DEUTSCHE BUNDESBANK EUROSYSTEM

III Course on Machine Learning and Central Banking

CEMLA and Deutsche Bundesbank – September 2022

Prof. Gabriela Alves Werb, Ph.D., Sebastian Seltmann

Agenda

- Structure and Organization of the Course
- Train, Test and Validation Samples
- Cross-Validation
- Break
- Confusion Matrix
- Model Validation Measures
- PR Curve
- Hands-On Exercises

Structure and Organization of the Course

Structure and Organization of the Course

- **Day 1: September, 12**
  - Introduction
  - Break
  - Introduction II (Confusion Matrix, Evaluation Measures, PR Curve)
- **Day 2: September, 13**
  - Q&A
  - Tree-Based Methods (CART and Conditional Inference Trees)
  - Break
  - Introduction to Ensemble Methods (Bagging and Boosting)
- **Day 3: September, 14**
  - Q&A
  - Ensemble Methods I (Random Forest and Causal Random Forests)
  - Break
  - Ensemble Methods II (Gradient Boosting)

Structure and Organization of the Course

- **Day 4: September, 15**
  - Q&A
  - Support Vector Machines
  - Break
  - Machine Learning Use Cases in Central Banking
  - Wrap Up

- Each module is followed by a hands-on exercise.
- Let‘s keep it interactive: please ask questions at any time!

Main Idea: Toolbox

Train, Validation, and Test Subsamples
Splitting the Data

Train, Validation, and Test Subsamples
Splitting the Data

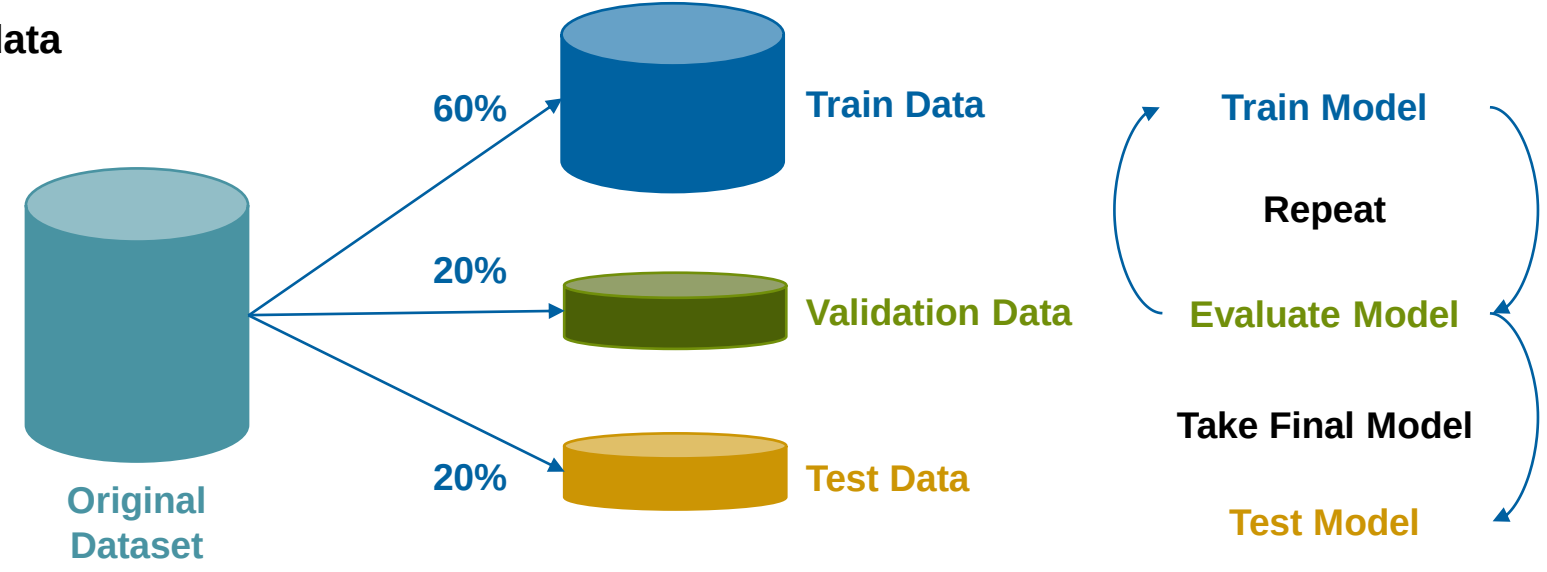
- Golden rule in Machine Learning: evaluate the model on d**ata** that **was not used** to **train** it
- Most common: **randomly** split the dataset into **train and test** data
  - E.g., when modeling loan default, some borrowers land in the train, others land in the test data
  - Usually: **75%-80%** train, **25%-20%** test
- Better approach: **randomly** split dataset into **train**, **validation**, and **test** data
  - Usually: **60%** train, **20%** validation, **20%** test
- Even better in some settings: also have **out-of-time** data
  - **Completely separate** dataset, e.g., collected a few months later

Train, Validation, and Test Subsamples
Splitting the Data

- **Train data**
  - Data used to train the model
  - The model learns from the underlying patterns and relationships in this subsample
- **Validation data**
  - After **optimizing several models** in the train data, see how they perform in the validation data
  - Go back to **training and iterate** until you are happy with the results
  - The results in this subsample motivate future modeling decisions
  - Validation data is “contaminated” → **No unbiased estimate** of the **error** on truly new data

