BIDS-Training 2024

ScaDS.AI DRESDEN LEIPZIG

CENTER FOR SCALABLE DATA ANALYTICS AND ARTIFICIAL INTELLIGENCE

Day 2, Session 3: Machine Learning for Pixel and Object Segmentation

SPEAKER: Christian Martin, Anja Neumann

DATE: 14-05-2024

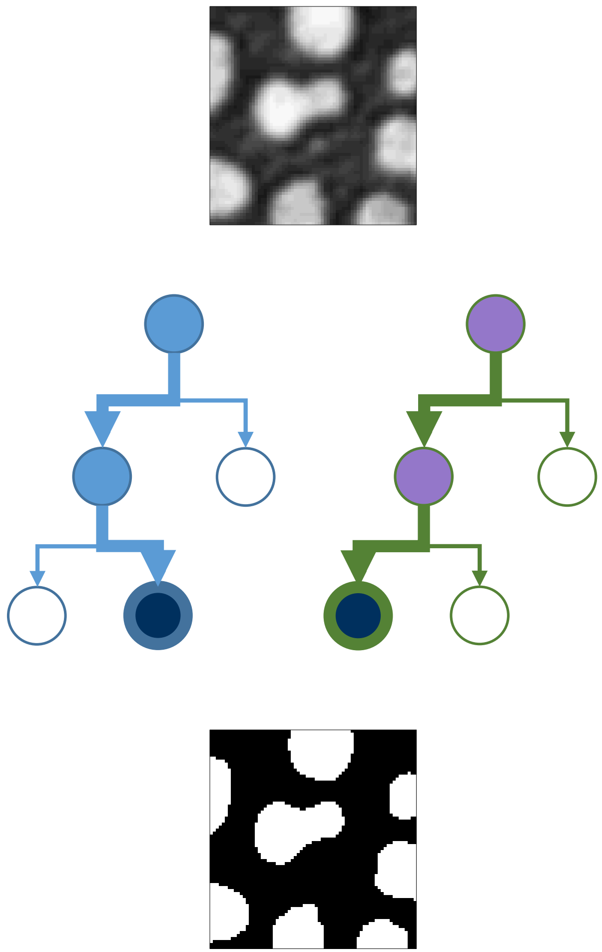
Overview

Machine Learning

- Introduciton
- Decision Tree / Random Forest
- Image Segmentation using thresholding
- Image Segmentation using machine learning
- Object classification
- Segmentation quality
- Model validation
- Outlook

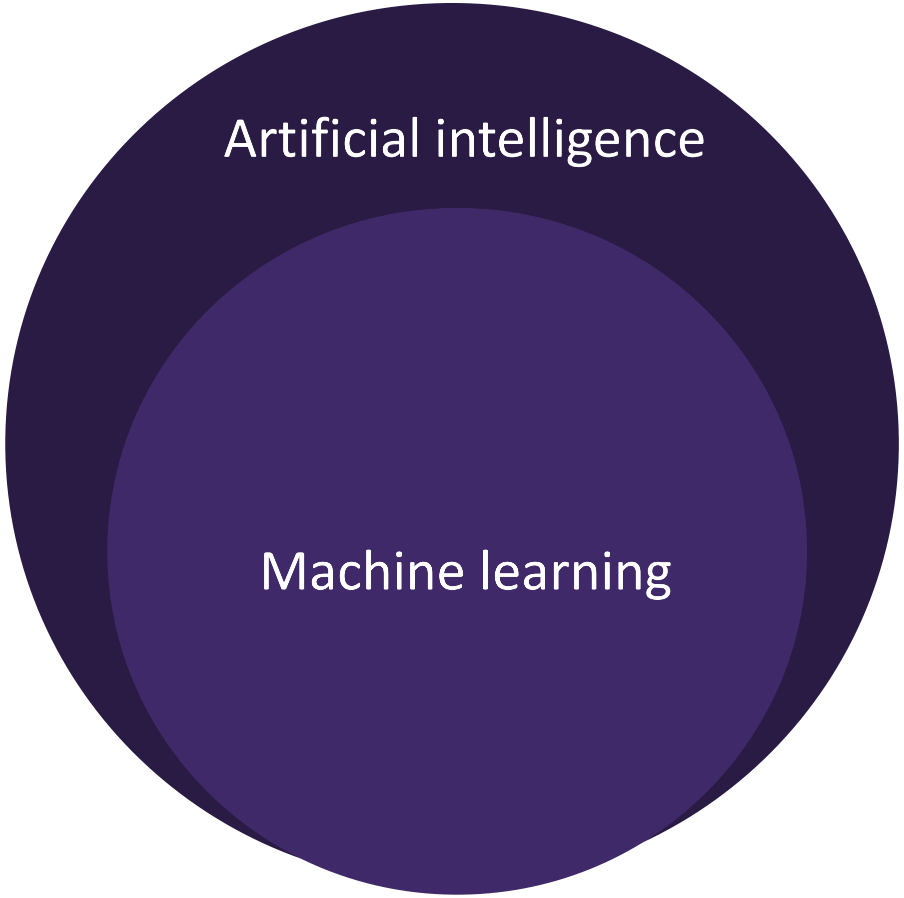
Practical part with Python

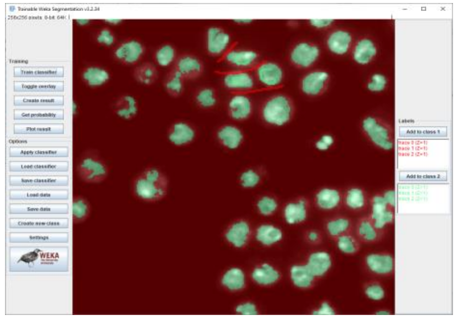
- scikit-learn / napari
- Accelerated pixel and object classification (APOC)



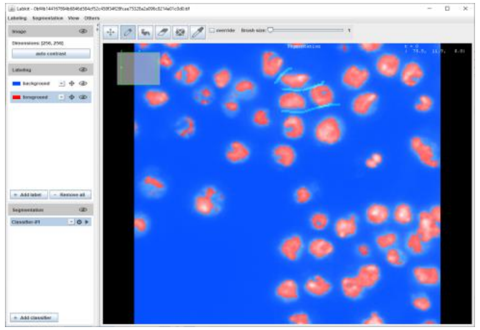
Machine learning

- A research field in computer science
- Finds more and more applications, also in life sciences.

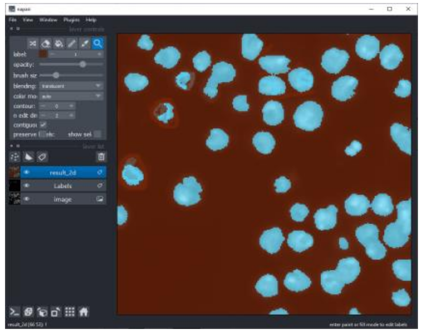




Trainable Weka Segmentation
https://imagej.net/plugins/tws/



LabKit
https://imagej.net/plugins/labkit/

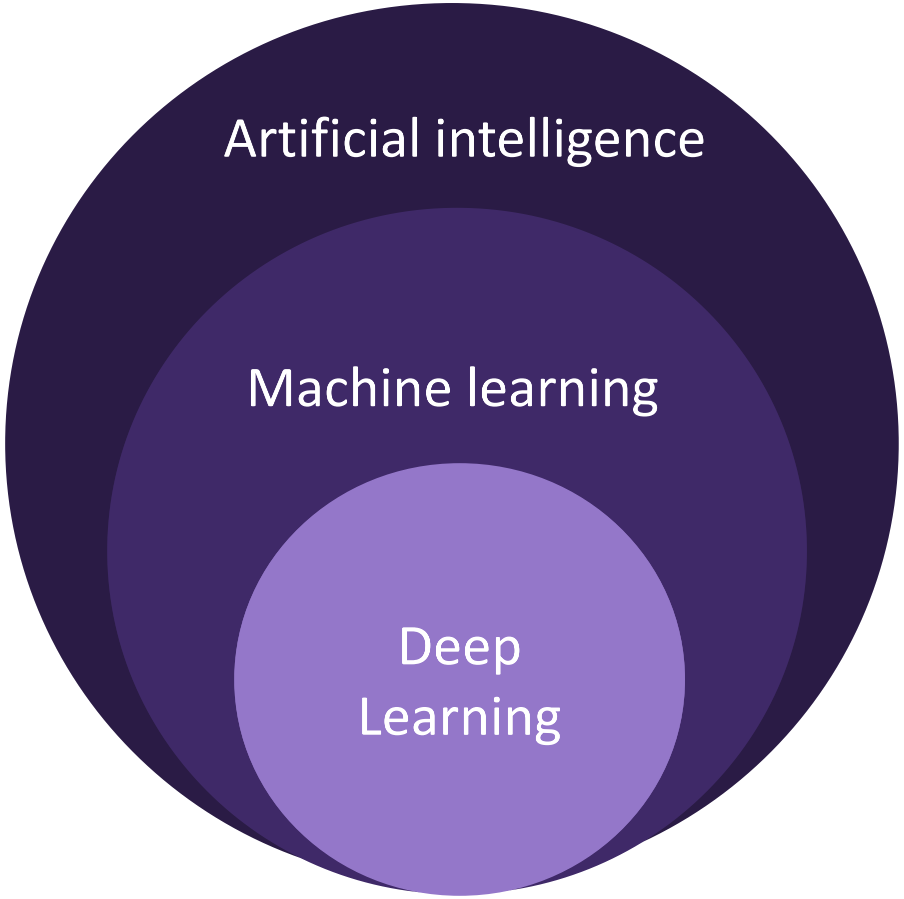


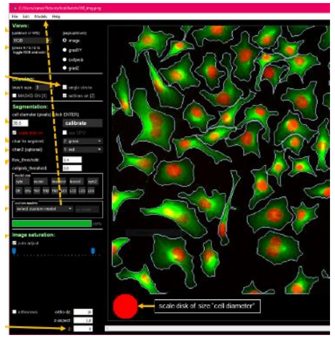
Python/scikit-learn/napari/apoc

Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

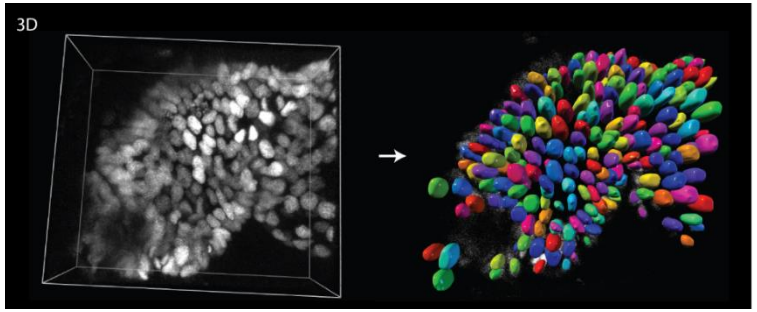
Machine learning

- A research field in computer science
- Finds more and more applications, also in life sciences.

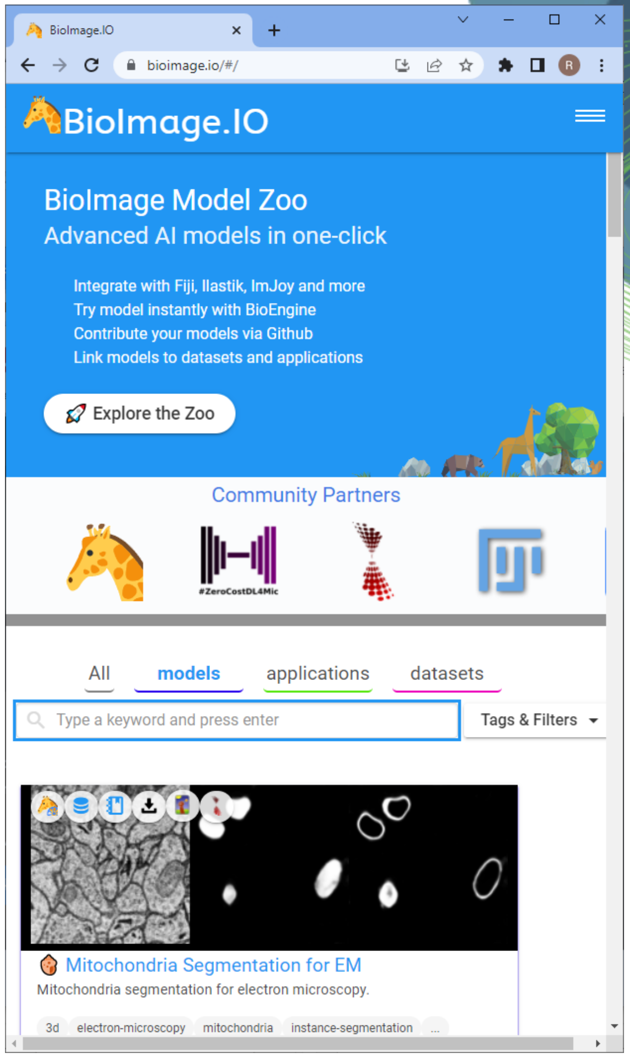




www.cellpose.org/



https://github.com/stardist/stardist



https://bioimage.io/

Logos and screenshots are taken from the github repositories / websites provided under BSD and MIT licenses.

Machine Learning

Machine Learning

- subfield of Artificial Intelligence
- Automatic construction of predictive models from given data
- Learning from Data (data-driven approach)
- Input Data: m items of n dimensions
- If available, ground truth for each item
  → classified data

id	dim1	dim 2	…	dim n	class
1	69	23.5	…	4.3	A
2	54	27.4	…	2.7	C
3	81	22.4	…	5.2	B
4	72	31.5	…	1.5	C
5	69	25.4	…	4.8	A
…	…	…	…	…	…
m	78	15.7	…	5.1	C

Main Topics

- Data preprocessing
    - Annotation
    - Missing Values
- Unsupervised Learning
    - Clustering
    - Data Visualization
- Supervised Learning
    - Classification (predict a class)
    - Regression (predict a value)
- Feature Engineering
- Feature Selection
- Dimension Reduction / Embedding

