# Bagging and Random Forests

## Background: Resampling Techniques

We briefly review a technique called *bootstrap* on which bagging and random forests are based.

### Bootstrap

**Bootstrap** (or **resampling**) is a technique for improving the quality of estimators.

Resampling = sampling from the empirical distribution

### Application to ensemble methods

- We will use resampling to generate weak learners for classification.
- We discuss two classifiers which use resampling: Bagging and random forests.
- Before we do so, we consider the traditional application of Bootstrap, namely improving estimators.

## Bootstrap: Basic algorithm

### Given

- A sample $\tilde{\mathbf{x}}_1, \dots, \tilde{\mathbf{x}}_n$.
- An estimator $\hat{S}$ for a statistic $S$.

### Bootstrap algorithm

1. Generate $B$ **bootstrap samples** $\mathcal{B}_1, \dots, \mathcal{B}_B$. Each bootstrap sample is obtained by sampling $n$ times with replacement from the sample data. (Note: Data points can appear multiple times in any $\mathcal{B}_b$.)
2. Evaluate the estimator on each bootstrap sample:
$$\hat{S}_b := \hat{S}(\mathcal{B}_b)$$
(That is: We estimate $S$ pretending that $\mathcal{B}_b$ is the data.)
3. Compute the **bootstrap estimate** of $S$ by averaging over all bootstrap samples:
$$\hat{S}_{\text{BS}} := \frac{1}{B}\sum_{b=1}^{B}\hat{S}_b$$

## Example

### Recall: Plug-in estimators for mean and variance

$$\hat{\mu} := \frac{1}{n}\sum_{i=1}^{n} \tilde{\mathbf{x}}_i \qquad \hat{\sigma}^2 := \frac{1}{n}\sum_{i=1}^{n} (\tilde{\mathbf{x}}_i - \hat{\mu})^2$$

### Bootstrap Variance Estimate

1. For $b = 1, \dots, B$, generate a boostrap sample $\mathcal{B}_b$. In detail:
   For $i = 1, \dots, n$:
   - Sample an index $j \in \{1, \dots, n\}$.
   - Set $\tilde{\mathbf{x}}_i^{(b)} := \tilde{\mathbf{x}}_j$ and add it to $\mathcal{B}_b$.
2. For each $b$, compute mean and variance estimates:

$$\hat{\mu}_b := \frac{1}{n}\sum_{i=1}^{n} \tilde{\mathbf{x}}_i^{(b)} \qquad \hat{\sigma}_b^2 := \frac{1}{n}\sum_{i=1}^{n} (\tilde{\mathbf{x}}_i^{(b)} - \hat{\mu}_b)^2$$

3. Compute the bootstrap estimate:

$$\hat{\sigma}_{\mathrm{BS}}^2 := \frac{1}{B}\sum_{b=1}^{B} \hat{\sigma}_b^2$$

Not examinable.

## How Often Do We See Each Sample?

Sample $\{\tilde{\mathbf{x}}_1, ..., \tilde{\mathbf{x}}_n\}$, bootstrap resamples $\mathcal{B}_1, ..., \mathcal{B}_B$.

### In how many sets does a given $\mathbf{x}_i$ occur?

Probability for $\mathbf{x}_i$ *not* to occur in $n$ draws:

$$\Pr\{\tilde{\mathbf{x}}_i \notin \mathcal{B}_b\} = (1 - \frac{1}{n})^n$$

For large $n$:

$$\lim_{n\to\infty}\left(1 - \frac{1}{n}\right)^n = \frac{1}{e} \approx 0.3679$$

- Asymptotically, any $\tilde{\mathbf{x}}_i$ will appear in $\sim 63\%$ of the bootstrap resamples.
- Multiple occurrences possible.

### How often is $\tilde{\mathbf{x}}_i$ expected to occur?

The *expected* number of occurences of each $\tilde{\mathbf{x}}_i$ is $B$.

Bootstrap estimate averages over reshuffled samples.

Not examinable.

# Bootstrap: Applications in Statistics

## Estimate variance of estimators

- Since estimator $\hat{S}$ depends on (random) data, it is a random variable.
- The more this variable scatters, the less we can trust our estimate.
- If scatter is high, we can expect the values $\hat{S}_b$ to scatter as well.
- In previous example, this means: Estimating the variance of the variance estimator.

## Variance reduction

- Averaging over the individual bootstrap samples can reduce the variance in $\hat{S}$.
- In other words: $\hat{S}_{\text{BS}}$ typically has lower variance than $\hat{S}$.
- This is the property we will use for classicifation in the following.

## As alternative to cross validation

To estimate prediction error of classifier:

- For each $b$, train on $\mathcal{B}_b$, estimate risk on points not in $\mathcal{B}_b$.
- Average risk estimates over bootstrap samples.

Not examinable.