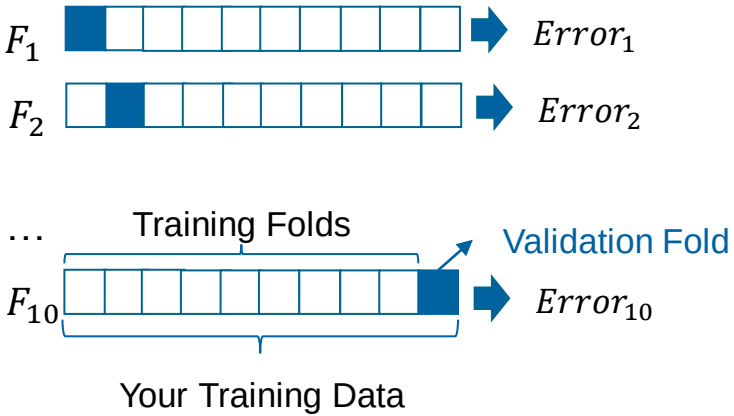
Train, Validation, and Test Subsamples
Splitting the Data

- **Test data**
  - Only to be used in the very end
  - Performance of the model in this subsample provides an **unbiased estimate** of the **error** on **new data**



Model Validation and Evaluation Measures
Cross-Validation

- Idea: use **many subsamples of the data** to estimate model error
- K subsamples = K-Fold cross-validation
- Measure error (e.g., error rate for classification or Sum of Squared Residuals for regression)
- Average the errors or sum them (e.g., “risk” measure in RPART)



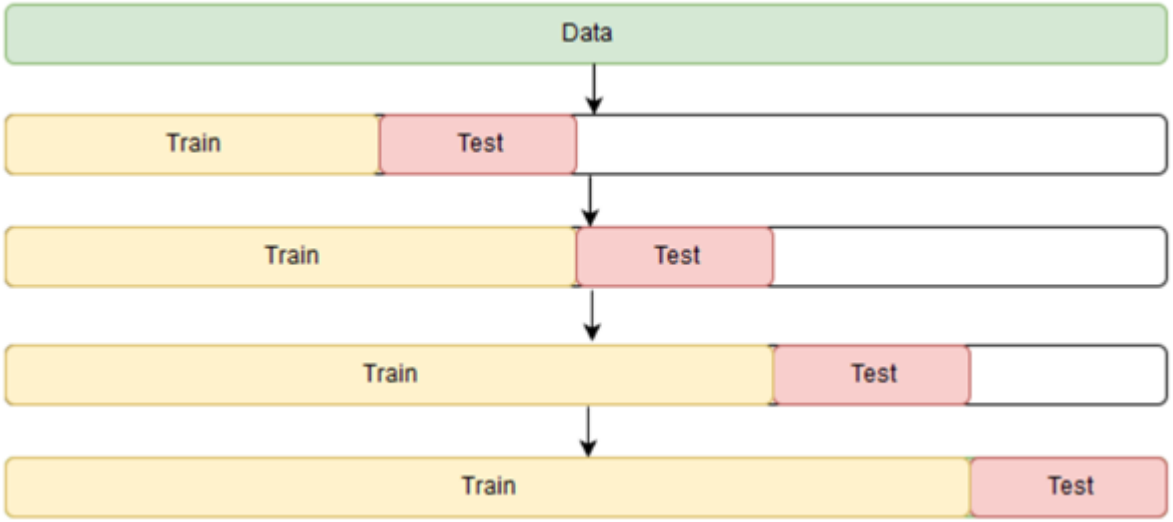
$$\widehat{Error} = \sum_{k=1}^{10} Error_k$$

or

$$\widehat{Error} = \sum_{k=1}^{10} \frac{Error_k}{10}$$

Model Validation and Evaluation Measures
Cross-Validation

- Time Series Cross Validation
  - The model must not be trained with data “from the future”



Model Validation and Evaluation Measures
Cross-Validation

- But, beware!
  - If you **preselect variables** that flow in the model, then **cross-validation** must be also applied at the **variable selection step** (not only later)!
  - **Otherwise**, cross-validation does **not accurately** estimate the **prediction error**

Model Validation and Evaluation Measures

Model Validation and Evaluation Measures
Confusion Matrix for Binary Classification

Confusion Matrix

		Actual Data	
		No Default	Default
Predictions	No Default	TN	FN
	Default	FP	TP

- Example: Loan Default Prediction
- TN: true negatives / TP: true positives
- FN: false negatives / FP: false positives
- Based on them, we can **compute several other metrics**

Model Validation and Evaluation Measures
Accuracy, Recall

- What is the share of **correctly predicted** cases?
  - Accuracy = (TN + TP) / Total
  - Total = TP+TN+FP+FN

Confusion Matrix

		Actual Data	
		No Default	Default
Predictions	No Default	TN	FN
	Default	FP	TP

- Which share of the **true default** cases is correctly predicted?
  - Sensitivity (or Recall) = TP / (TP + FN)

Confusion Matrix

		Actual Data	
		No Default	Default
Predictions	No Default	TN	FN
	Default	FP	TP

- Which share of the **true non-default** cases is correctly predicted?
  - Specificity = TN / (TN + FP)

Confusion Matrix

		Actual Data	
		No Default	Default
Predictions	No Default	TN	FN
	Default	FP	TP

Model Validation and Evaluation Measures
Precision, F1-Score

- What is the share of **correct "default" predictions**?
  - Precision = TP / (TP + FP)

