

PATTERN RECOGNITION AND MACHINE LEARNING

Slide Set 3: Detection Theory

October 2019

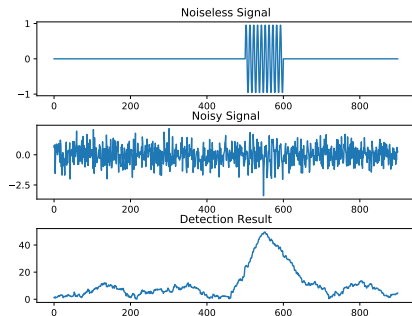
Heikki Huttunen
heikki.huttunen@tuni.fi
Signal Processing
Tampere University

Detection theory

- In this section, we will briefly consider detection theory.
- Detection theory has many common topics with machine learning.
- The methods are based on estimation theory and attempt to answer questions such as
 - Is a signal of specific model present in our time series? E.g., detection of noisy sinusoid; beep or no beep?
 - Is the transmitted pulse present at radar signal at time t ?
 - Does the mean level of a signal change at time t ?
 - After calculating the mean change in pixel values of subsequent frames in video, is there something moving in the scene?
 - Is there a person in this video frame?
- The area is closely related to *hypothesis testing*, which is widely used e.g., in medicine: Is the response in patients due to the new drug or due to random fluctuations?

Detection theory

- Consider the detection of a sinusoidal waveform



Detection theory

- In our case, the hypotheses could be

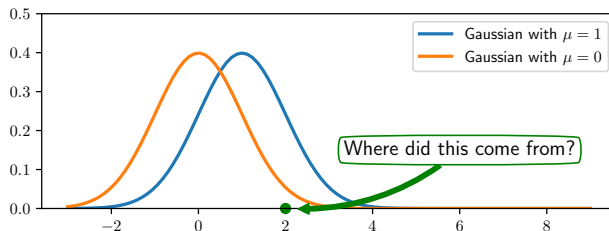
$$\mathcal{H}_1 : x[n] = A \cos(2\pi f_0 n + \phi) + w[n]$$

$$\mathcal{H}_0 : x[n] = w[n]$$

- This example corresponds to detection of noisy sinusoid.
- The hypothesis \mathcal{H}_1 corresponds to the case that the sinusoid is present and is called *alternative hypothesis*.
- The hypothesis \mathcal{H}_0 corresponds to the case that the measurements consists of noise only and is called *null hypothesis*.

Introductory Example

- Consider a simplistic detection problem, where we observe one sample $x[0]$ from one of two densities: $\mathcal{N}(0, 1)$ or $\mathcal{N}(1, 1)$.
- The task is to choose the correct density in an optimal manner.



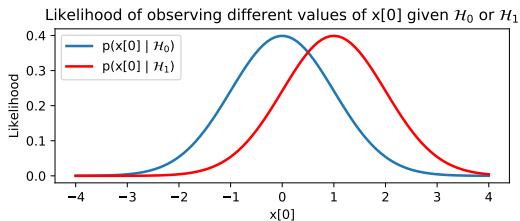
Introductory Example

- Our hypotheses are now

$$\mathcal{H}_1 : \mu = 1,$$

$$\mathcal{H}_0 : \mu = 0,$$

and the corresponding likelihoods are plotted below.



Introductory Example

- An obvious approach for deciding the density would choose the one, which is higher for a particular $x[0]$.
- More specifically, study the likelihoods and choose the more likely one.
- The likelihoods are

$$\mathcal{H}_1 : p(x[0] \mid \mu = 1) = \frac{1}{\sqrt{2\pi}} \exp \left(-\frac{(x[0] - 1)^2}{2} \right).$$

$$\mathcal{H}_0 : p(x[0] \mid \mu = 0) = \frac{1}{\sqrt{2\pi}} \exp \left(-\frac{(x[0])^2}{2} \right).$$

- One should select \mathcal{H}_1 if " $\mu = 1$ " is more likely than " $\mu = 0$ ".
- In other words, $p(x[0] \mid \mu = 1) > p(x[0] \mid \mu = 0)$.

