S6-ESE-AI

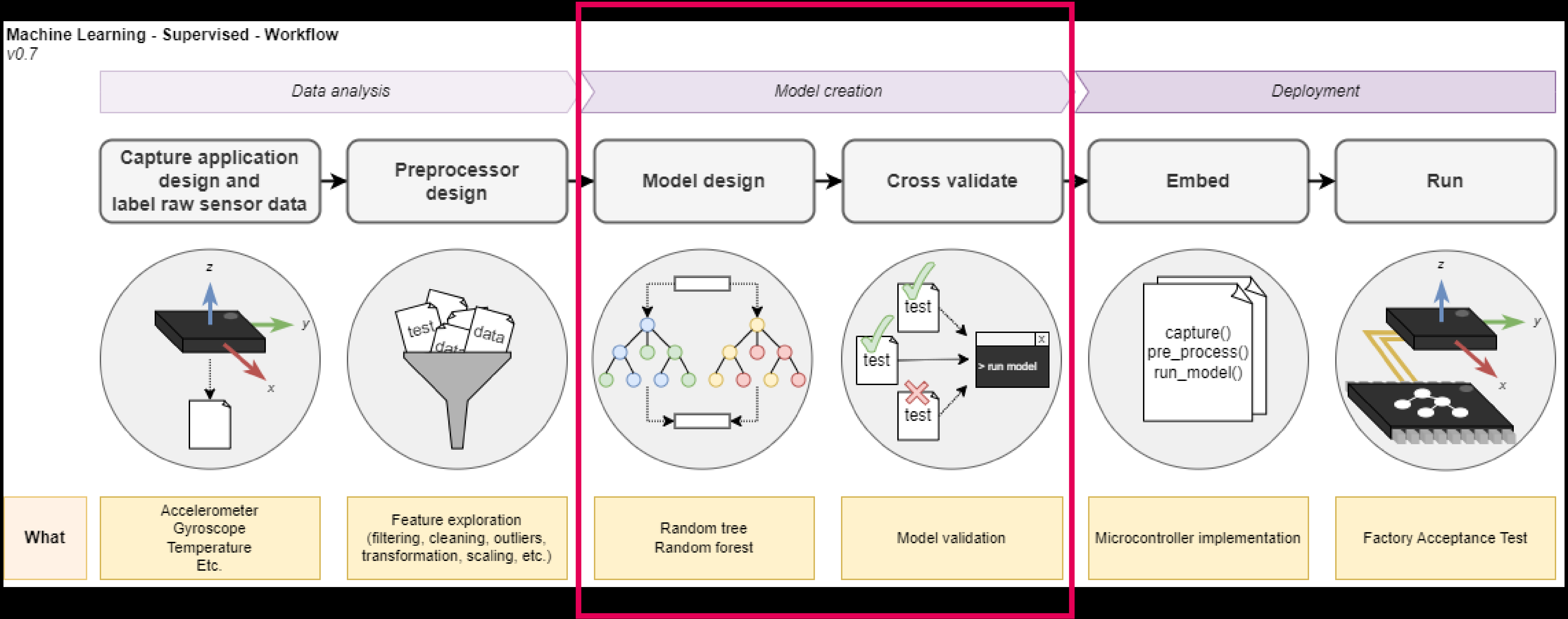
MODELS

JEROEN VEEN

HUGO ARENDS

HAN_UNIVERSITY OF APPLIED SCIENCES

WORKFLOW



Machine Learning - Supervised - Workflow

v0.7

Data analysis → *Model creation* → *Deployment*

Capture application design and label raw sensor data → Preprocessor design → Model design → Cross validate → Embed → Run

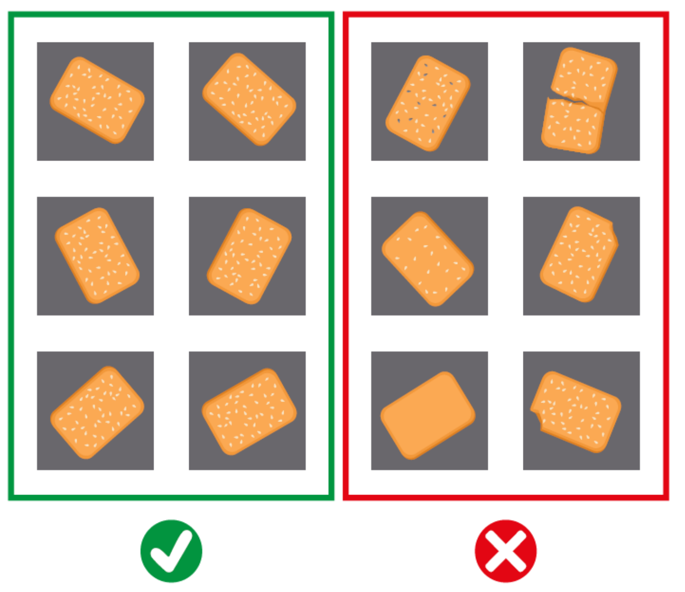
What	Accelerometer Gyroscope Temperature Etc.	Feature exploration (filtering, cleaning, outliers, transformation, scaling, etc.)	Random tree Random forest	Model validation	Microcontroller implementation	Factory Acceptance Test

AGENDA

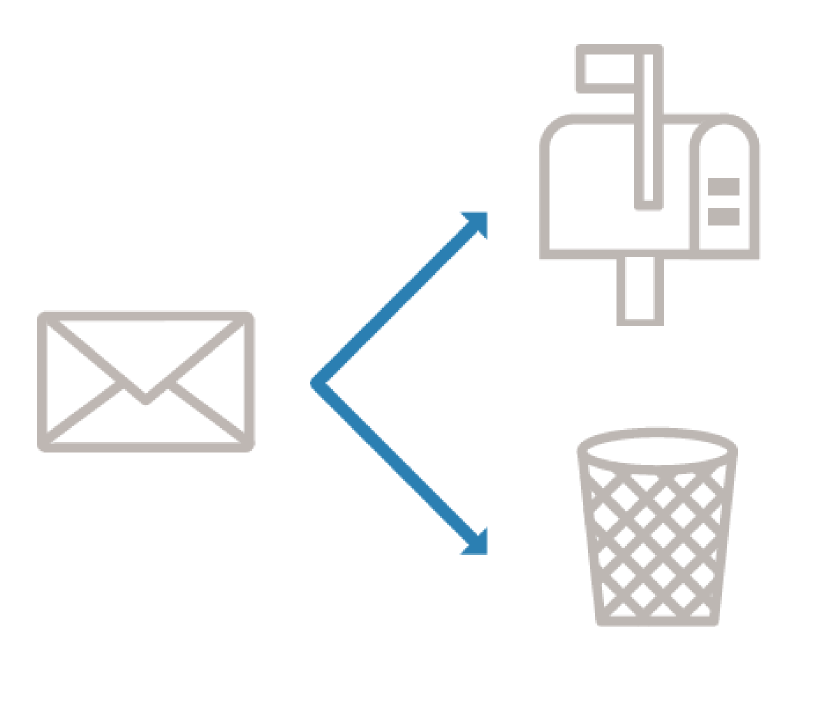
- Classification
- K nearest neighbors (KNN)
- Support vector machine (SVM)
- Decision tree
- Ensemble learning
- Model design

