Machine Learning

A Brief Introduction

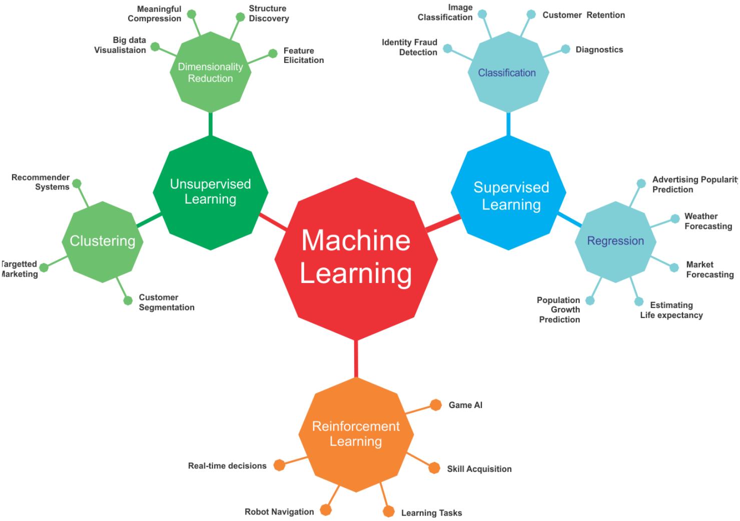
Marvin N. Wright

Leibniz Institute for Prevention Research & Epidemiology – BIPS
University of Bremen
University of Copenhagen

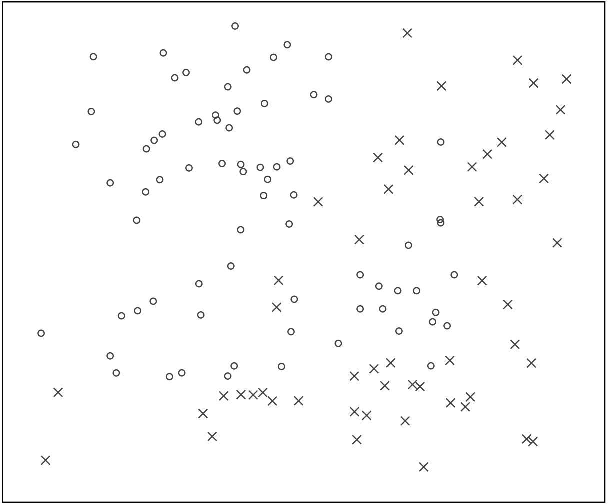
December 2023

Outline

Machine Learning

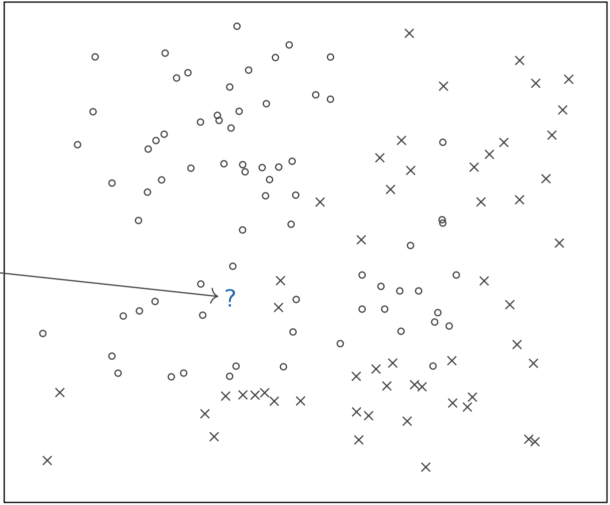


k-Nearest Neighbors



k-Nearest Neighbors

new
observation

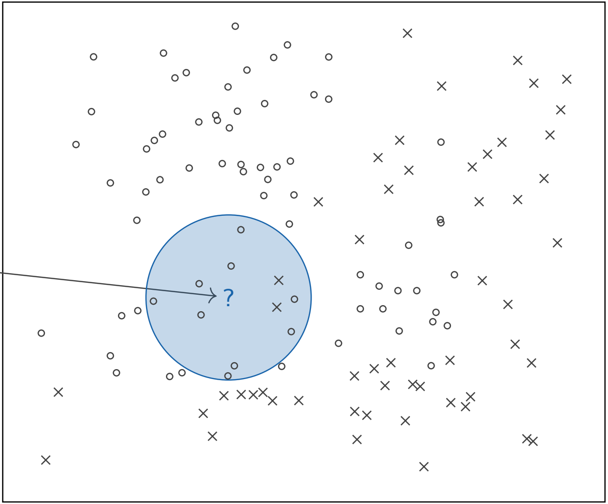


k-Nearest Neighbors

new
observation

11 nearest
neighbors

2x x
9x o

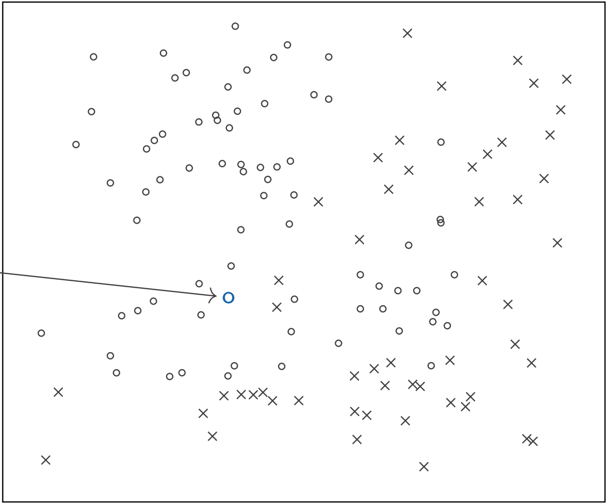


k-Nearest Neighbors

new
observation

11 nearest
neighbors

2x x
9x o



Example: House Prices

Predict the price for a house in a certain area

Features x				Target y
square footage of the house	**number of bedrooms**	**swimming pool (yes/no)**	**...**	**house price in US$**
1,180	3	0	...	221,900
2,570	3	1	...	538,000
770	2	0	...	180,000
1,960	4	1	...	604,000

Example: Length of Hospital Stay

Predict days a patient has to stay in hospital

Features x					Target y
diagnosis category	**admission type**	**gender**	**age**	**...**	**Length-of-stay in the hospital in days**
heart disease	elective	male	75	...	4.6
injury	emergency	male	22	...	2.6
psychosis	newborn	female	0	...	8
pneumonia	urgent	female	67	...	5.5

Example: Life Insurance

Predict risk category for a life insurance customer

Features x				Target y
job type	**age**	**smoker**	**...**	**risk group**
carpenter	34	1	...	3
stuntman	25	0	...	5
student	23	0	...	1
white-collar worker	39	0	...	2

Supervised Learning

Learn a functional relationship between **features** x and **target** y

Features x		**Target** y
People in Office (Feature 1) x_1	**Salary (Feature 2)** x_2	**Worked Minutes Week (Target Variable)**
4	4300 €	2220
12	2700 €	1800
5	3100 €	1920

$n = 3$ $\quad$ $p = 2$ $\quad$ $x_1^{(2)}$ $\quad$ $x_2^{(1)}$ $\quad$ $y^{(3)}$

Supervised Learning

Use labeled data to learn a model f

Use model f to predict target y of new data

Supervised Learning

Model

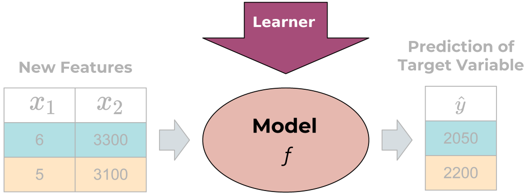
Functional relationship between **features** x and **target** y

