17-18 July 2019 | Bayer AG R&D, Berlin, Germany

classical machine learning algorithms

Marco Chierici
with Margherita Francescatto

Fondazione Bruno Kessler
chierici@fbk.eu

BAYER

FONDAZIONE BRUNO KESSLER | ICT CENTER FOR COMMUNICATION AND INFORMATION TECHNOLOGY

MPBA PREDICTIVE MODELS FOR BIOMEDICINE & ENVIRONMENT

outline

- Recap: yesterday's main points
- Simple classification algorithms/approaches
  - Random forest
  - Support Vector Machines
- Concepts of parameter tuning
- Concepts of feature selection
- More on reproducibility & the MAQC Data Analysis Plan

recap
yesterday's take-home messages

learning

- supervised (teacher)
- unsupervised (no teacher)
- reinforcement (learn from experience)

model

- knowledge
- algorithms

training data

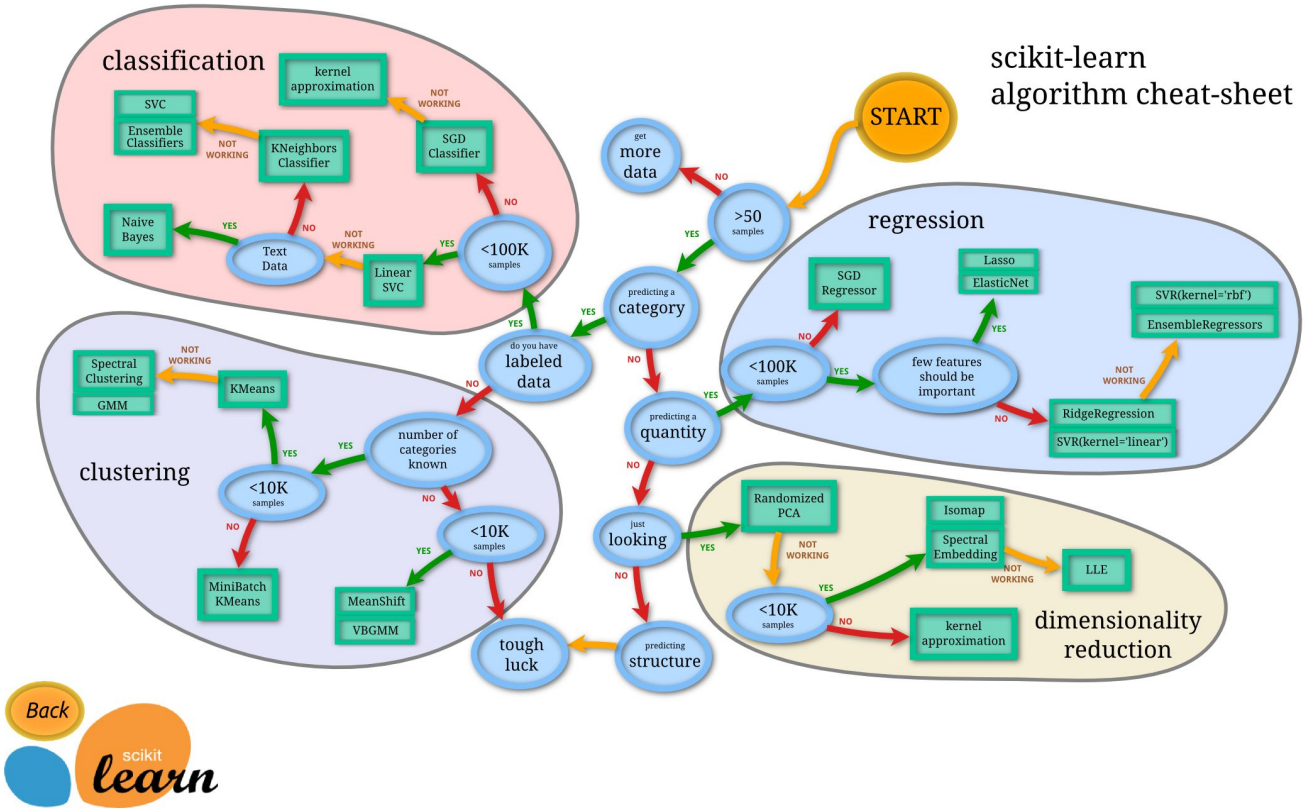
- a set of examples represented by feature vectors
- [optional] target values for each example

generalization

reasonable / unbiased predictions on unseen data

beyond k-NN

- k-NN is about as straightforward as it can get
- There are many more approaches more sophisticated than that



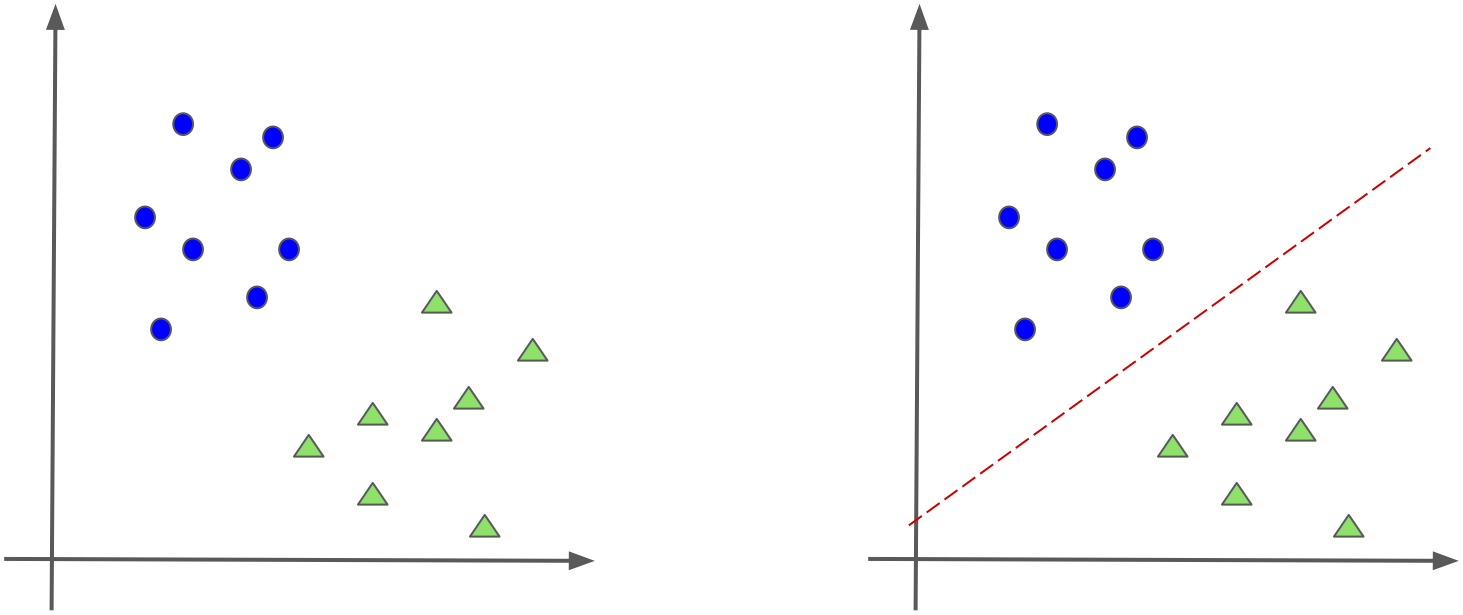
We want to talk about something more elaborate and widely used in the field of classification