BINARY CLASSIFICATION

- Sample falls in either of 2 classes



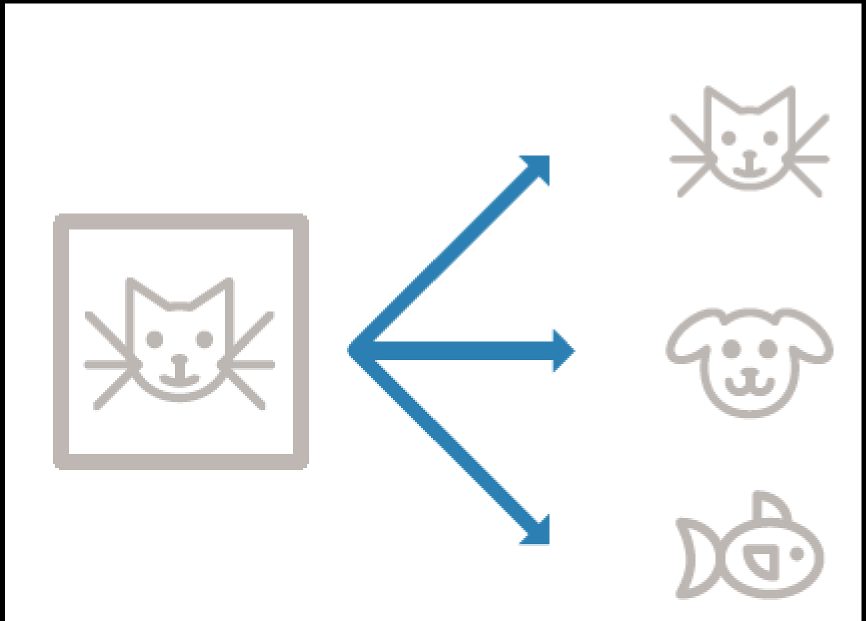
Source: Basler, Artificial Intelligence in Image Processing



Source: Mathworks, Applying Supervised Learning

MULTI-CLASS CLASSIFICATION

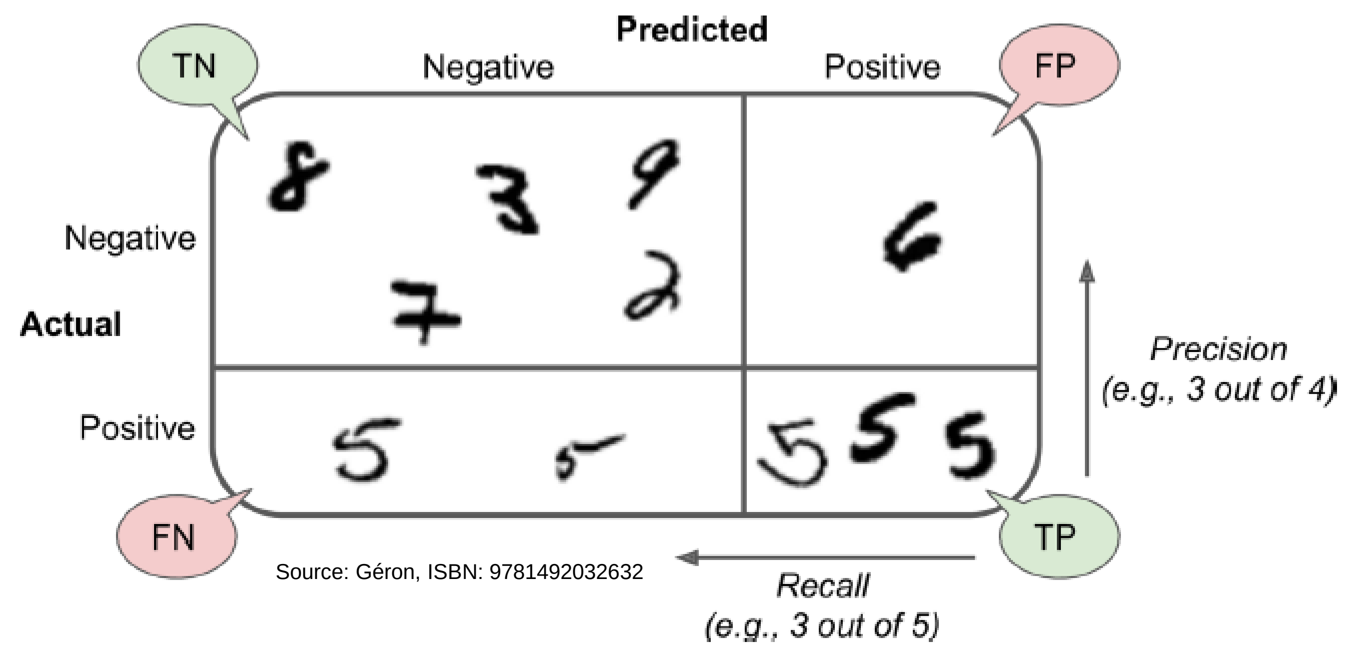
- Sample falls in either of 3 or more classes



Source: Mathworks, Applying Supervised Learning

CLASSIFICATION PERFORMANCE

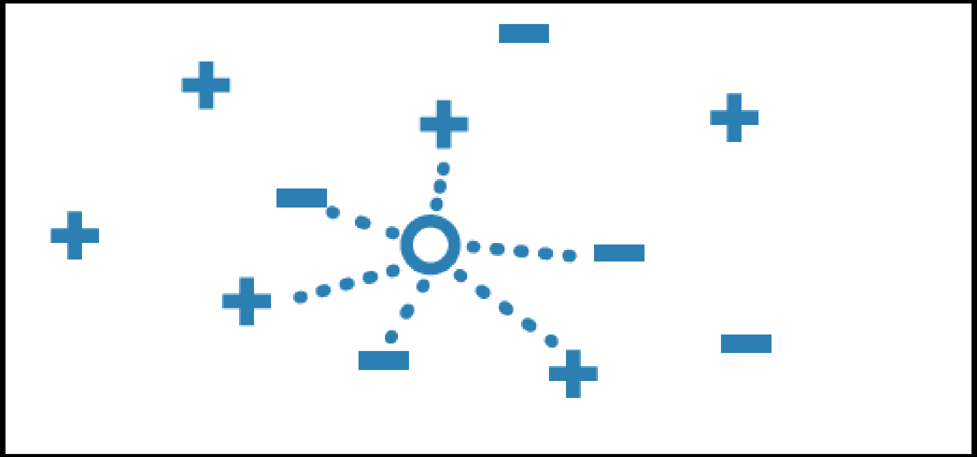
- Confusion matrix



Source: Géron, ISBN: 9781492032632

K NEAREST NEIGHBORS (KNN)