Learner (or inducer)

Algorithm for finding model

Train Set

y	x_1	x_2
2200	4	4300
1800	12	2700
1920	15	3100



Example

- Learner: Artificial neural network (as a concept)
- Model: Actual network with learned weights

Models differ in size and complexity

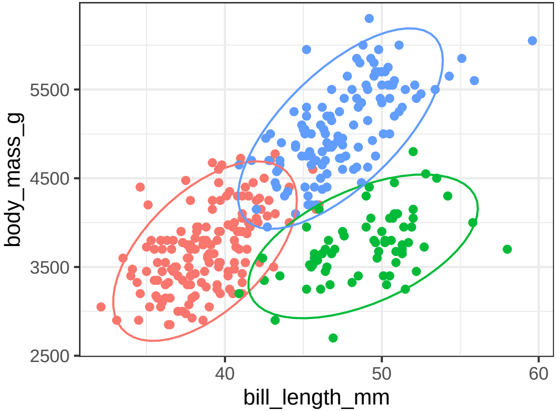
- Linear model: Coefficients β
- Neural network: Weights for all units in all layers
- Decision trees: Many binary splits
- k-nearest neighbors: Complete training data

Supervised Learning

Unsupervised Learning

No **target** y available

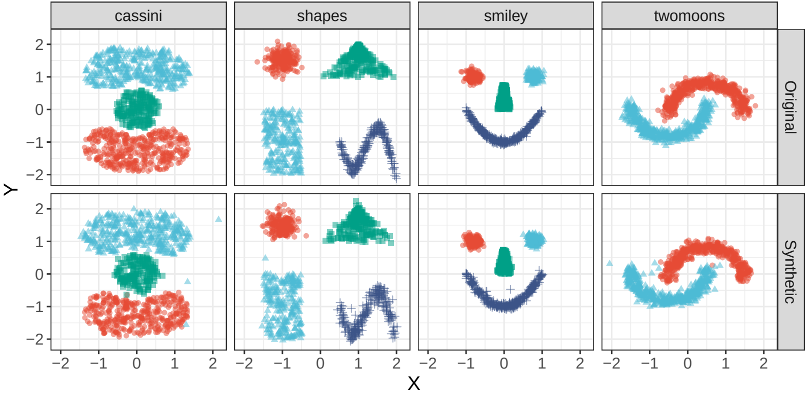
Search for patterns in the data x, e.g. clustering:



Generative Modeling

Learn data distribution (joint density)

Generate new data:



Watson et al. 2023 Proc Mach Learn Res 206:5357

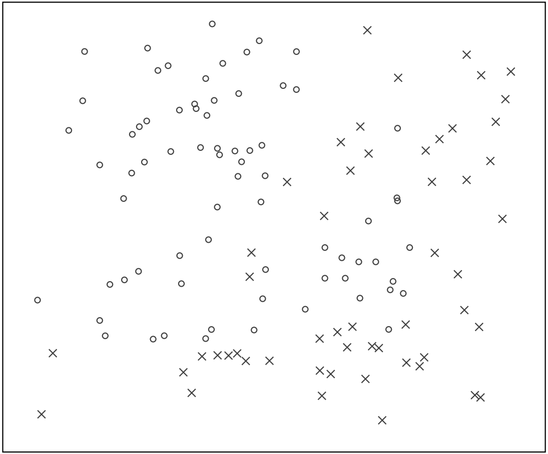
Supervised Learning

Use labeled data to learn a model f

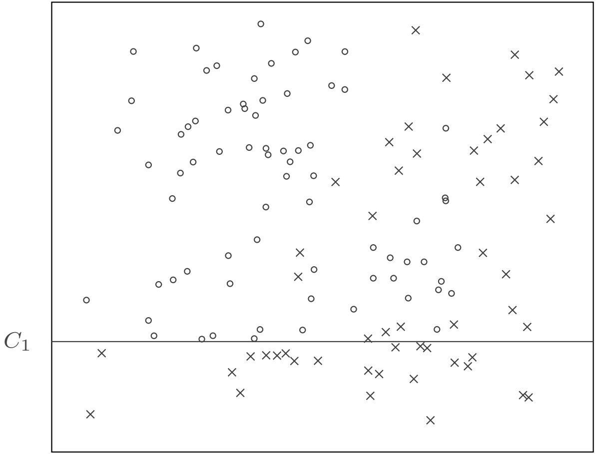
Use model f to predict target y of new data