Introductory Example

- Let's state this in terms of $x[0]$:

$$\begin{aligned} p(x[0] \mid \mu = 1) &> p(x[0] \mid \mu = 0) \\ \Leftrightarrow \frac{p(x[0] \mid \mu = 1)}{p(x[0] \mid \mu = 0)} &> 1 \\ \Leftrightarrow \frac{\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x[0]-1)^2}{2}\right)}{\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x[0])^2}{2}\right)} &> 1 \\ \Leftrightarrow \exp\left(-\frac{(x[0]-1)^2 - x[0]^2}{2}\right) &> 1 \end{aligned}$$

Introductory Example

$$\Leftrightarrow (x[0]^2 - (x[0] - 1)^2) > 0$$

$$\Leftrightarrow 2x[0] - 1 > 0$$

$$\Leftrightarrow x[0] > \frac{1}{2}.$$

- In other words, choose \mathcal{H}_1 if $x[0] > 0.5$ and \mathcal{H}_0 if $x[0] < 0.5$.
- Studying the ratio of likelihoods (second row of the previous derivation) is the key:

$$\frac{p(x[0] \mid \mu = 1)}{p(x[0] \mid \mu = 0)} > 1$$

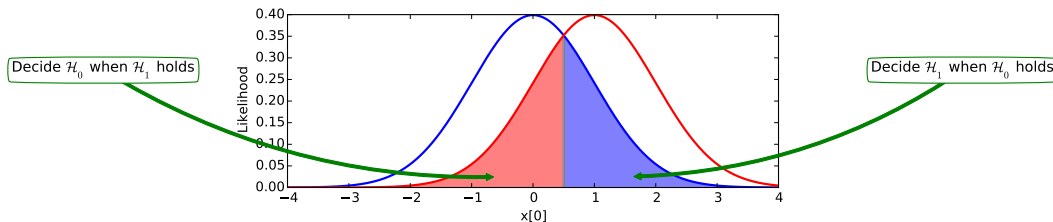
- This ratio is called *likelihood ratio*, and comparison to a threshold (here $\gamma = 1$) is called *likelihood ratio test* (LRT).
- Of course the *detection threshold* γ may be chosen other than $\gamma = 1$.

Error Types

- It might be that the detection problem is not symmetric and some errors are more costly than others.
- For example, when detecting a disease, a missed detection is more costly than a false alarm.
- The tradeoff between misses and false alarms can be adjusted using the threshold of the LRT.

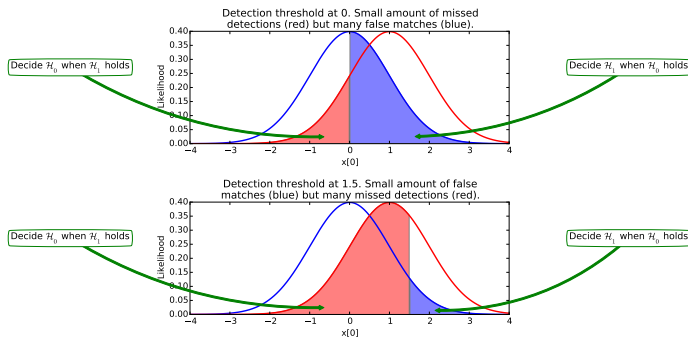
Error Types

- The below figure illustrates the probabilities of the two kinds of errors.
- The blue area on the left corresponds to the probability of choosing \mathcal{H}_1 while \mathcal{H}_0 would hold (false match).
- The red area is the probability of choosing \mathcal{H}_0 while \mathcal{H}_1 would hold (missed detection).



Error Types

- It can be seen that we can decrease either probability arbitrarily small by adjusting the detection threshold.



Error Types

- For example, suppose the threshold is $\gamma = 1.5$. What are P_{FA} and P_D ?
- Probability of false alarm is found by integrating over the blue area:

$$P_{FA} = P(x[0] > \gamma \mid \mu = 0) = \int_{1.5}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x[0])^2}{2}\right) dx[0] \approx 0.0668.$$

- Probability of missed detection is the area marked in red:

$$P_M = P(x[0] < \gamma \mid \mu = 1) = \int_{-\infty}^{1.5} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x[0] - 1)^2}{2}\right) dx[0] \approx 0.6915.$$

- An equivalent, but more useful term is the complement of P_M : probability of detection:

$$P_D = 1 - P_M = \int_{1.5}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x[0] - 1)^2}{2}\right) dx[0] \approx 0.3085.$$