- Instance-based classification
- The simplest classifier
- Categorizes objects based on the classes of their nearest neighbors
- No training required
- Intuitive
- Benchmark



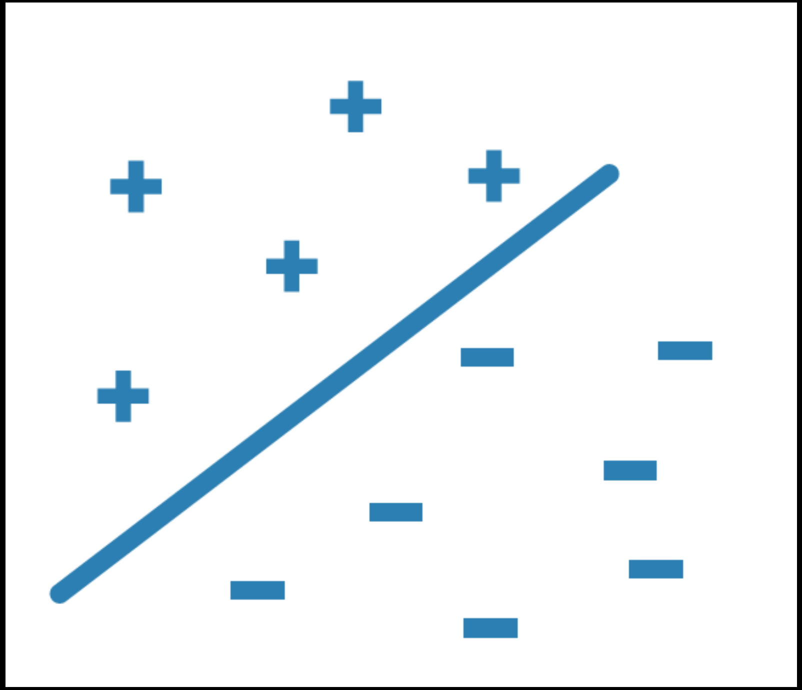
Source: Mathworks, Applying Supervised Learning

"Tell me who your neighbors are, and I'll tell you who you are"

LINEAR SEPARATION

- Model-based classification

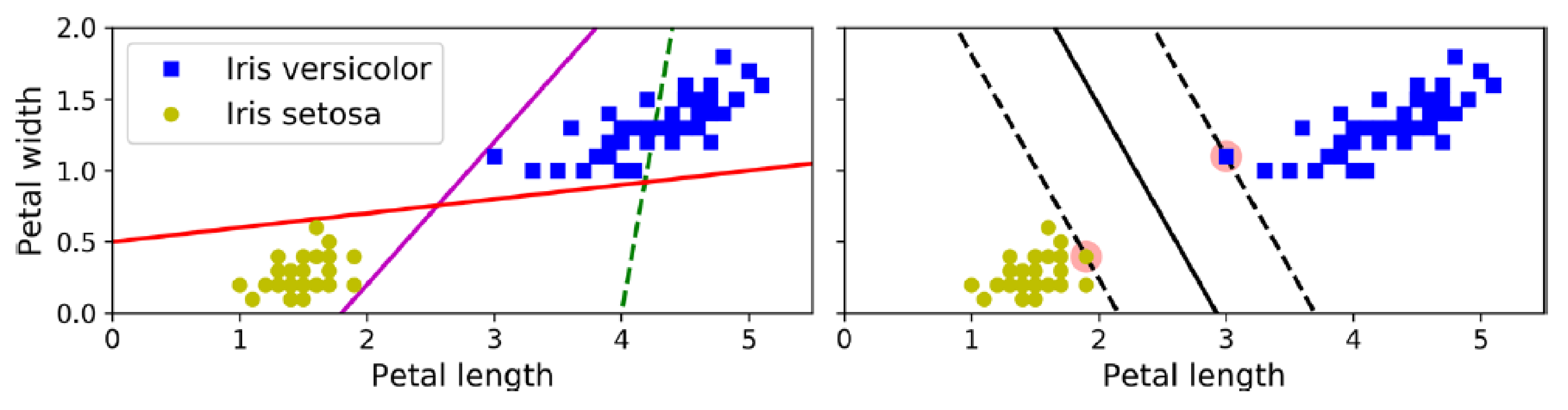
- Finding the linear decision boundary that separates all data points of one class from those of the other class.



Source: Mathworks, Applying Supervised Learning

SUPPORT VECTOR MACHINE (SVM)

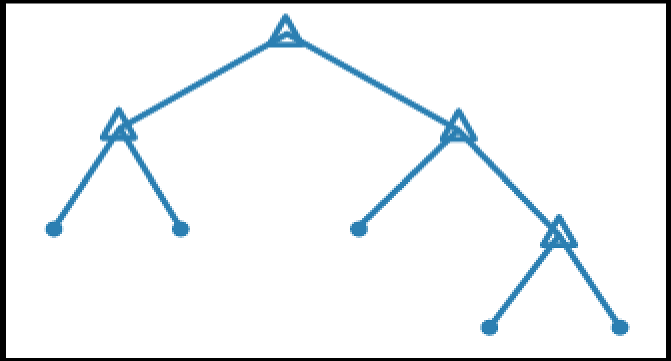
- Binary classifier
- Simple and easy to interpret



Source: Géron, ISBN: 9781492032632

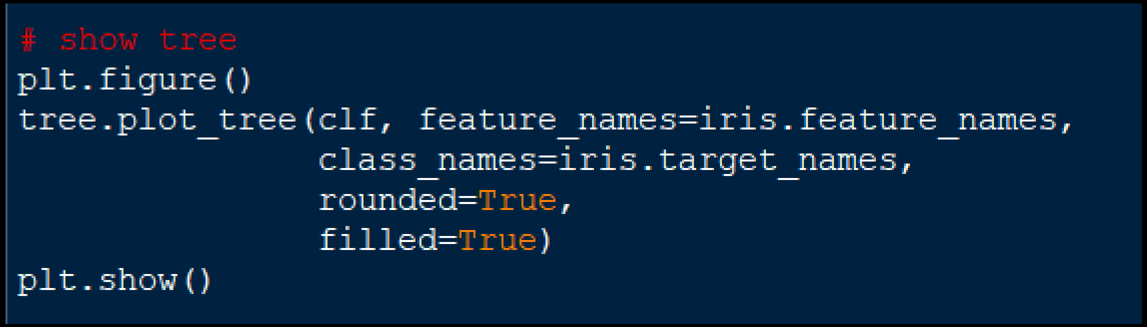
DECISION TREE

- Predict responses to data by following the decisions in the tree from the down to a leaf node.
- Easy to interpret
- White box model
- Fast to fit
- Minimize memory usage