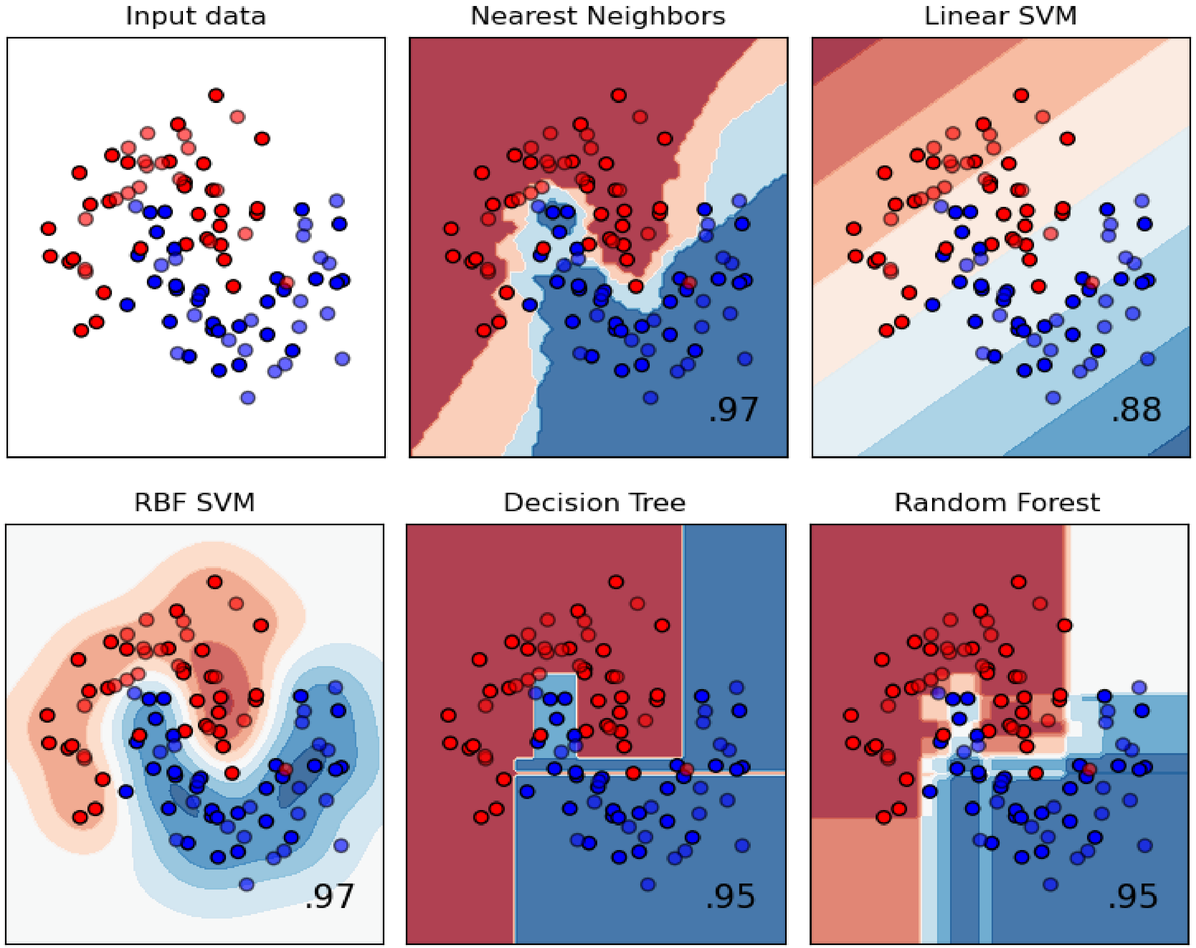
Machine Learning

Supervised Learning

- Train model on training data
  - paint feature space
- Evaluate model on test data
  - estimate class from position of sample in feature space
- Apply model on new data

Supervised Learning Methods

- k-nearest neighbor (knn)
- Linear Regression
- Logistic Regression
- Support Vector Machines (SVM)
- Decision Trees / Random Forests
- Gaussian Process
- Naïve Bayes
- Neural Networks
- …



Adapted from https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

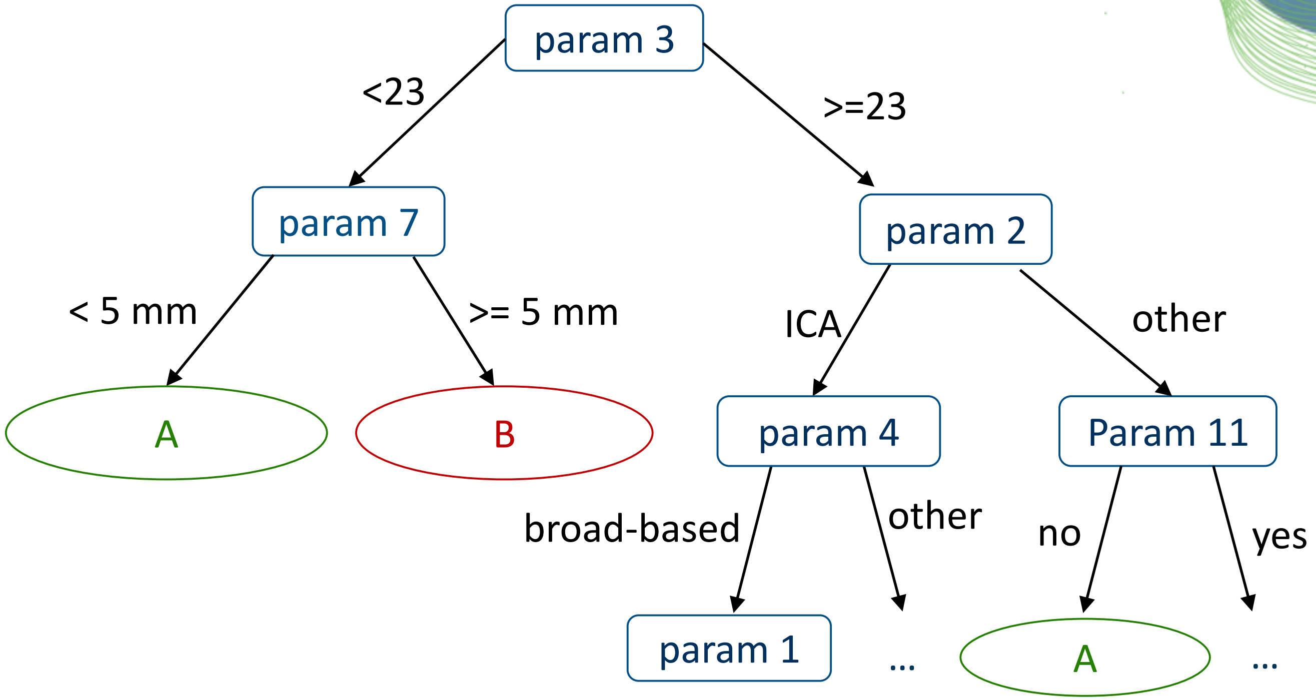
Decision Tree

Introduction

- machine learning algorithm
- data can be interval-scaled, categorical, or mixed
- classification: predict a class
- regression: predict a value
- shows good performance on tabular data
  (5-100 parameters, 50-1000 data points)
- model (tree) is computed based on training data

Preparation

- Divide data in training data / test data
- Use 5-fold cross validation
  - 4/5 of data is training data
  - 1/5 of data is test data
  - Repeat 5 times
- Never train and test trained model on same data!



Decision Tree

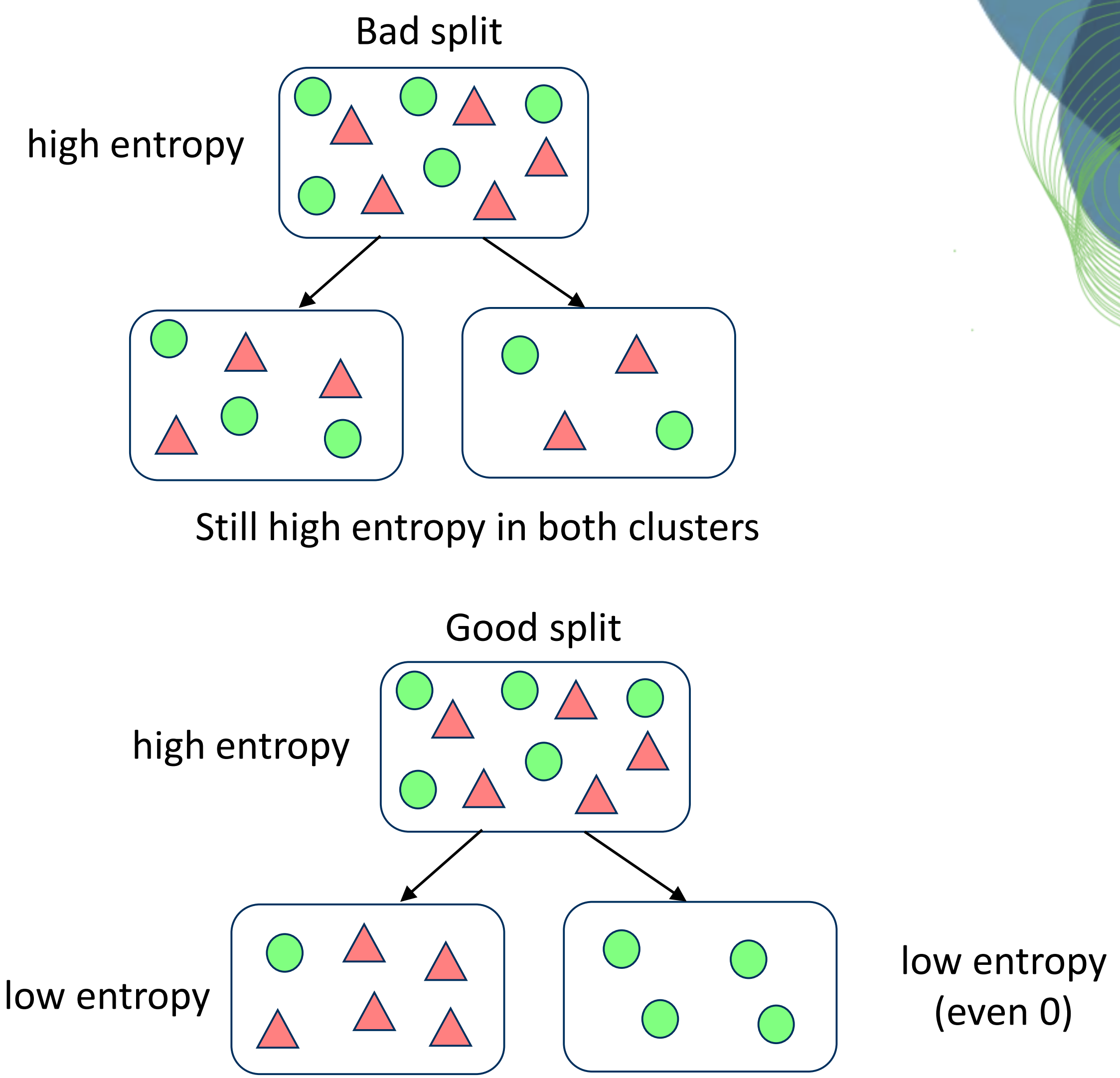
Decision Tree - Training

Training

- Start with complete data of training dataset
- For each step
  - choose parameter and threshold to minimize entropy in remaining clusters (leaves in tree)
  - Split cluster accordingly
- Entropy
  - measure for disorder
- Stopping criteria
  - Maximal depth reached (e.g. 10)
  - Minimal samples in leaf reached (e.g. 5)

Classification / Application

- Apply tree on
  - test data (for testing) or
  - new data (for application)



Decision Trees and Random Forests

Drawback of Decision Trees

- Problem
  - allow few levels → only few parameters are considered
  - allow many levels → overfitting
- Solution: Random Forests

Random Forests

- Idea: train many decision trees with part of the data
- for each tree
  - use only part of the data items
  - use only part of the parameters
- Train n different trees ()
- Result: n slightly different decision trees
- Application: combine results using majority-voting

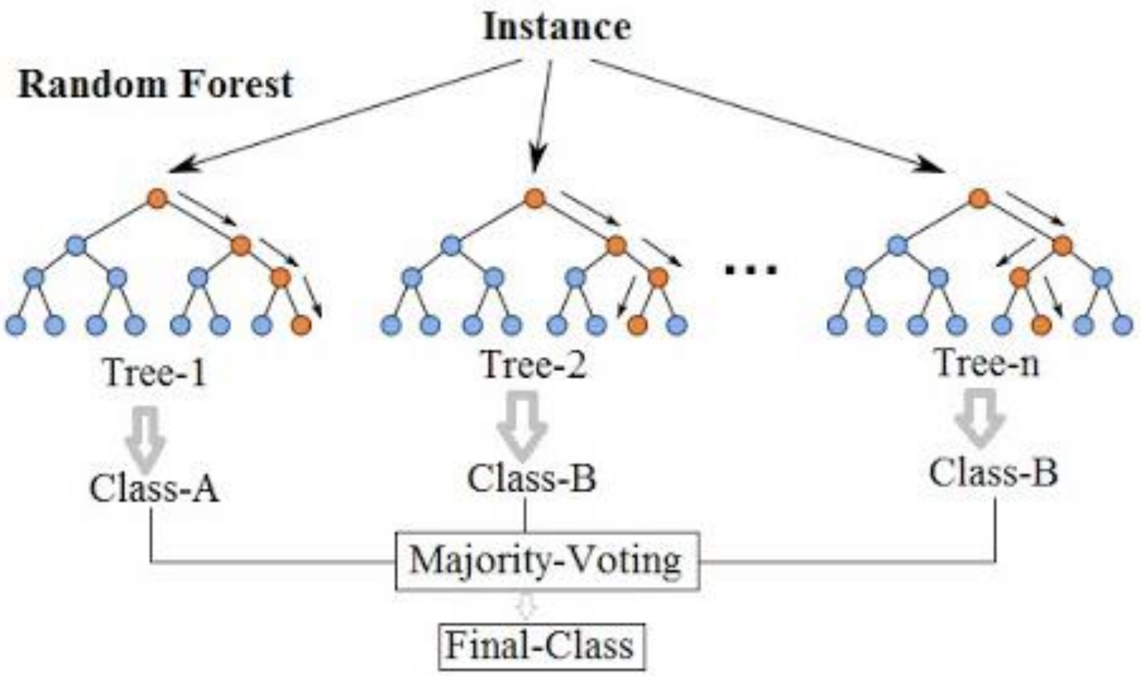


Image Segmentation

Image segmentation using thresholding

- Recap: Finding the right workflow towards a good segmentation takes time

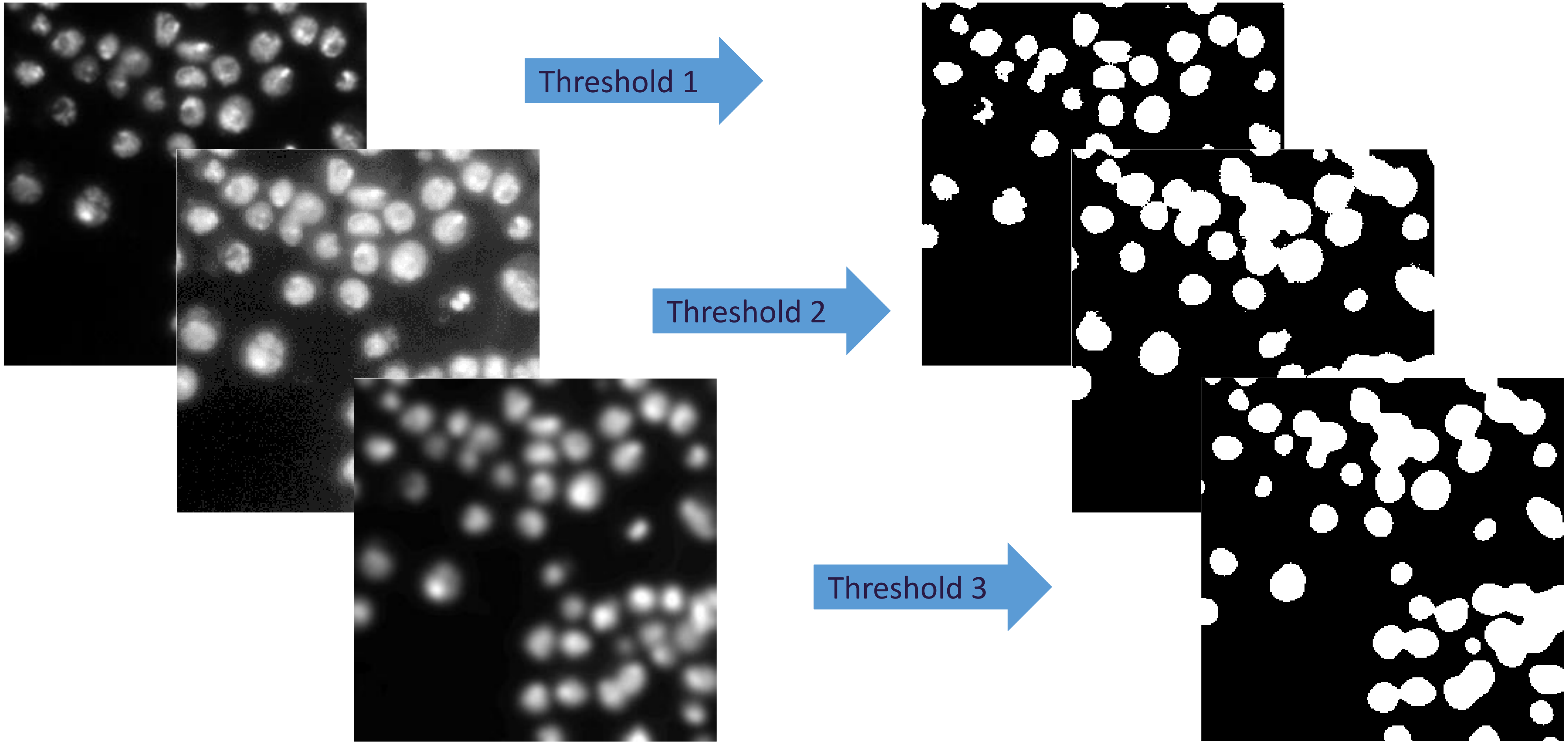


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

Image segmentation using thresholding

- Recap: Recap: Combining images, e.g. using Difference of Gaussian (DoG)