## BAGGING

### Idea

- Recall Boosting: Weak learners are deterministic, but selected to exhibit high variance.
- Strategy now: Randomly distort data set by resampling.
- Train weak learners on resampled training sets.
- Resulting algorithm: **Bagging** (= **B**ootstrap **agg**regation)

## Representation of class labels

For Bagging with $K$ classes, we represent class labels as vectors:

$$\mathbf{x}_i \text{ in class } k \qquad \text{as} \qquad y_i = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \begin{array}{l} \\ \\ \\ \longleftarrow k\text{th entry} \\ \\ \\ \\ \end{array}$$

This way, we can average together multiple class labels:

$$\frac{1}{n}(y_1 + \ldots + y_n) = \begin{pmatrix} p_1 \\ \vdots \\ p_k \\ \vdots \\ p_K \end{pmatrix}$$

We can interpret $p_k$ as the probability that one of the $n$ points is in class $k$.

# BAGGING: ALGORITHM

## Training

For $b = 1, \dots, B$:

1. Draw a bootstrap sample $\mathcal{B}_b$ of size $n$ from training data.
2. Train a classifier $f_b$ on $\mathcal{B}_b$.

## Classification

- Compute

$$f_{\text{avg}}(\mathbf{x}) := \frac{1}{B} \sum_{b=1}^{B} f_b(\mathbf{x})$$

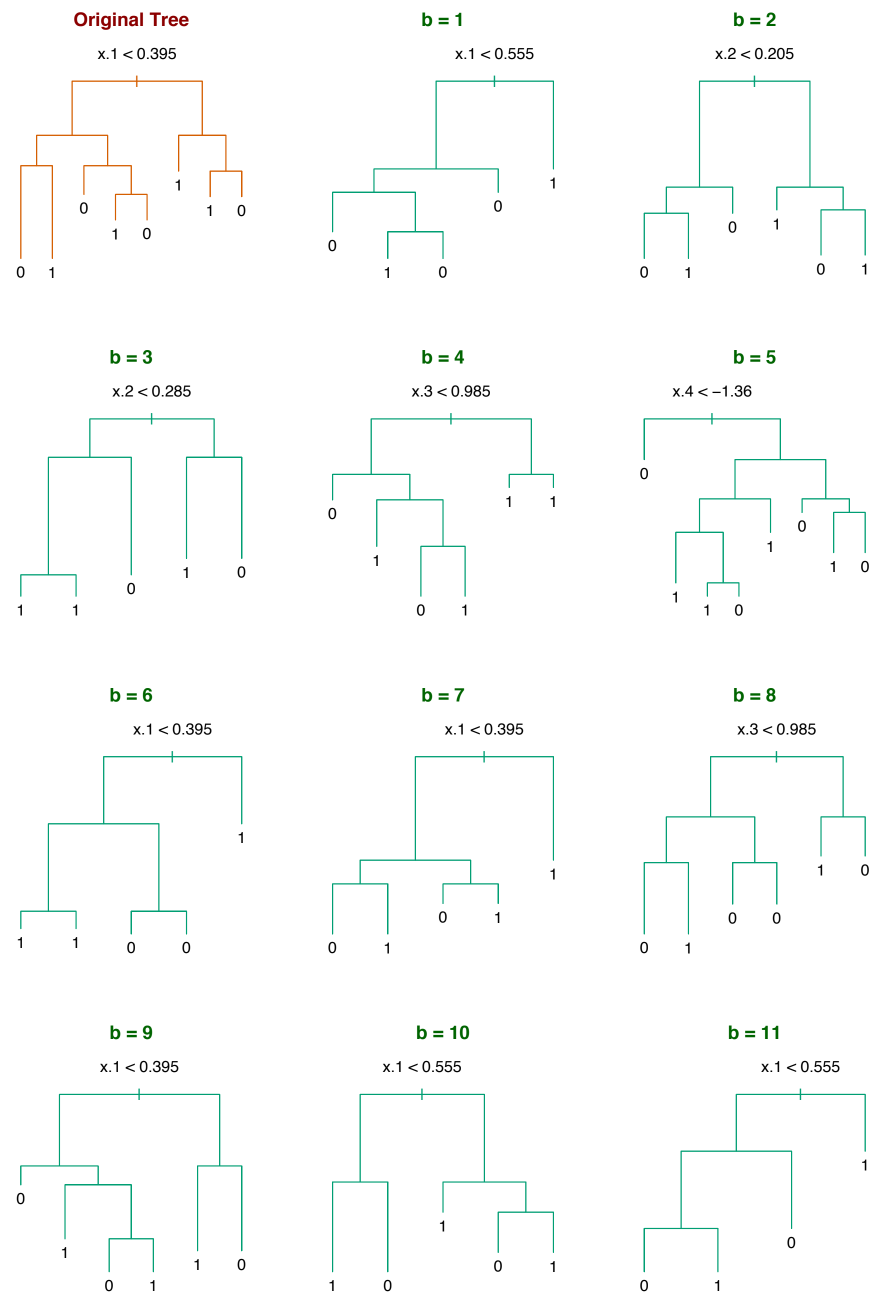
  This is a vector of the form $f_{\text{avg}}(\mathbf{x}) = (p_1(\mathbf{x}), \dots, p_k(\mathbf{x}))$.