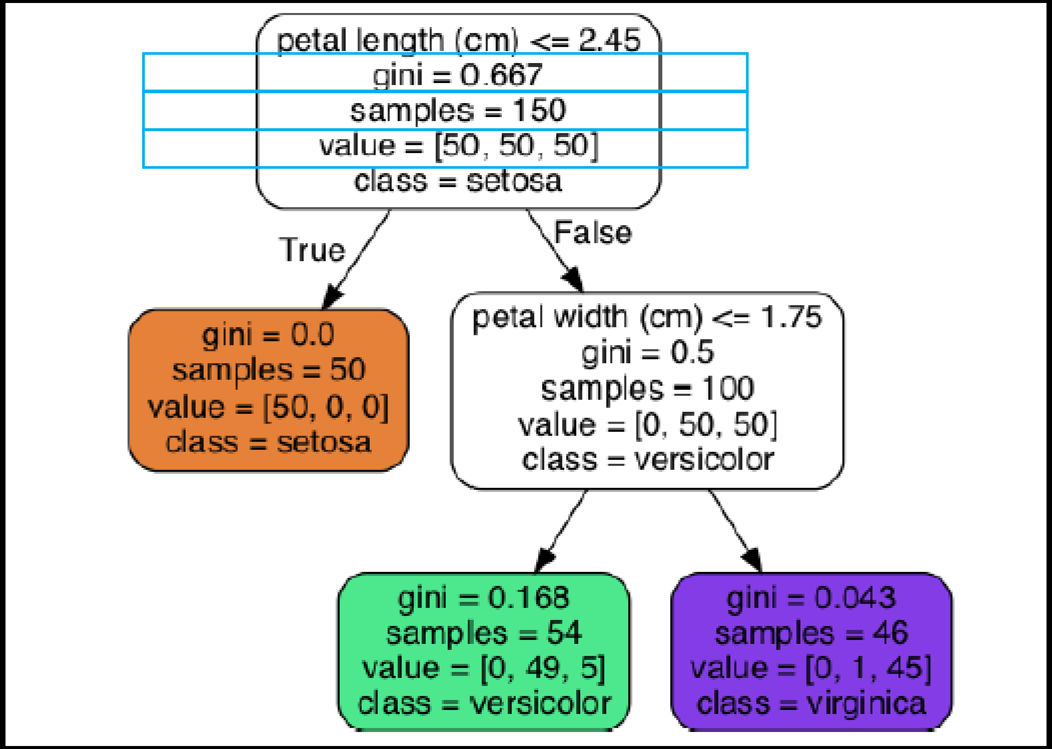


Source: Mathworks, Applying Supervised Learning

MAKING PREDICTIONS



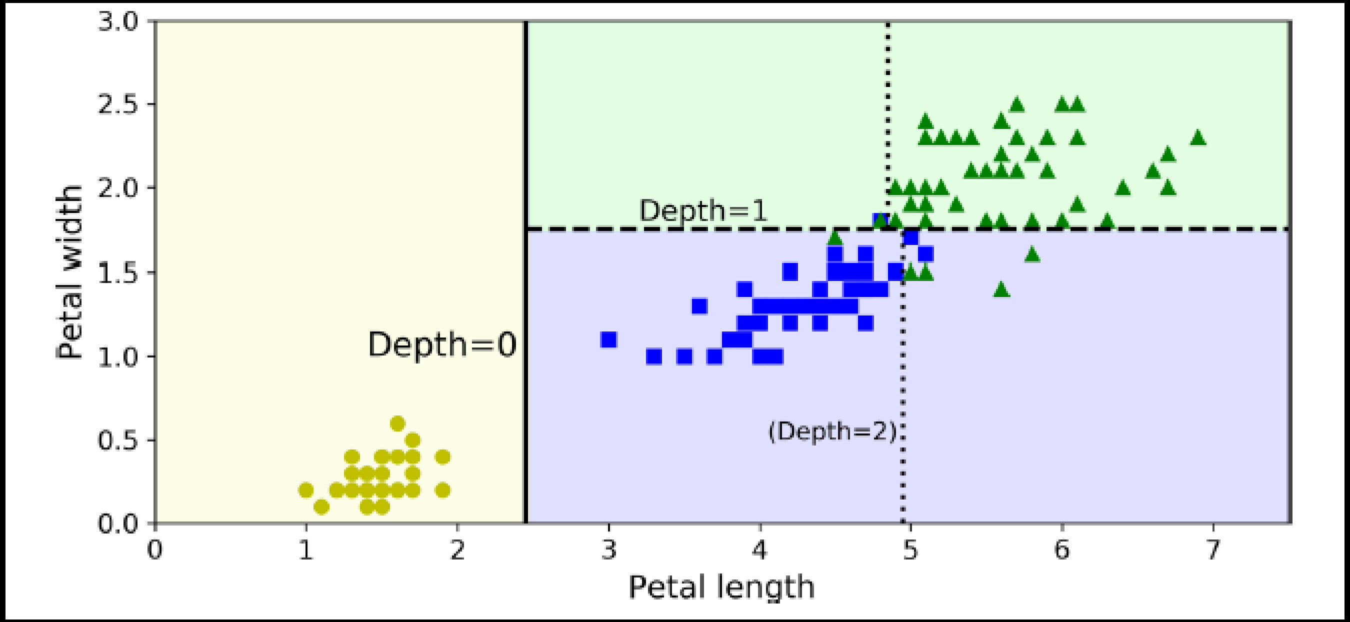
```
# show tree
plt.figure()
tree.plot_tree(clf, feature_names=iris.feature_names,
               class_names=iris.target_names,
               rounded=True,
               filled=True)
plt.show()
```