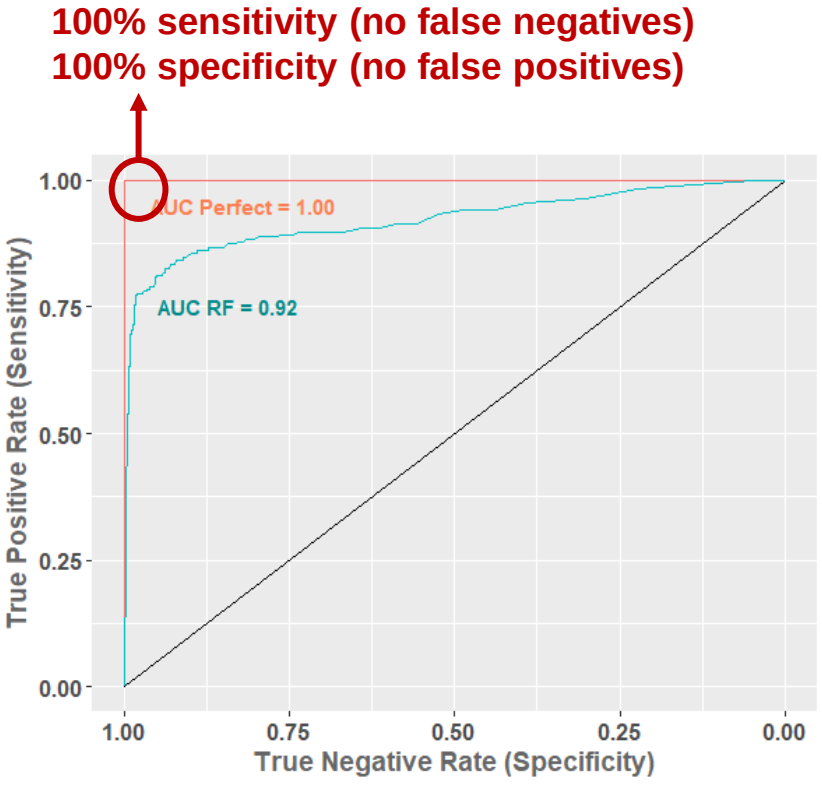
Confusion Matrix

		Actual Data	
		No Default	Default
Predictions	No Default	TN	FN
	Default	FP	TP

- Can the model identify **true "default"** cases without many **false alarms**?
  - F1-Score = $2 * \frac{Precision * Recall}{Precision + Recall}$
  - F1-Score provides a tradeoff between precision and recall

Model Validation and Evaluation Measures
ROC Curve

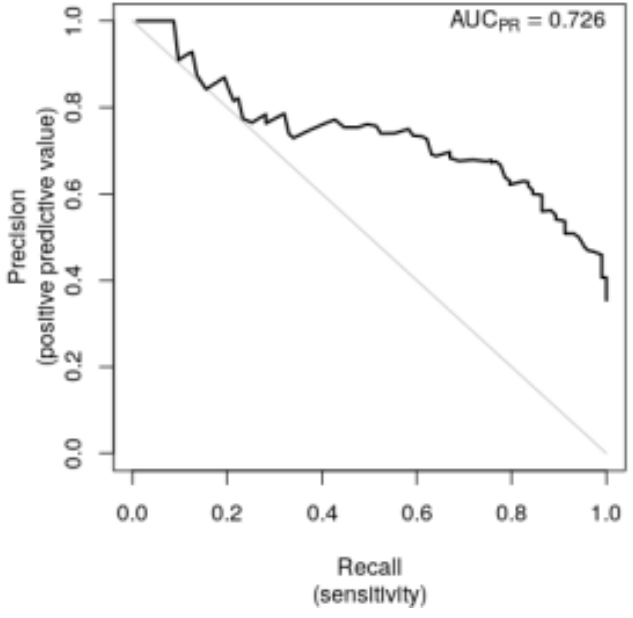
- Receiver Operating Curve (ROC)
  - Plots the **tradeoff** between **Sensitivity and Specificity** for different probability **thresholds**
  - **Decrease threshold**: increase **TP**, but also FP
  - **Increase threshold**: increase **TN**, but also FN
- **AUC** (Area under the Receiver Operating Curve)
  - **Probability** that a randomly chosen **positive** case (e.g., “default”) receives from the model a **higher score** (predicted probability) than a randomly chosen **negative** case (e.g., “non-default”)



Model Validation and Evaluation Measures
Precision-Recall (PR) Curve

- Precision-Recall (PR) Curve
  - Plots the tradeoff between **Sensitivity (Recall)** and **Precision** for different probability **thresholds**

- Comparison to ROC curve:
  - **ROC** curves are appropriate **for balanced datasets**
  - **PR** curves are also appropriate for **imbalanced datasets**



Model Validation and Evaluation Measures
Example: Load Default Prediction

- Dependent variable: default (yes / no)
- 30 independent variables:
  - Loan characteristics (e.g., funded amount, term, loan amount)
  - Borrower characteristics (e.g., employment length, annual income, home ownership)

	Original Data		Train Data		Validation Data		Test Data	
	# Obs.	% of Dataset	# Obs.	% of Dataset	# Obs.	% of Dataset	# Obs.	% of Dataset
Default	100,209	20.3%	68,773	20.6%	10,417	19.8%	10,417	19.1%
No Default	393,818	**79.7%**	264,528	**79.4%**	85,076	**80.2%**	44,214	**80.9%**
Total	494,027	100%	333,301	100%	106,095	100%	54,631	100%
Share of Original Data		100%		67.5%		21.5%		11%