- The Bagging classifier is given by

$$f_{\text{Bagging}}(\mathbf{x}) := \arg\max_k \{p_1(\mathbf{x}), \dots, p_k(\mathbf{x})\} \ ,$$

  i.e. we predict the class label which most weak learners have voted for.

## EXAMPLE: BAGGING TREES



- Two classes, each with Gaussian distribution in $\mathbb{R}^5$.
- Note the variance between bootstrapped trees.

## Bagging vs. Boosting

- Bagging works particularly well for trees, since trees have high variance.
- Boosting typically outperforms bagging with trees.
- The main culprit is usually dependence: Boosting is better at reducing correlation between the trees than bagging is.

## Random Forests

Modification of bagging with trees designed to further reduce correlation.

- Tree training optimizes each split over all dimensions.
- Random forests choose a different subset of dimensions *at each split*.
- Optimal split is chosen within the subset.
- The subset is chosen at random out of all dimensions $\{1, \dots, d\}$.

## Random Forests: Algorithm

### Training

Input parameter: $m$ (positive integer with $m < d$)

For $b = 1, \dots, B$:

1. Draw a bootstrap sample $\mathcal{B}_b$ of size $n$ from training data.
2. Train a tree classifier $f_b$ on $\mathcal{B}_b$, where each split is computed as follows:
    - Select $m$ axes in $\mathbb{R}_d$ at random.
    - Find the best split $(j^*, t^*)$ on this subset of dimensions.
    - Split current node along axis $j^*$ at $t^*$.

### Classification

Exactly as for bagging: Classify by majority vote among the $B$ trees. More precisely:

- Compute $f_{\text{avg}}(\mathbf{x}) := (p_1(\mathbf{x}), \dots, p_k(\mathbf{x})) := \frac{1}{B}\sum_{b=1}^{B} f_b(\mathbf{x})$
- The Random Forest classification rule is

$$f_{\text{Bagging}}(\mathbf{x}) := \arg\max_k \{p_1(\mathbf{x}), \dots, p_k(\mathbf{x})\}$$

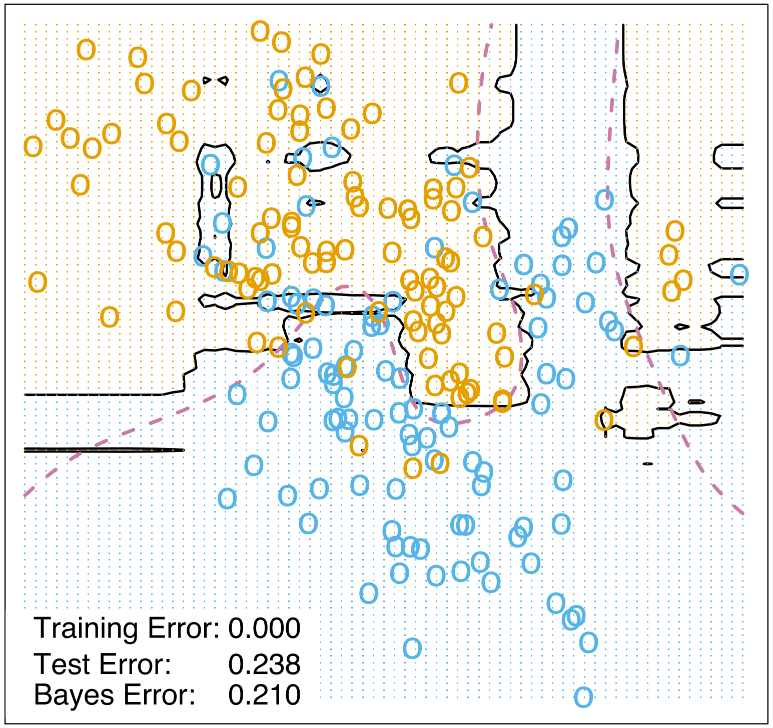
## Random Forests

### Remarks

- Recommended value for $m$ is $m = \lfloor\sqrt{d}\rfloor$ or smaller.
- RF typically achieve similar results as boosting. Implemented in most packages, often as standard classifier.

### Example: Synthetic Data

- This is the RF classification boundary on the synthetic data we have already seen a few times.
- Note the bias towards axis-parallel alignment.