Source: Géron, ISBN: 9781492032632

DECISION TREE BOUNDARIES

- white box models



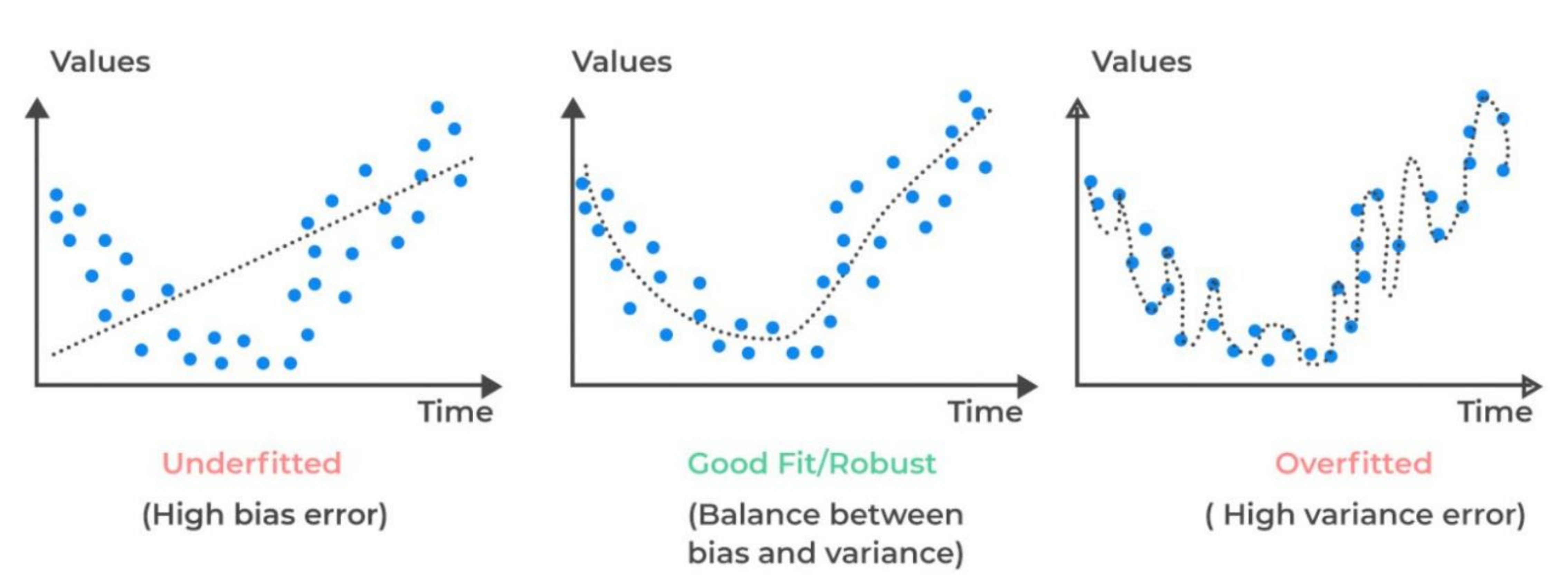
Source: Géron, ISBN: 9781492032632

HAN_UNIVERSITY
OF APPLIED SCIENCES

DISADVANTAGES OF DECISION TREES

- Trees become biased if some classes dominate
  -> balance dataset before fitting
- Trees are prone to overfitting
  -> use feature selection, pruning, set max depth, set min samples per leaf
- Trees can be unstable
  -> use ensemble learning, or random forest
- Trees cannot learn all problems
  -> sometimes other models perform better or more efficient

GENERALIZATION, UNDERFITTING AND OVERFITTING

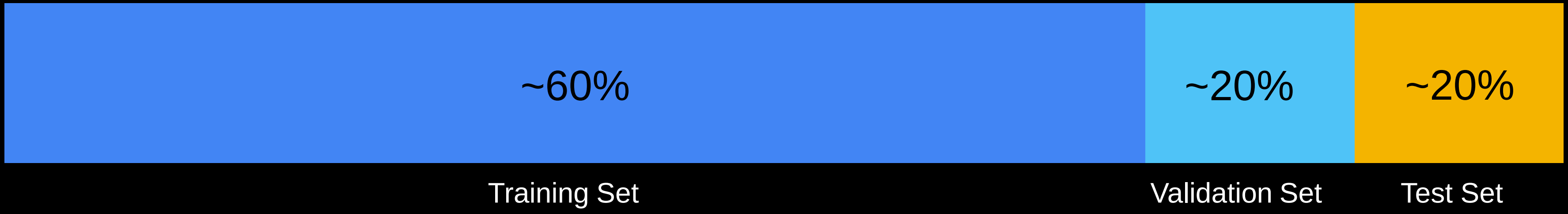


TIPS

- Visualize your tree as you are training.
- Follow the decision path of samples of interest.
- Use max_depth to control the size of the tree to prevent overfitting.
- Use min_samples_split or min_samples_leaf to ensure that multiple samples inform every decision in the tree.
- Optionally apply exhaustive or randomized hyperparameter search
- Optionally apply post pruning based on cost complexity

SPLITTING DATA

- Slice data into three subsets: Training, validation and test data



~60%

~20%

~20%

Training Set

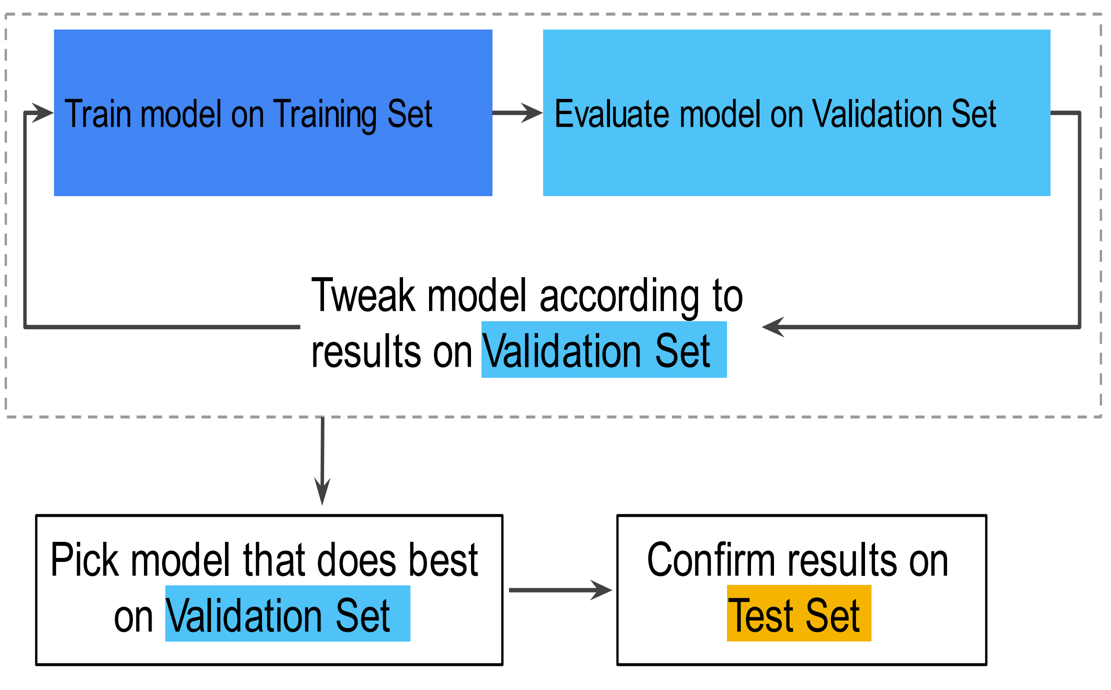
Validation Set

Test Set

- Make sure that your subsets meet the following conditions:
  - Large enough to yield statistically meaningful results.
  - Representative of the data set as a whole.
    E.g. don't pick a test set with different characteristics than the training set.