Outline

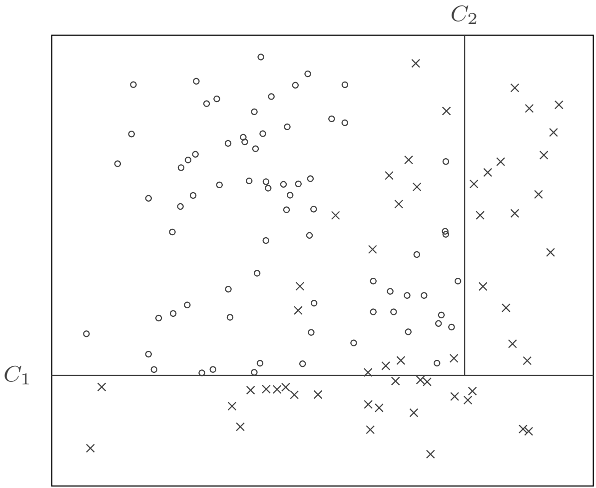
Decision Trees



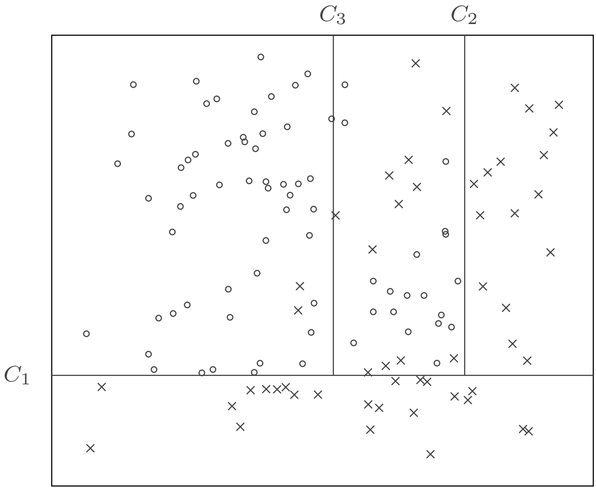
Decision Trees



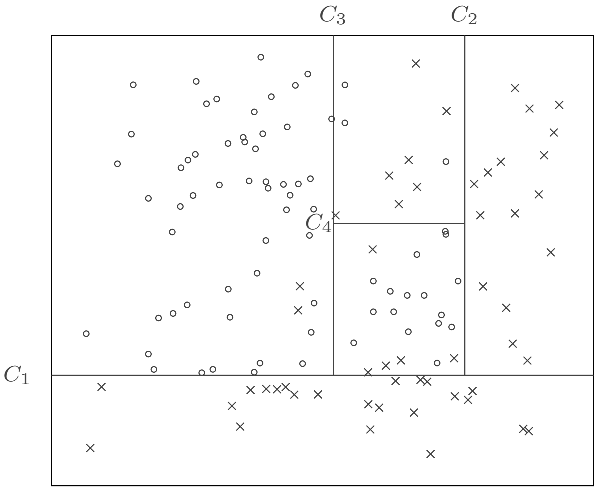
Decision Trees



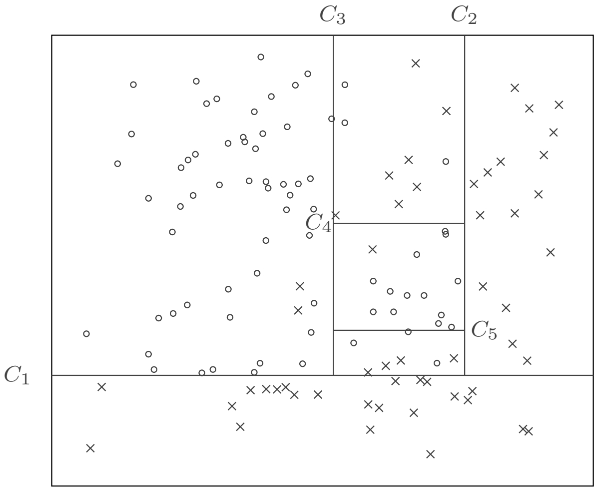
Decision Trees



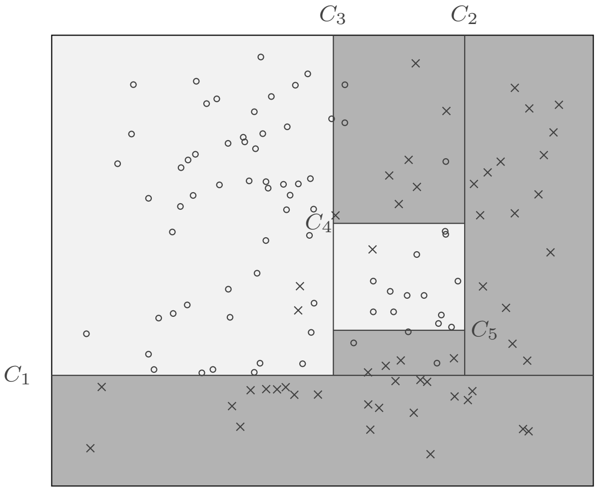
Decision Trees



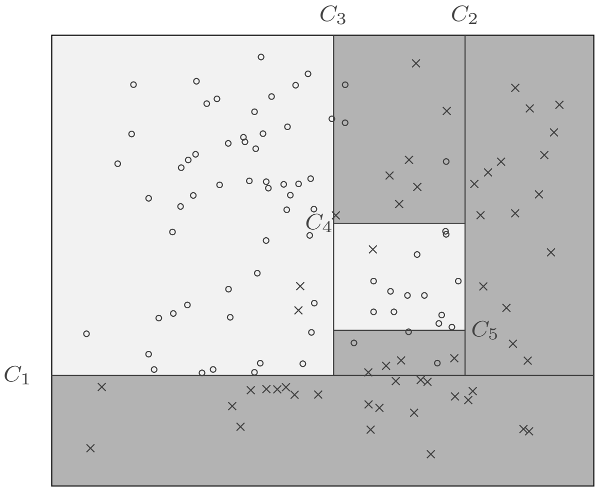
Decision Trees

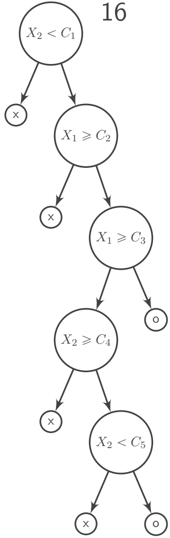


Decision Trees

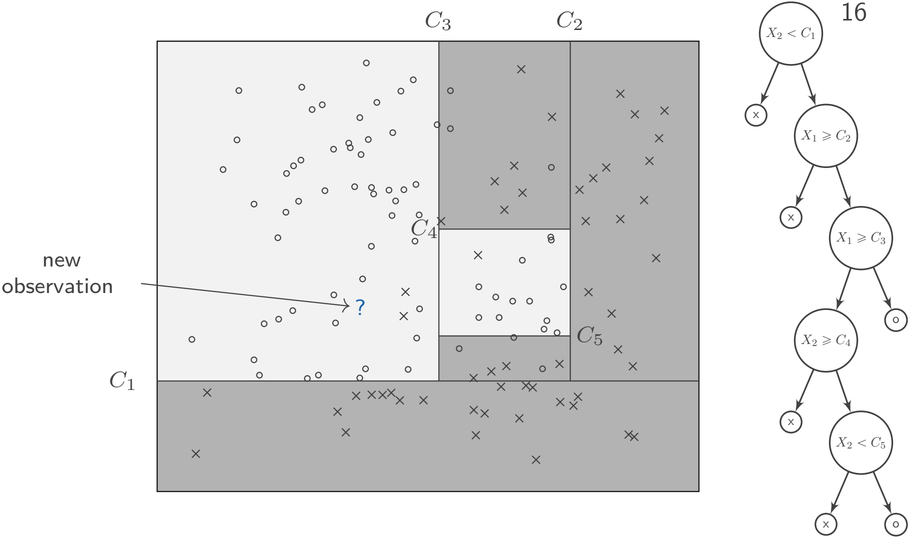


Decision Trees

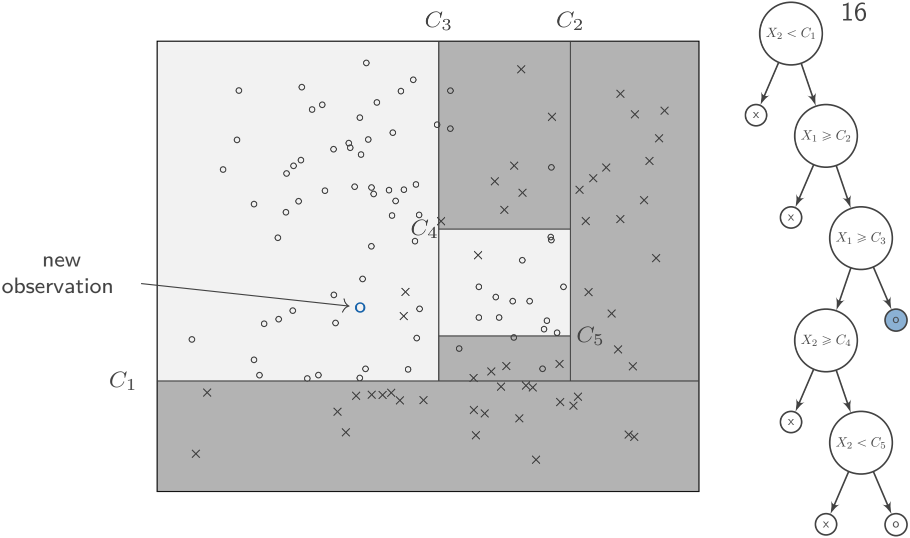




Decision Trees



Decision Trees



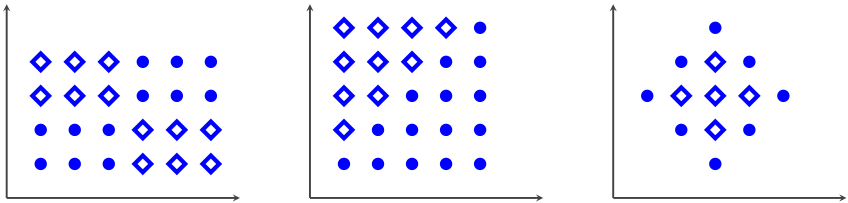
Decision Trees

Advantages of decision trees

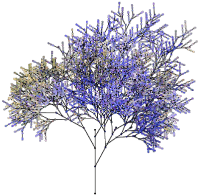
- Procedure intuitive
- Small trees simple to interpret
- Intrinsic variable selection
- Simple handling of outliers
- Fast training
- Usually better prediction performance than kNN