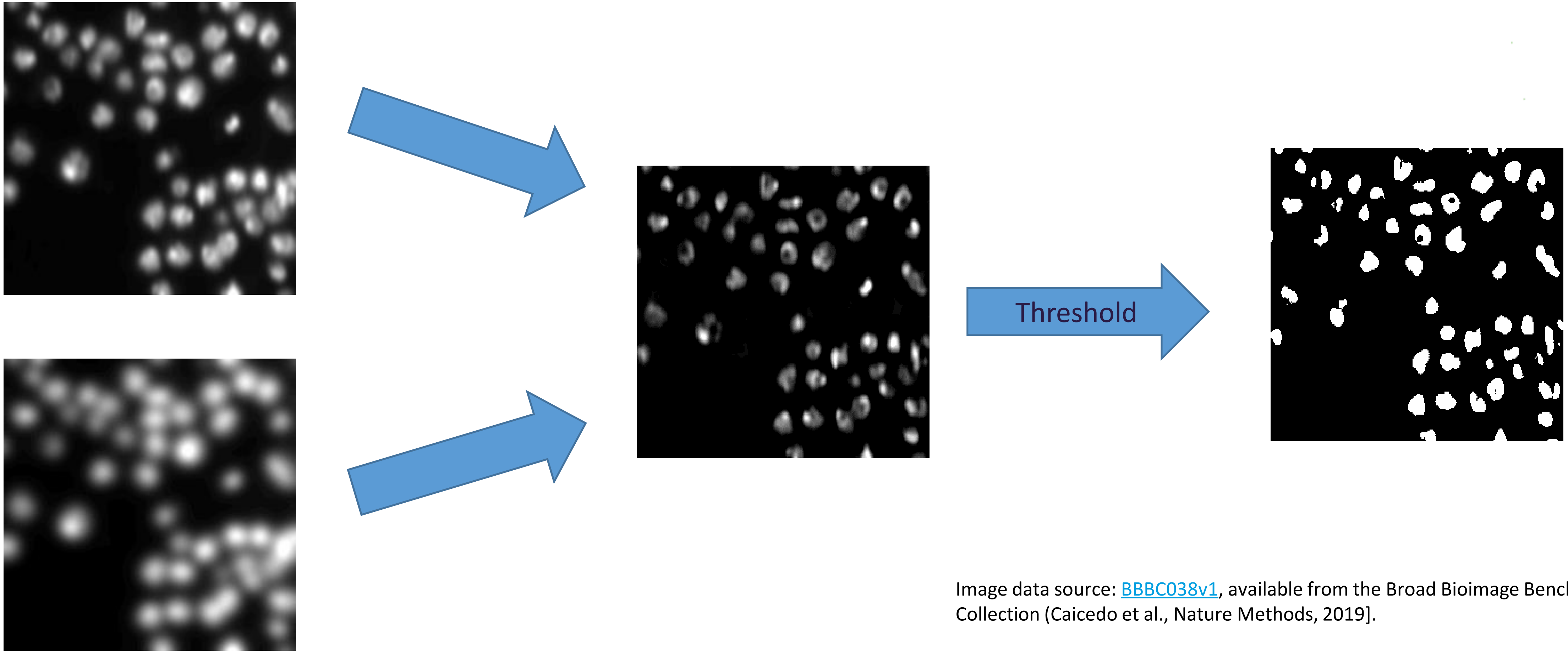


Image data source: [BBBC038v1](), available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

Image segmentation using thresholding

- Might there be a technology for optimization which combination of images can be used to get the best segmentation result?

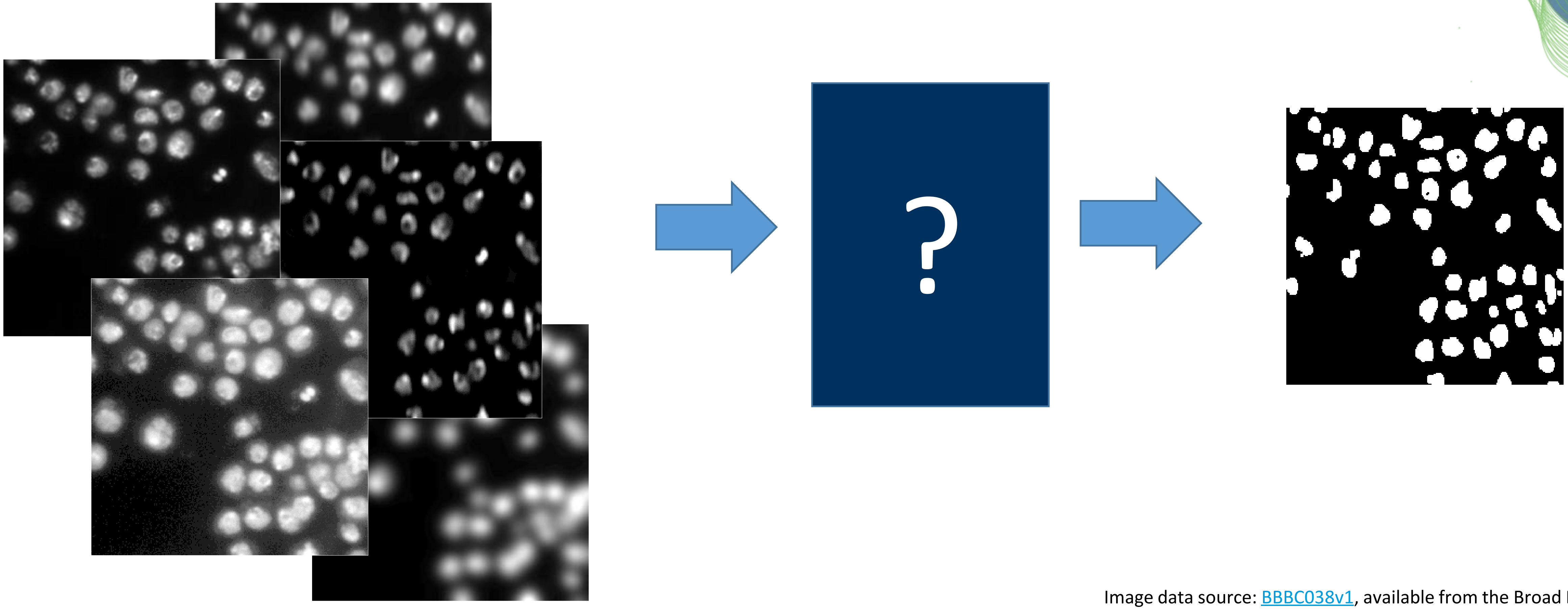


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

Image segmentation using thresholding

- Might there be a technology for optimization which combination of images can be used to get the best segmentation result?

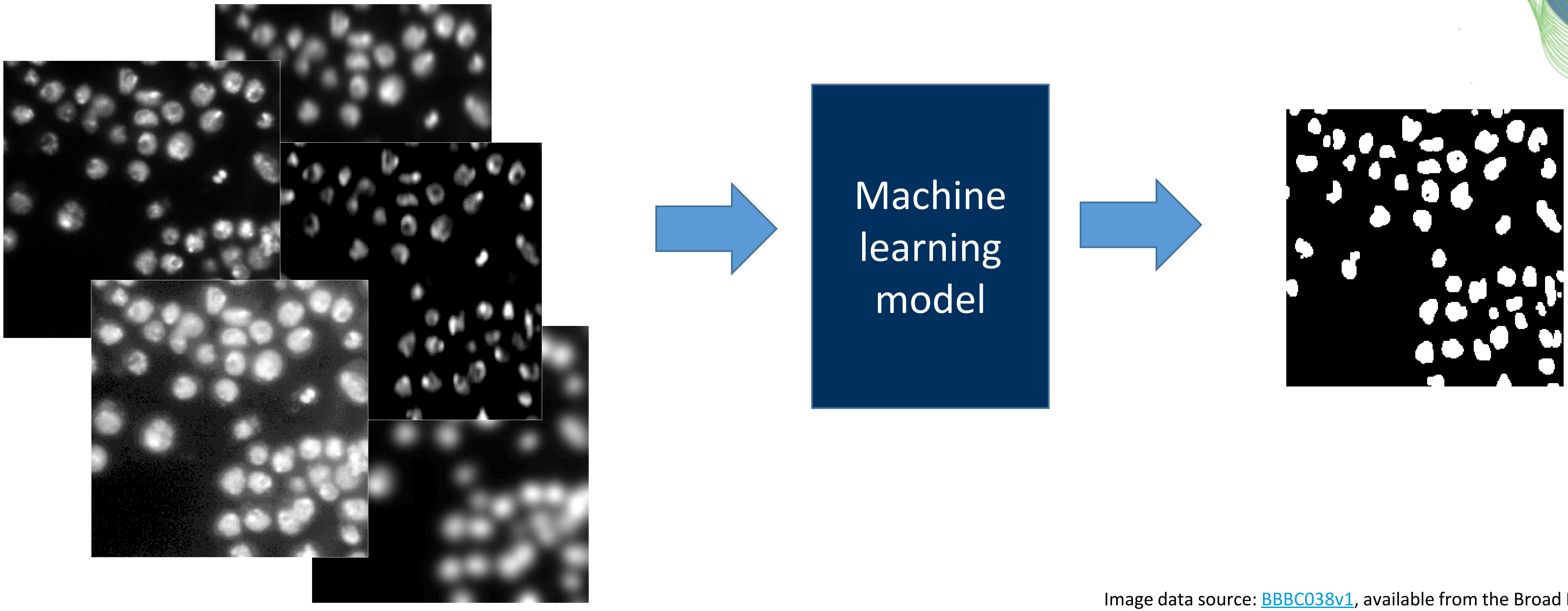


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

Machine learning for image segmentation

- *Supervised* machine learning: We give the computer some ground truth to learn from
- The computer derives a *model* or a *classifier* which can judge if a pixel should be foreground (white) or background (black)
- Example: Binary classifier

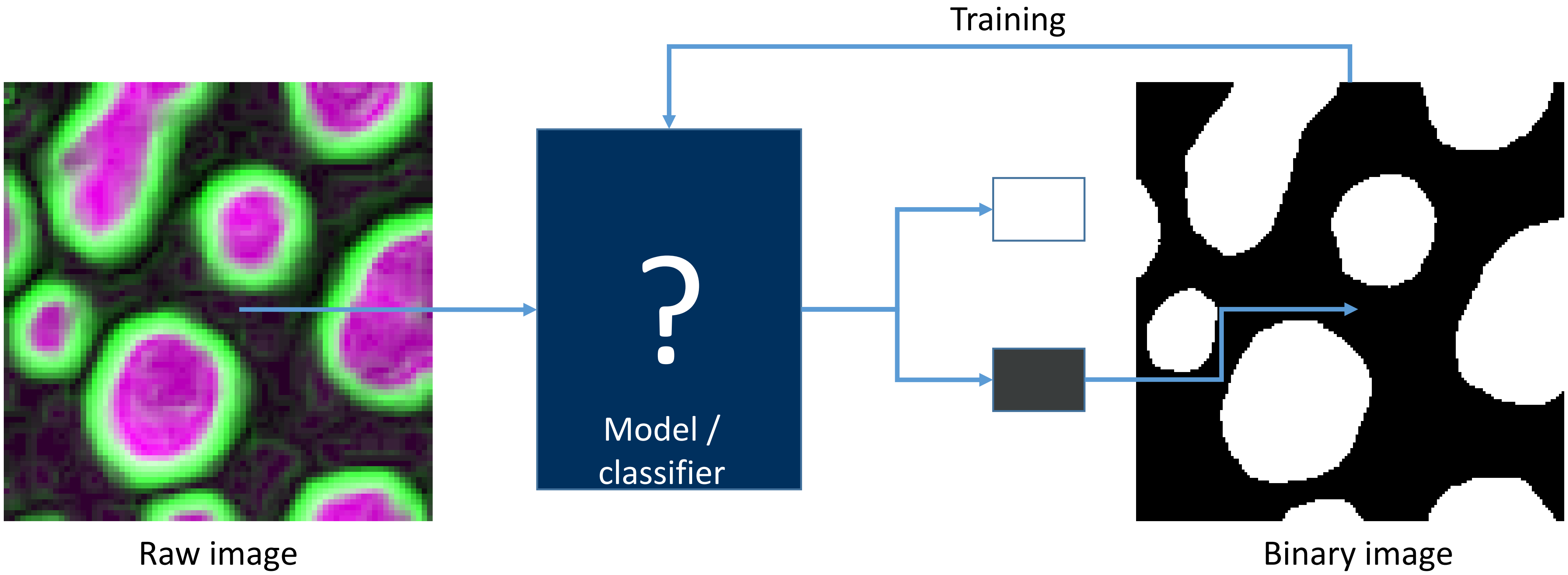
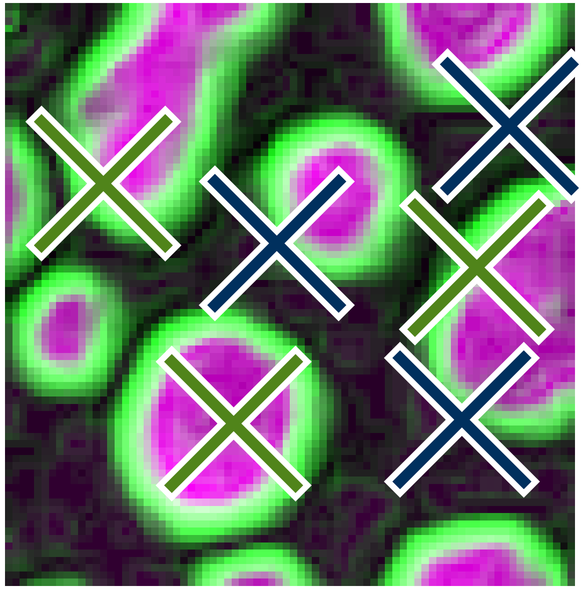
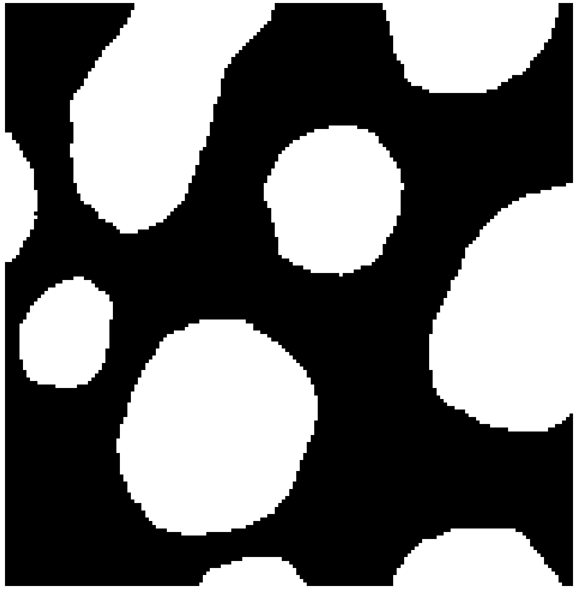