TRAINING, VALIDATION, TESTING

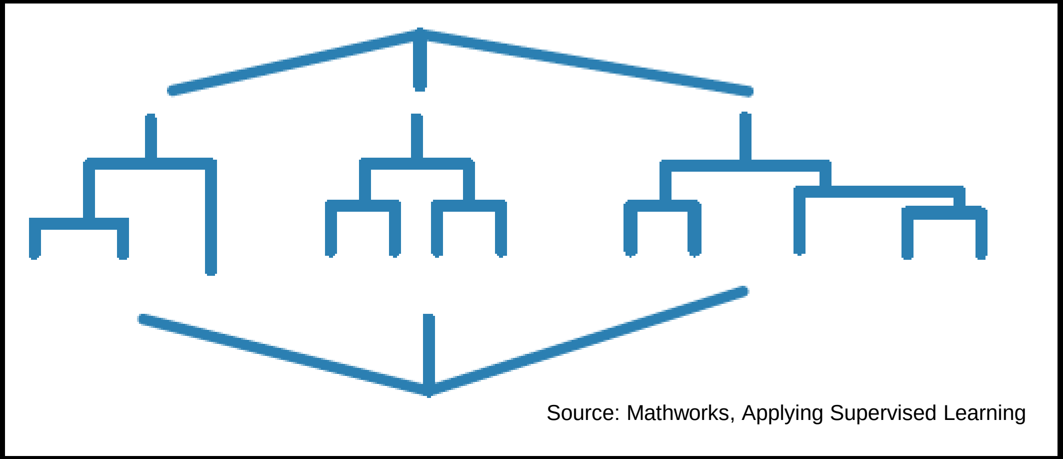
- Never train on test data!



Train model on Training Set
Evaluate model on Validation Set
Tweak model according to results on Validation Set
Pick model that does best on Validation Set
Confirm results on Test Set