Support Vector Machines

Random Forests

support vector machines /1

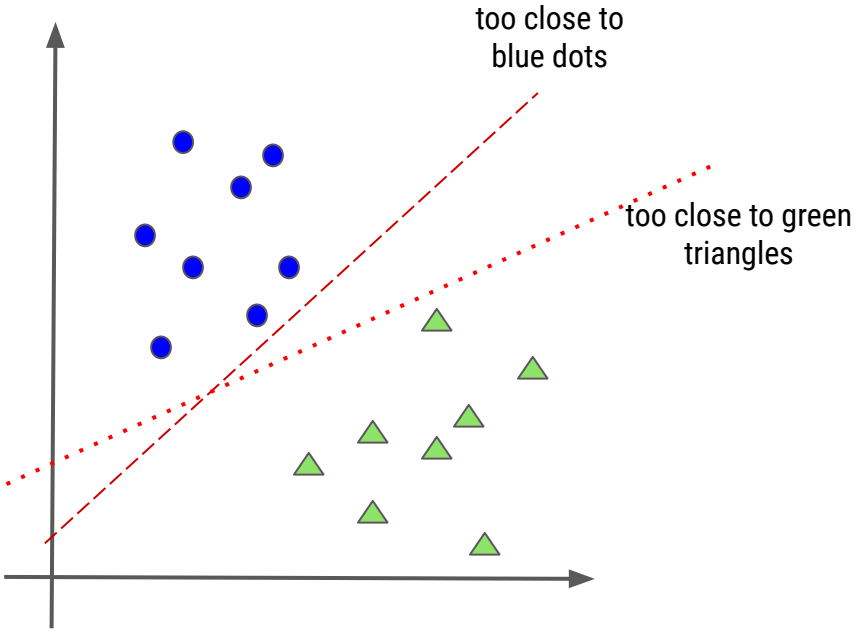
SVMs formalize and generalize the idea of “drawing a line” to separate samples

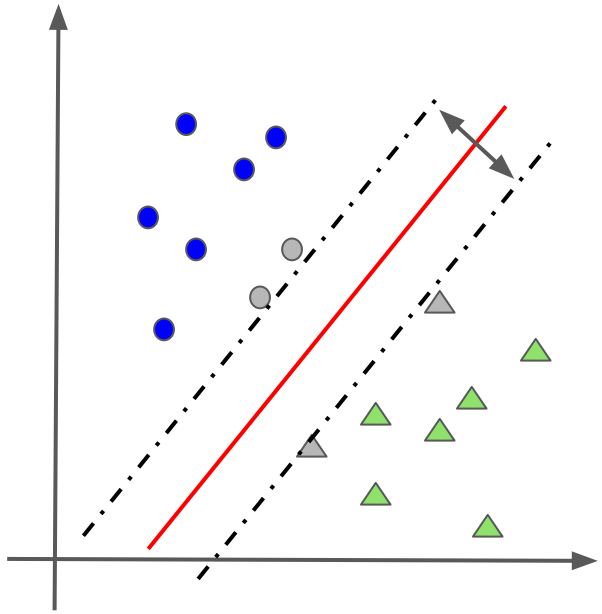


The original SVM algorithm was invented by Vladimir N. Vapnik and Alexey Ya. Chervonenkis in 1963

support vector machines /2

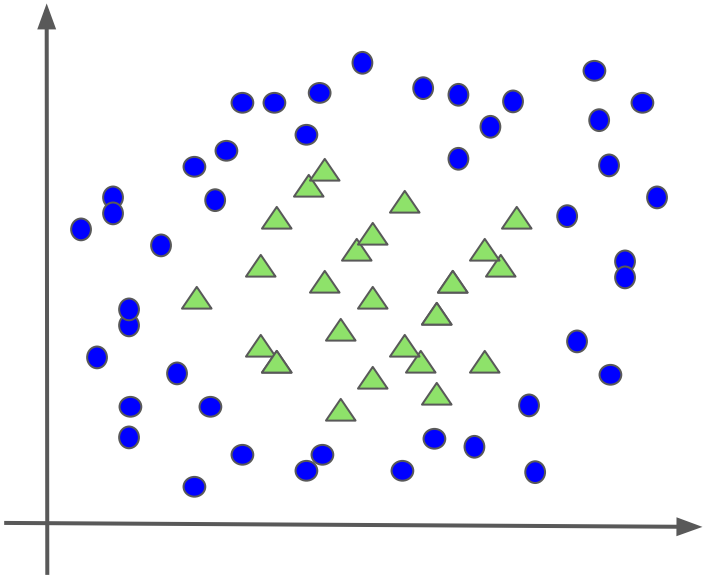
- In general, many solutions possible
- SVMs find the optimal one: technically they maximize the **margin**, i.e. the street around the separating line
- Decision function usually defined by a small subset of training examples, the **support vectors**