Disadvantages of decision trees

- Trees unstable
- Pruning can be computationally intensive
- Usually worse prediction performance than random forests (covered later) and boosted trees
- Problematic data sets

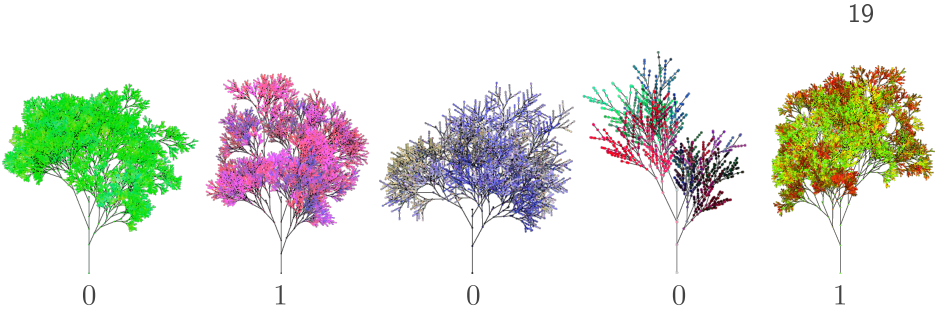


Random Forests

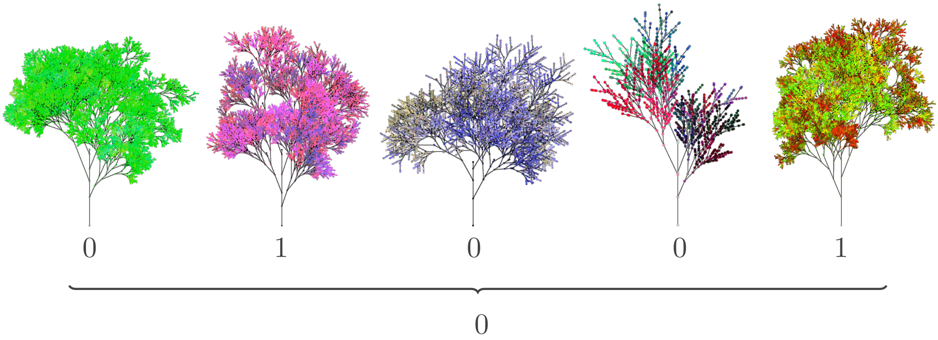


0

Random Forests



Random Forests

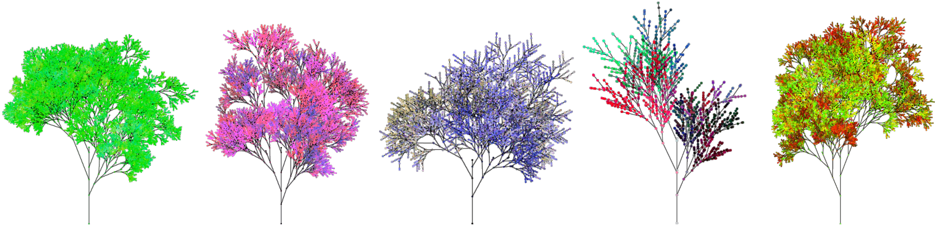


Classification: **majority vote** over all trees

Breiman 2001 Mach Learn 45:5-32 • Malley et al. 2012 Methods Inf Med 51:74-81 • luc.devroye.org

Random Forests

19



0 1 0 0 1

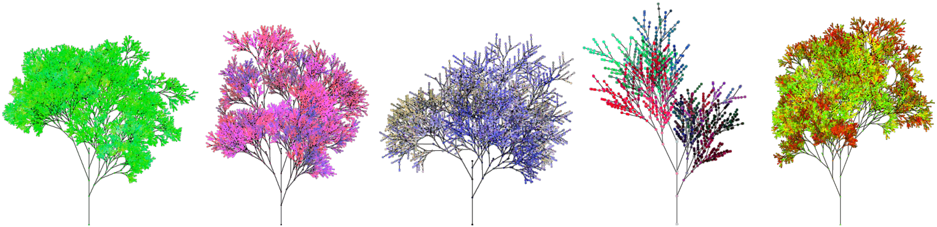
$$\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}_{0.4 < 0.5 \Rightarrow 0}$$

Classification: **majority vote** over all trees
Identical to average over all trees, cut point 0.5

Breiman 2001 Mach Learn 45:5-32 • Malley et al. 2012 Methods Inf Med 51:74-81 • luc.devroye.org

Random Forests

19



8.7% 52.6% 21.3% 3.1% 69.6%

$$\underbrace{8.7\% \quad 52.6\% \quad 21.3\% \quad 3.1\% \quad 69.6\%}_{31.1\%}$$

Probability estimation: Average over all trees

Random Forests

Two components of randomization

- Data manipulation in rows: bootstrapping / subsampling
- Data manipulation in columns: feature subsampling

Breiman 2001 Mach Learn 45:5

Random Forests

Bootstrap aggregating (bagging)

- Ensemble = committee of experts
- Single weak learner = single committee member
- Ensemble decision = committee decision

Fundamental idea of bagging (bootstrap aggregating)

Any learner can be used as *base learner*, e.g. kNN or tree
→ **Ensemble learning** (covered later)

Breiman 2001 Mach Learn 45:5-32

Bootstrapping

- Sampling **with** replacement
- Original sample size n, resampled sample size n
- On average $lim_{n\to\infty}\left(1-\frac{1}{n}\right)^n \approx 0.632 \approx 2/3$ resampled

Subsampling

- Sampling **without** replacement
- Original sample size n, resampled sample size $< n$
- Standard: resampling of $0.632n$

Breiman 2001 Mach Learn 45:5-32

Random Forests

Feature subsampling

At a node consider only subset of features

- Trees vary
- “Experts” differ in their opinion
- Reduce correlation between trees

Number of features considered at split

$\texttt{mtry} = \sqrt{d}$, $\ln d$ or $d/3 \rightarrow$ Tuning possible (later)

Random forest algorithm

For each tree

1. Draw bootstrap sample with replacement
2. Grow tree
   a) Use random subset of variables (`mtry`) at each node
   b) Stop if minimum node size reached
3. Determine proportion of ‘1’ in each terminal node

New subject

1. Drop down subject in each single tree
2. Store proportion from all trees
3. Average proportion of ‘1’s over all trees

Kruppa et al. 2013 Expert Sys Appl 40:5125-31

Advantages of random forests

- As with trees: Procedure intuitive, intrinsic variable selection, simple handling of outliers, fast training
- Work well with high dimensional data
- Work well without (or with only a little) tuning
- Usually better prediction performance than a single tree

Disadvantages of random forests

- Not simple to interpret
- Sometimes worse prediction performance than well tuned boosted trees
- Bad prediction performance on image, text and speech data

Outline

Model Evaluation

How goood is a prediction model?

Compare true target y with predicted target $\hat{y}$

Examples

- How many patients correctly diagnosed?
- How many emails correctly detected as ham or spam?
- How close is the predicted price of a house to the true value?
- How close is the length of hospitalization to the true value?