Image segmentation using pixel classification

- Idea: use different features of a pixel to classify it to background or foreground
- Each pixel is considered separately
- Features:
  - Intensity/color of original pixel
  - Gaussian blur image
  - DoG image
  - LoG image
  - Hessian
- Features from different images
- For efficient processing, we randomly *sample* our dataset
- Create a dataset with pixel features vectors that belong to the background and the foreground
- Use machine learning (e.g. Random Forest) to classify each pixel





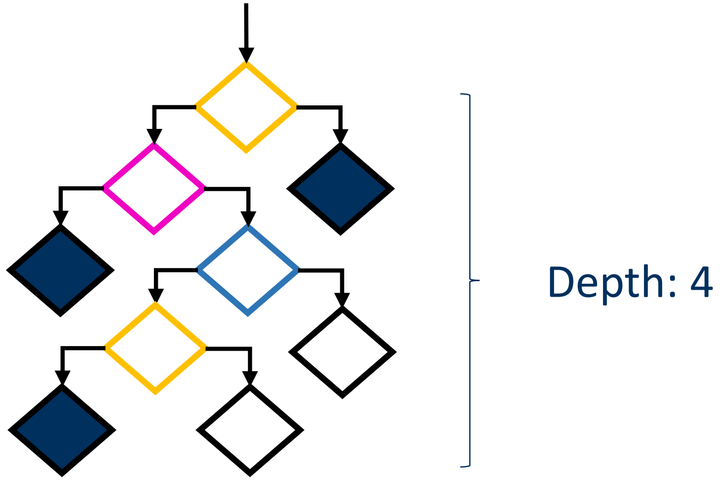
Background

Foreground

Random Forest Pixel Classifier

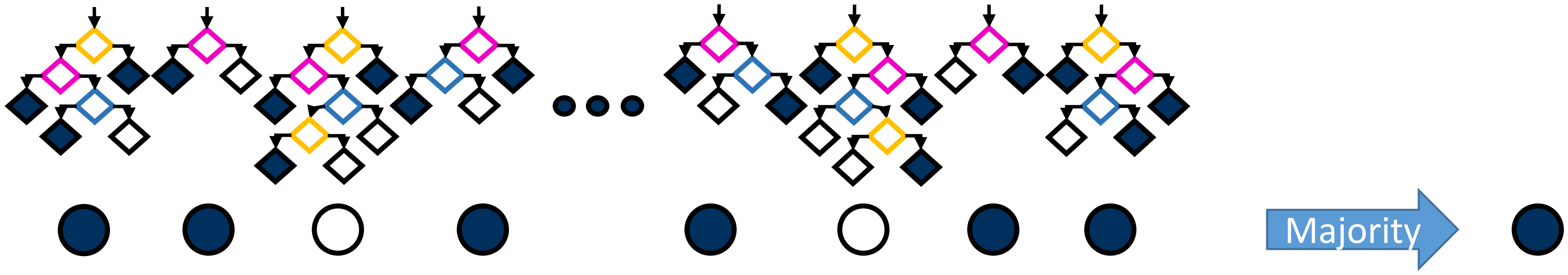
Available features: > 20

- Gaussian blur image
- DoG image
- LoG image
- Hessian
- ....



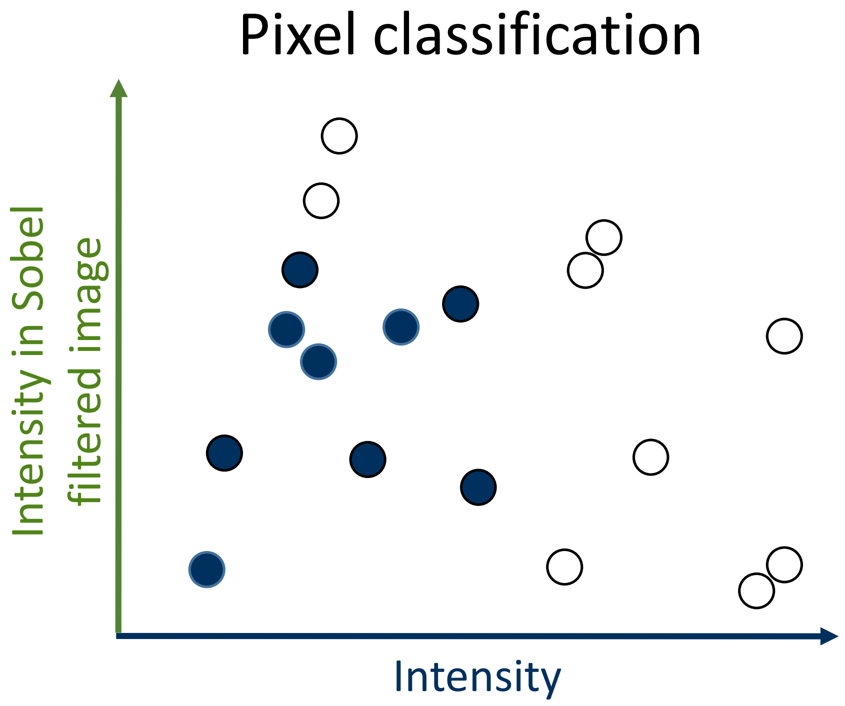
- Train t trees on selected features and sampled pixels -> t different trees
- Combination of different tree decisions by max/mean voting

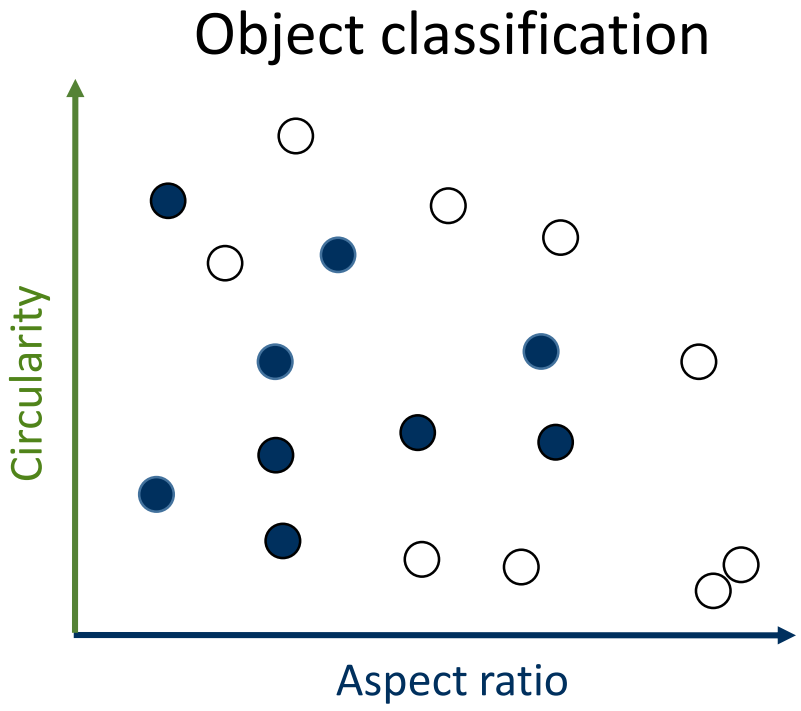
t > 100 trees

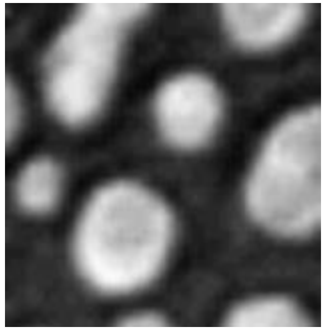


Object classification

- Use object features instead of pixel features (e.g. size, aspect ratio, shape, circularity)
- The algorithms work the same

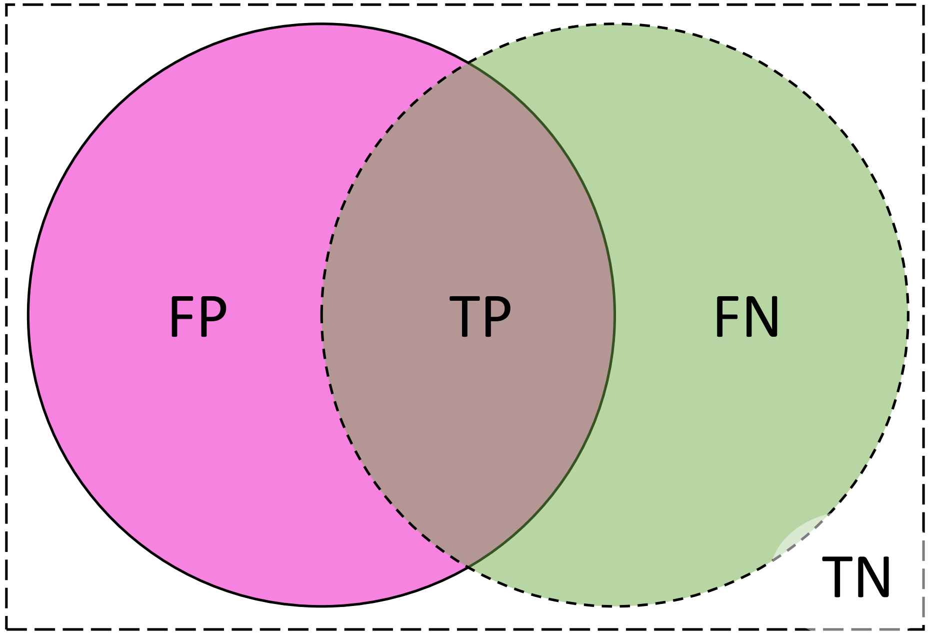






Segmentation quality estimation

- In general
  - Define what's positive and what's negative.
  - Compare with a reference to figure out what was true and false
  - Welcome to the <u>Theory of Sets</u>



A Prediction A

B Reference B (ground truth)

ROI Region of interest

TP True-positive

FN False-negative

FP False-positive

TN True-negative

Overlap (a.k.a. Jaccard index) $\frac{TP}{FP + TP + FN}$ How much do A and B overlap?

Precision $\frac{TP}{TP + FP}$ What fraction of points that were predicted as positives were really positive?

Recall (a.k.a. sensitivity) $\frac{TP}{TP + FN}$ What fraction of positives points were predicted as positives?

Model validation

- A good classifier is trained on a hand full of datasets and works on thousands similarly well.
- In order to assess that, we split the ground truth into two set
  - Training set (80% of the available data)
  - Test set (20% of the available data)

Typically done with hundreds or thousands of cells / images / objects / whatever.

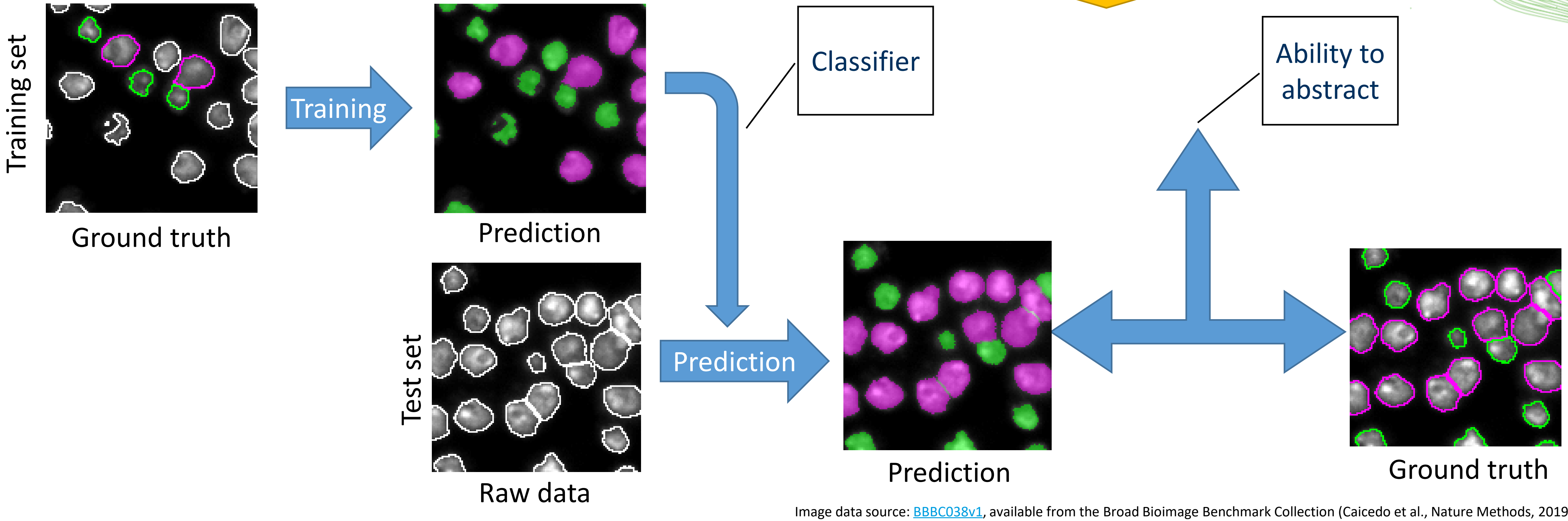


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

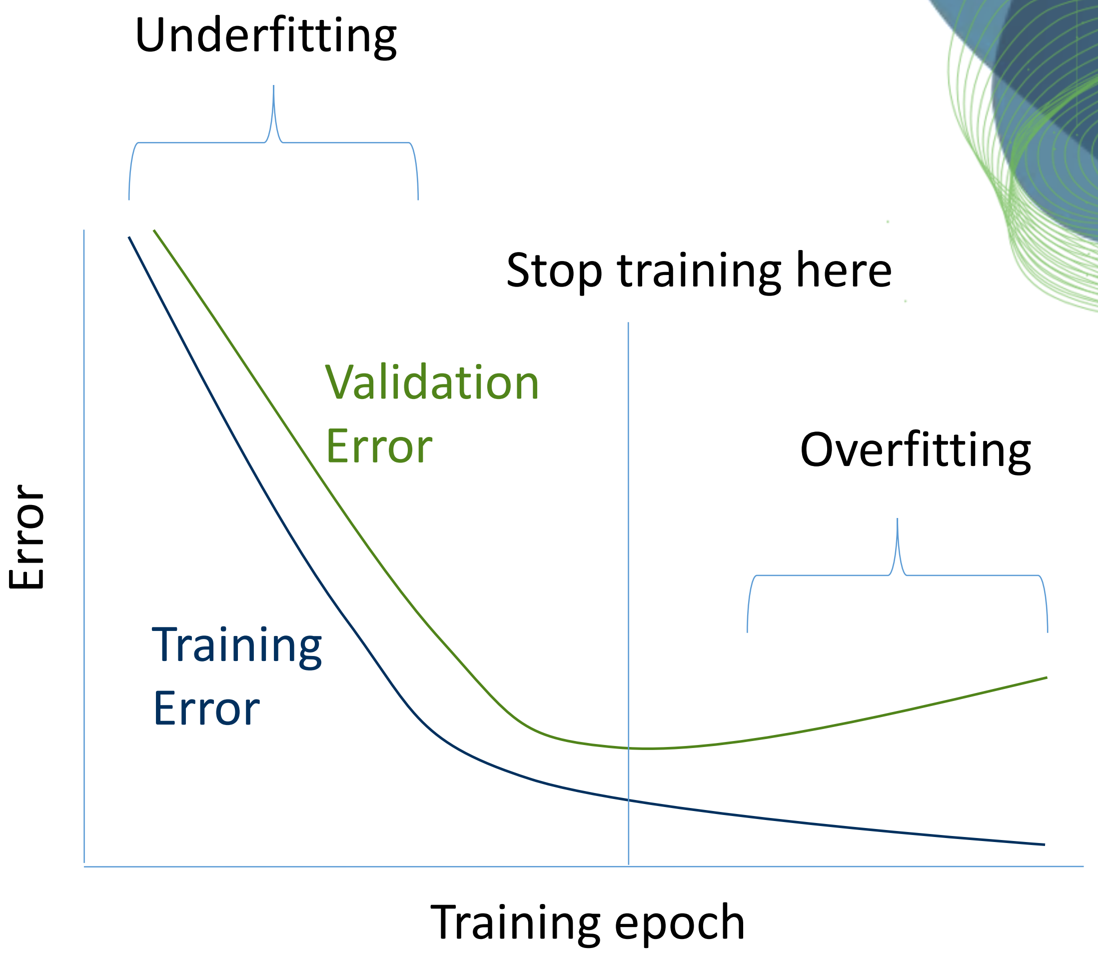
Model validation

Split data in

- Training dataset (80% of the data):
  used for training the model
- Validation dataset (10% of the data):
  after each iteration, see if the model overfits
- Test dataset (10% of the data):
  final evaluation after training is finished