Choosing the threshold

- Often we don't want to define the threshold, but rather the amount of false alarms we can accept.
- For example, suppose we want to find the best detector for our introductory example, and we can tolerate 10% false alarms ($P_{FA} = 0.1$).
- The likelihood ratio detection rule is:

$$\text{Select } \mathcal{H}_1 \text{ if } \frac{p(x | \mu = 1)}{p(x | \mu = 0)} > \gamma$$

The only thing to find out now is the threshold γ such that

$$\int_{\gamma}^{\infty} p(x | \mu = 0) dx = 0.1.$$

Choosing the threshold

- This can be done with Python function `isf`, which solves the inverse cumulative distribution function.

```
>>> import scipy.stats as stats  
  
>>> # Compute threshold such that  $P_{FA} = 0.1$   
>>> T = stats.norm.isf(0.1, loc = 0, scale = 1)  
>>> print T  
1.28155156554
```

- The parameters `loc` and `scale` are the mean and standard deviation of the Gaussian density, respectively.

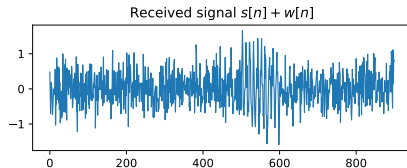
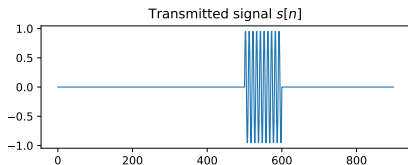
Detector for a known waveform

- An important special case is that of a known waveform $s[n]$ embedded in WGN sequence $w[n]$:

$$\mathcal{H}_1 : x[n] = s[n] + w[n]$$

$$\mathcal{H}_0 : x[n] = w[n].$$

- An example of a case where the waveform is known could be detection of radar signals, where a pulse $s[n]$ transmitted by us is reflected back after some propagation time.



Detector for a known waveform

- For this case the likelihoods are

$$p(\mathbf{x} | \mathcal{H}_1) = \prod_{n=0}^{N-1} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x[n] - s[n])^2}{2\sigma^2}\right),$$
$$p(\mathbf{x} | \mathcal{H}_0) = \prod_{n=0}^{N-1} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x[n])^2}{2\sigma^2}\right).$$

- The likelihood ratio test is easily obtained as

$$\frac{p(\mathbf{x} | \mathcal{H}_1)}{p(\mathbf{x} | \mathcal{H}_0)} = \exp\left[-\frac{1}{2\sigma^2} \left(\sum_{n=0}^{N-1} (x[n] - s[n])^2 - \sum_{n=0}^{N-1} (x[n])^2\right)\right] > \gamma.$$

Detector for a known waveform

- This simplifies by taking the logarithm from both sides:

$$-\frac{1}{2\sigma^2} \left(\sum_{n=0}^{N-1} (x[n] - s[n])^2 - \sum_{n=0}^{N-1} (x[n])^2 \right) > \ln \gamma.$$

- This further simplifies into

$$\frac{1}{\sigma^2} \sum_{n=0}^{N-1} x[n]s[n] - \frac{1}{2\sigma^2} \sum_{n=0}^{N-1} (s[n])^2 > \ln \gamma.$$

Detector for a known waveform

- Since $s[n]$ is a known waveform (= constant), we can simplify the procedure by moving it to the right hand side and combining it with the threshold:

$$\sum_{n=0}^{N-1} x[n]s[n] > \sigma^2 \ln \gamma + \frac{1}{2} \sum_{n=0}^{N-1} (s[n])^2.$$

We can equivalently call the right hand side as our threshold (say γ') to get the final decision rule

$$\sum_{n=0}^{N-1} x[n]s[n] > \gamma'.$$

Example

- The detector for a sinusoid in WGN is

$$\sum_{n=0}^{N-1} x[n] A \cos(2\pi f_0 n + \phi) > \gamma \Rightarrow A \sum_{n=0}^{N-1} x[n] \cos(2\pi f_0 n + \phi) > \gamma.$$