Model Evaluation

Dichotomous (binary) outcome

- Proportion of correct classifications (PC); also accuracy: $\widehat{PC} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{1}_{y_i = \hat{y}_i}$
- Sensitivity, specificity, ROC, AUC: $\hat{\mathbb{P}}(y = 1 \mid x)$
- Brier score (BS), i.e., MSE of probability estimates; also probability score (PS): $\widehat{BS} = \frac{1}{n}\sum_{i=1}^{n} \left(y_i - \hat{\mathbb{P}}\left(y_i = 1 \mid x_i\right)\right)^2$

Multicategory outcome

- Proportion of correct classifications (PC)
- Averaged class-wise PC
- ROC, AUC: several extensions

Model Evaluation

Continuous outcome

- MSE: $\widehat{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
- MAE: $\widehat{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$
- RMSE: $\widehat{RMSE} = \sqrt{\widehat{MSE}}$
- Explained variance: $\hat{R}^2 = \frac{1-\widehat{MSE}}{\widehat{\mathbb{V}ar}(y)}$

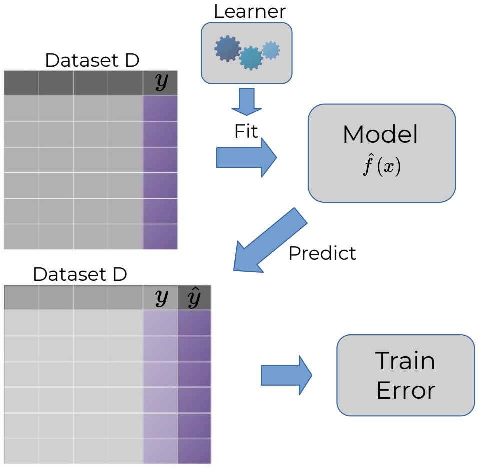
Survival outcome

- Time-dependent Brier Score
- Integrated Brier score
- C-Index

Model Evaluation

Training error

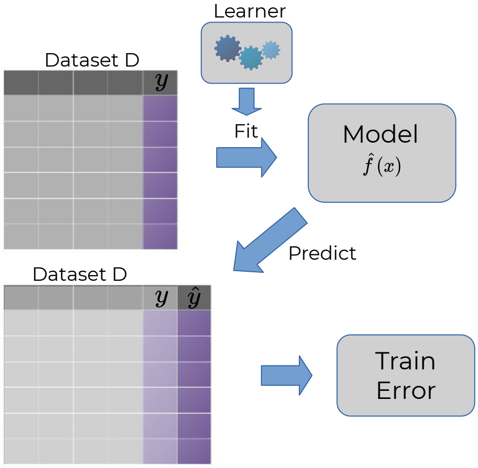
Evaluate performance on training data



Model Evaluation

Training error

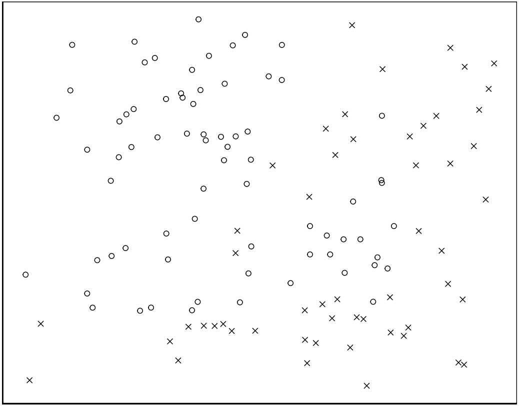
Evaluate performance on training data



Problem: Overfitting

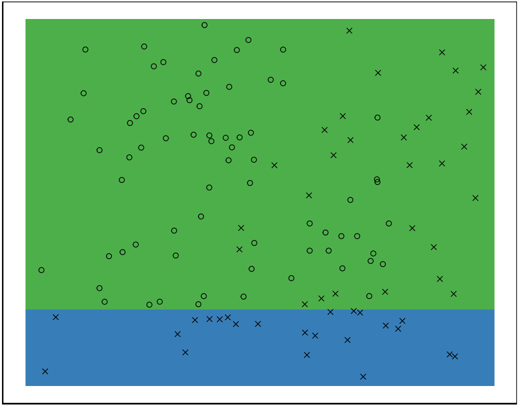
Model Evaluation

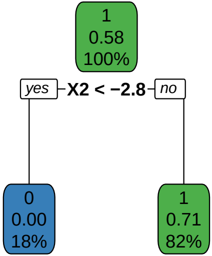
Overfitting



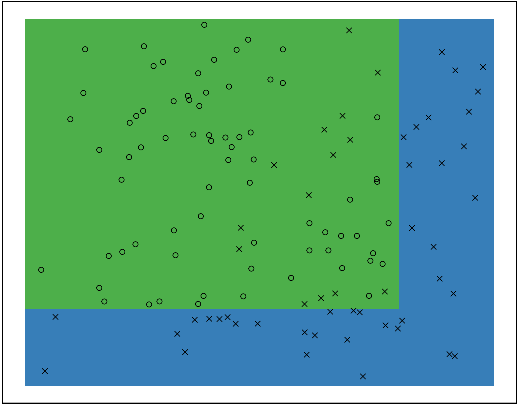
Model Evaluation

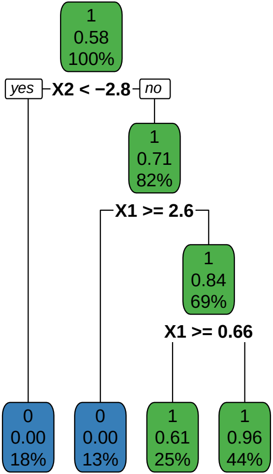
Overfitting





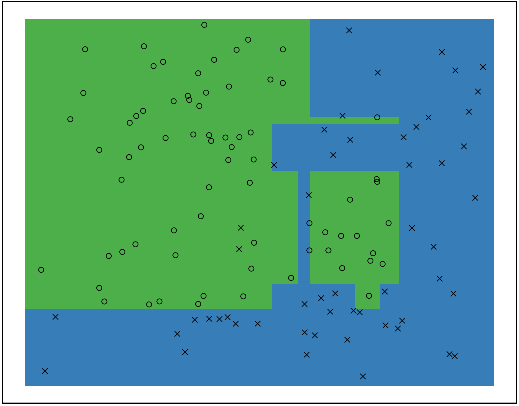
Overfitting

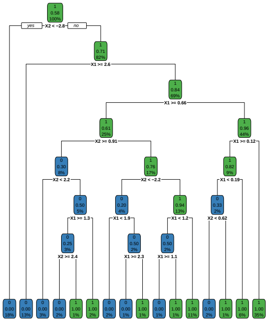




Model Evaluation

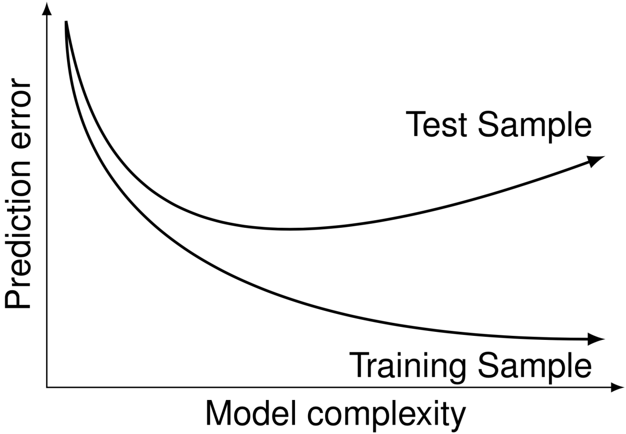
Overfitting





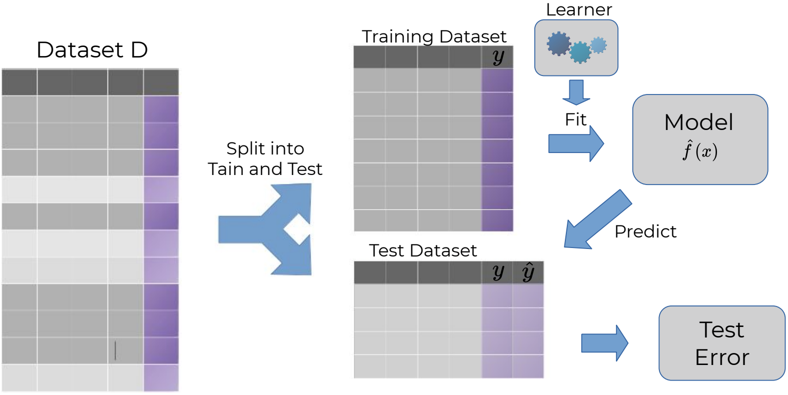
Model Evaluation

Overfitting



Model Evaluation

Test error



Model Evaluation

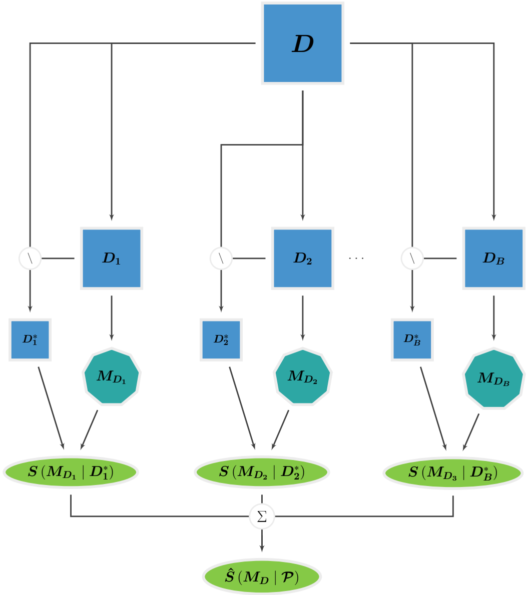
Training and test error

- Training error heavily biased
- Test error (almost) unbiased but variance unknown

Resampling