Training

- Find spot with lowest validation error
- Avoid Underfitting: A model that is not trained long enough to capture the structure of the data
- Avoid Overfitting: A model that has been trained too long, has memorized the training data, but is not able to generalize on new data



https://towardsdatascience.com/how-to-split-data-into-three-sets-train-validation-and-test-and-why-e50d22d3e54c

Outlook: Machine learning for image analysis

- In classical machine learning, we typically select features for training our classifier

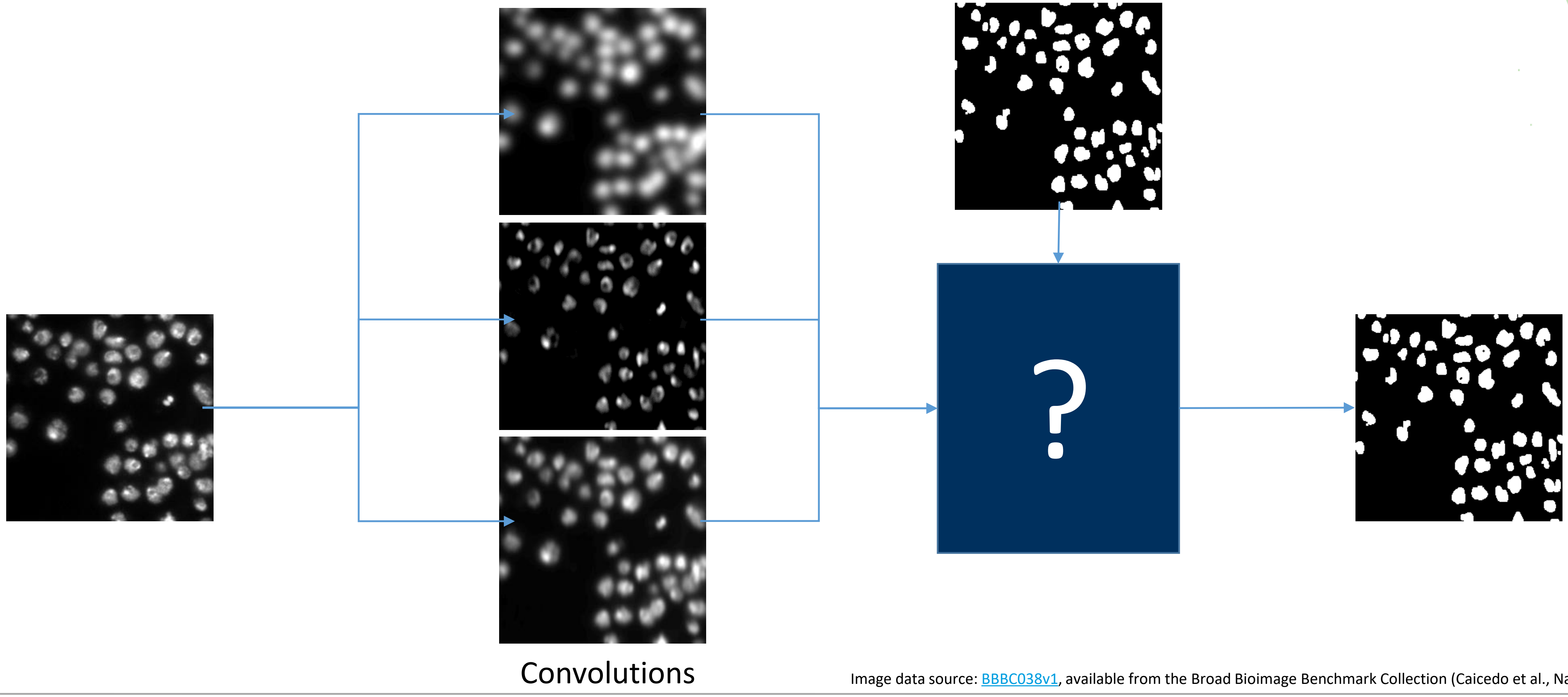
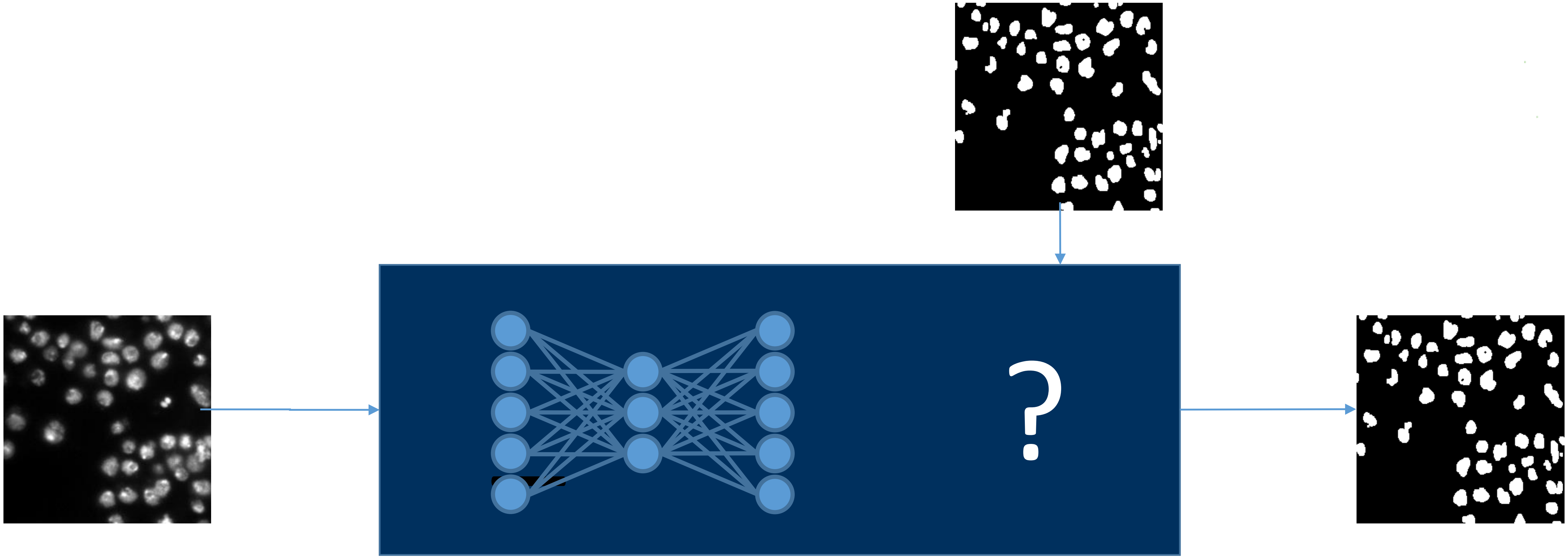


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

Outlook: Deep learning for image analysis

- In deep learning, this is done automatically by the neural network



Convolutional neural networks

Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

Pixel classification using scikit-learn

With material from

Robert Haase

Tabular object classification

- Classify objects starting from feature vectors (table columns)

Raw data

	area	elongation
0	3.950088	2.848643
1	4.955912	3.390093
2	7.469852	5.575289
3	2.544467	3.017479
4	3.465662	1.463756
5	3.156507	3.232181
6	9.978705	6.676372
7	6.001683	5.047063
8	2.457139	3.416050
9	3.672295	3.407462
10	9.413702	7.598608

"Ground truth" annotation

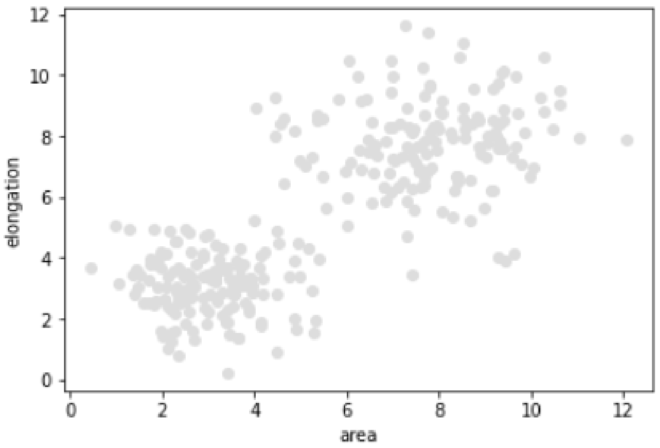
```
annotation = [1, 1, 2, 1, 1, 1, 2, 2,
```

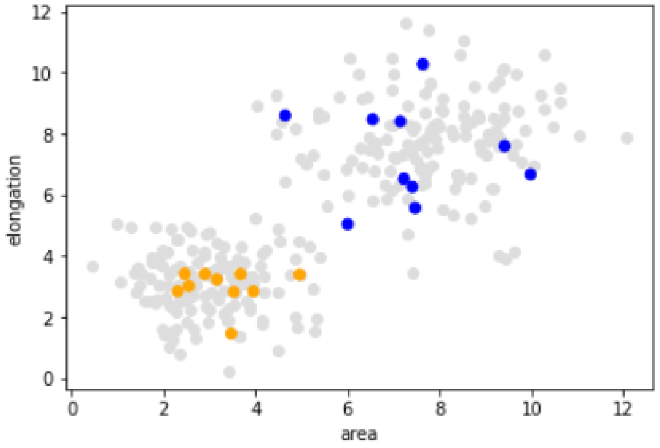
Classifier training

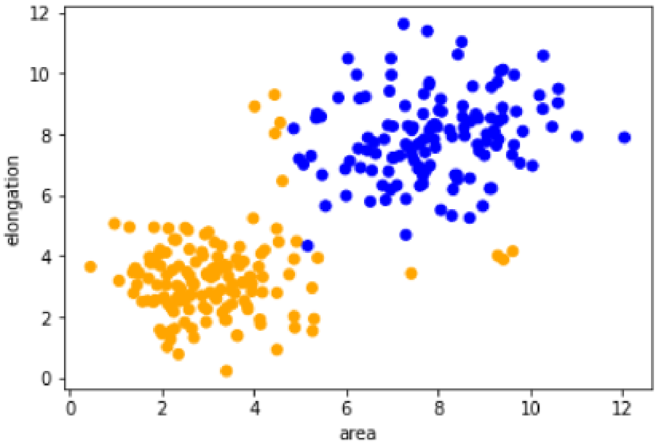
```
classifier = RandomForestClassifier()
classifier.fit(train_data, train_annotation)
```

Classifier prediction

```
result = classifier.predict(validation_data)
```







https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/09_machine_learning/01_supervised_machine_learning.ipynb

Interactive pixel classification

- Prepare an empty layer for annotations and keep a reference

```
labels = viewer.add_labels(
    np.zeros(image.shape).astype(int))
```

- Read annotations

```
manual_annotations = labels.data

from skimage.io import imshow
imshow(manual_annotations,
             vmin=0, vmax=2)
```

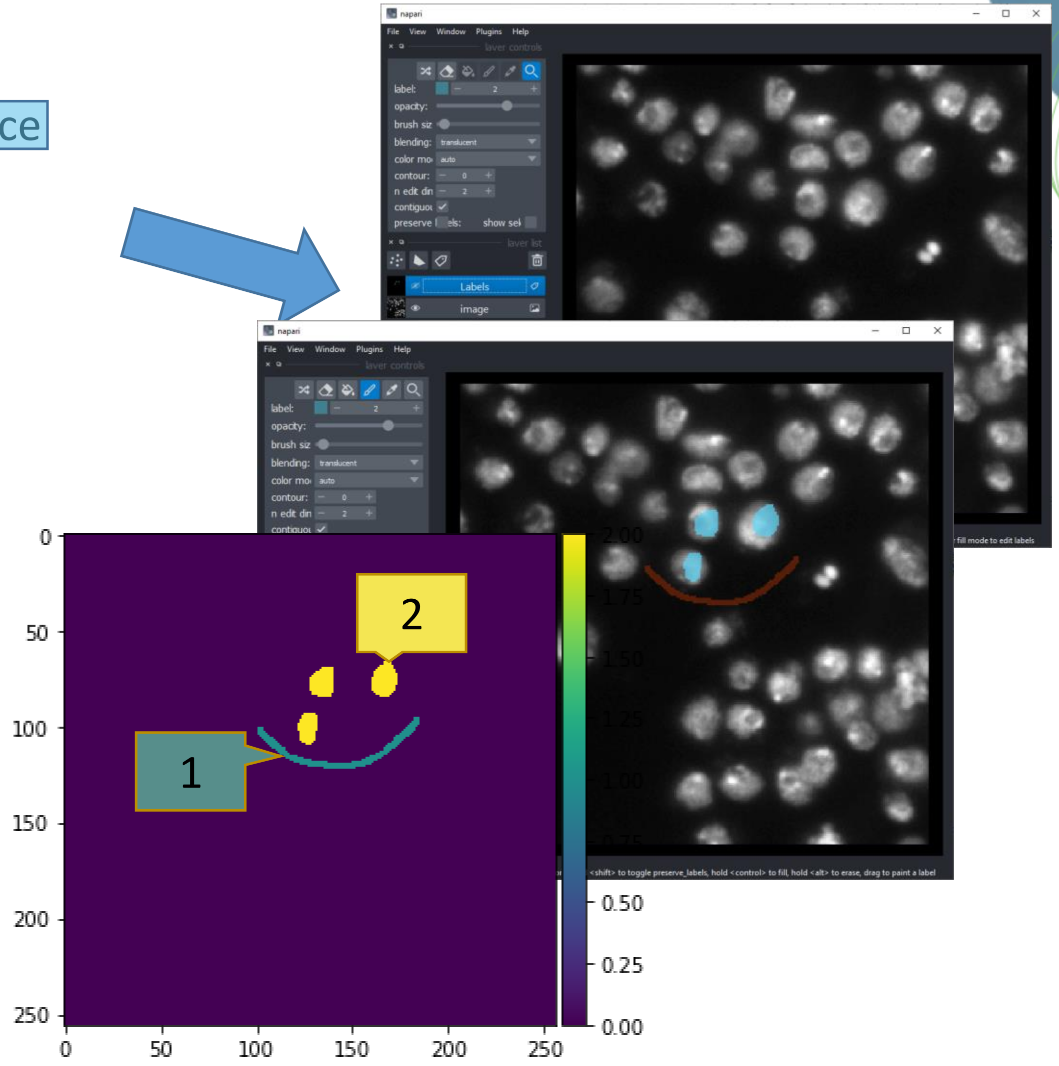


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].
https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/09_machine_learning/02_scikit_learn_random_forest_pixel_classifier.ipynb