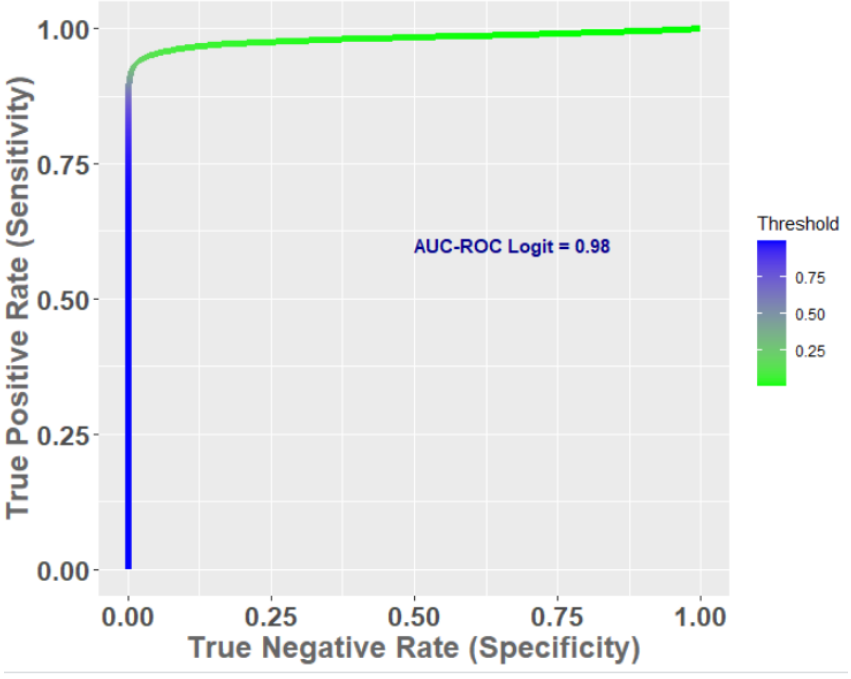
This is the “no information rate” in the data

Model Validation and Evaluation Measures
Example: Load Default Prediction

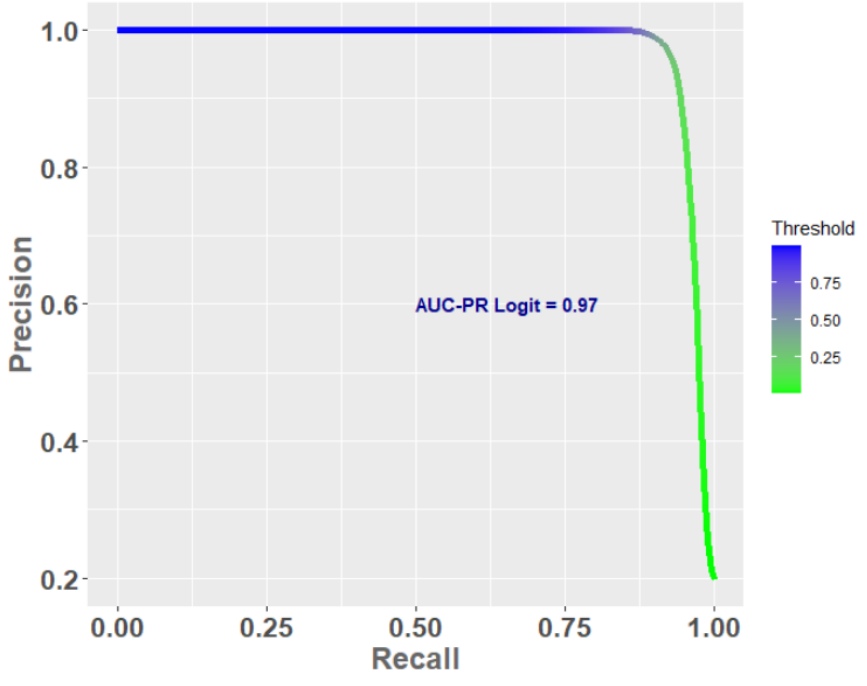
- No information rate: **accuracy** if we simply **predict** that **all observations** belong to the **most frequent class** in the data
- Example here: predict “no-default” to all observations
- This rate an important benchmark to assess the accuracy of our models

Model Validation and Evaluation Measures
Example: Load Default Prediction

AUC Curve



PR Curve



Hands-on Exercises

References

Davis, J., & Goadrich, M. (2006). The Relationship Between Precision-Recall and ROC Curves. In *Proceedings of the 23rd international conference on Machine learning* (pp. 233-240).

Dietterich, T. G. (1998), Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms, Neural Computation, 10(7), 1895-1923.

Raschka, S. (2018), Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning, Working Paper.

Day 2
Tree-Based Methods & Introduction to Ensembles

Agenda

- Introduction
- Decision Trees (CART)
- Bootstrapping
- Break
- Introduction to Ensemble Methods
- Hands-on Exercise

Decision Trees (CART)

Decision Trees (CART)
Are They Really a New Method?

- Decision trees are essentially not a brand new idea
- First algorithms were developed decades ago
  - AID (Morgan & Sonquist, **1963**)
  - CHAID (Kass, 1980)
  - CART (Breiman et al., 1984)
  - ID3 (Precursor of C5.0) (Quinlan, 1986)
- But now they are more widely known and used (why?)

Decision Trees (CART)
What Are Decision Trees?

- Belong to “divide and conquer” algorithms (greedy algorithms)
  - Divide the “big” problem into smaller “subproblems”
  - Recursively solve the “subproblems” and combine the solutions
- Recursive partitioning of the training data in smaller subsets until:
  - All subsets **predominantly** belong to **one value of Y** (**homogeneous** nodes), or
  - A **pre-specified parameter** (stop criterion) is achieved – e.g. minimum number of observations in a node
- Resulting tree represents a **set of rules** – “**if this, than that**”

Decision Trees (CART)
What Are Decision Trees?

- **Classification Trees – Our focus today**
  - **Y** is a **categorical** variable, e.g. represents **categories** or **binary** choices
  - E.g. transportation mode (car, bike, subway), **default** (y/n), **buy** (y/n)
- Regression Trees
  - **Y** is a **numerical** variable, e.g. represents **discrete** or **continuous** quantities
  - E.g., **number of products** purchased, **duration** of customer relationship

Decision Trees (CART)
Example: Will a Borrower Default?