- Repeated training/test splits (subsampling)
- Cross validation
- Repeated cross validation
- Bootstrap

Resampling


D
$\setminus$
D_1
D_2
D_B
$\cdots$
D_1^*
D_2^*
D_B^*
M_{D_1}
M_{D_2}
M_{D_B}
$S\left(M_{D_1} \mid D_1^*\right)$
$S\left(M_{D_2} \mid D_2^*\right)$
$S\left(M_{D_3} \mid D_B^*\right)$
Σ
$\hat{S}\left(M_D \mid \mathcal{P}\right)$

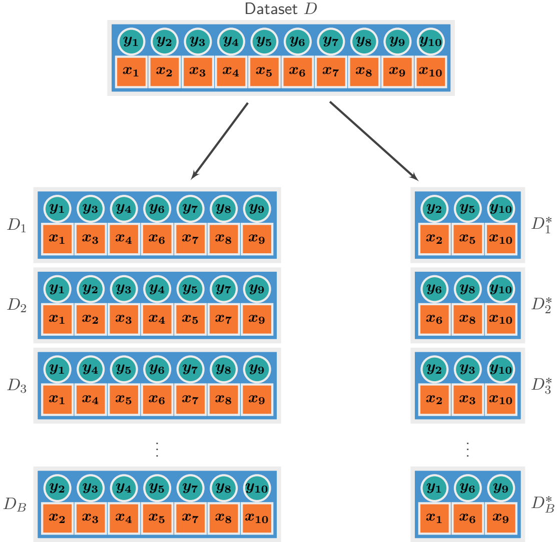
Resampling

- Estimate performance on independent data
- Used for
  - Performance estimation
  - Hyperparameter tuning
  - Model selection
- Resampling based performance estimation
  1. Split dataset in several (smaller) datasets D_b
  2. On each dataset D_b:
     2.1 Train learner
     2.2 Estimate performance on $D_b^* = D \setminus D_b$
  3. Aggregate performance estimates

Simon 2007 ISBN 9780387475080 • Bischl et al. 2012 Evol Comput 20:249–75

Resampling

Subsampling



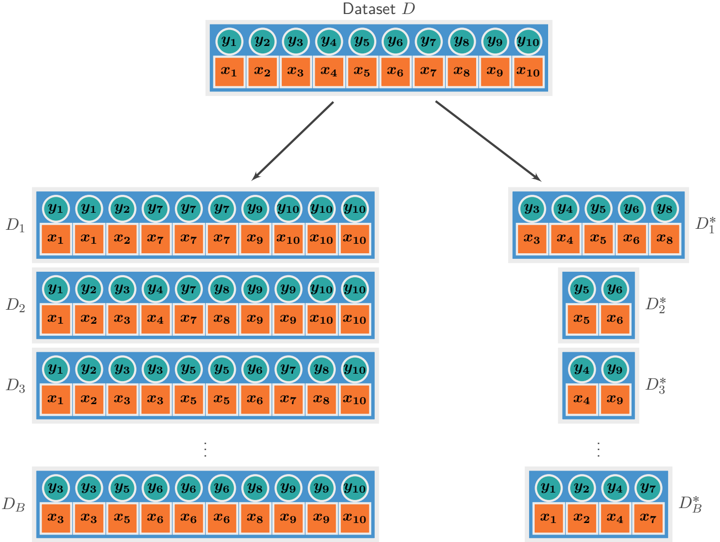
Subsampling

- Sample B training datasets D_b from D without replacement, usually $n_b = \frac{2}{3}n$
- Use $D_b^* = D \setminus D_b$ as test datasets
- D_b and D_b^* disjunct
- D_1 and D_2 not disjunct
- D_1^* and D_2^* not disjunct
- Performance estimator biased
- No optimal B, usually $100 < B < 1000$
- Special case with $B = 1$: Single train/test split (holdout)

Molinaro et al. 2005 Bioinformatics 21:3301–07 • Bischl et al. 2012 Evol Comput 20:249–75

Resampling

Bootstrapping

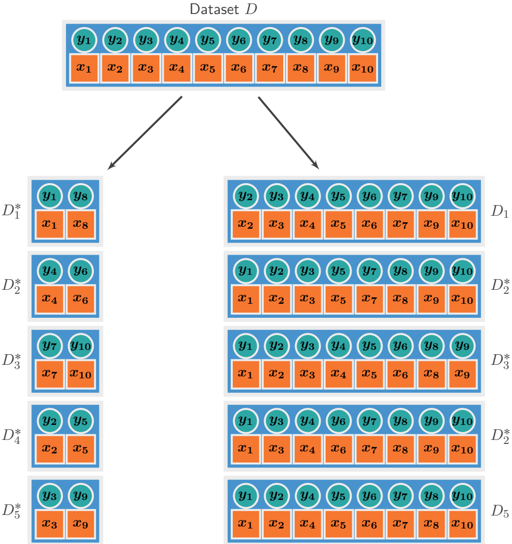


Bootstrapping

- Sample B training datasets D_b from D with replacement, usually $n_b = n$
- Use $D_b^* = D \setminus D_b$ as test datasets
- D_b and D_b^* disjunct
- D_1 and D_2 not disjunct
- D_1^* and D_2^* not disjunct
- Performance estimator biased
- Adaptive weighting to reduce bias (.632+ bootstrap)
- No optimal B, usually $100 < B < 1000$

Molinaro et al. 2005 Bioinformatics 21:3301–07 • Efron & Tibshirani 1997 JASA 92:548–60 • Bischl et al. 2012 Evol Comput 20:249–75

Cross validation (CV)



Cross validation (CV)