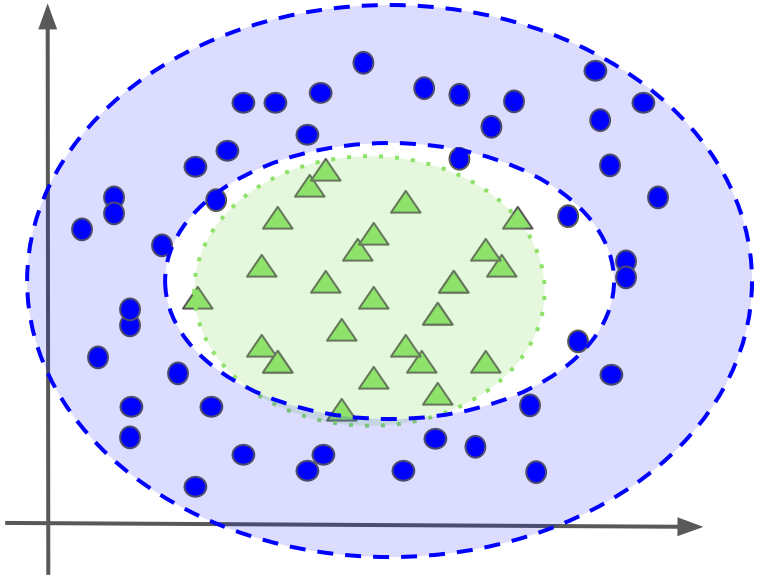


example: linear SVM

beyond linear SVMs



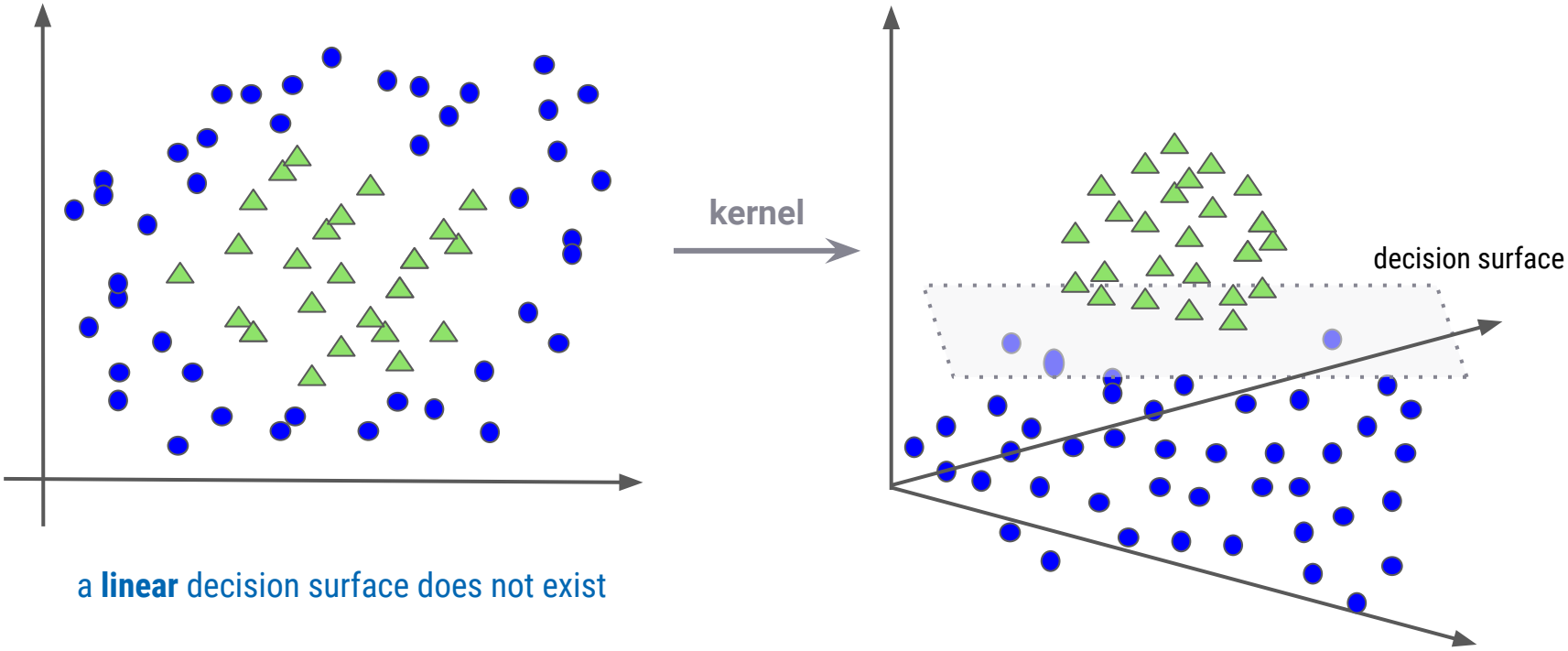
No good line to split groups



Alternative “shapes” would capture the distributions

⇒ Non-linear SVM

SVMs & kernels



a **linear** decision surface does not exist

- map the data into a much higher dimensional space (feature space) where a decision surface can be found
- the mapping is achieved by a mathematical projection (**kernel**)

SVMs pros & cons

pros

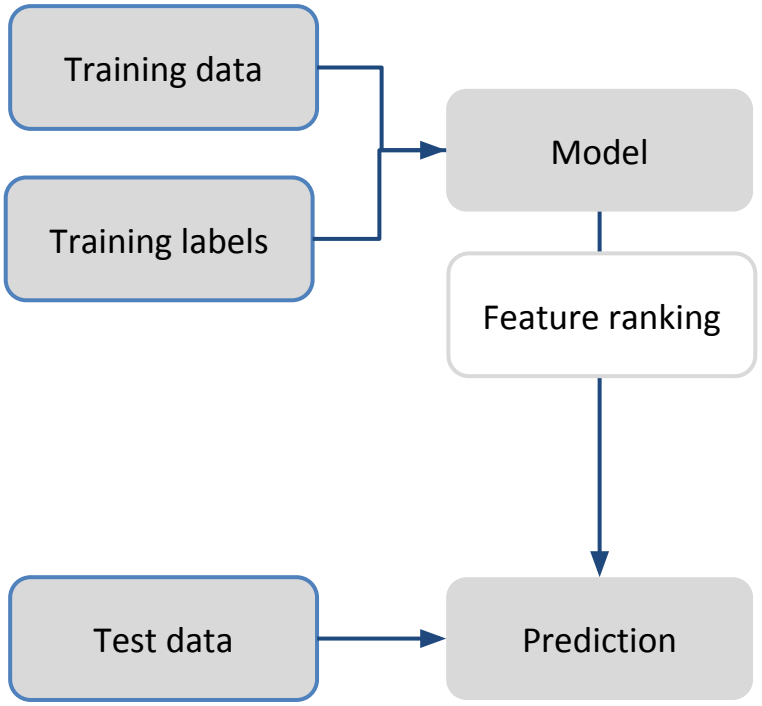
- effective in high dimensional spaces
- Still effective in cases where number of dimensions is greater than the number of samples.
- Use of support vectors makes them memory efficient
- Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels

cons

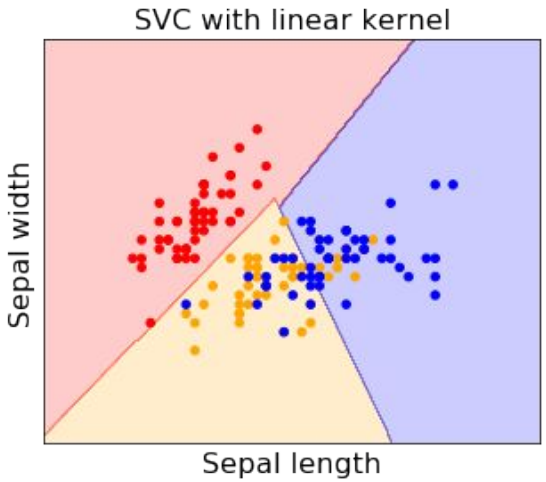
- If #features >> #samples, avoid over-fitting in choosing Kernels and parameters is crucial
- SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation

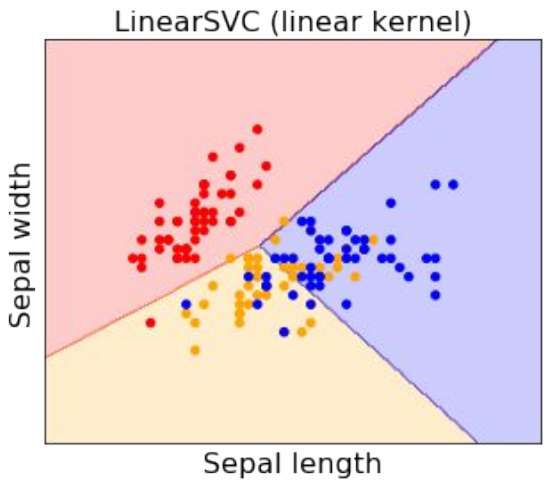
SVMs
in scikit-learn

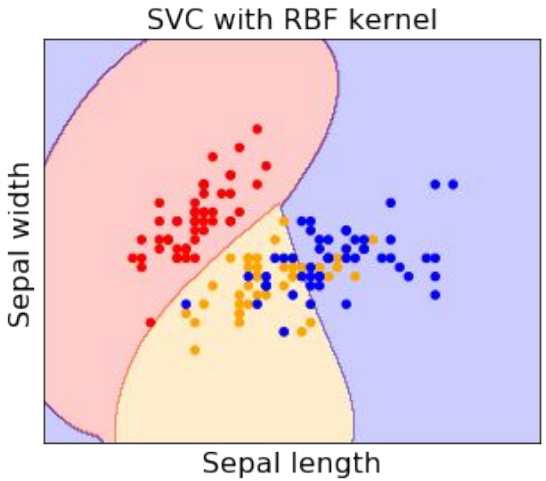
```
from sklearn.svm import SVC

clf = SVC(kernel='linear', C=100)

clf.fit(x_tr, y_tr)

y_pred = clf.predict(x_ts)
```

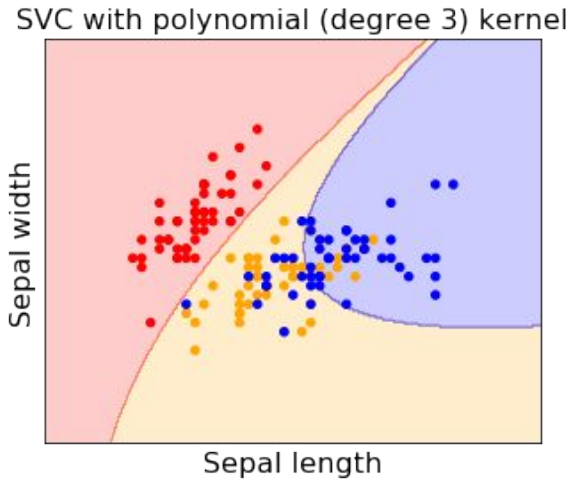


SVM on Iris data









SVM ‘poly’ on Iris data
C = 0.01, degree = 3

Metrics for SVM classification on **sepal width/length**:

Accuracy = 0.83
Sensitivity = 0.83
MCC = 0.75

	precision	recall	f1-score	support
0	1.00	1.00	1.00	10
1	0.70	0.78	0.74	9
2	0.80	0.73	0.76	11
avg / total	0.84	0.83	0.83	30

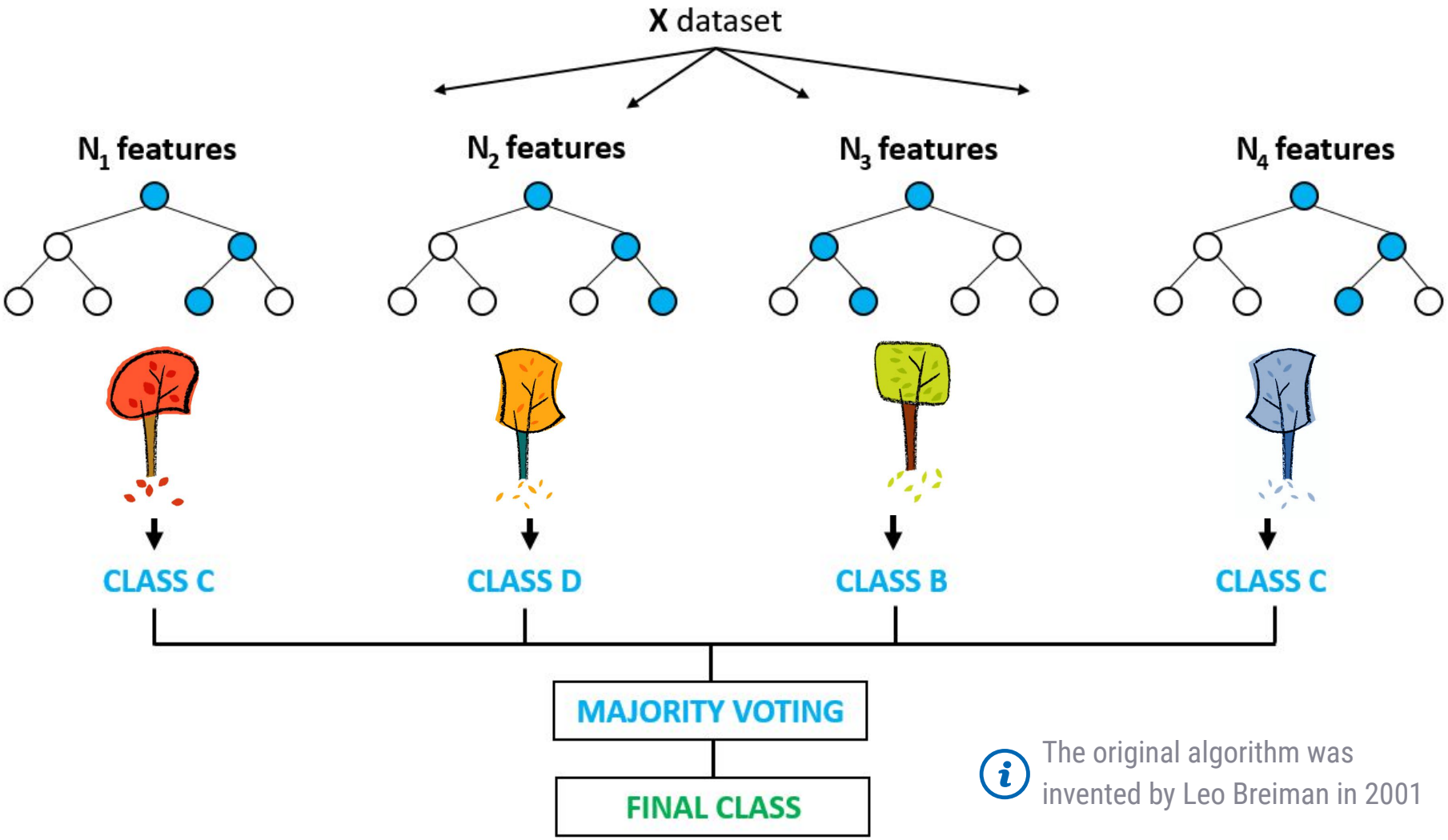
Metrics for SVM classification on **petal width/length**:

Accuracy = 1.0
Sensitivity = 1.0
MCC = 1.0

	precision	recall	f1-score	support
0	1.00	1.00	1.00	10
1	1.00	1.00	1.00	9
2	1.00	1.00	1.00	11
avg / total	1.00	1.00	1.00	30

random forests

Basic idea: generate multiple small decision trees from random subsets of the data



The original algorithm was invented by Leo Breiman in 2001

RFs pros & cons

pros

- one of the most accurate learning algorithms available
- estimates feature importances
- maintains accuracy even when a large proportion of data is missing
- insensitive to feature distribution: no need to rescale the data

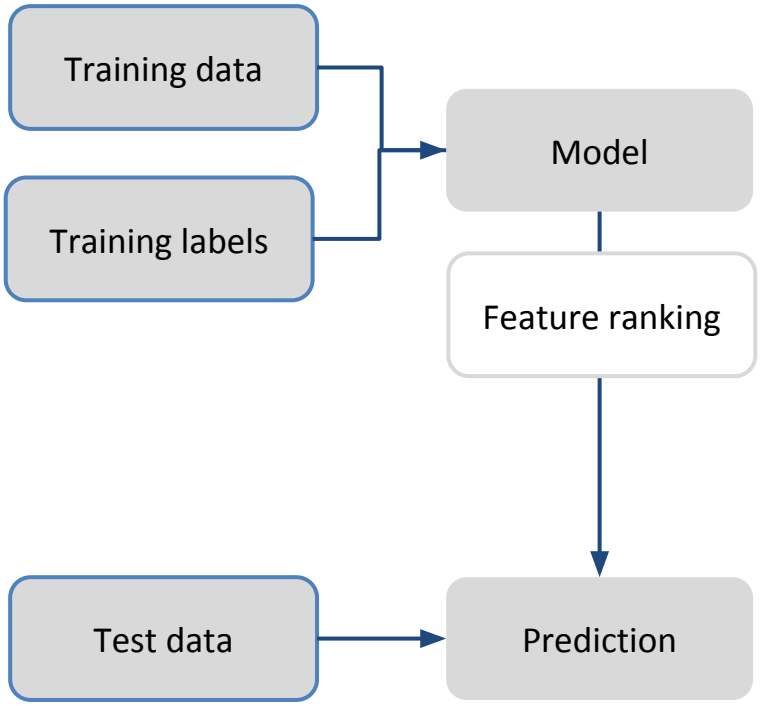
cons

- can overfit datasets with noisy classification tasks
- classification more difficult for humans to interpret

random forests
in scikit-learn

```
from sklearn.ensemble import RandomForestClassifier

clf = RandomForestClassifier(n_estimators=500)
```

```
clf.fit(x_tr, y_tr)
```




```
y_pred = clf.predict(x_ts)
```

RF on Iris data
n_estimators = 10

Metrics for RF classification on **sepal width/length**:

Accuracy = 0.8

Specificity (recall score) = 0.8

MCC = 0.71

	precision	recall	f1-score	support
0	1.00	1.00	1.00	10
1	0.64	0.78	0.70	9
2	0.78	0.64	0.70	11
avg / total	0.84	0.83	0.83	30

Metrics for RF classification on **petal width/length**:

Accuracy = 1.0

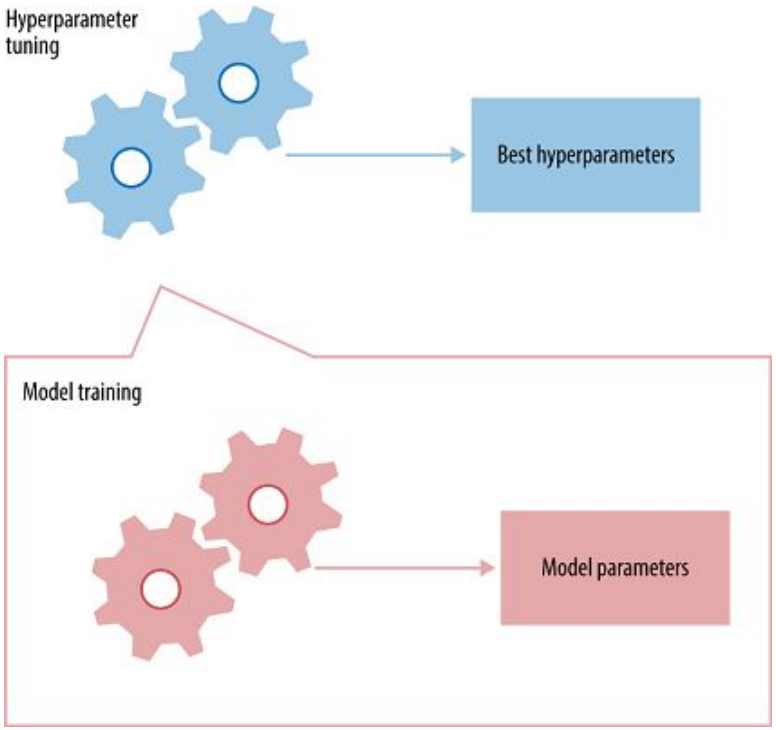
Sensitivity = 1.0

MCC = 1.0

	precision	recall	f1-score	support
0	1.00	1.00	1.00	10
1	1.00	1.00	1.00	9
2	1.00	1.00	1.00	11
avg / total	1.00	1.00	1.00	30

parameter tuning

- Parameter tuning (or optimization): identifying a set of optimal parameters, or “hyperparameters”
  - “Hyperparameters” indicate parameters that are not learned during the training
- Hyperparameter optimization: finding a set of optimal hyperparameters that define a model minimizing a predefined **loss function** on given independent data
- Scikit-learn offers an extremely functional model that gives the possibility to search the best parameters within a pre-defined grid: **GridSearchCV**



parameter tuning SVM

$$\min_{\mathbf{w}\in\mathbb{R}^d,\xi_i\in\mathbb{R}+} ||\mathbf{w}||^2 + C\sum_i^N \xi_i$$

- Linear: 1 parameter

  $k(x, x_i) = x \cdot x_i$

  - C: penalty parameter of the error term; controls the **trade off** between smooth decision boundary and classifying the training points correctly
- Gaussian: 2 parameters

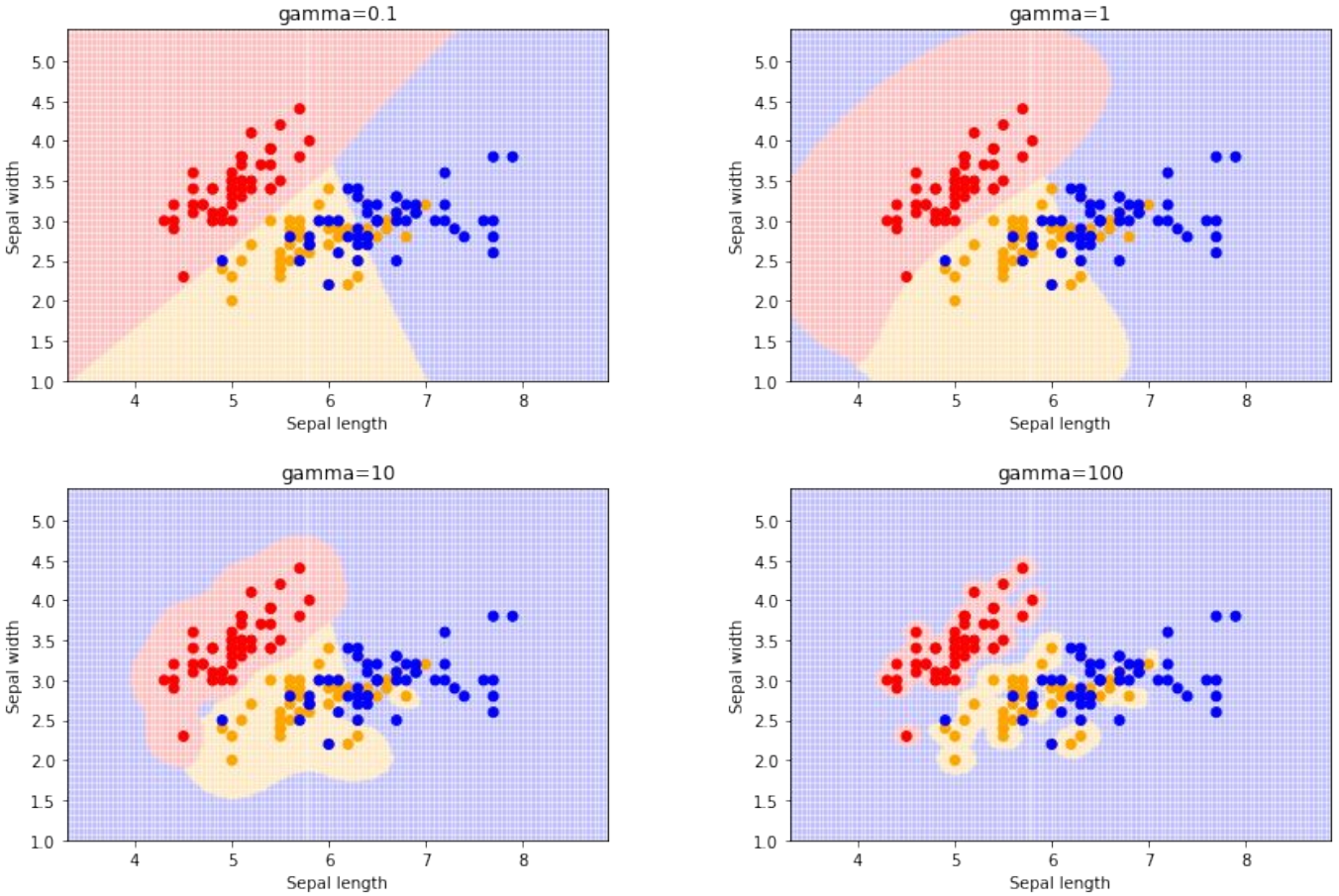
  $k(x, x_i) = \exp(-\text{gamma} * \text{sum}(x-x_i)^2)$

  - C
  - gamma: parameter for non-linear hyperplanes; the higher gamma, the tighter the fit to the training data
- Polynomial: 4 parameters