- Example: Will a borrower default?
- Classification tree: predict default (yes/no)
- Method: CART (RPART in R)

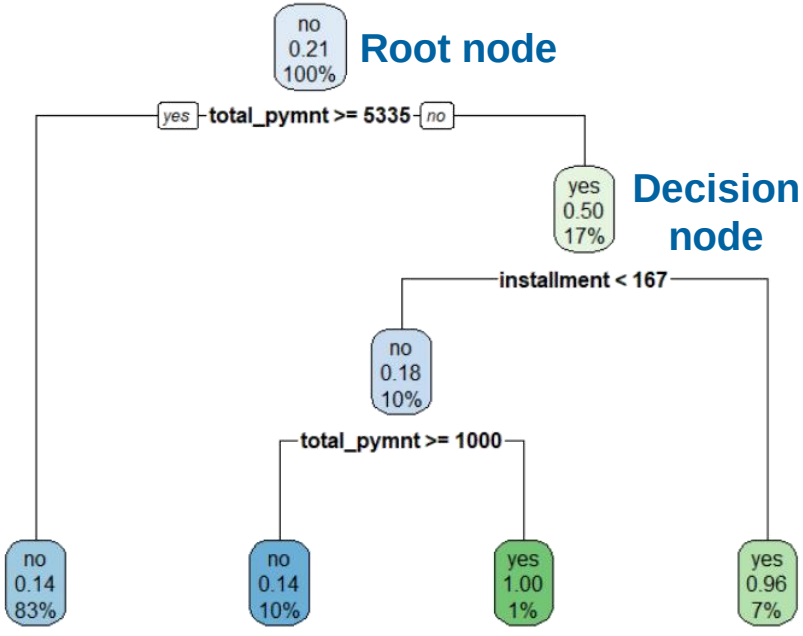
Root node: beginning of tree

Decision nodes: node in which a choice is made – lead either to outcome or to another decision node

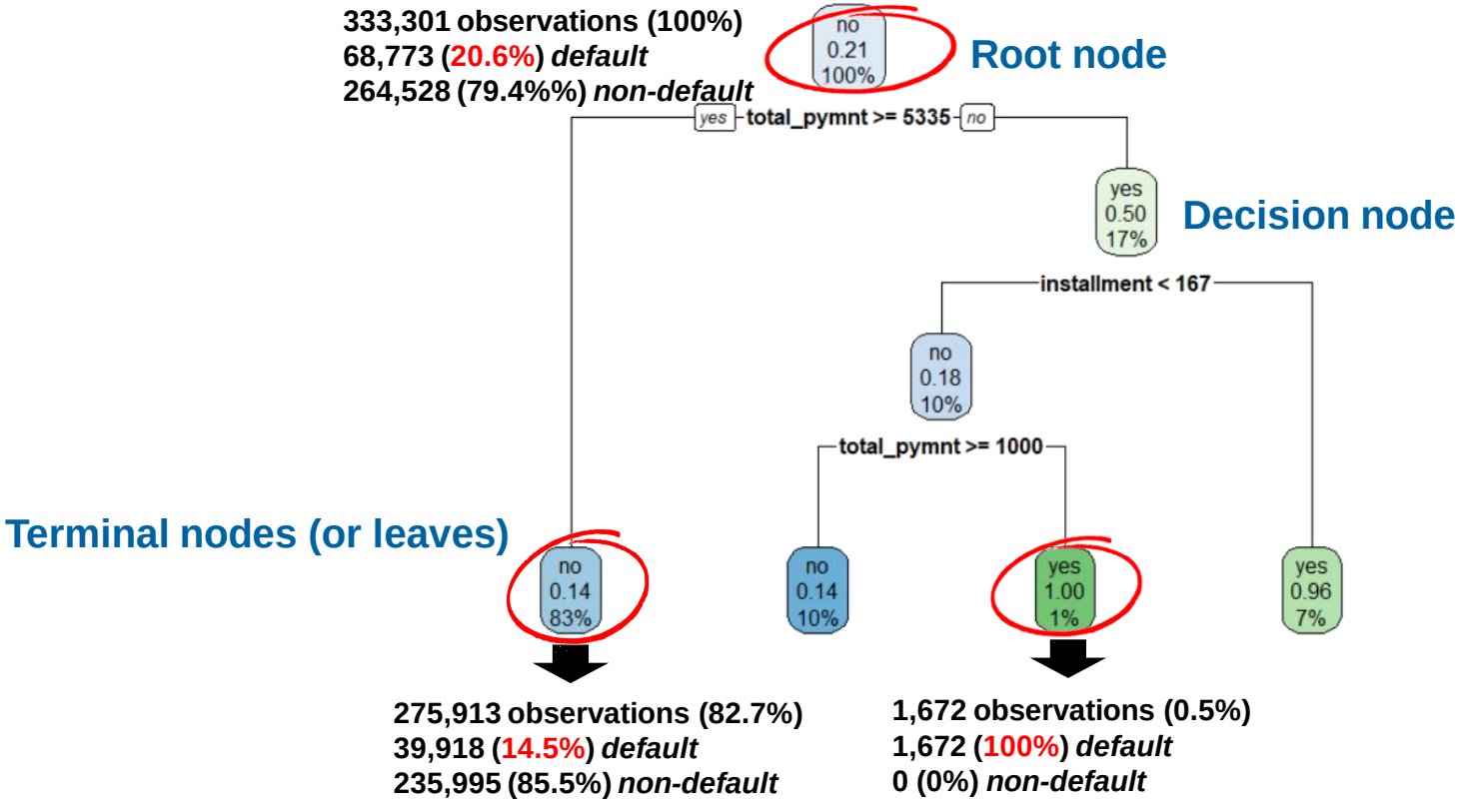
Leaf/terminal nodes: outcome

Question formats

- $x \geq a$ (or $x < a$)
- $x = a$
- $x \in A$, where A are partitions of the values x takes in the training data



Decision Trees (CART)
Example: Will a Borrower Default?