## Application: Cancer Diagnosis

## Overview

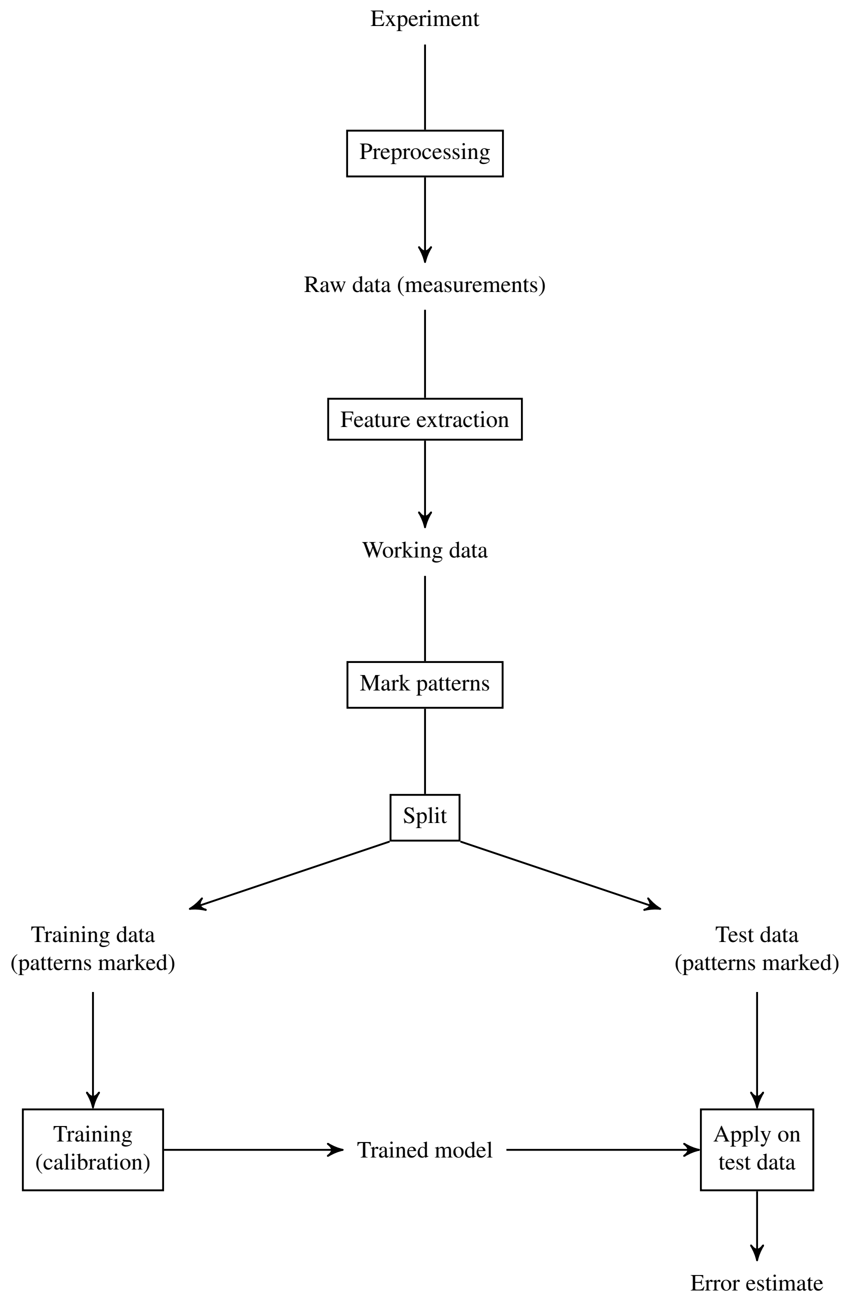
### Kidney cancer diagnosis: Clinical procedure

- Take tissue sample from patient’s kidney
- Preprocess sample and photograph under microscope
- A pathologist looks at the image and diagnosis patient on scale from healthy to advanced stage cancer