  $k(x, x_i) = (x \cdot x_i + \text{coef})^d$

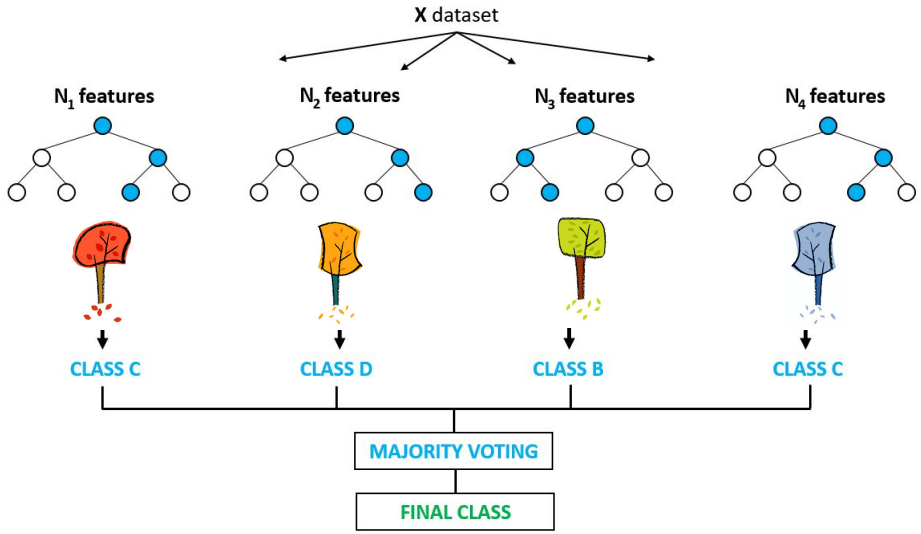
  - C
  - gamma
  - degree: the degree of the polynomial used to find the hyperplane
  - coef: independent term

impact of parameter tuning
SVM / on iris

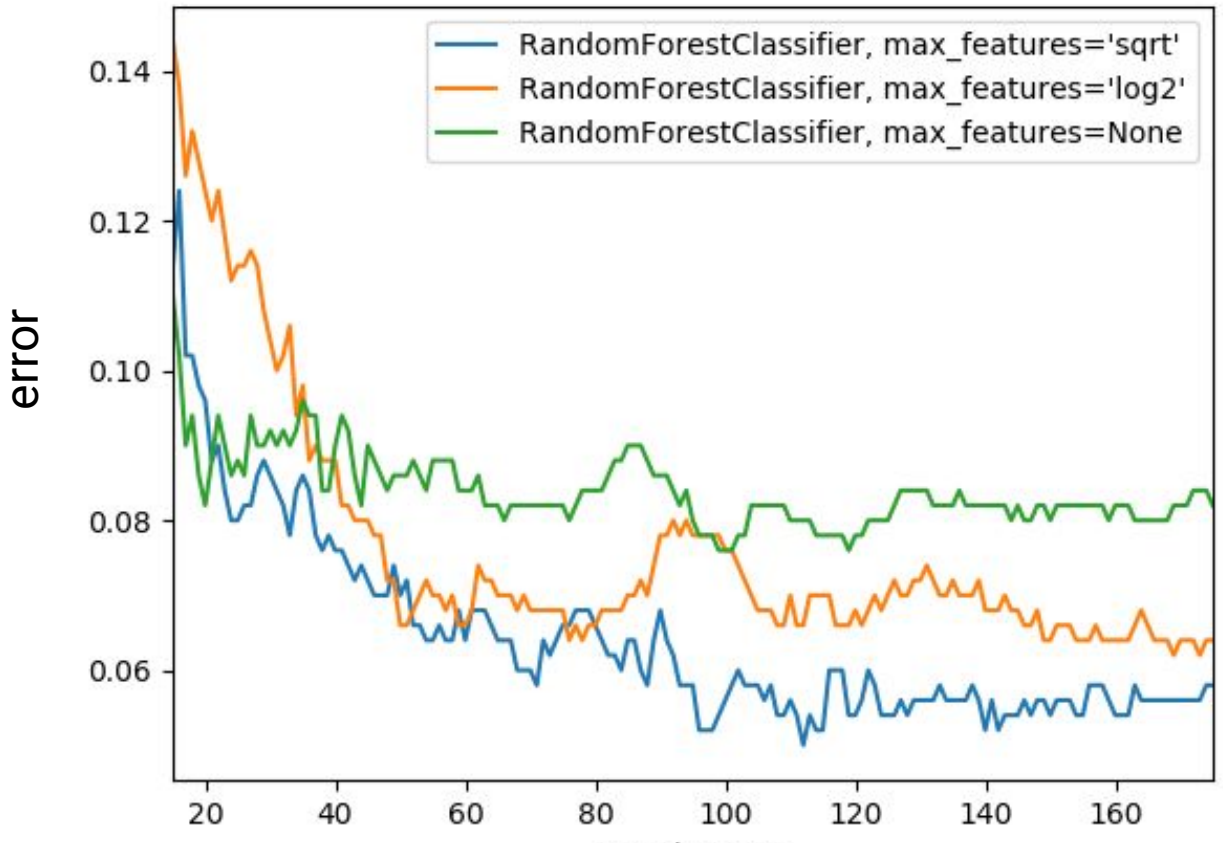


parameter tuning
RF

- Number of trees in the forest
- Maximum depth of each tree in the forest
- Minimum number of samples required to split an internal node
- Minimum number of samples required to be at a leaf node
- Maximum number of features to consider when evaluating best split

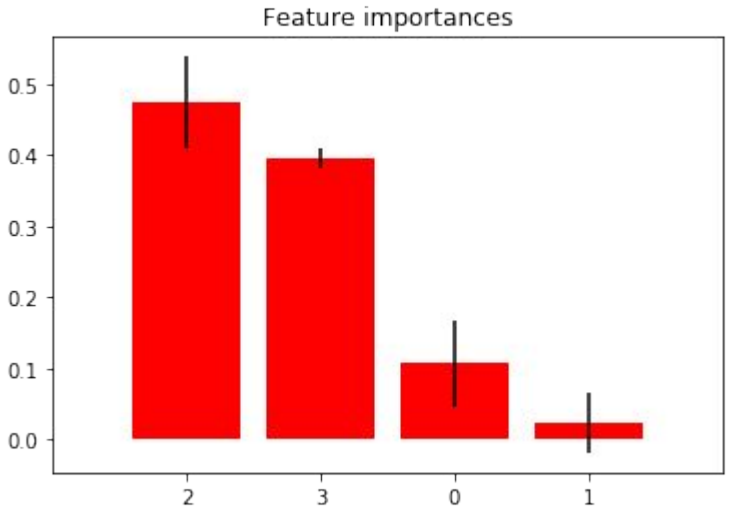


impact of parameter tuning
RF



feature ranking

- Not necessarily all the features are relevant for classification
- Idea: ranking features by their importance in classifying the data
- Some models natively provide a feature ranking scheme: e.g. importance/entropy (RFs) or model weights (linear SVMs)

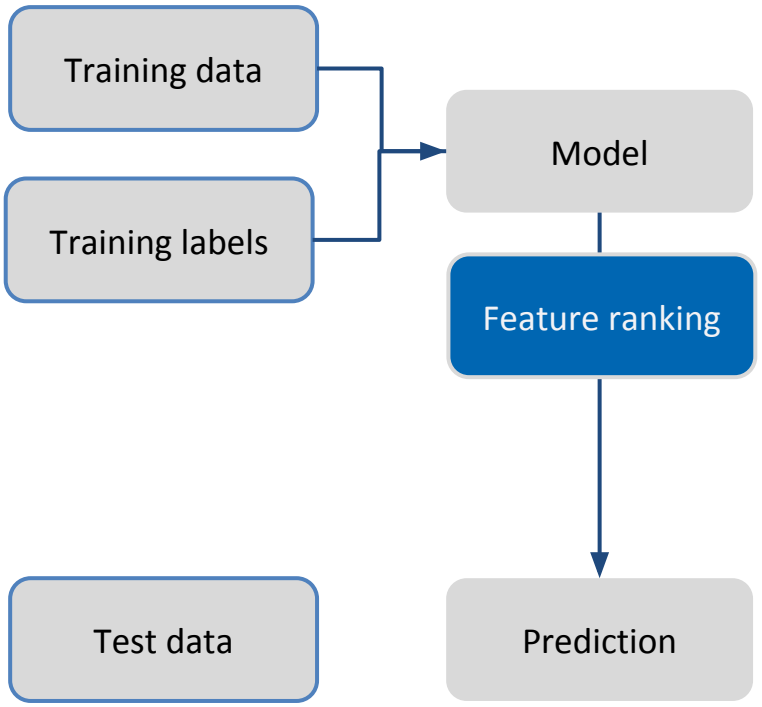


Features:
0. sepal length (cm)
1. sepal width (cm)
2. petal length (cm)
3. petal width (cm)