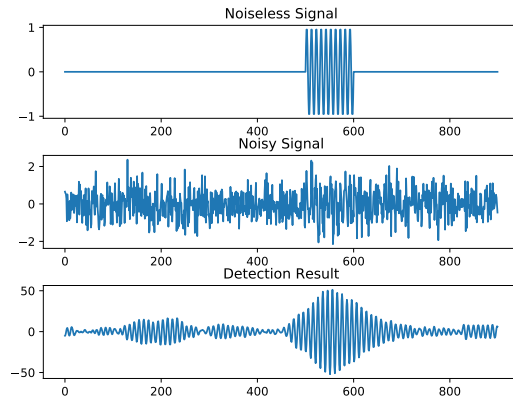
- Again we can divide by A to get

$$\sum_{n=0}^{N-1} x[n] \cos(2\pi f_0 n + \phi) > \gamma'.$$

- In other words, we check the correlation with the sinusoid. Note that the amplitude A does not affect our statistic, only the threshold which is anyway selected according to the fixed P_{FA} rate.

Example

- As an example, the picture shows the detection process with $\sigma = 0.5$.
- Note, that we apply the detector with a sliding window; *i.e.*, we perform the hypothesis test at every window of length 100.



Detection of random signals

- The problem with the previous approach was that the model was too restrictive; the results depend on how well the phases match.
- The model can be relaxed by considering *random signals*, whose exact form is unknown, but the correlation structure is known. Since the correlation captures the frequency (but not the phase), this is exactly what we want.
- In general, the detection of a random signal can be formulated as follows.
- Suppose $\mathbf{s} \sim \mathcal{N}(0, \mathbf{C}_s)$ and $\mathbf{w} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$. Then the detection problem is a hypothesis test

$$\mathcal{H}_0 : \mathbf{x} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

$$\mathcal{H}_1 : \mathbf{x} \sim \mathcal{N}(0, \mathbf{C}_s + \sigma^2 \mathbf{I})$$

Detection of random signals

- It can be shown, that the decision rule becomes

Decide \mathcal{H}_1 , if $\mathbf{x}^T \hat{\mathbf{s}} > \gamma$,

where

$$\hat{\mathbf{s}} = \mathbf{C}_s(\mathbf{C}_s + \sigma^2 \mathbf{I})^{-1} \mathbf{x}.$$

Example of Random Signal Detection

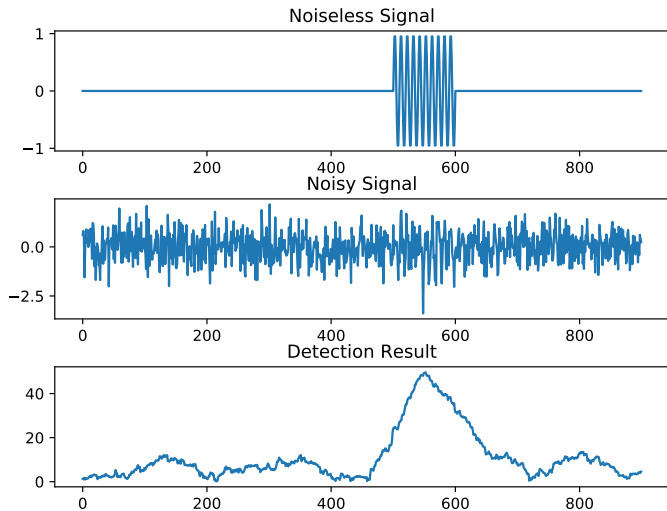
- Without going into the details, let's jump directly to the derived decision rule for the sinusoid:

$$\left| \sum_{n=0}^{N-1} x[n] \exp(-2\pi i f_0 n) \right| > \gamma.$$

- As an example, the below picture shows the detection process with $\sigma = 0.5$.
- Note the simplicity of Python implementation:

```
import numpy as np
h = np.exp(-2 * np.pi * 1j * f0 * n)
y = np.abs(np.convolve(h, xn, 'same'))
```


Example of Random Signal Detection



Receiver Operating Characteristics

- A usual way of illustrating the detector performance is the *Receiver Operating Characteristics* curve (ROC curve).
- This describes the relationship between P_{FA} and P_D for all possible values of the threshold γ .
- The functional relationship between P_{FA} and P_D depends on the problem and the selected detector.