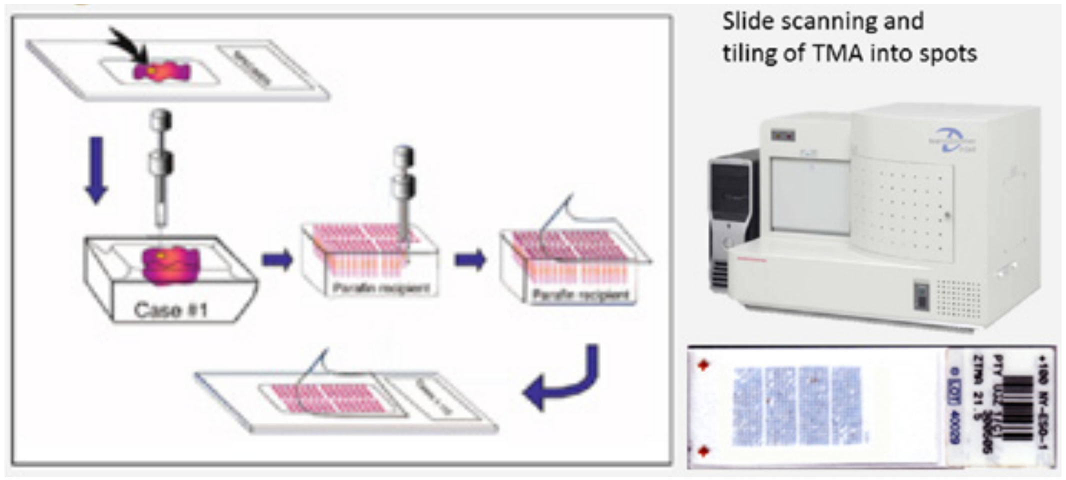
### Task

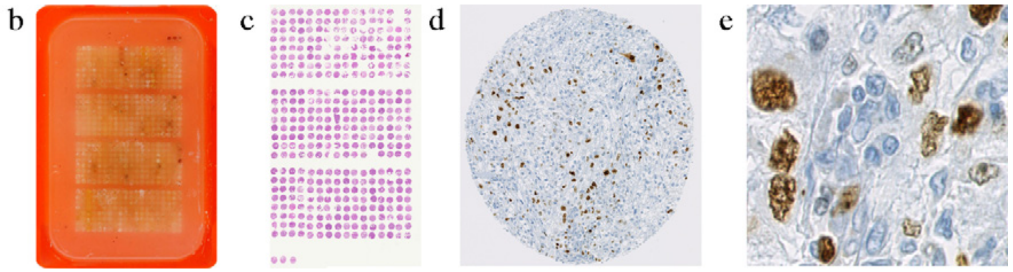
- Empirically, the results vary significantly between pathologists
- The objective is to build a classifier that produces a diagnosis using the same scale as the pathologist, hopefully with more stable results.

## Data Analysis Pipeline



# PREPROCESSING

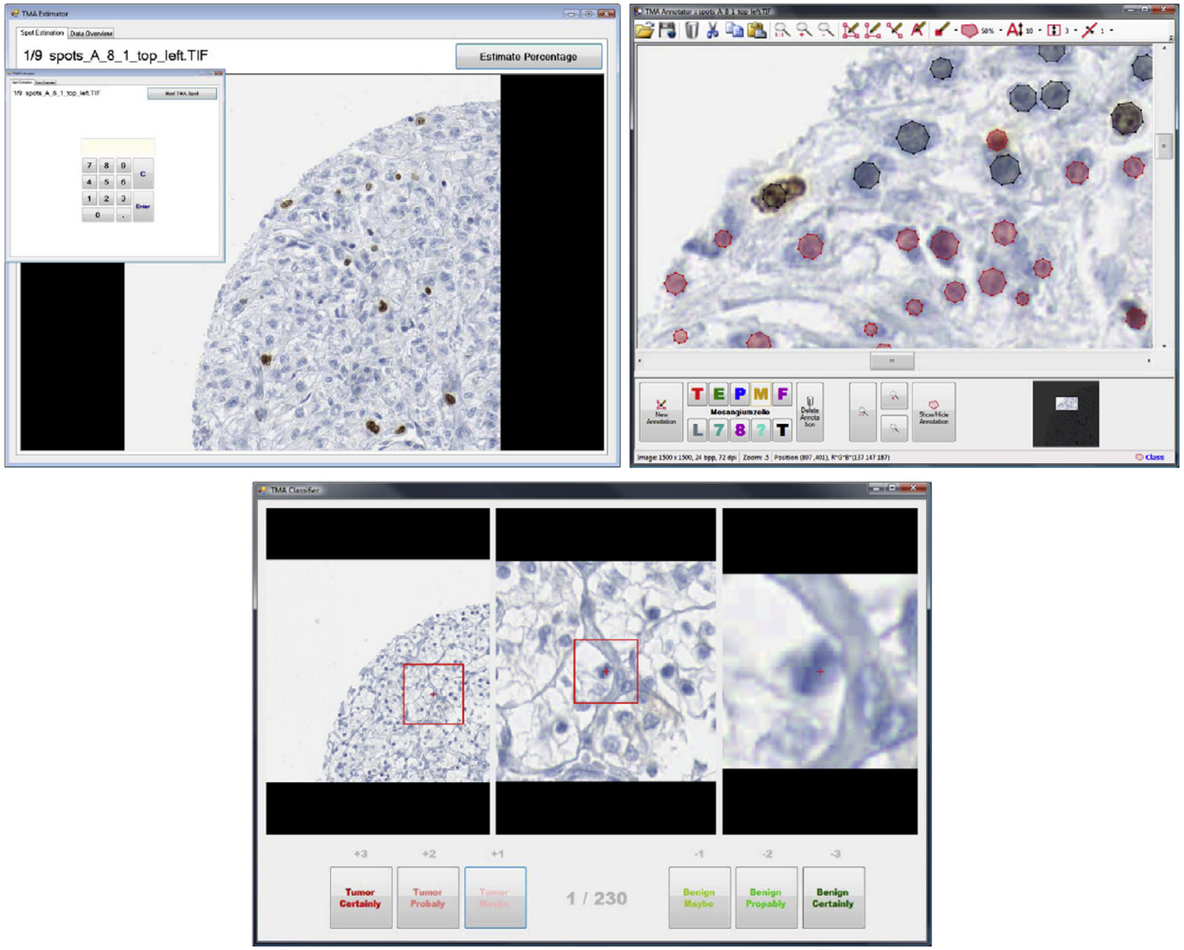




1. Tissue sample is “stained” with marking fluid
2. Thin slice is cut and placed on microscope slide
3. Sample is photographed under microscope
4. Tumor cells absorb more marker fluid and tend to be darker