- Split D in B test datasets D_b^*
- Use $D_b = D \backslash D_b^*$ as training datasets
- D_b and D_b^* disjunct
- D_1 and D_2 not disjunct
- D_1^* and D_2^* disjunct
- Special case with $B = n$: Leave-one-out CV (LOOCV)
  $\rightarrow$ Long runtime
- No optimal B, usually $B = 5, 10$
  $\rightarrow$ Lowest B of all resampling methods $\rightarrow$ fast computation

Stone 1974 J Roy Stat Soc B Met 36:111–47 • Molinaro et al. 2005 Bioinformatics 21:3301–07
Bischl et al. 2012 Evol Comput 20:249–75

Outline

Penalized Regression

Generalized linear model

$$\begin{aligned} g(\mathbb{E}(Y)) &= \beta_0 + \beta_1 \cdot X_1 + \ldots + \beta_p \cdot X_p \\ &= X\beta \end{aligned}$$

g: Link function

Penalized Regression

Generalized linear model

$$\begin{aligned} g(\mathbb{E}(Y)) &= \beta_0 + \beta_1 \cdot X_1 + \ldots + \beta_p \cdot X_p \\ &= X\beta \end{aligned}$$

g: Link function

Linear model

$$\mathbb{E}(Y) = X\beta$$

Ordinary least squares

Minimize squared differences

$$\begin{aligned} L_{\mathsf{OLS}} &= \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \\ &= \|y - X\beta\|_2^2 \\ &= (y - X\beta)'(y - X\beta) \end{aligned}$$

Ordinary least squares

Minimize squared differences

$$\begin{aligned} L_{\mathsf{OLS}} &= \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \\ &= \|y - X\beta\|_2^2 \\ &= (y - X\beta)'(y - X\beta) \end{aligned}$$

Solution:

$$\beta_{\mathsf{OLS}} = \left(X'X\right)^{-1} X'y$$

Penalized Regression

Ridge regression

Penalize large parameter estimates (L2 regularization)

$$L_{\text{Ridge}} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{m} \beta_j^2$$
$$= \|y - X\beta\|_2^2 + \lambda\|\beta\|_2^2$$

Penalized Regression

Ridge regression

Penalize large parameter estimates (L2 regularization)

$$L_{\text{Ridge}} = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{m} \beta_j^2$$
$$= \|y - X\beta\|_2^2 + \lambda\|\beta\|_2^2$$

Solution:

$$\beta_{\text{Ridge}} = \left(X'X + \lambda I\right)^{-1} X'y$$

Penalized Regression

Ridge regression

Penalize large parameter estimates (L2 regularization)

$$
\begin{aligned}
L_{\text{Ridge}} &= \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{m} \beta_j^2 \\
&= \|y - X\beta\|_2^2 + \lambda\|\beta\|_2^2
\end{aligned}
$$

Solution:

$$
\beta_{\text{Ridge}} = \left(X'X + \lambda I\right)^{-1} X'y
$$

Shrink parameter estimates towards zero

Penalized Regression

How to find best λ?

Minimize L_{Ridge} in cross validation

$\rightarrow$ Hyperparameter tuning

Penalized Regression

LASSO: Least absolute shrinkage and selection operator

Penalize large parameter estimates (L1 regularization)

$$\begin{aligned} L_{\mathsf{LASSO}} &= \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{m}|\beta_j| \\ &= \|y - X\beta\|_2^2 + \lambda\|\beta\|_1 \end{aligned}$$

Penalized Regression

LASSO: Least absolute shrinkage and selection operator

Penalize large parameter estimates (L1 regularization)

$$
\begin{aligned}
L_{\text{LASSO}} &= \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{m} |\beta_j| \\
&= \|y - X\beta\|_2^2 + \lambda \|\beta\|_1
\end{aligned}
$$

No closed-form solution

LASSO: Least absolute shrinkage and selection operator

Penalize large parameter estimates (L1 regularization)

$$\begin{aligned} L_{\text{LASSO}} &= \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{m}|\beta_j| \\ &= \|y - X\beta\|_2^2 + \lambda\|\beta\|_1 \end{aligned}$$

No closed-form solution

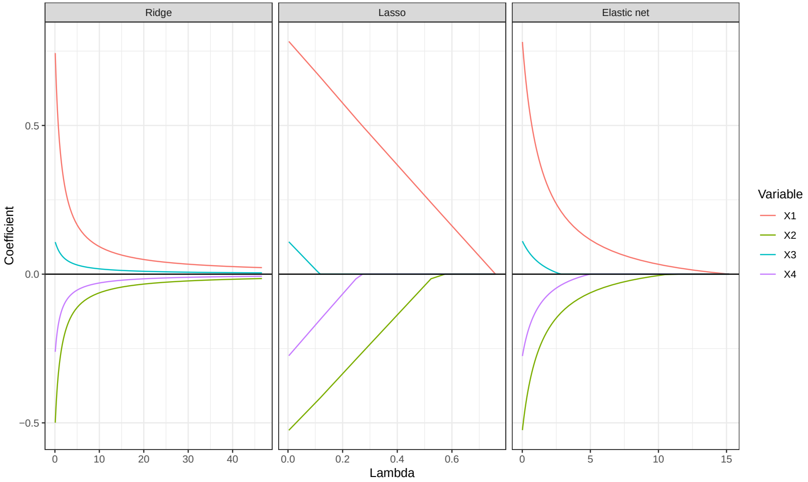
Shrink parameter estimates to (exactly) zero

Elastic net: Combination of Ridge and LASSO

L1 and L2 regularization

$$
\begin{aligned}
L_{\mathsf{Elnet}} &= \sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda_1 \sum_{j=1}^{m}|\beta_j| + \lambda_2 \sum_{j=1}^{m}\beta_j^2 \\
&= \|y - X\beta\|_2^2 + \lambda_1\|\beta\|_1 + \lambda_2\|\beta\|_2^2
\end{aligned}
$$

Penalized Regression



Penalized Regression

Advantages of penalized regression

- Reduces overfitting
- Avoid multicolinarity issues of (non-penalized) regression models
  - → Work well with high-dimensional data
- Same general concept of (non-penalized) regression models
  - → Interpretable model
- Better prediction performance than non-penalized regression (less variance)
- Implicit variable selection (LASSO)

Disadvantages of penalized regression

- Biased parameter estimates
- Cannot use statistical inference methods used in non-penalized regression
- Interactions and non-linear effects have to be explicitly specified
- Often worse prediction performance than (other) machine learning algorithms

Ensemble Learning

Averaging

Train several learners, average results

Majority voting

Train several learners, predict class with most votes
→ hard classification only

Bootstrap Aggregating

Bootstrap aggregating (bagging)

Averaging combined with bootstrapping: Train each learner on different bootstrap sample

Bootstrap Aggregating

Bootstrap aggregating (bagging)

Averaging combined with bootstrapping: Train each learner on different bootstrap sample

Problem

Some learners perform better than others, but all get equal weight
$\rightarrow$ Same problem with averaging and majority voting

Boosting

Boosting

Iterative procedure: Learn from previous mistakes

Gradient boosting

1. Train a model using any learner (often shallow tree)
2. Compute residuals (more general: any loss function)
3. Learn the residuals with another learner
4. Repeat 3. many times

Stacking

Combine different learning algorithms

- Base learners use different learning algorithms
- Combiner or meta-learner: Learner that uses predictions of base learners as features

Example

- Base learners: Random forest, penalized regression, neural network
- Combiner: Penalized regression

Wolpert 1992 Neural Networks 5:241

Stacking

Avoid overfitting

Combine stacking with cross validation: Use cross-validated predictions as combiner features

Stacking

Avoid overfitting

Combine stacking with cross validation: Use cross-validated predictions as combiner features

Nested cross validation

Evaluating cross-validated stacking with cross validation
→ Nested cross validation