Feature ranking:
1. feature 2 (0.473662)
2. feature 3 (0.395515)
3. feature 0 (0.107411)
4. feature 1 (0.023413)

RF feature ranking in scikit-learn

```
from sklearn.ensemble import RandomForestClassifier

clf = RandomForestClassifier(n_estimators=500)


clf.fit(x_tr, y_tr)


importances = clf.feature_importances_
indices = np.argsort(importances)[::-1]



y_pred = clf.predict(x_ts)
```



need for guidelines!
to ensure unbiased model estimates

```
                    from sklearn.ensemble import RandomForestClassifier
                    From sklearn import metrics
```

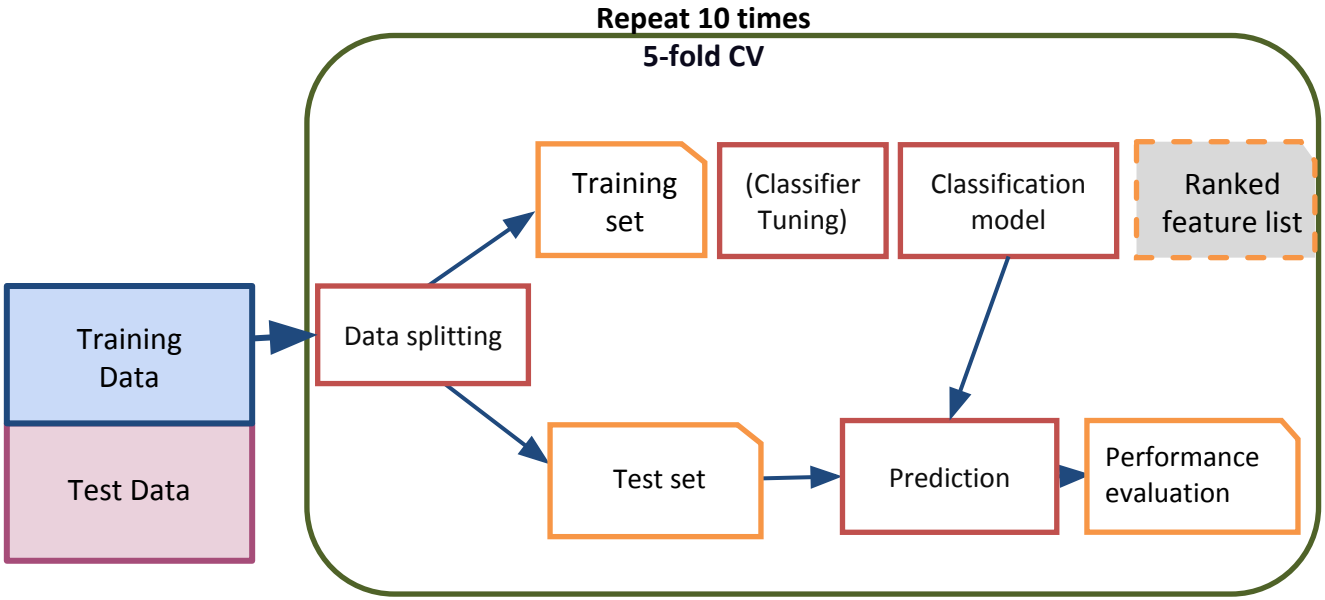

```
y_pred = clf.predict(
                                                                    True, random_state=0)
    for idx_tr, idx_ts in skf:
        X_train, Y_train = x_tr[idx_tr], y_tr[idx_tr]
        X_test, Y_test = x_tr[idx_ts], y_tr[idx_ts]


    metrics.matthews_corrcoef(y_ts, y_pred)
```

reproducibility
MAQC-II

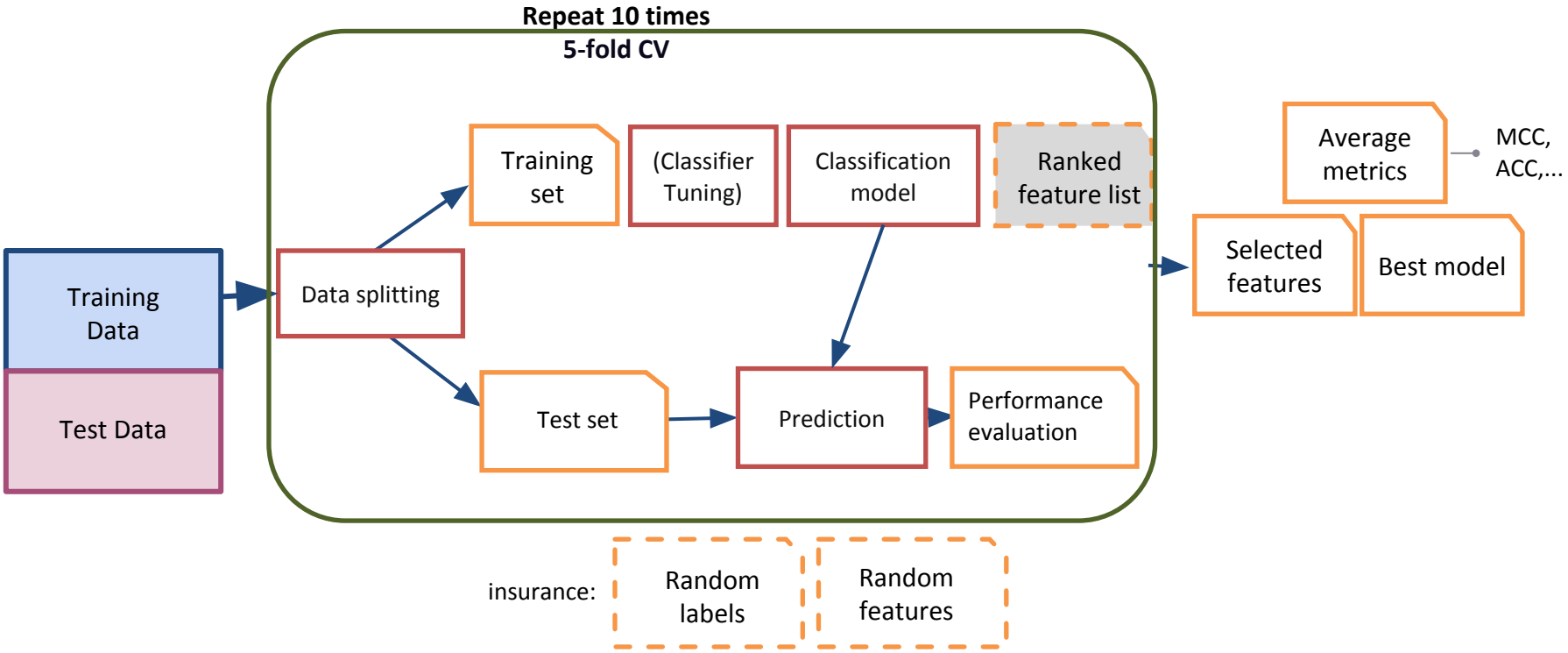
Identifying a set of guidelines for predictive profiling* (microarray and NGS) *through a suite of testbed datasets

1. Predictive models can be derived from high-throughput data,
2. But they need to be carefully developed and independently tested
3. Reproducibility requires substantial effort.