Receiver Operating Characteristics

- For example, in the DC level example,

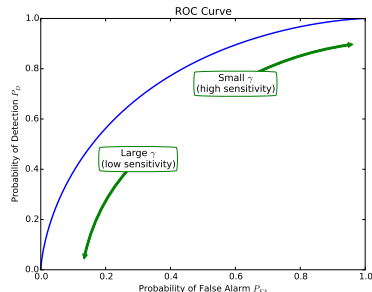
$$P_D(\gamma) = \int_{\gamma}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x-1)^2}{2}\right) dx$$

$$P_{FA}(\gamma) = \int_{\gamma}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) dx$$

- It is easy to see the relationship:

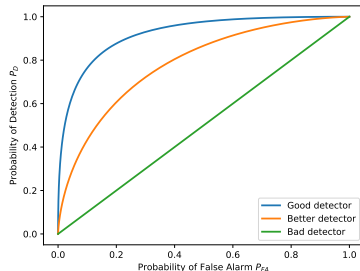
$$P_D(\gamma) = \int_{\gamma-1}^{\infty} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right) dx = P_{FA}(\gamma-1).$$

- Plotting the ROC curve for all γ is shown on the right.



Receiver Operating Characteristics

- The higher the ROC curve, the better the performance.
- A random guess has diagonal ROC curve.
- This gives rise to a widely used measure for detector performance: the *Area Under (ROC) Curve*, or AUC criterion.
- The benefit of AUC is that it is threshold independent, and tests the accuracy for *all* thresholds.

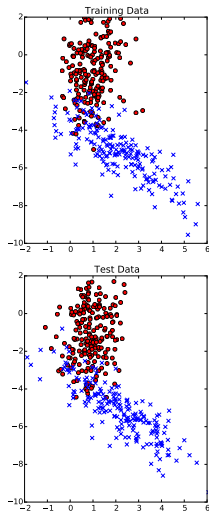


Empirical AUC

- Initially, AUC and ROC stem from radar and radio detection problems.
- More recently, AUC has become one of the standard measures of classification performance, as well.
- Usually a closed form expression for P_D and P_{FA} can not be derived.
- Thus, ROC and AUC are most often computed empirically; *i.e.*, by evaluating the prediction results on a holdout test set.

Classification Example—ROC and AUC

- For example, consider the 2-dimensional dataset on the right.
- The data is split to *training* and *test* sets, which are *similar* but not exactly the *same*.
- Let's train 4 classifiers on the upper data and compute the ROC for each on the bottom data.

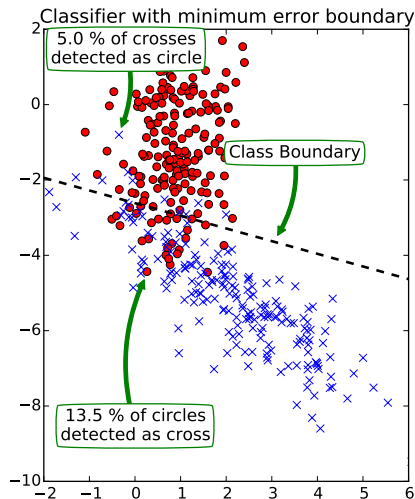


Classification Example—ROC and AUC

- A linear classifier trained with the training data produces the shown class boundary.
- The class boundary has the orientation and location that minimizes the overall classification error for the training data.
- The boundary is defined by:

$$y = c_1x + c_0$$

with parameters c_1 and c_0 learned from data.

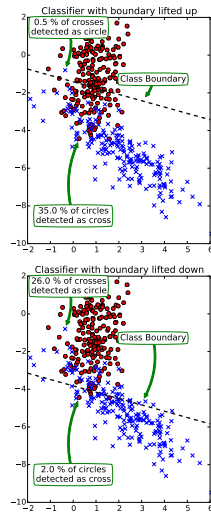


Classification Example—ROC and AUC

- We can adjust the sensitivity of classification by moving the decision boundary up or down.
- In other words, slide the parameter c_0 in

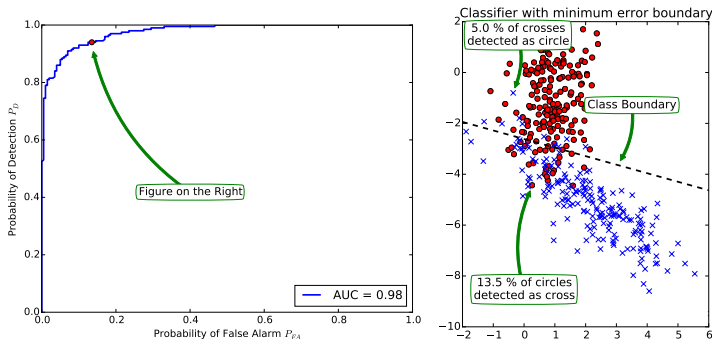
$$y = c_1x + c_0$$

- This can be seen as a *tuning parameter* for plotting the ROC curve.