Interactive pixel classification

- Pixel classification using scikit-learn
  - Expects one-dimensional arrays for
    - every feature individually
    - ground truth

```
# train classifier
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(max_depth=2, random_state=0)
classifier.fit(X, y)
```

Image data

Ground truth / annotation

```
y_ = classifier.predict(X)
```

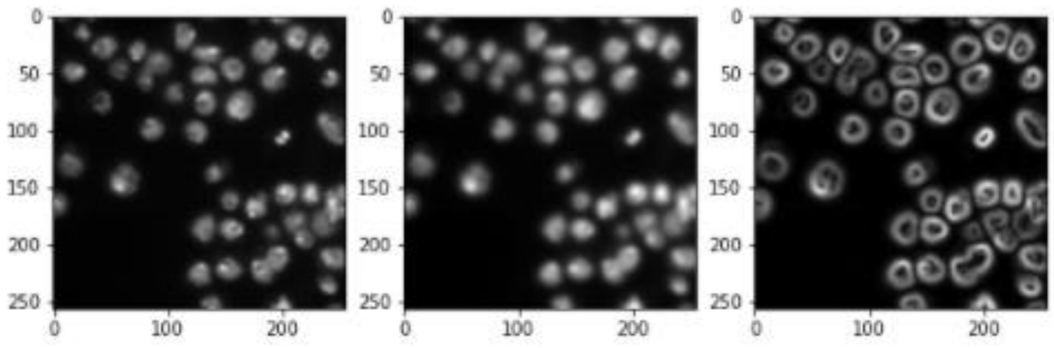
Image data

prediction

https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/09_machine_learning/02_scikit_learn_random_forest_pixel_classifier.ipynb

Interactive pixel classification

- Pixel classification using scikit-learn
  - Expects one-dimensional arrays for
    - every feature individually
    - ground truth




```
# for training, we need to generate features
feature_stack = generate_feature_stack(image)
X, y = format_data(feature_stack, manual_annotations)

# train classifier
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(max_depth=2, random_state=0)
classifier.fit(X, y)
```

Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/09_machine_learning/02_scikit_learn_random_forest_pixel_classifier.ipynb

Interactive pixel classification

- Pixel classification using scikit-learn

```
# process the whole image and show result

result_1d =
classifier.predict(feature_stack.T)

result_2d = result_1d.reshape(image.shape)


viewer.add_labels(result_2d)
```

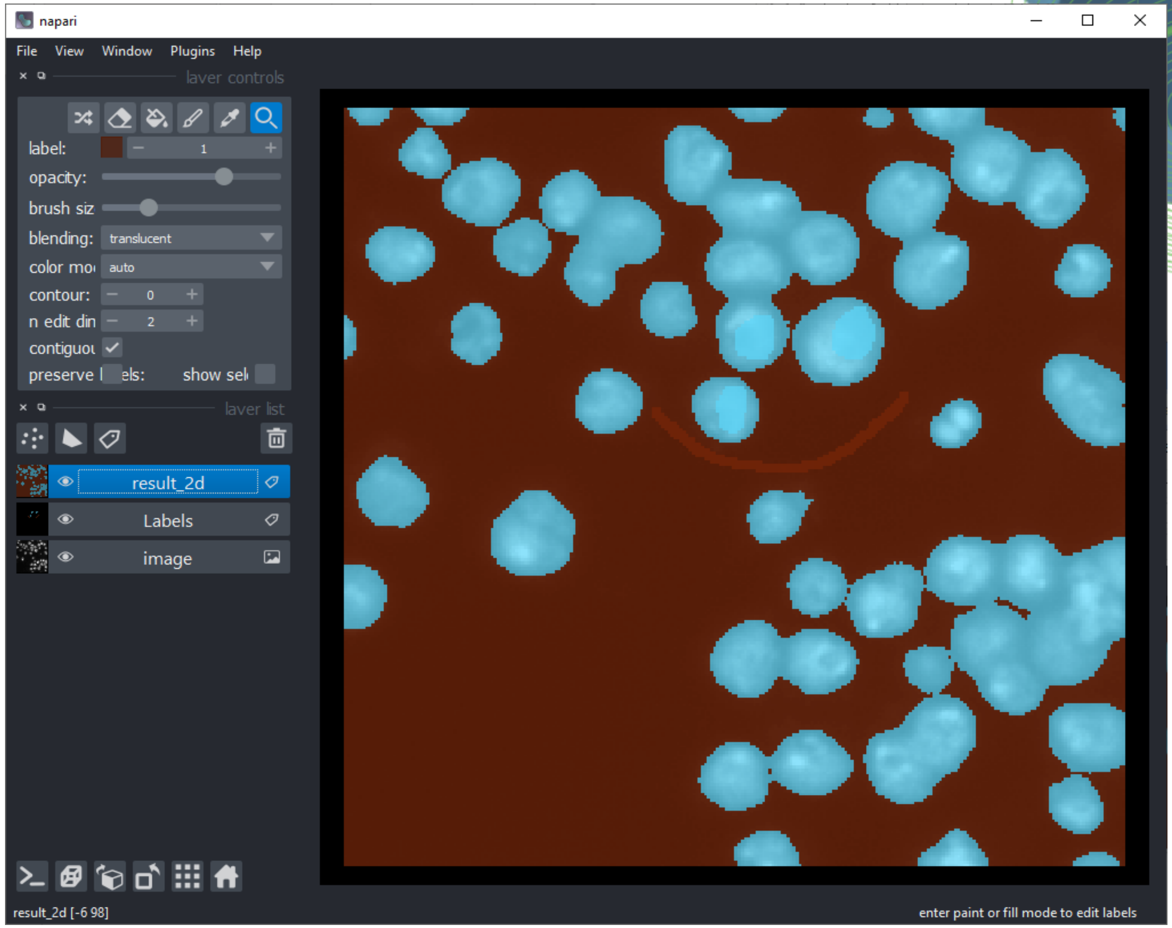


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].
https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/09_machine_learning/02_scikit_learn_random_forest_pixel_classifier.ipynb

Interactive pixel classification

- Jupyter notebooks and napari side-by-side

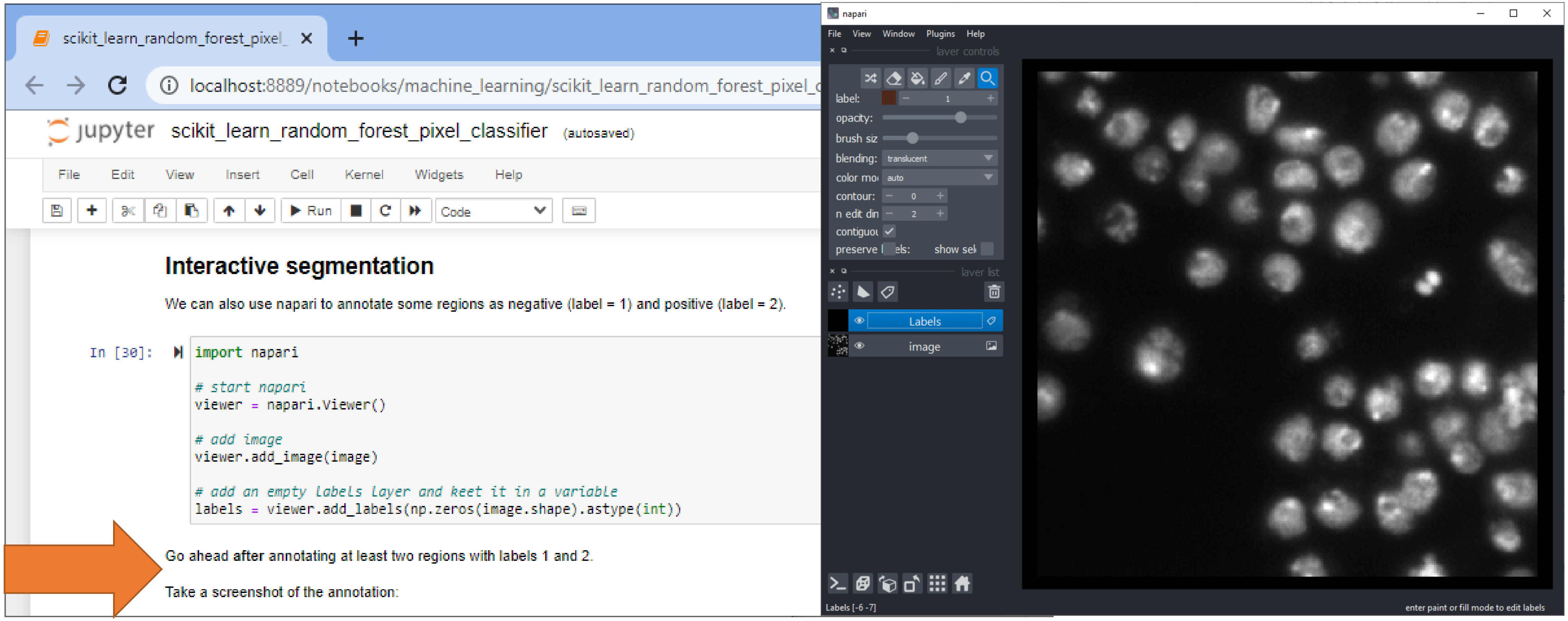


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/09_machine_learning/02_scikit_learn_random_forest_pixel_classifier.ipynb

Interactive pixel classification

- Jupyter notebooks and napari side-by-side

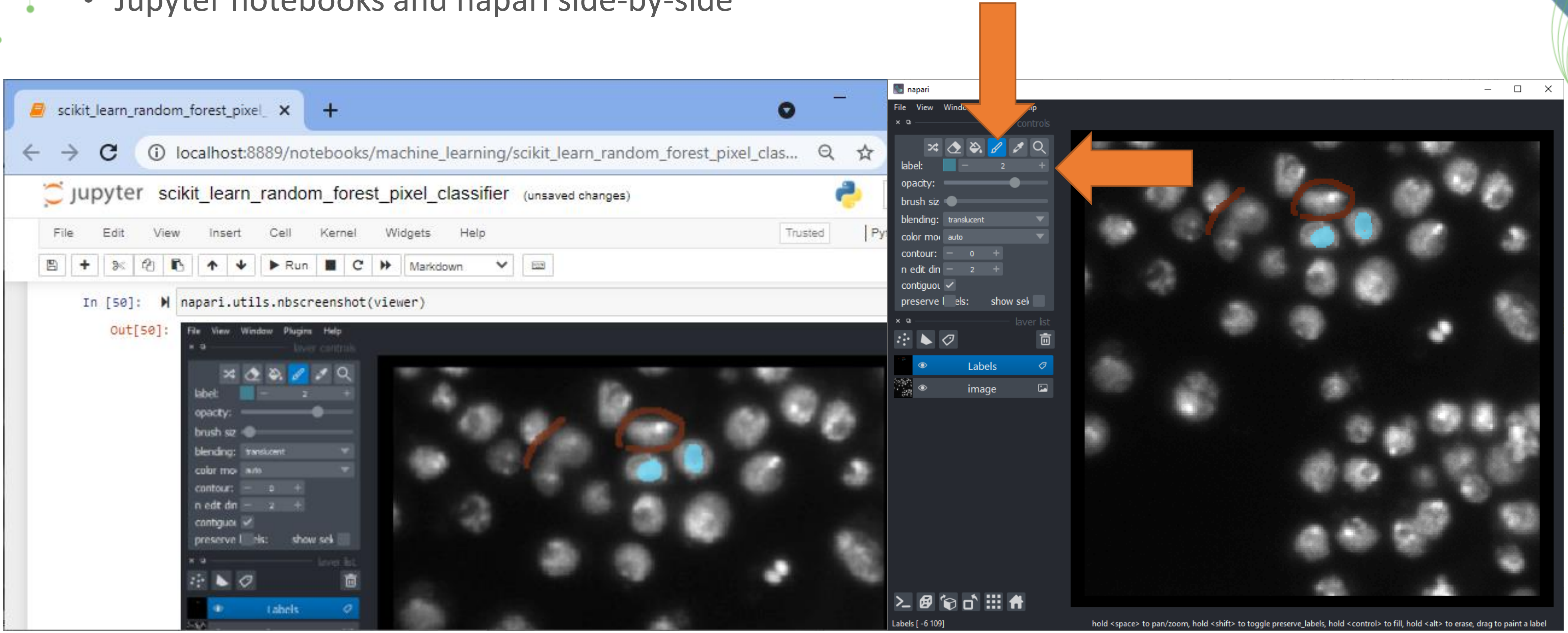


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].
https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/09_machine_learning/02_scikit_learn_random_forest_pixel_classifier.ipynb

Interactive pixel classification

- Jupyter notebooks and napari side-by-side

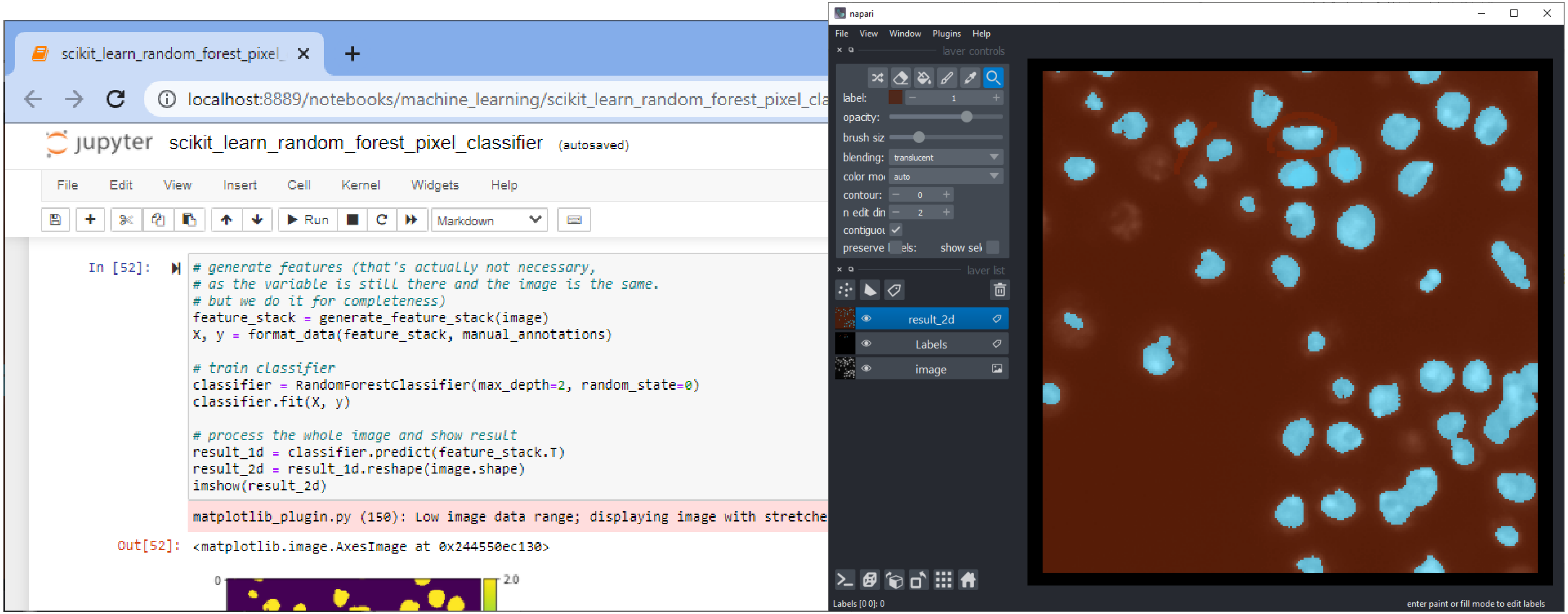


Image data source: BBBC038v1, available from the Broad Bioimage Benchmark Collection (Caicedo et al., Nature Methods, 2019].

