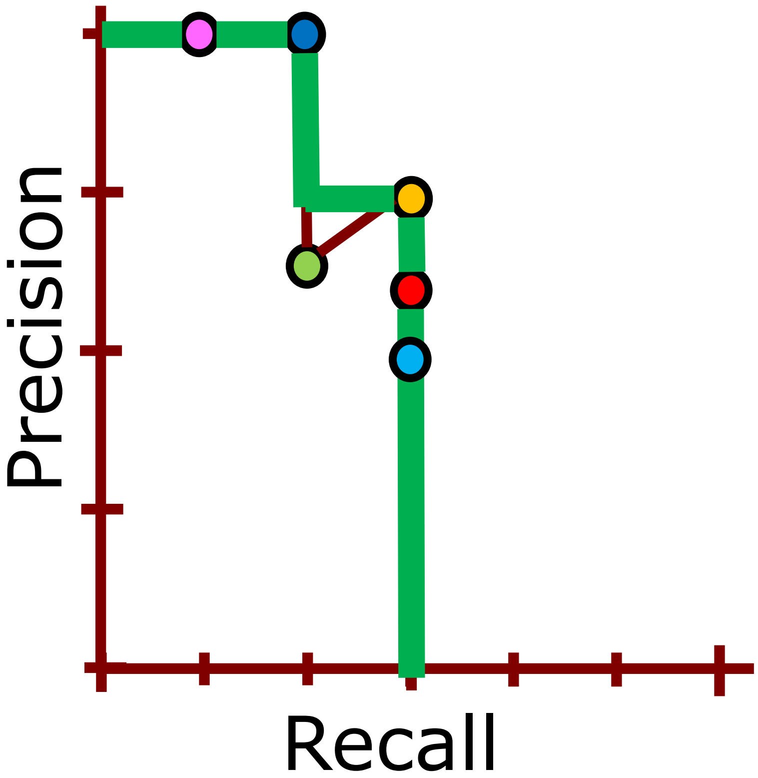
Precision Recall Curve



Confidence	TP	P	R
0.93	✓	1/1	1/6
0.87	✓	2/2	2/6
0.74	✗	2/3	2/6
0.71	✓	3/4	3/6
0.62	✗	3/5	3/6
0.51	✗	3/6	3/6

- Add Precision/Recall for a single class

Interpolated PR Curve

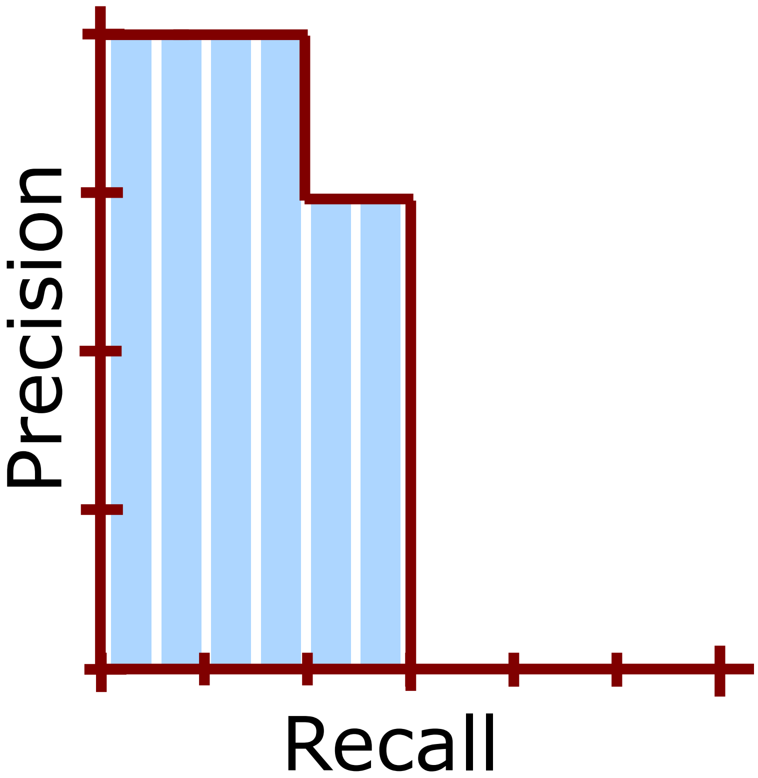


Confidence	TP	P	R
0.98	✓	1/1	1/6
0.83	✓	2/2	2/6
0.78	✗	2/3	2/6
0.73	✓	3/4	3/6
0.67	✗	3/5	3/6
0.53	✗	3/6	3/6

Interpolated PR Curve

- Interpolated PR-Curve: take maximum precision at higher recalls

Mean Average Precision (mAP)



$$\text{AP} = \frac{1}{N}\sum_i \text{Precision}\left(\frac{i}{N}\right)$$

Typical values:

- N = 11, 40 (KITTI)
- N = 101 (COCO)

- **Average Precision**: Area under PR Curve
- mAP = mean over all class APs
- COCO: mean mAP at multiple IoU levels θ_{IoU}

Questions?

See you next week!