ENSEMBLE LEARNING

- Wisdom of the crowd
- Group of predictors
- Random forest

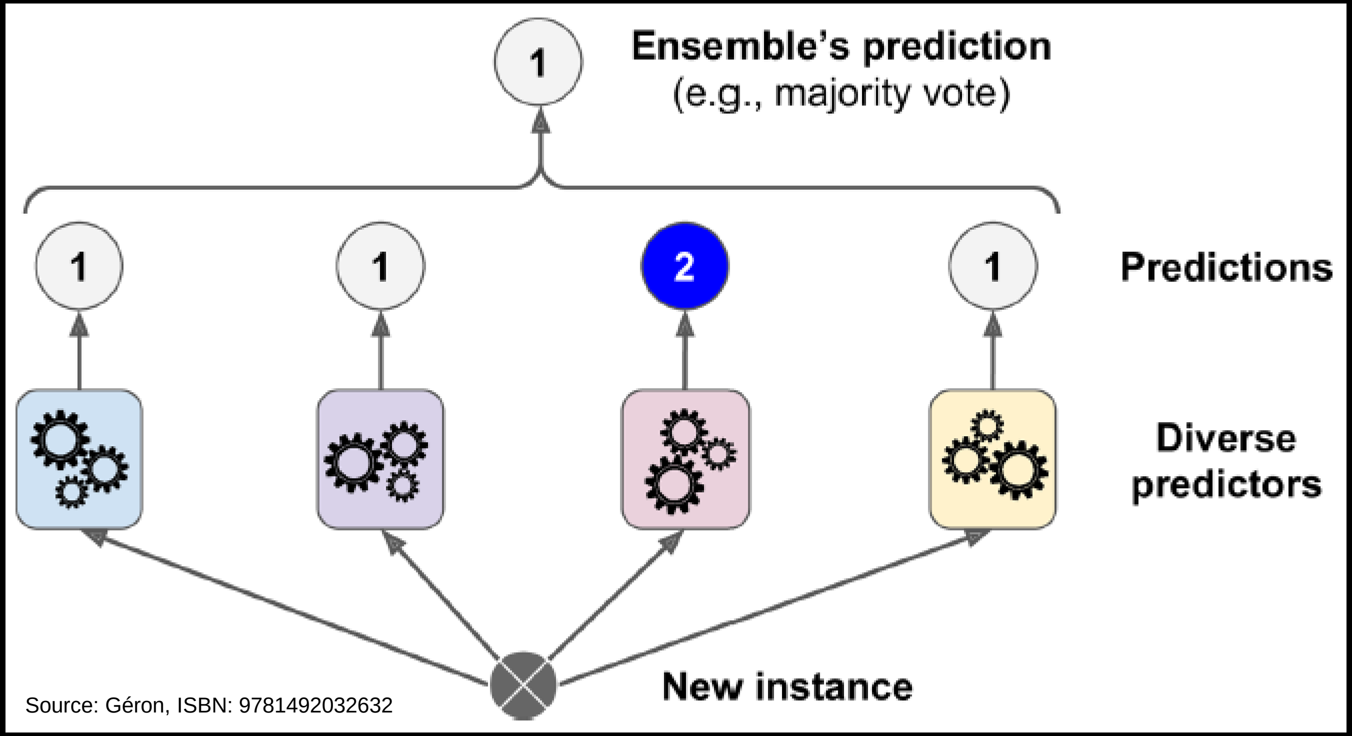


Source: Mathworks, Applying Supervised Learning

- Several “weaker” decision trees are combined into a “stronger” ensemble

HARD VOTING CLASSIFIER

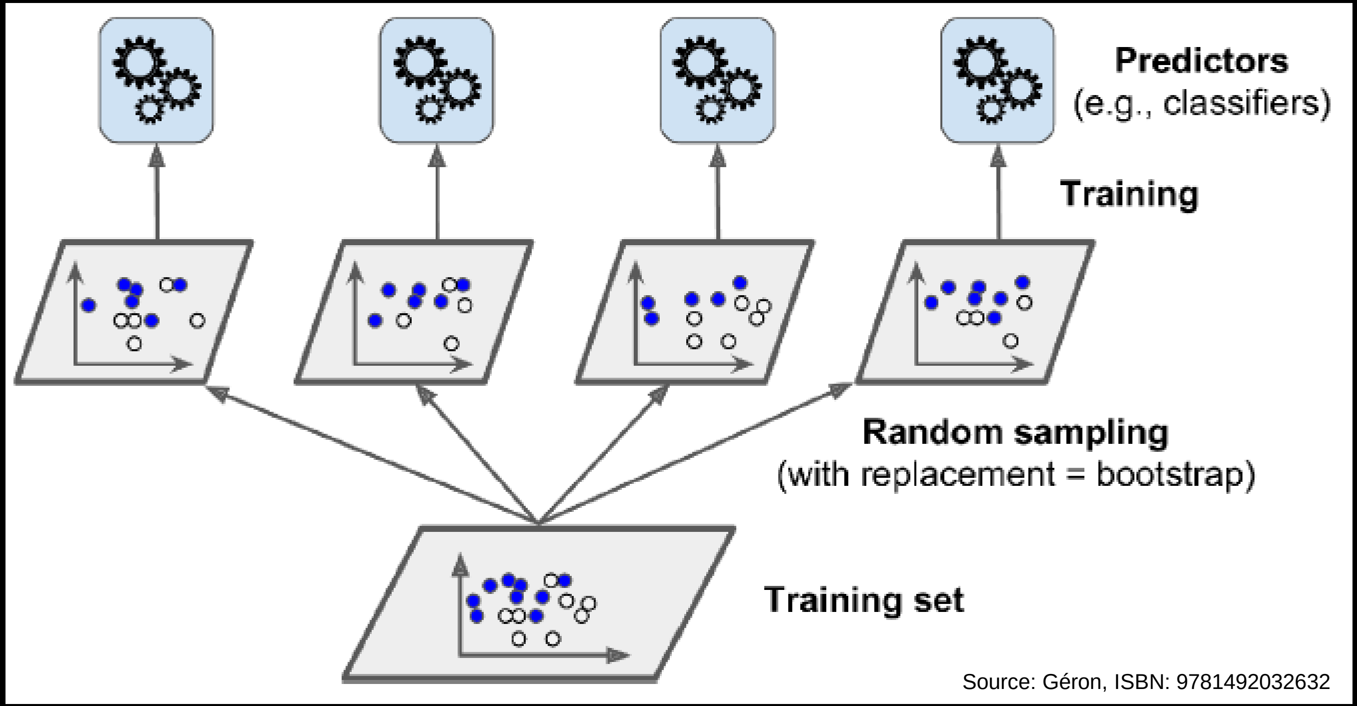
- Majority-vote can be strong given sufficient diversity



Source: Géron, ISBN: 9781492032632

BAGGING PREDICTORS

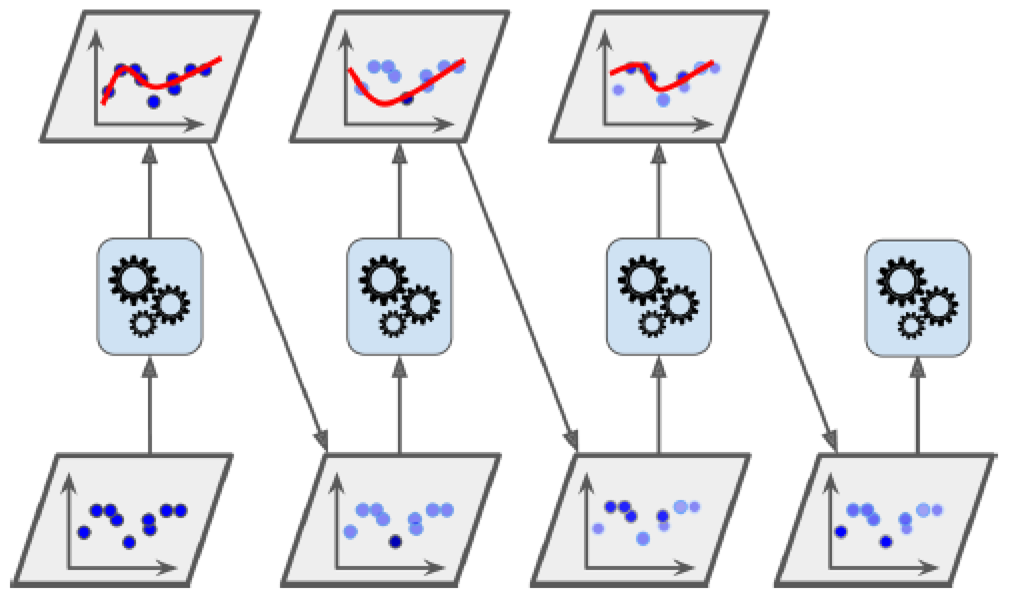
- Trees are trained independently on bootstrapped data



Source: Géron, ISBN: 9781492032632

BOOSTING

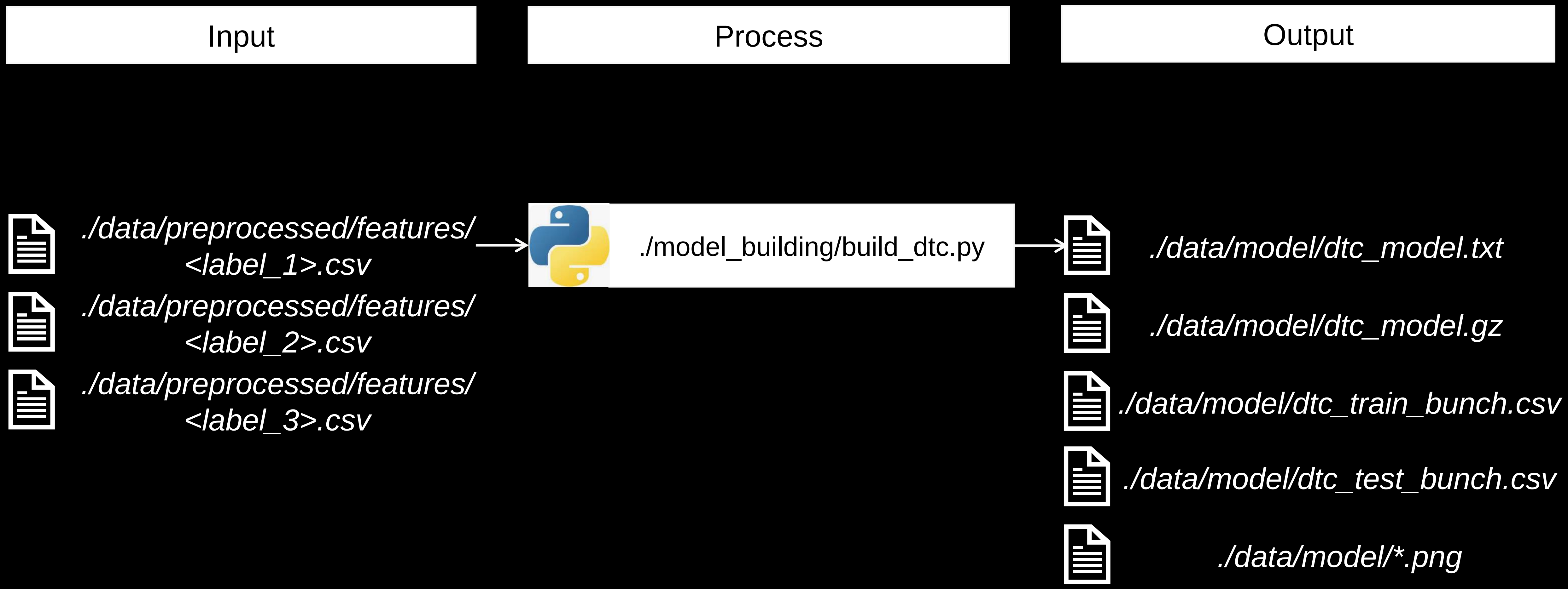
- Sequentially adding predictors to an ensemble, each one correcting its predecessor