https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/09_machine_learning/02_scikit_learn_random_forest_pixel_classifier.ipynb

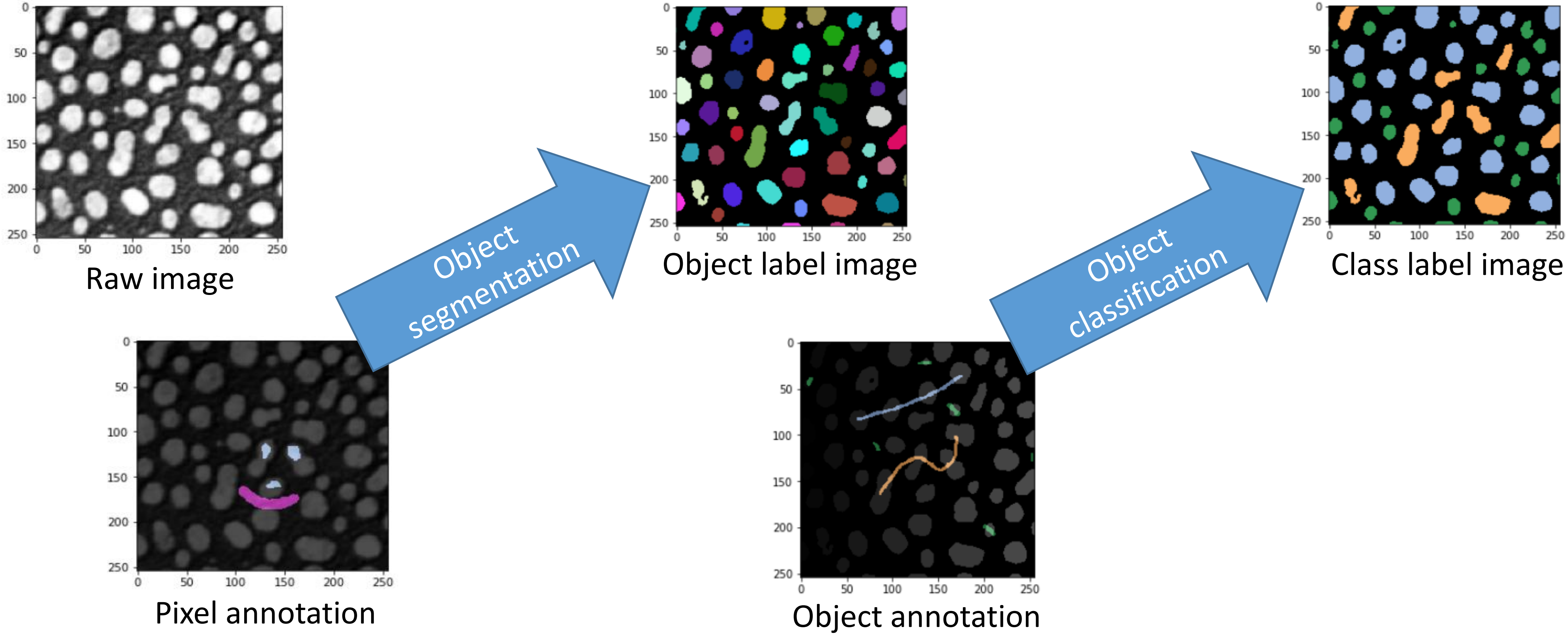
Accelerated pixel and object classification (APOC)

With material from

Robert Haase

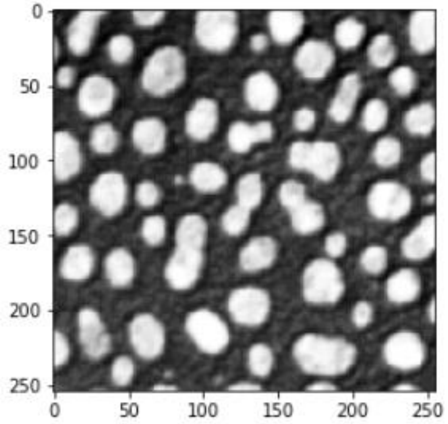
Accelerated pixel and object classification

- APOC is a python library that makes use of OpenCL-compatible Graphics Cards to accelerate pixel and object classification

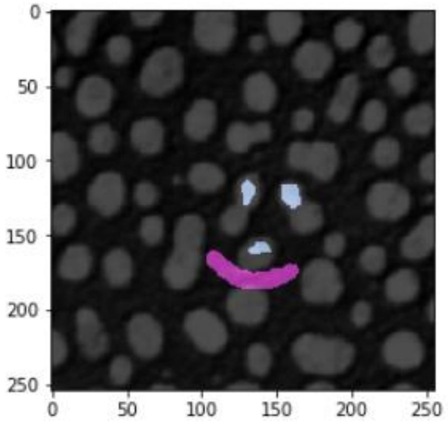


Object segmentation

- Pixel classification + connected component labeling



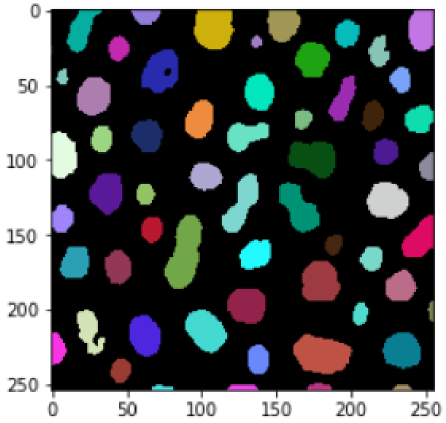
Raw image



Pixel annotation

```
# define features
features = "gaussian_blur=1 gaussian_blur=5 sobel_of_gaussian_blur=1"

# this is where the model will be saved
cl_filename = 'my_object_segmenter.cl'

# delete classifier in case the file exists already
apoc.erase_classifier(cl_filename)

# train classifier
clf = apoc.ObjectSegmenter(opencl_filename=cl_filename, positive_class_identifier=2)
clf.train(features, manual_annotations, image)

segmentation_result = clf.predict(features=features, image=image)
cle.imshow(segmentation_result, labels=True)
```

Object segmentation

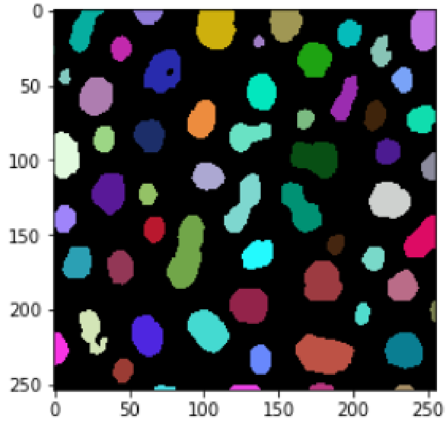


Object label image

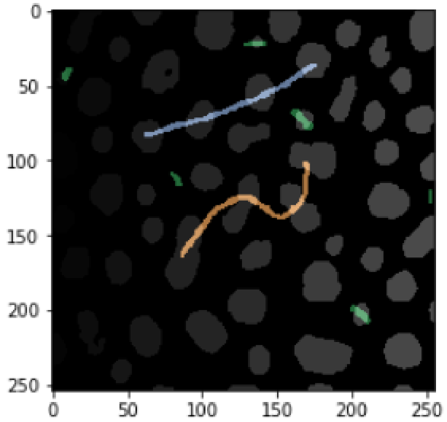
https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/09_machine_learning/03_apoc_object_segmenter.ipynb

Object classification

- Feature extraction + tabular classification



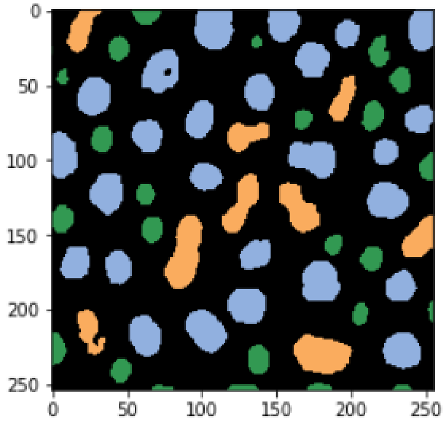
Object label image



Object annotation

```python
# for the classification we define size and shape as criteria
features = 'area mean_max_distance_to_centroid_ratio'

# This is where the model will be saved
cl_filename_object_classifier = "my_object_classifier.cl"

# delete classifier in case the file exists already
apoc.erase_classifier(cl_filename_object_classifier)

# train the classifier
classifier = apoc.ObjectClassifier(cl_filename_object_classifier)
classifier.train(features, segmentation_result, annotation, image)

# determine object classification
classification_result = classifier.predict(segmentation_result, image)
cle.imshow(classification_result, labels=True)
```

Object classification

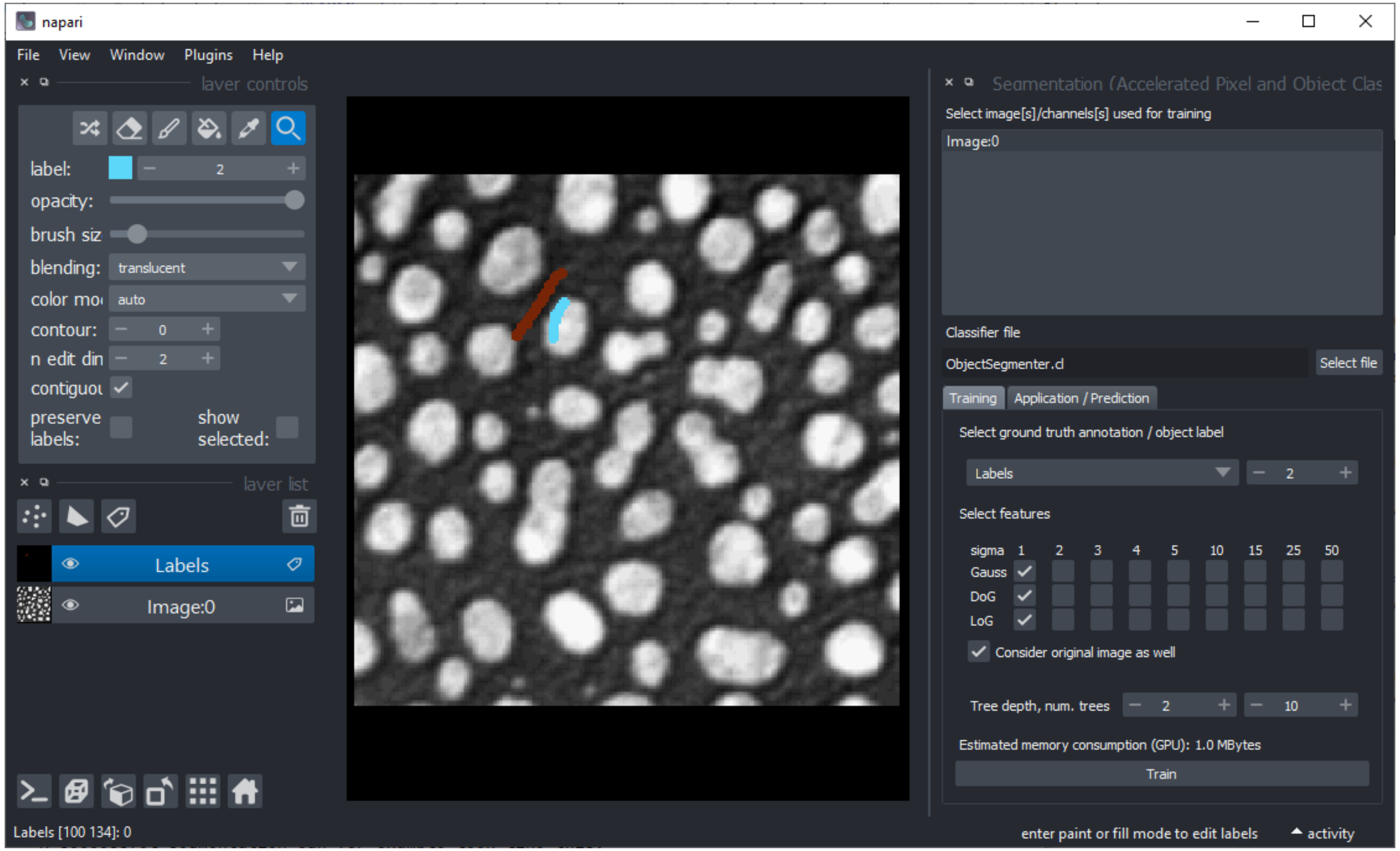


Class label image

https://github.com/BiAPoL/Bio-image_Analysis_with_Python/blob/main/09_machine_learning/03_apoc_object_segmenter.ipynb

Graphical user interface

- Object segmentation
- https://github.com/haesleinhuepf/napari-accelerated-pixel-and-object-classification#object-and-semantic-segmentation