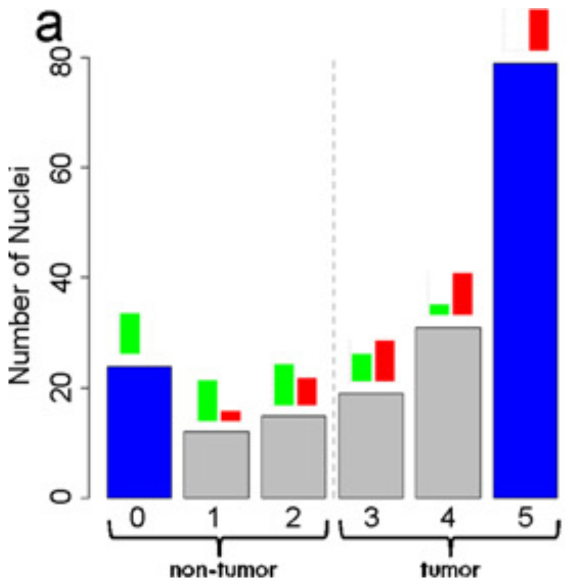
From: Fuchs & Buhmann, Computerized Medical Imaging and Graphics Vol. 35, 2011.

# LABELING



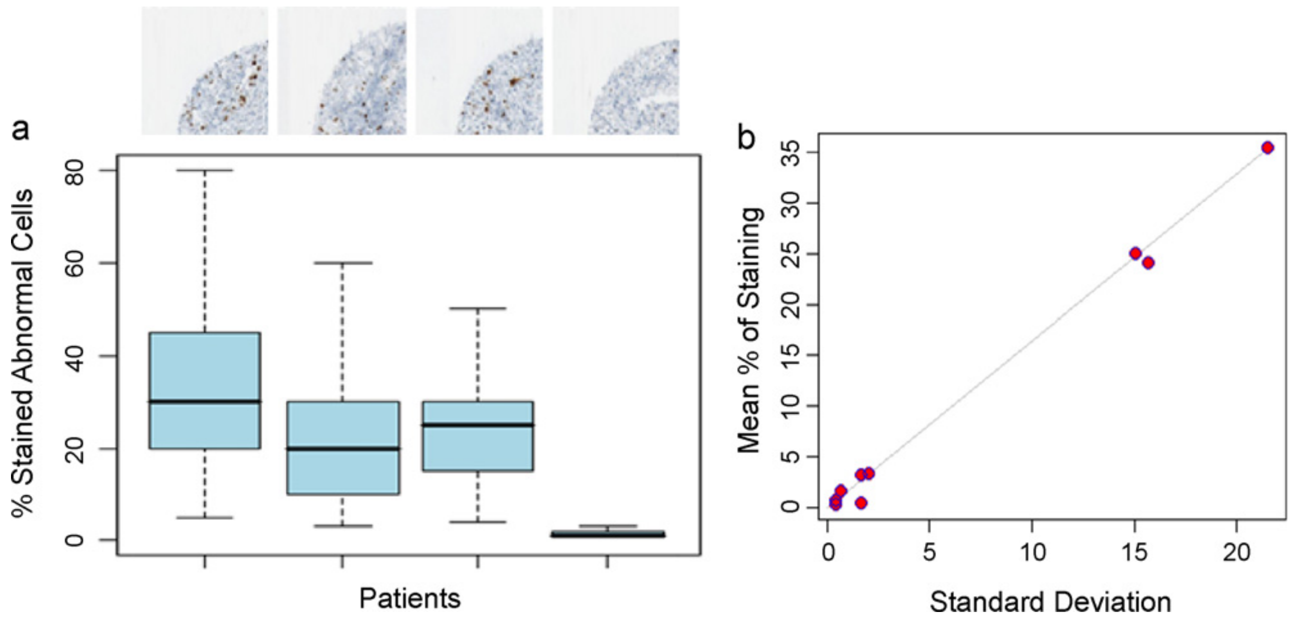
A pathologist uses a software with a graphical user interface to (1) mark the locations of nuclei and (2) label nuclei as healthy/cancerous.

## COMPARISON BETWEEN EXPERTS (1)



- Five experts label the same set of nuclei (180 in total)
- For each data point (nucleus), count the number of votes (0, . . . , 5) in favor of "tumor"
- The diagram above is a histogram of the vote counts for the 180 data points
- All five experts agree if the count is 0 (all say healthy) or 5 (all say tumor)
- (The small red/green bars are the vote proportions, so they encode the same information as the numbers at the bottom.)

## Comparison Between Experts (2)



Results for 14 pathologists labeling four patients.

- Each box in the boxplot represents one patient.
- Box plot: The line in the middle of each box is the median. The upper and lower box boundary are the third and first quartile, respectively. The horizontal bars at either end of the dashed vertical line represent one standard deviation around the mean.
- In three of the four cases, disagreement between experts is substantial.
- Plot on the right: The standard deviation increases linearly with the overall number of stained nuclei. (Roughly, the more cancer cells there are, the more volatile the diagnosis becomes.)