Decision Trees (CART)
How to Build a Decision Tree?

Fundamental steps to build a decision tree:

1.Which **split criterion** to choose?

2.For **every decision node**:

- on which **independent variable** to split?
- on which **value to split** (i.e. which value is the cut-off value)?

3. **Depth** of tree (i.e. number of decision nodes)?

4.What **value to predict** at each leaf node?

Decision Trees (CART)
1. Which Split Criterion to Choose?

- Impurity using the Gini Index: **CART** and its implementation in R, **RPART**

- $i(t) = 1 - \sum_{j=1}^{k} \left(p(j|t)\right)^2$

$\boldsymbol{p(j|t)}$: relative frequency of category *j* in node *t*
k: Number of categories in the sample

- **Minimum value**: all observations belong to **one category** (**"pure"** node)

- **Maximum value**: observations are **equally distributed** across all categories (**maximal dispersion**)

- Other criteria (we won't discuss these in details)
  - Significance tests: χ^2 (CHAID), Permutation tests (CTREE)
  - Entropy (C5.0)
  - Sum of Squares (for regression)

Decision Trees (CART)
2. On Which Value to Split?

- **Goal:** generate **child nodes** that are **"purer"** than their **parents**
- **Split** finds the **independent variable and its cut-off** value that **maximizes** the **decrease in node impurity**
- Decrease in node impurity from split s in node t (Gini Index)
  - $\Delta i(s,t) = i(t) - \sum_{n=1}^{N} p_{t_n} i(t_n)$ for multiway splits (N child nodes)
  - $\Delta i(s,t) = i(t) - p_{t_l} i(t_l) - p_{t_r} i(t_r)$ for binary splits (t_l left node, t_r right node)
  - **Split minimizes** the **second term** of the equation

p_{t_n}: Proportion of observations in node t assigned to child node t_n. $\sum_{n=1}^{N} p_{t_n} = 1$.

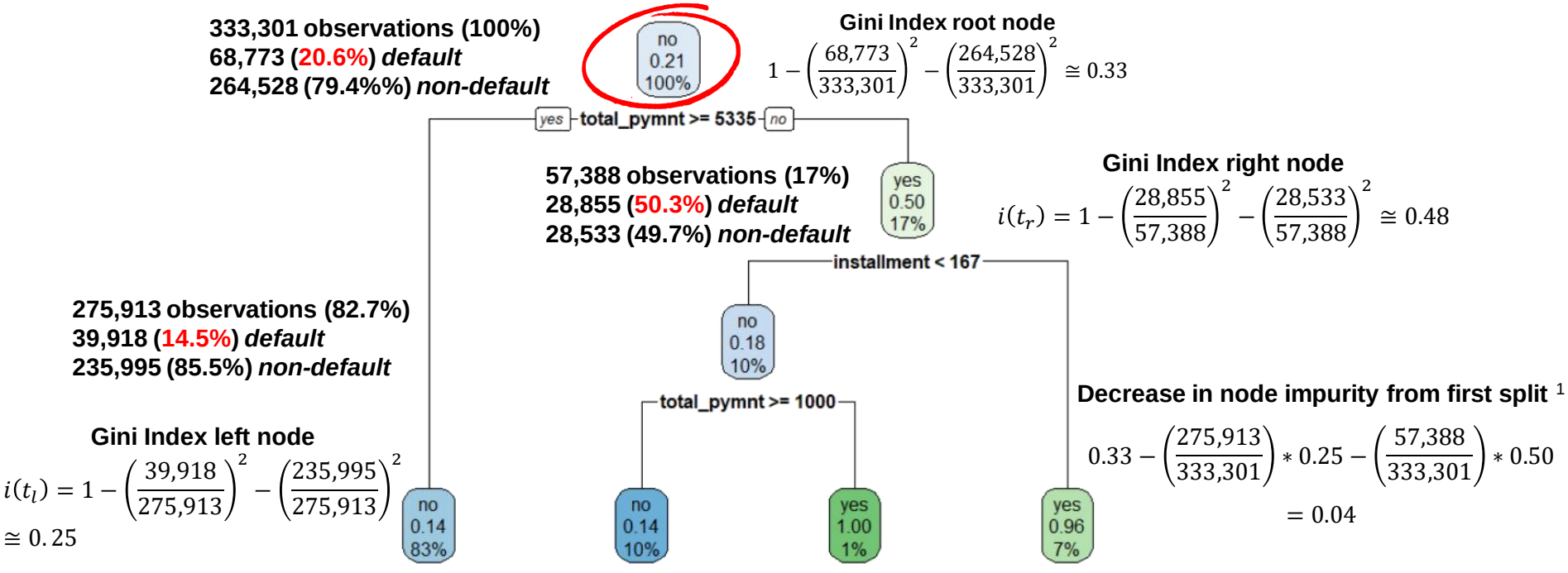
Decision Trees (CART)
2. On Which Value to Split - Example Loan Default Prediction

- Loan default prediction with RPART: Decrease in node impurity from the first split

333,301 observations (100%)
68,773 (20.6%) *default*
264,528 (79.4%%) *non-default*

Gini Index root node

$$1 - \left(\frac{68{,}773}{333{,}301}\right)^2 - \left(\frac{264{,}528}{333{,}301}\right)^2 \cong 0.33$$



57,388 observations (17%)
28,855 (50.3%) *default*
28,533 (49.7%) *non-default*

Gini Index right node

$$i(t_r) = 1 - \left(\frac{28{,}855}{57{,}388}\right)^2 - \left(\frac{28{,}533}{57{,}388}\right)^2 \cong 0.48$$

275,913 observations (82.7%)
39,918 (14.5%) *default*
235,995 (85.5%) *non-default*

Gini Index left node

$$i(t_l) = 1 - \left(\frac{39{,}918}{275{,}913}\right)^2 - \left(\frac{235{,}995}{275{,}913}\right)^2 \cong 0.25$$

Decrease in node impurity from first split [1]

$$0.33 - \left(\frac{275{,}913}{333{,}301}\right) * 0.25 - \left(\frac{57{,}388}{333{,}301}\right) * 0.50 = 0.04$$

[1] The "improve" reported under *summary()* in R shows the decrease in node impurity multiplied by the number of obs. in the parent node

Decision Trees (CART)
3. How Deep Should the Tree be?