Supervised machine learning for tissue classification

Random Forest Classifiers based on

- scikit-learn and
- clesperanto

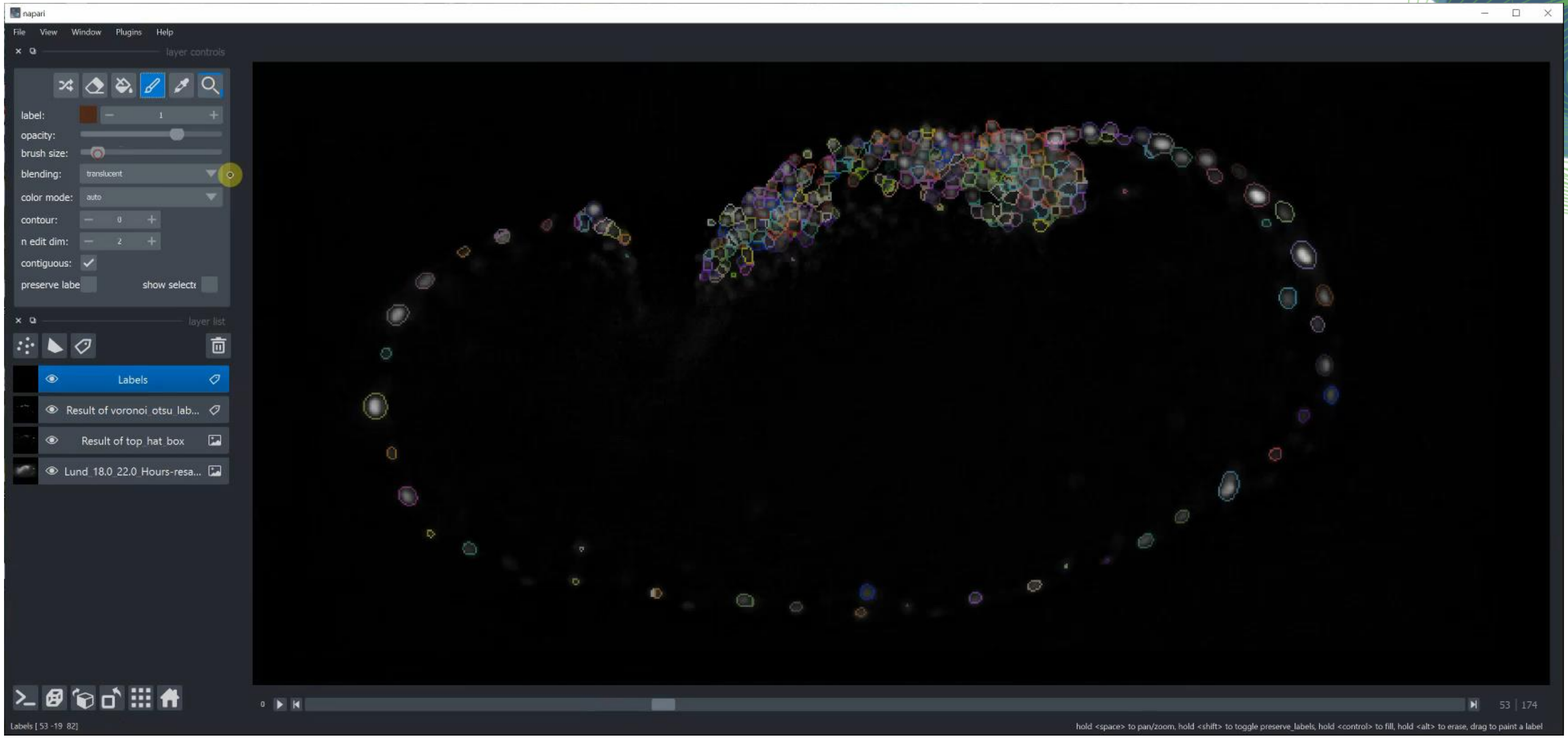
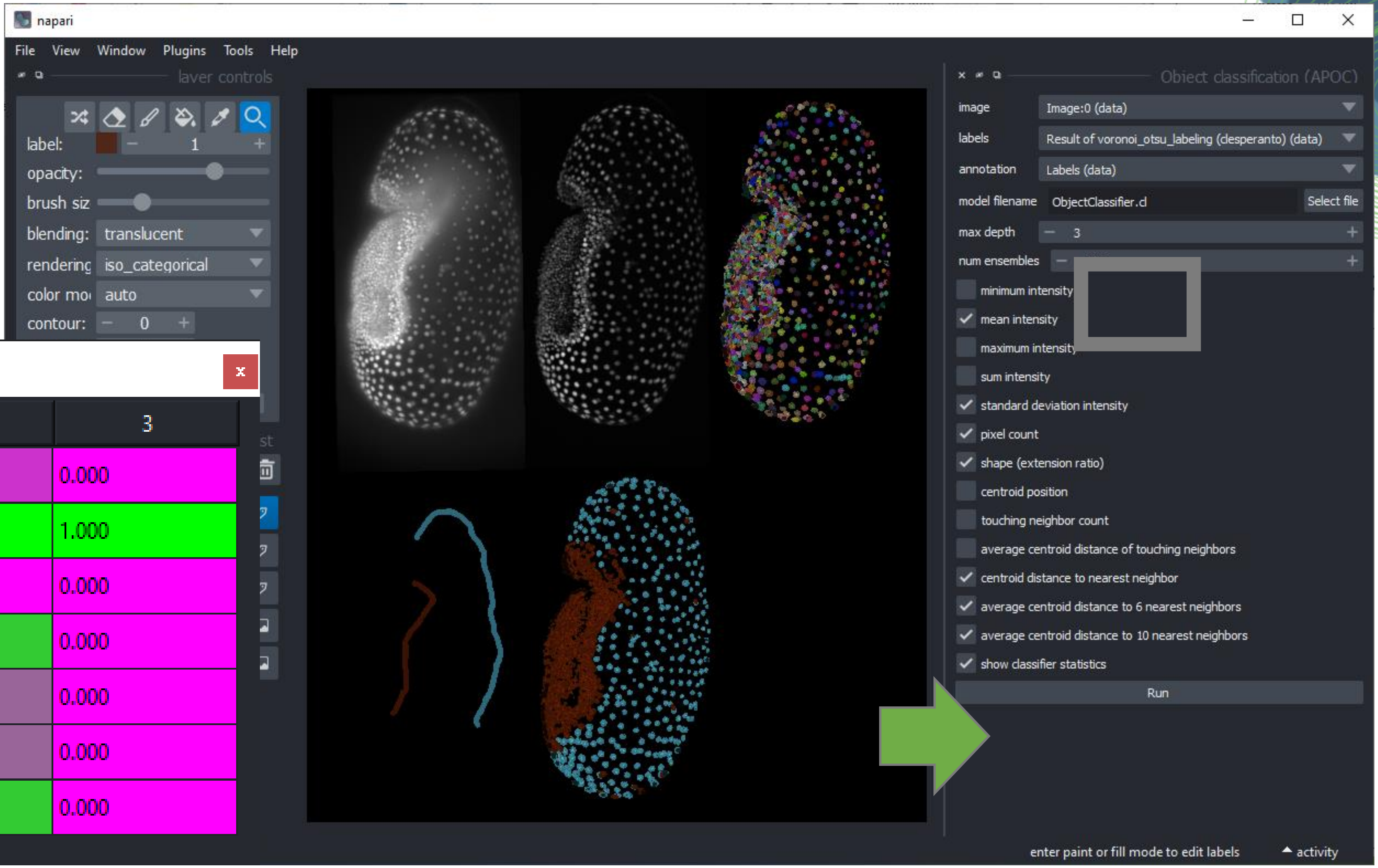


Image data source: Daniela Vorkel, Myers lab, MPI-CBG/CSBD
https://github.com/haesleinhuepf/napari-accelerated-pixel-and-object-classification

Data exploration / supervised machine learning

- Inspect how the random forest classifier makes decisions
- Note: Beware of correlated parameters!



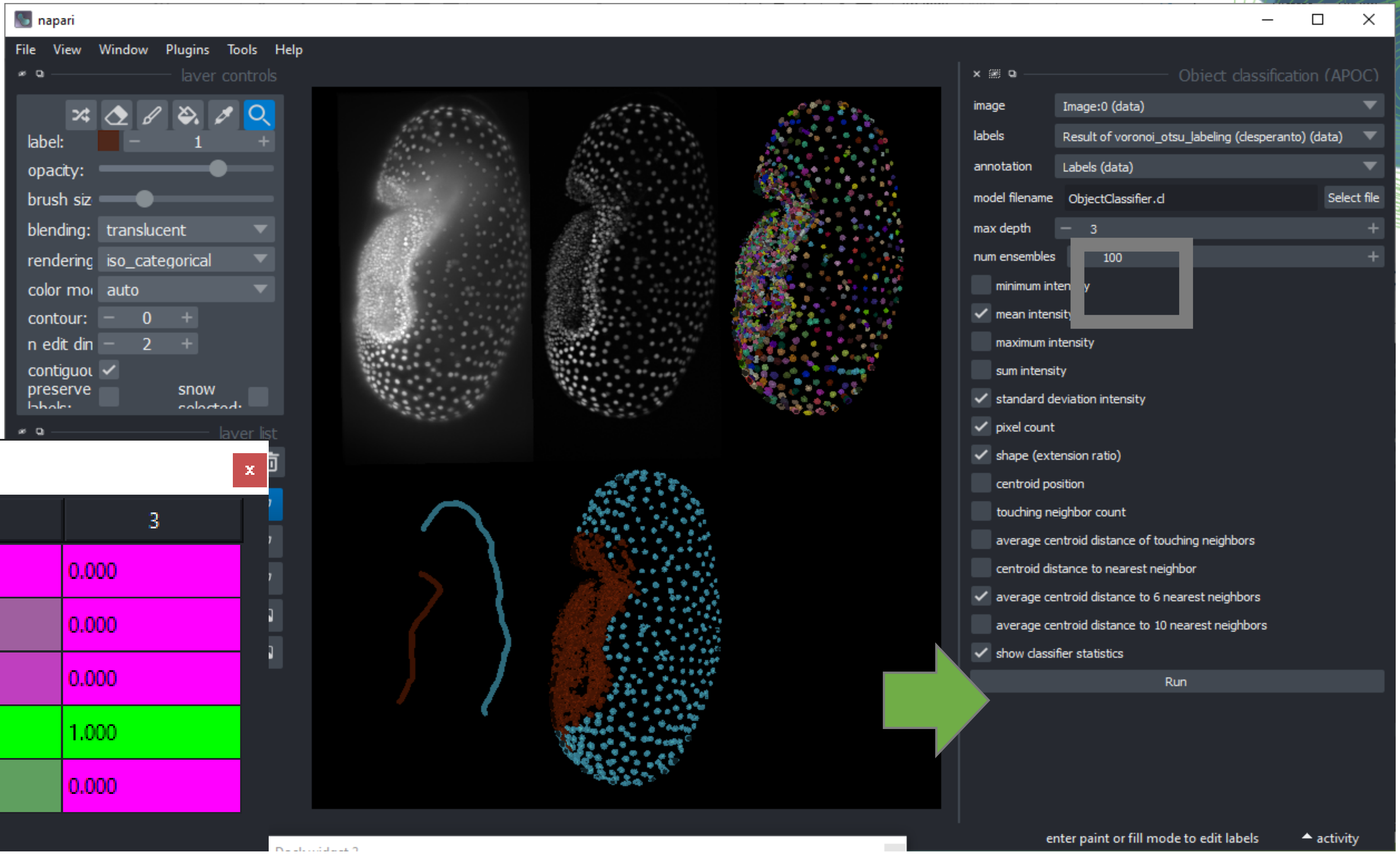
Dock widget 1

	1	2	3
area	0.010	0.056	0.000
mean_intensity	0.200	0.278	1.000
standard_deviation_intensity	0.030	0.000	0.000
mean_max_distance_to_centroid_ratio	0.270	0.222	0.000
average_distance_of_n_nearest_neighbors=1	0.120	0.111	0.000
average_distance_of_n_nearest_neighbors=6	0.170	0.111	0.000
average_distance_of_n_nearest_neighbors=10	0.200	0.222	0.000

Image data source: Daniela Vorkel, Myers lab, MPI-CBG/CSBD
https://github.com/haesleinhuepf/napari-accelerated-pixel-and-object-classification

Data exploration / supervised machine learning

- Inspect how the random forest classifier makes decisions
- Note: Beware of correlated parameters!



Dock widget 2

	1	2	3
area	0.060	0.000	0.000
mean_intensity	0.330	0.167	0.000
standard_deviation_intensity	0.040	0.111	0.000
mean_max_distance_to_centroid_ratio	0.260	0.444	1.000
average_distance_of_n_nearest_neighbors=6	0.310	0.278	0.000

Image data source: Daniela Vorkel, Myers lab, MPI-CBG/CSBD

https://github.com/haesleinhuepf/napari-accelerated-pixel-and-object-classification

Data exploration / supervised machine learning

- Inspect how the random forest classifier makes decisions
- Note: Beware of correlated parameters!

Dock widget 2

	1	2	3
area	0.060	0.000	0.000
mean_intensity	0.330	0.167	0.000
standard_deviation_intensity	0.040	0.111	0.000
mean_max_distance_to_centroid_ratio	0.260	0.444	1.000
average_distance_of_n_nearest_neighbors=6	0.310	0.278	0.000

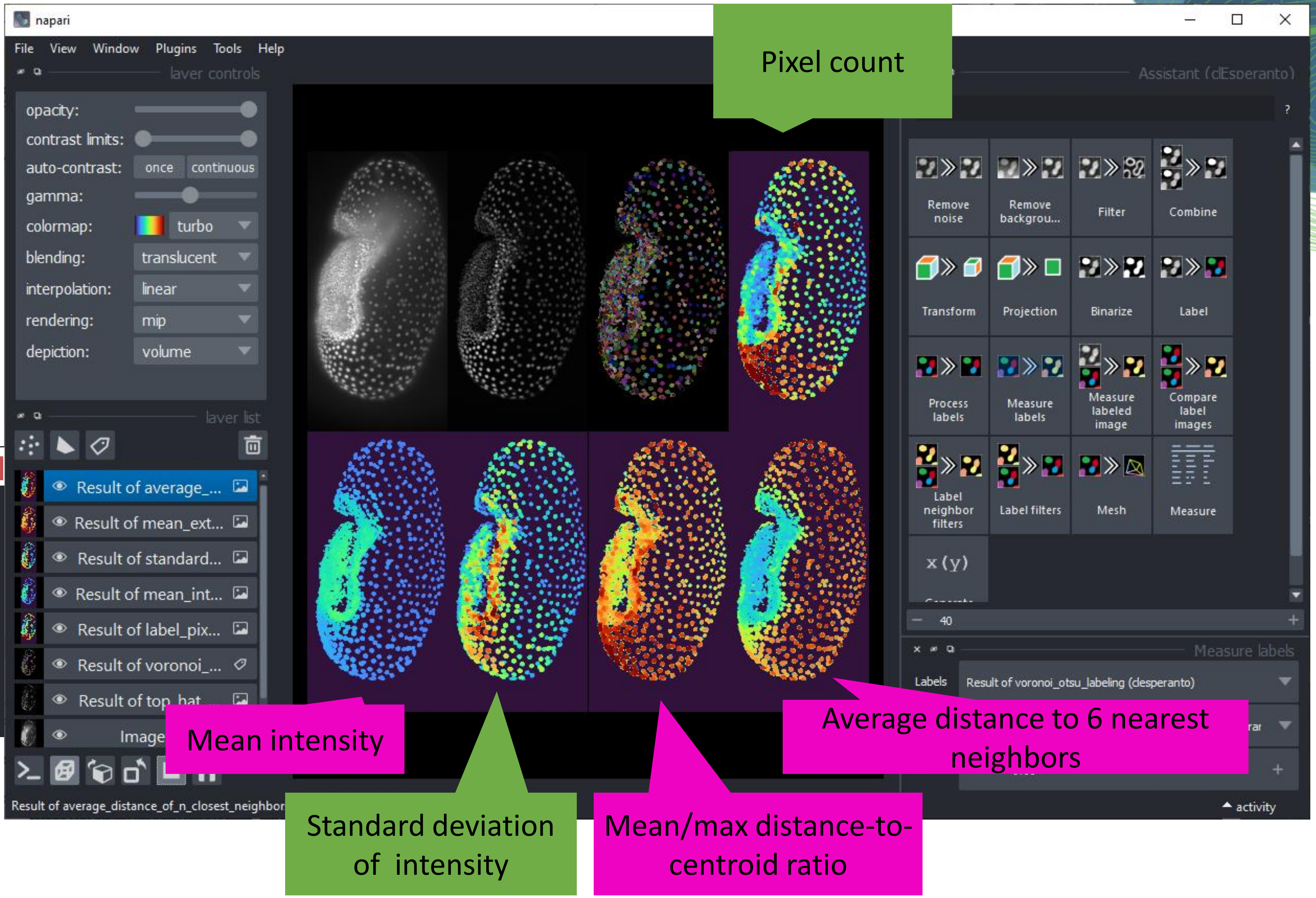
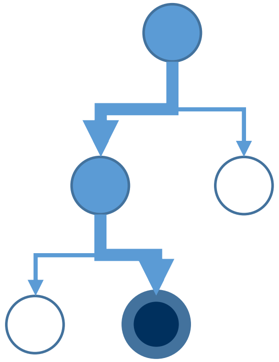
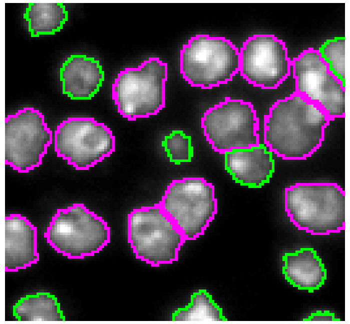


Image data source: Daniela Vorkel, Myers lab, MPI-CBG/CSBD
https://github.com/clEsperanto/napari_pyclesperanto_assistant

Thank you for your attention!





with material from

Robert Haase, ScaDS.AI, Leipzig University

Deborah Schmidt, Jug Lab, MPI CBG

Uwe Schmidt, Myers Lab, MPI CBG

Martin Weigert, EPFL

Ignacio Arganda-Carreras, Universidad del Pais Vasco

Carsen Stringer, HHMI Janelia

Wei Ouyang, KTH Royal Institute of Technology, Stockholm and

The Scikit-Learn community