Super Learner

Super learning = Stacking

Super Learner

Super learning = Stacking

Theoretical guarantee

Stacked ensemble performs at least as well as best base learner

Van der Laan et al. 2007 Stat Appl Genet Mol Biol 6:25

Automated Machine Learning

AutoML: Automated machine learning

Automate the whole machine learning pipeline

Outline

Hyperparameter Tuning

Hyperparameters

Learners have hyperparameters, e.g.:

- Number of nearest neighbors k
- Depth of a tree
- Number of features to consider in each split of a random forest (mtry)
- Architecture of neural network

Most learners have several hyperparameters

Have to be jointly optimized

Hyperparameter Tuning

Search entire parameter space

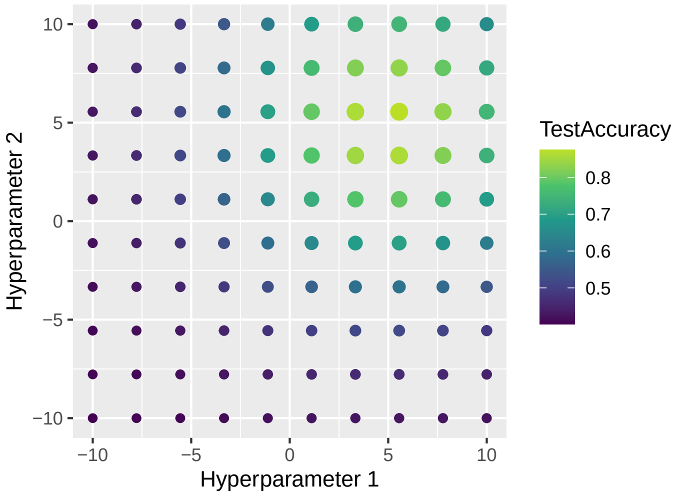
- All possible combinations
- Grid search
- Randomly select combinations
- Model-based optimization

Use resampling

- Evaluate each parameter combination on all resampling iterations/folds
- Choose parameter maximizing aggregated performance measure

Hyperparameter Tuning

Grid search



Hyperparameter Tuning

Grid search

Advantages

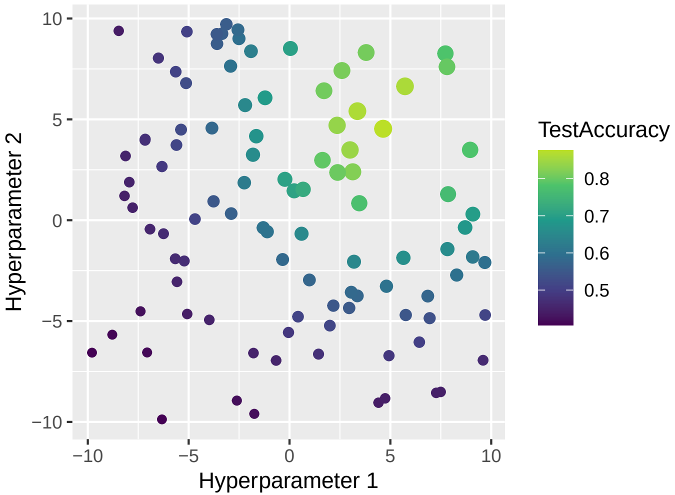
- Easy to implement
- All parameter types possible
- Easily parallelized

Disadvantages

- Computationally intensive
- Inefficient: Searches large irrelevant areas
- Arbitrary: Which values / discretization?

Hyperparameter Tuning

Random search



Hyperparameter Tuning

Random search

Advantages

- Same as grid search: Easy to implement, all parameter types possible, trivial parallelization
- Easy to adjust to computational budget
- No discretization
- Superior performance compared to grid search

Disadvantages

- Computationally intensive
- Inefficient: Searches large irrelevant areas

Hyperparameter Tuning

Model-based optimization

Surrogate model

Learn relationship between hyperparameters and prediction performance

Algorithm

1. Pick initial configuration (e.g. random)
2. Learn surrogate model
3. Predict new configuration with surrogate model
4. Repeat steps 2 and 3

Hyperparameter Tuning

Model-based optimization

Advantages

- All parameter types possible
- Efficient: Focus on promising areas
- Superior performance compared to grid and random search

Disadvantages

- Computationally intensive
- Non-trivial parallelization
- Harder to implement

Turner et al. 2020 NeurIPS PMLR 133:3

Benchmarking

How can performance be compared?

Be fair!

- Compare all learners and models on same data
- Tune parameters of all learners
- Don't overfit
- Don't publish over-optimistic results

Never learn, tune or evaluate on same data!

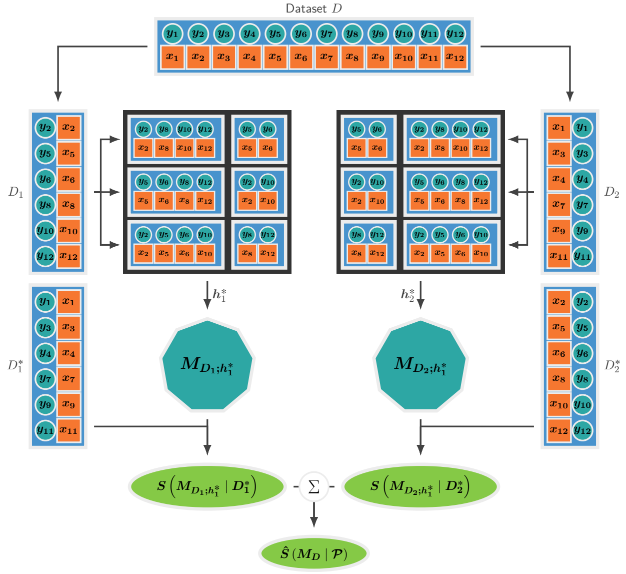
Hyperparameter tuning

- Optimize (tune) the hyperparameters
- Do not tune and evaluate on same data

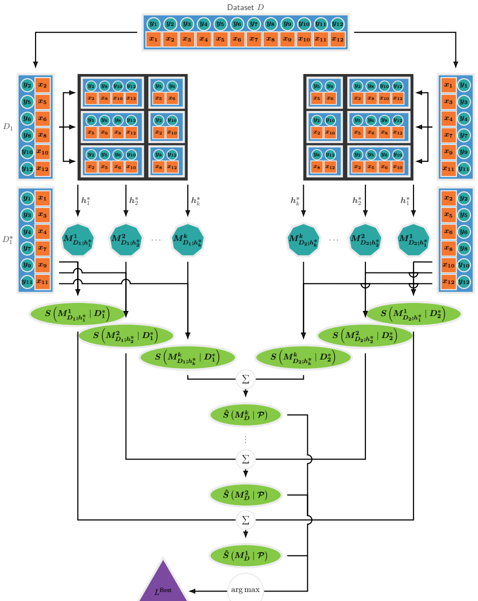
→ 3-fold split into training, validation, test

→ Nested resampling

Nested Resampling



Model Selection



How to build a final model?

1. Select best learner with nested resampling
2. Find optimal hyperparameters of best learner with resampling
3. Train best learner with optimal hyperparameters on full data

Discussion

Is there a single best learner?

No!

Learner recommendations

- Typically RF $\approx$ Boosting $>$ Tree $>$ kNN
- RF robust, easy to tune and fast
- Boosting often slightly better than RF on tabular data (when properly tuned)
- Support vector machine (SVM) good alternative for binary classification with numerical features (when properly tuned)
- Image, text and speech data $\rightarrow$ Deep Learning
- Consider ensembles, e.g. stacking / Super learner