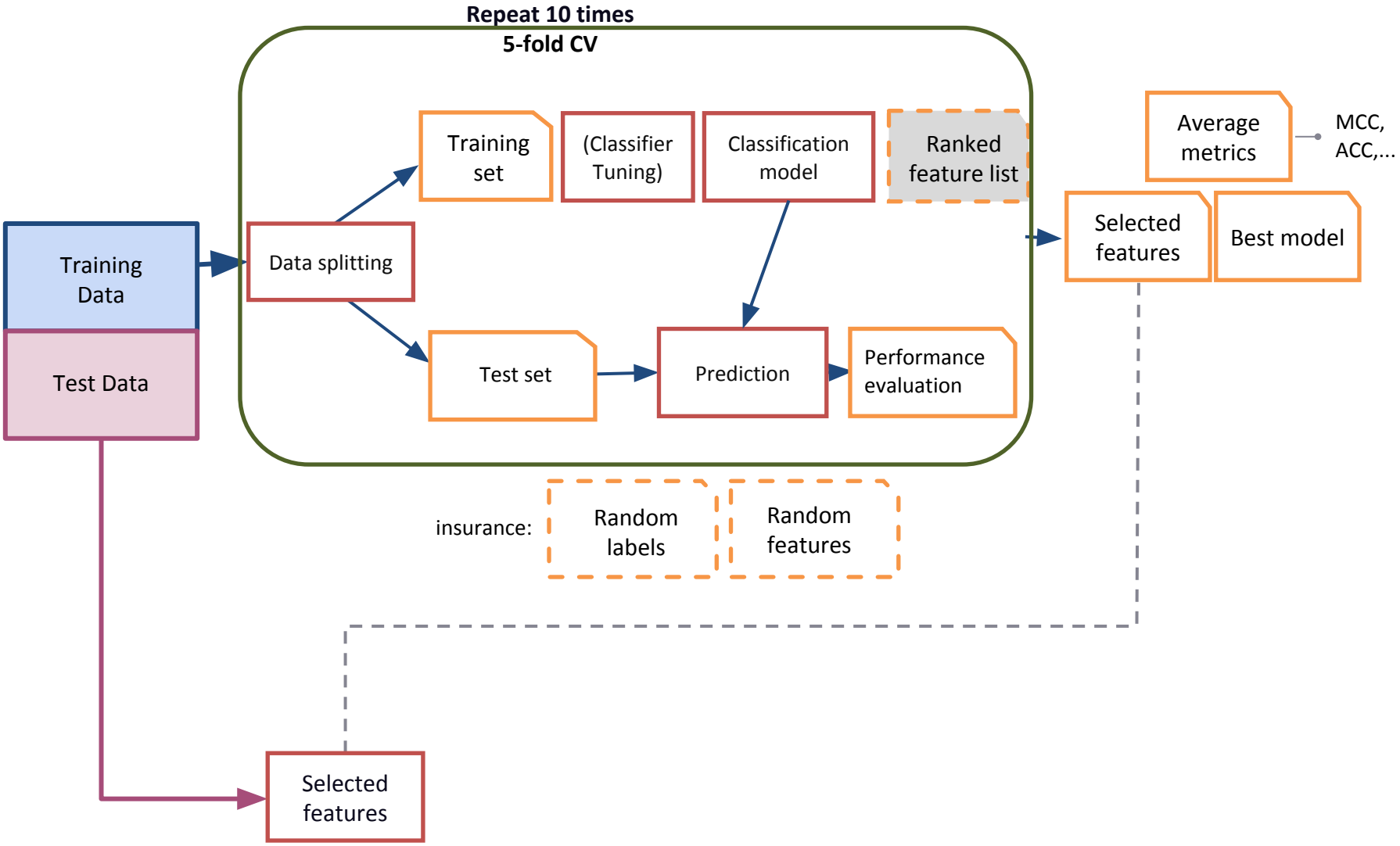
The MAQC-II/SEQC Data Analysis Plan



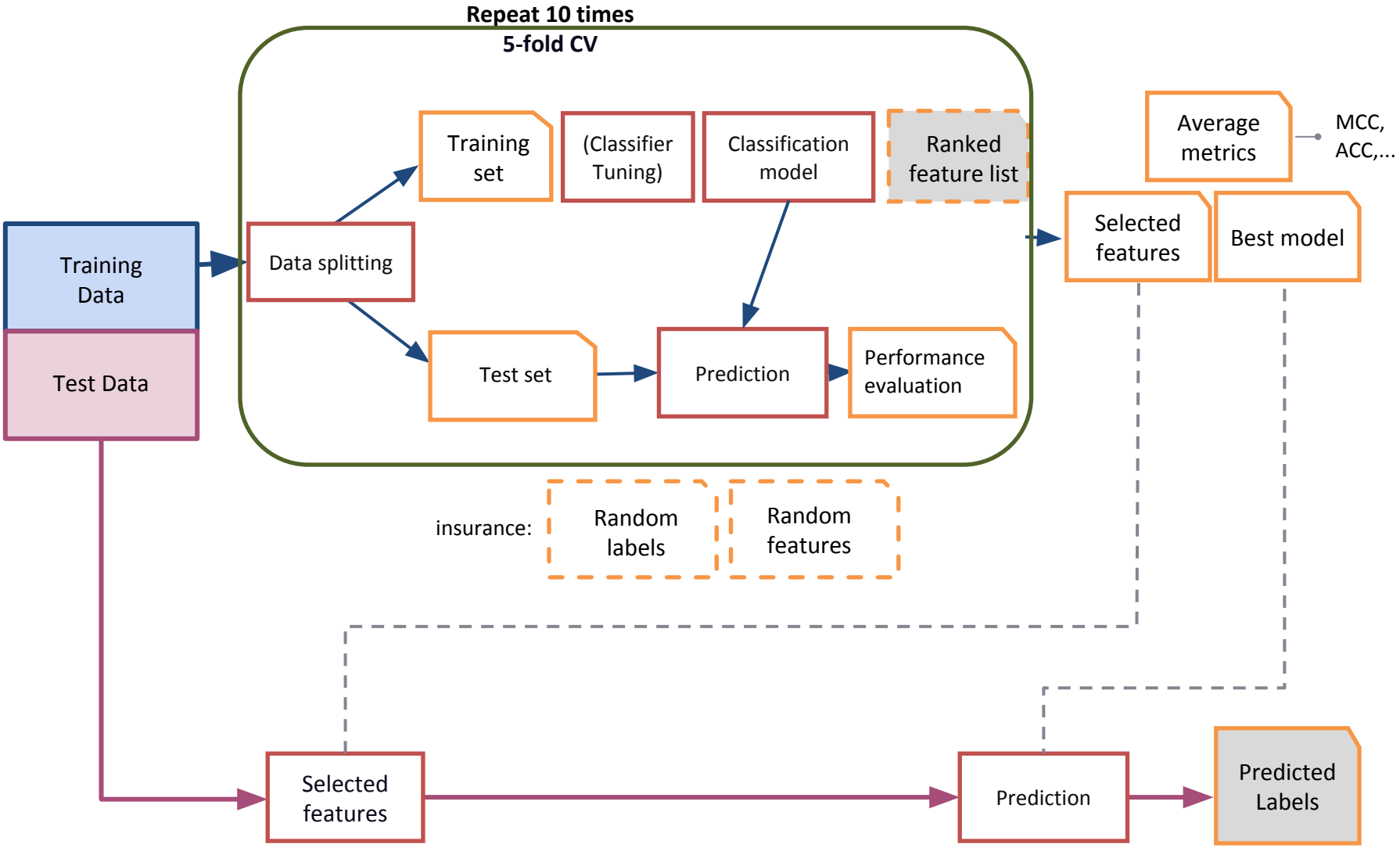
The MAQC-II/SEQC Data Analysis Plan



The MAQC-II/SEQC Data Analysis Plan



The MAQC-II/SEQC Data Analysis Plan



thank you / danke / grazie !

Acks

Giuseppe Jurman
Margherita Francescatto
Cesare Furlanello