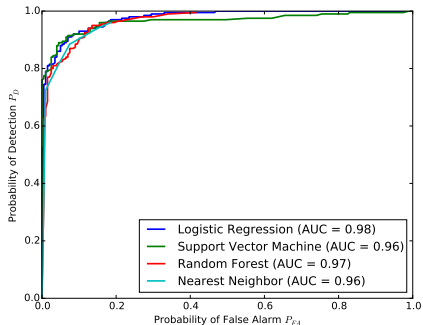
Classification Example—ROC and AUC

- When the boundary slides from bottom to top, we plot the *empirical ROC curve*.
- Plotting starts from upper right corner.
- Every time the boundary passes a blue cross, the curve moves left.
- Every time the boundary passes a red circle, the curve moves down.



Classification Example—ROC and AUC

- Real usage is for comparing classifiers.
- Below is a plot of ROC curves for 4 widely used classifiers.
- Each classifier produces a class membership score over which the tuning parameter slides.



ROC and AUC code in Python

```
classifiers = [(LogisticRegression(), "Logistic Regression"),
                (SVC(probability = True), "Support Vector Machine"),
                (RandomForestClassifier(n_estimators = 100), "Random Forest"),
                (KNeighborsClassifier(), "Nearest Neighbor")]

for clf, name in classifiers:
    clf.fit(X, y)

    ROC = []
    for gamma in np.linspace(0, 1, 1000):

        err1 = np.count_nonzero(clf.predict_proba(X_test[y_test == 0, :])[:,1] <= gamma)
        err2 = np.count_nonzero(clf.predict_proba(X_test[y_test == 1, :])[:,1] > gamma)

        err1 = float(err1) / np.count_nonzero(y_test == 0)
        err2 = float(err2) / np.count_nonzero(y_test == 1)

        ROC.append([err1, err2])
    ROC = np.array(ROC)

    ROC = ROC[::-1, :]
    auc = roc_auc_score(y_test, clf.predict_proba(X_test)[:,1])

    plt.plot(1-ROC[:, 0], ROC[:, 1], linewidth = 2, label="%s (AUC = %.2f)" % (name, auc))
```

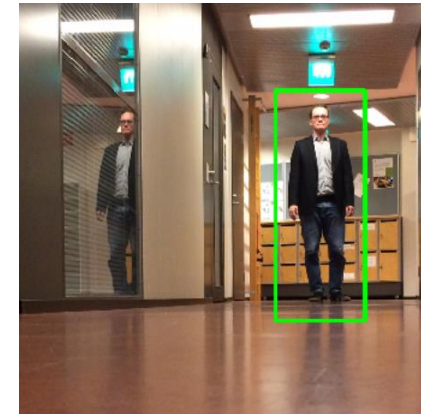
Precision and Recall

- Computing the P_{FA} from the negative examples may not always be feasible: For example, in the attached pictures, there are millions of negative examples (locations).
- Thus, in *search and retrieval*, other metrics are preferred:
 - Detector **recall** is defined as the proportion of true objects found.
 - Detector **precision** is the proportion of true objects from all found objects.
- For example, in the bottom figure:
 - Recall = #found true objects / #all true objects = $1/1 = 100\%$
 - Precision = #found true objects / #found all objects = $1 / 5 = 20\%$