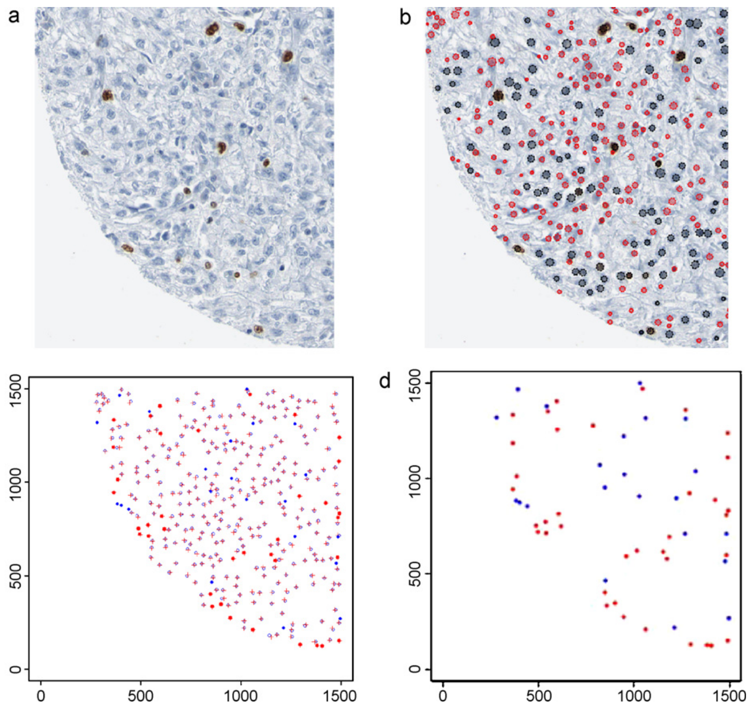
## EXAMPLE

input image

locations marked by one expert

classification by one expert

disagreement between two experts



## RANDOM FOREST CLASSIFIER

### Data

"Relative images" of individual cell nuclei, i.e. the difference between a nucleus image and a background patch from the same tissue sample image. That makes results less sensitive to variations between tissue samples.