Source: Géron, ISBN: 9781492032632

HAN_UNIVERSITY OF APPLIED SCIENCES

MODEL DESIGN



Input

Process

Output

./data/preprocessed/features/<label_1>.csv

./data/preprocessed/features/<label_2>.csv

./data/preprocessed/features/<label_3>.csv

./model_building/build_dtc.py

./data/model/dtc_model.txt

./data/model/dtc_model.gz

./data/model/dtc_train_bunch.csv

./data/model/dtc_test_bunch.csv

./data/model/*.png

MODEL DESIGN

./data/model/dtc_train_bunch.csv

```
label,timestamp1,timestamp2,x_out_fir_rescale_variance,y_out_fir_rescale_variance,z_out_fir_rescale_variance
left_right,,,0.080616518855509491,0.0011965184239670634,2.7622331799648236e-06
up_down,,,0.0010762271704152226,0.08740243315696716,8.584440365666524e-06
up_down,,,0.0014844088582322001,0.06616755574941635,7.747937161184382e-06
up_down,,,0.0265665240585804,0.0593581758439540 86,0.03963468596339226
stationary,,,6.266510155228389e-08,1.3307550261743017e-07,1.228962958066404e-07
stationary,,,9.401501444017413e-08,1.4411538984404615e-07,1.432363490039279e-07
up_down,,,0.002021343447268009,0.07682900130748749,6.288621989369858e-06
stationary,,,0.025755632668733597,1.0166539254896634e-07,0.025553688406944275
stationary,,,7.357205333846665e-08,8.371668513973418e-08,8.276586527244945e-08
left_right,,,0.10078003257513046,0.0047892015427351,4.227960744174197e-06
up_down,,,0.0024098153226077557,0.0726759135723114,7.771175660309382e-06
```

MODEL DESIGN

./data/model/dtc_test_bunch.csv

```
label,timestamp1,timestamp2,x_out_fir_rescale_variance,y_out_fir_rescale_variance,z_out_fir_rescale_variance
stationary,,,1.1347064798883366e-07,1.2575644348089554e-07,1.2267376803265506e-07
left_right,,,0.0840372741222381б,0.0015222649089992046,2.3235679691424593e-06
up_down,,,0.0018794061616063118,0.07356412708759308,5.5476843954238575e-06
stationary,,,4.993831126398618e-08,7.614894315111087e-08,1.7571814225902926e-07
stationary,,,5.802025526691068e-08,5.298928940078440e-08,1.366340285358092e-07
```

MODEL DESIGN

./data/model/dtc_model.gz

Dump of the Python object containing the trained Decision Tree Classifier model

MODEL DESIGN

./data/model/dtc_model.txt

```
Decision tree plot:
-----------------------------------------------------------
|--- y_out_fir_rescale_variance <= 0.00
|   |--- class: stationary
|--- y_out_fir_rescale_variance >  0.00
|   |--- x_out_fir_rescale_variance <= 0.03
|   |   |--- class: up_down
|   |--- x_out_fir_rescale_variance >  0.03
|   |   |--- class: left_right
```

Decision tree plot

MODEL DESIGN

./data/model/dtc_model.txt

```
Training report:
--------------------------------------------------------------
              precision    recall  f1-score   support

  left_right       1.00      1.00      1.00         7
  stationary       1.00      1.00      1.00         8
     up_down       1.00      1.00      1.00         7

    accuracy                           1.00        22
   macro avg       1.00      1.00      1.00        22
weighted avg       1.00      1.00      1.00        22

Training accuracy scores (cross-validated over 5 splits): 1.0 0.8 1.0 1.0 1.0
Average training accuracy: 0.9600 +/- 0.0800
```

Training report

MODEL DESIGN

./data/model/dtc_model.txt

```
Test report:
-------------------------------------------------------------
              precision    recall  f1-score   support

  left_right       1.00      1.00      1.00         3
  stationary       1.00      1.00      1.00         2
     up_down       1.00      1.00      1.00         3

    accuracy                           1.00         8
   macro avg       1.00      1.00      1.00         8
weighted avg       1.00      1.00      1.00         8

Test accuracy score: 1.0000
```

Test report

MODEL DESIGN

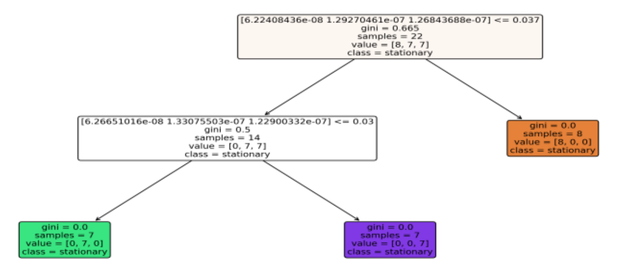
./data/model/dtc_model.txt

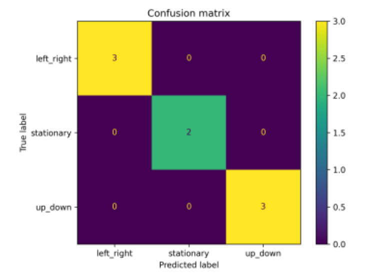
```
Confusion matrix:
-------------------------------------------------------------
[[3 0 0]
 [0 2 0]
 [0 0 3]]
```

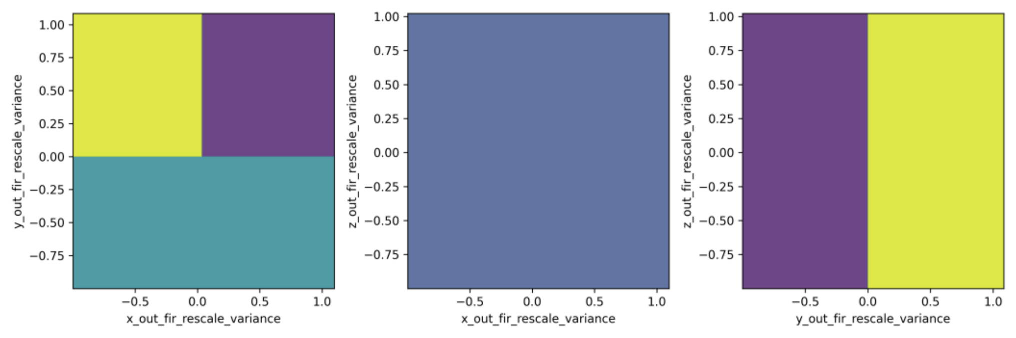
Confusion matrix

MODEL DESIGN

./data/model/.png*







tree
confusion_matrix
decision_surfaces