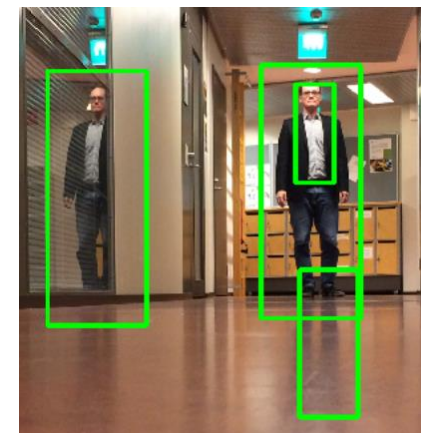
R = 0%
P = 100%



R = 100%
P = 100%

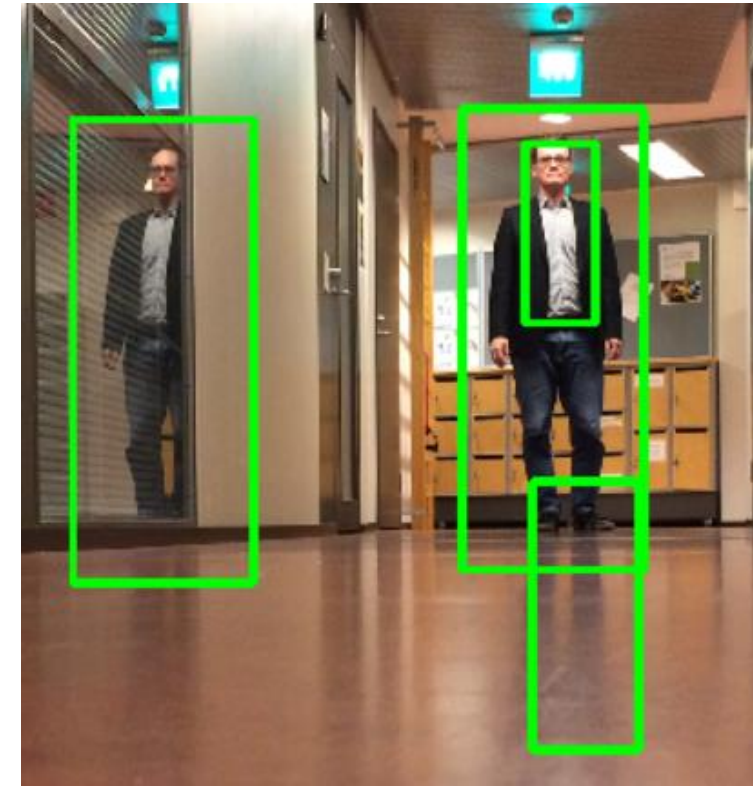
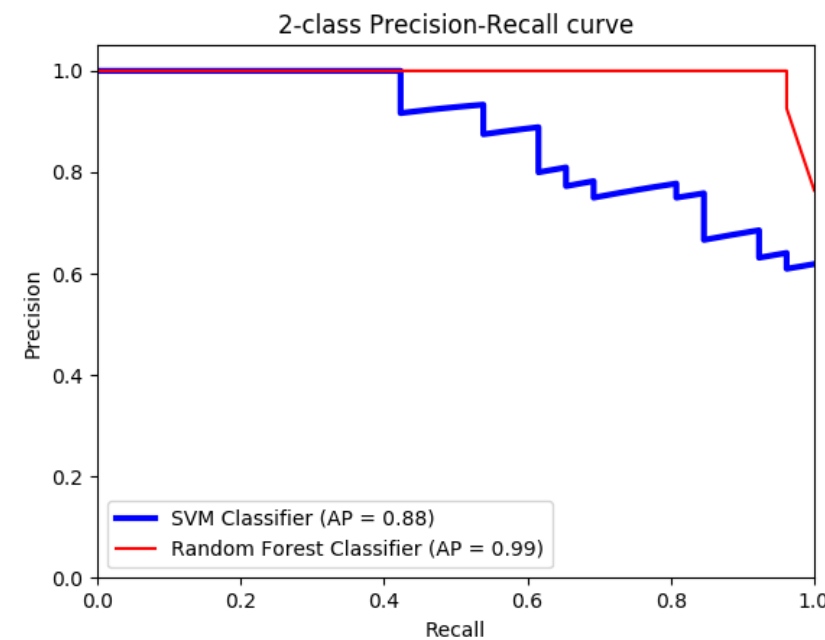