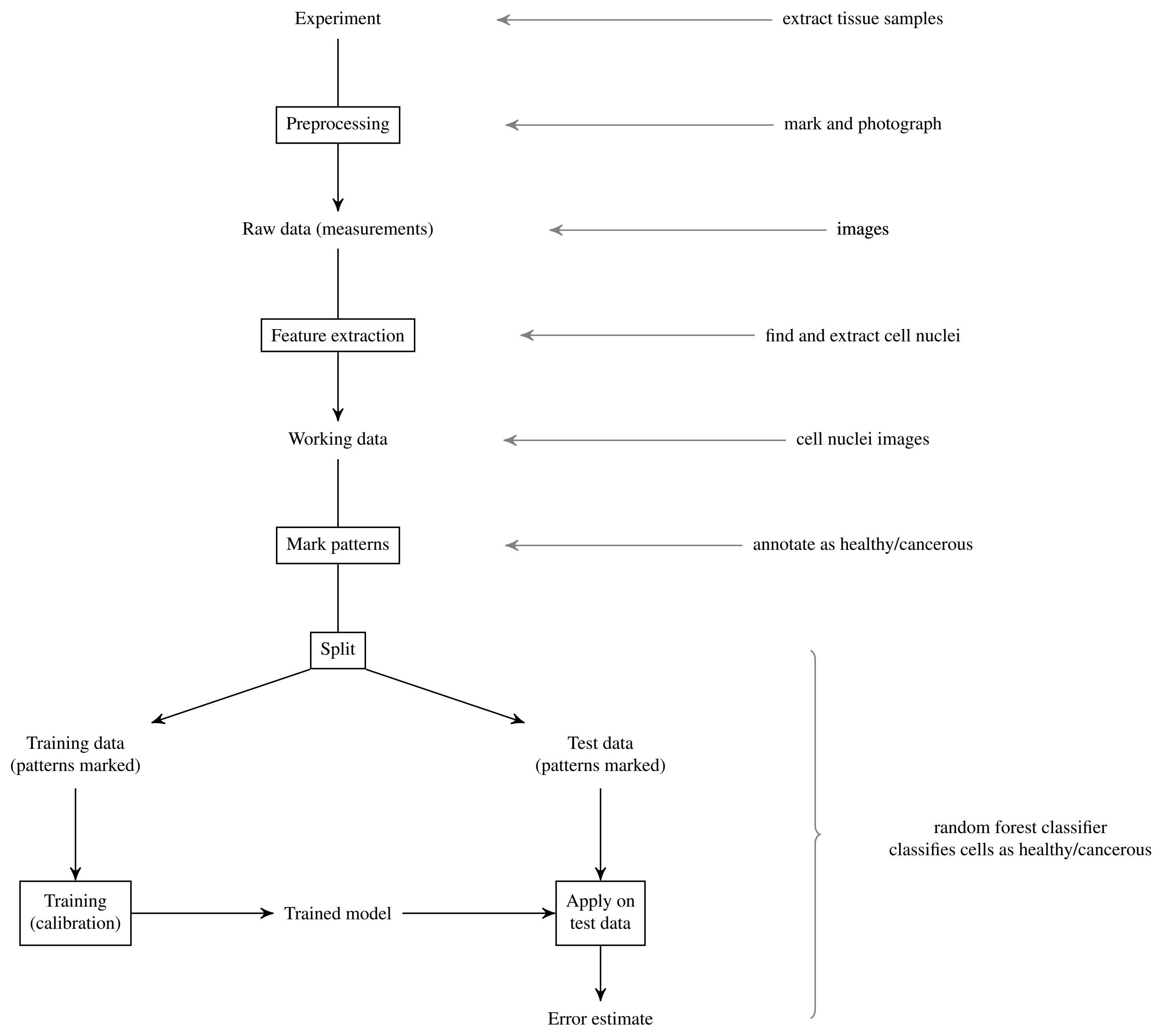
### Training

1. Nuclei are hand-labeled by pathologists.
2. A random forest classifier is trained on the relative images (as training data points) and the labels (as training labels).

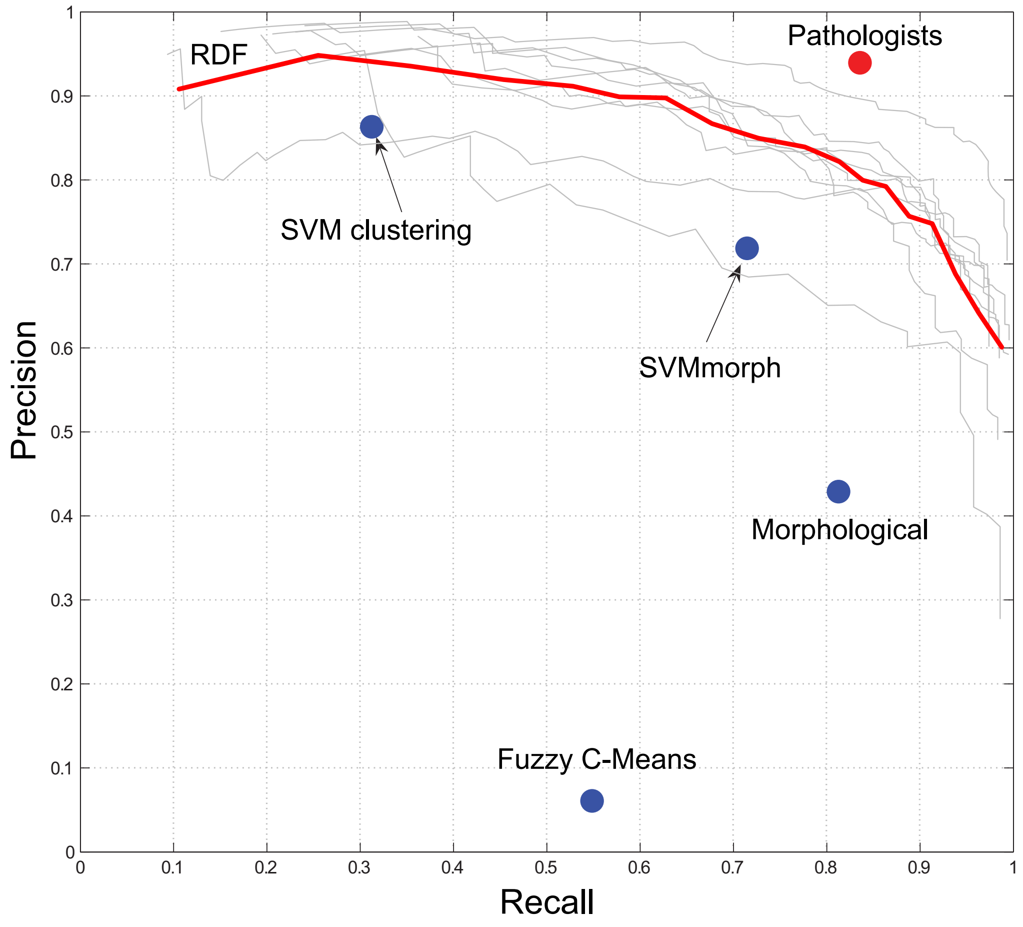
### Diagnosis of a new tissue sample image

1. Input: Entire tissue sample image.
2. Find nuclei using an image segmentation algorithm.
3. Extract subimages of these nuclei and apply random forest classifier.
4. Diagnose according to ratio of healthy to cancerous cells.

# DATA ANALYSIS PIPELINE



# Results



Precision/Recall plot for the random forest method (“RDF”) compared to other classifiers. The “true label” for each data point is a randomly selected pathologist. The performance of pathologists (red dot) is the average of the aggregate result for all remaining pathologists.

## REMARKS

This application illustrates a number of challenges encountered in applications:

- Generating label information is work-intensive.
- The comparison experiments show that the training/test labels themselves have limited reliability.
- These methods are now several years old. Neural networks developed in the last few years might be able to improve the feature extraction step. (More on neural networks and feature extraction later.)