- Without a stop criterion (e.g., minimum number of observations in a node)
  - Tree can potentially grow until **all observations** either have
    - The same value for the dependent variable (**perfect prediction**)
    - The same value for the independent variables
  - Extreme case: each observation has ist own leaf node → **overfitting!**
  - **Predictive power** is high on traning data, but **poor on new data** – model is **not generalizable**
  - **Solution: pruning** (removing branches from the tree)

Decision Trees (CART)
3. How Deep Should the Tree be?

Pre-pruning: stop tree growth **during the tree building process**

- Pre-define thresholds either in
  - Split criterion (e.g., minimum improvement in split criterion)
  - Tree characteristics (e.g., number of leaf nodes, number of splits, minimum number of observations in leaf nodes)

Decision Trees (CART)
3. How Deep Should the Tree be?

Post-pruning: grow the tree to its maximum and then trim the nodes in a bottom-up fashion

- **Merge leaf nodes while considering prediction error (shouldn't increase much)**
  - **Cost-complexity pruning** with cross-validation: CART
  - **Pessimistic error-based pruning** with binomial confidence limit: C5.0
  - Usually possible to **weight the prediction errors differently**
    - E.g., it might be **worse** to **wrongly predict** a borrower who **ends up defaulting** than to predict that a borrower will default, but she doesn't

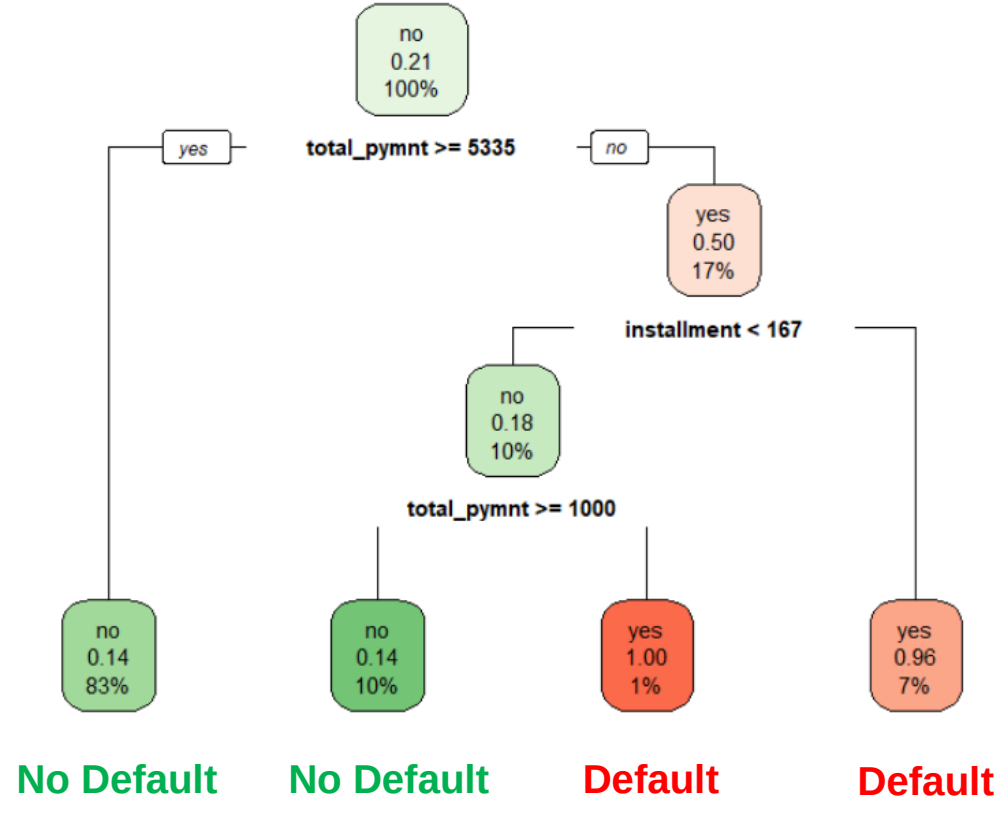
Decision Trees (CART)
4. What Value to Predict at Each Leaf Node?

Predictions for every leaf node

- **Majority voting:** category with the **most "votes"** within the leaf node is the prediction (mode of dependent variable)
- **Assumption:** Equal cost of **type I and type II errors**
  - When is this (not) a **reasonable** assumption?
- **Predicted probabilities** (relative frequency of category in the leaf node)

Source: twitter.com/freakonometrics

Decision Trees (CART)
4. What Value to Predict at Each Leaf Node - Example Loan Default Prediction