R = 100%
P = 20%



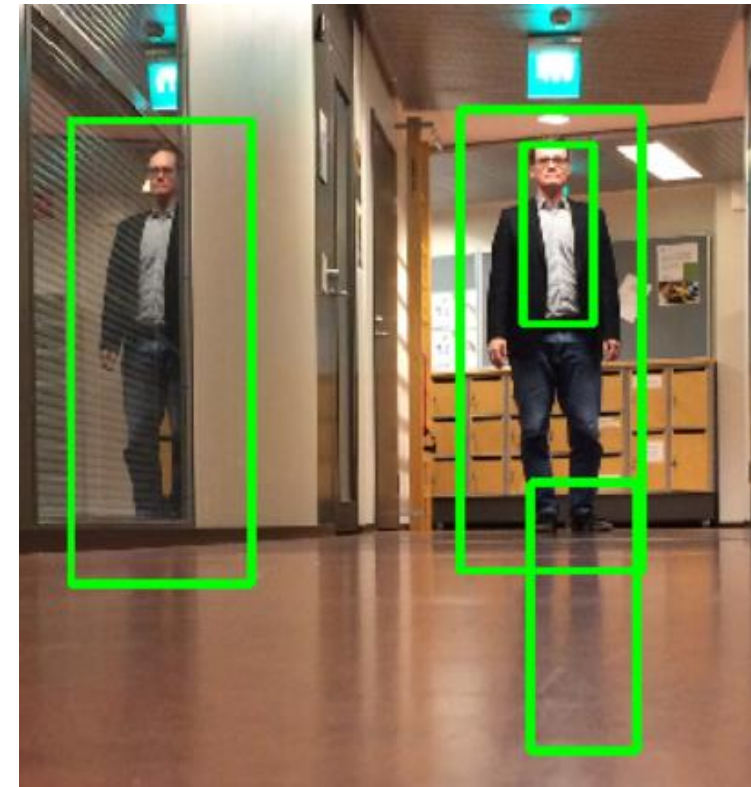
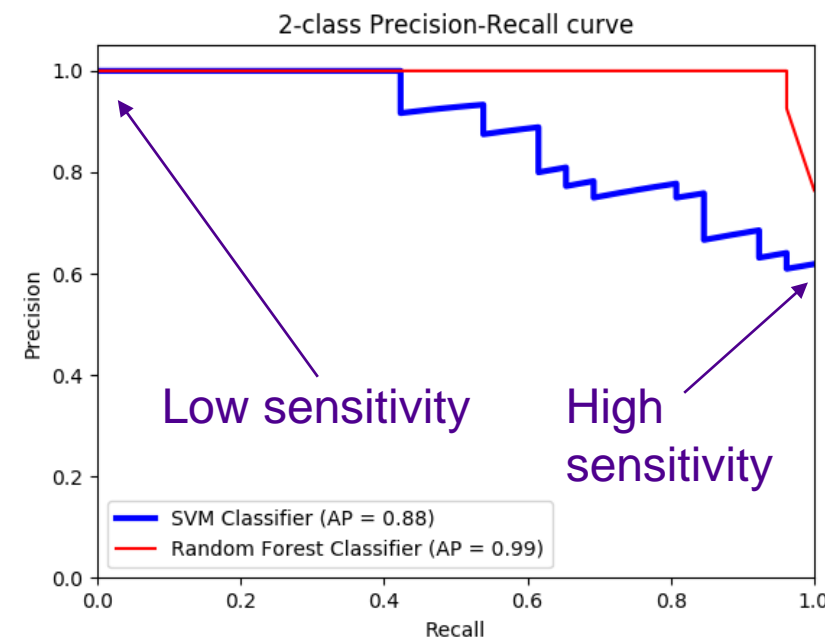
Precision-Recall tradeoff

- So, we can adjust the sensitivity (find many objects) and precision (find only true objects) by tuning the detection threshold.
- How could we study the detector itself without the need to speculate about the sensitivity.
- This can be done by plotting the precision versus recall for all relevant thresholds.
- An example of precision-recall curve is shown on the right.
- It is evident that the RF classifier (red curve) is superior to the SVM classifier (blue curve).
- If we looked only at an individual operating point (e.g., recall = 0.3, precision = 1.0), we might fail to see the difference.



Average Precision

- The precision recall curve also gives rise to a single metric of detector performance: **average precision (AP)**.
- Average precision is simply the average of precision over all recalls.
- Roughly the same as the **area under precision-recall curve**.
- The details of computation may vary; for example the tested thresholds, etc.



Average Precision

- The precision recall curve also gives rise to a single metric of detector performance: **average precision (AP)**.
- Average precision is simply the average of precision over all recalls.
- Roughly the same as the **area under precision-recall curve**.
- The details of computation may vary; for example the tested thresholds, etc.