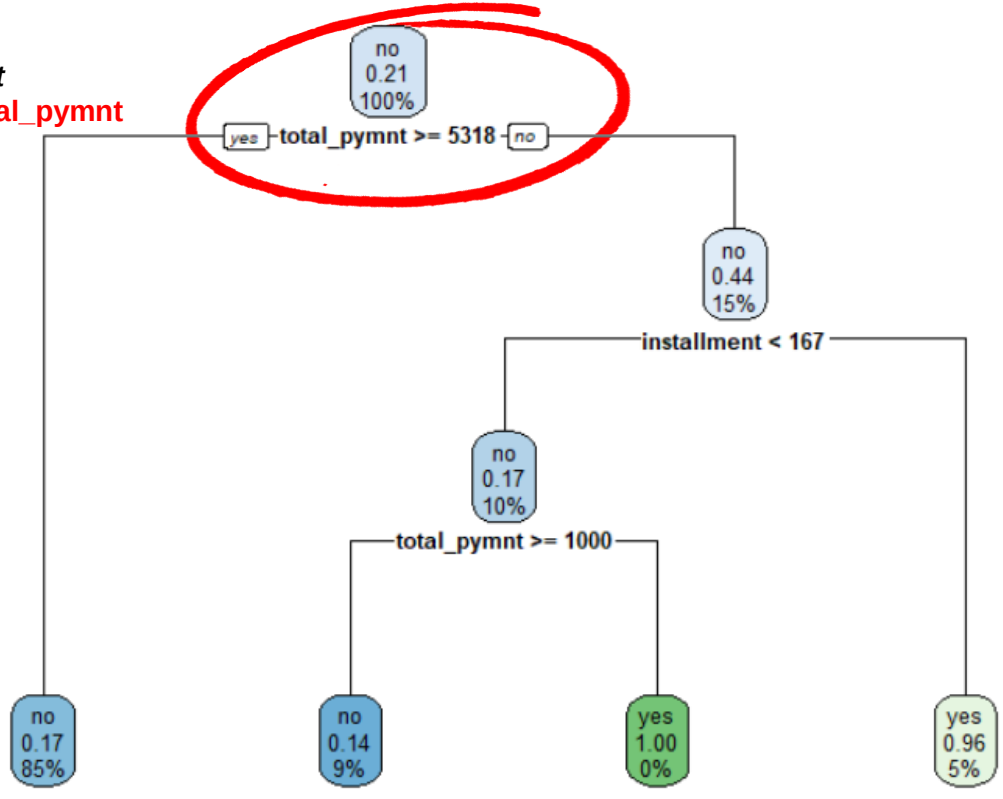


Decision Trees (CART)
Handling Missing Values

- Surrogate variables
  - **Tree grows** with the selected splits (primary splits) using **observations with no missing values** on the **split variables**
  - Weakness: CART is **biased** toward selecting **variables** with many **missing values** for a **primary split** (Kim & Loh, 2001)
- When a **split variable is missing** for an observation:
  - **Surrogate split**: use a **surrogate variable** instead of the primary split variable
  - A **surrogate variable** is **another independent variable and cut-off value** whose split most resembles the primary split
  - Ideally, they should **send exactly the same observations to the each child node**

Decision Trees (CART)
Handling Missing Values – Example Loan Default Prediction

333,301 observations (100%)
68,773 (20.6%) *default*
264,528 (79.4%%) *non-default*
Now: 100,000 missings in total_pymnt



- Randomly force **100,000 values of total_pymnt to be missing**
- Surrogate variables are used when we set na.action=na.rpart

Decision Trees (CART)
Handling Missing Values – Example Loan Default Prediction

- Output from R – using the summary() command:

```
Primary splits:
   total_pymnt < 5317.716 to the right, improve=8636.406, (100,000
missing)
 Surrogate splits:
   loan_amnt < 4712.5 to the right, agree=0.917, adj=0.515, (100,000
split)
```

Observations with a **missing in total_pymnt** are **split** on **loan_amnt** instead

Decision Trees (CART)
Handling Missing Values – Example Loan Default Prediction

- Agreement: **proportion of observations** sent to the **"correct" leaf node** when the **surrogate is used** (instead of the primary split)

$$Agree = \frac{correct\ surrogate}{\#\ obs\ in\ parent\ node}$$

- Adjusted agreement: **deducts** the surrogate **agreement from the "go with the majority"** baseline

$$\mathrm{Adj} = \frac{(correct\ surrogate\ -\ correct\ "go\ with\ majority")}{(\#\ obs\ in\ parent\ node\ -\ correct\ "go\ with\ majority")}$$

Decision Trees (CART)
Discussion

- Impurity Criterion
  - **Gini Index** is **biased** toward selecting split variables with **many missing values**

- Required Assumptions
  - Observations are independent
  - **Joint distribution** of **X** and **Y** in the **training** data is the **same** as in the **test** data

- Overfitting
  - Single trees have **high variance**: unstable predictions
  - Small perturbation in the data → large changes in leaf nodes
  - Solution? **Ensemble learning** (more on that after the break!)

Conditional Inference Trees

Conditional Inference Trees
Using significance as the splitting criterion

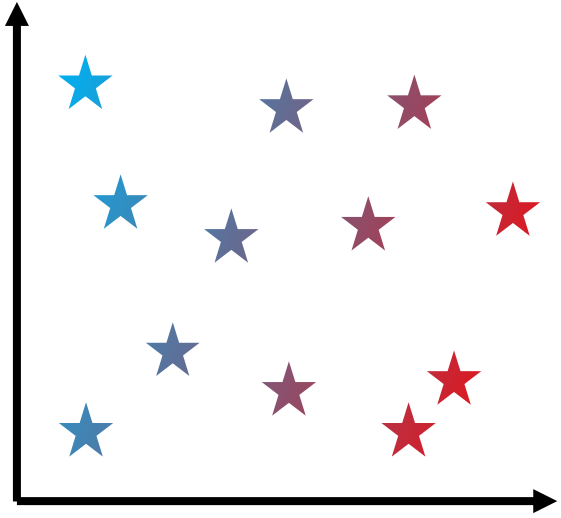
- Instead of node-impurity, use the p-value to choose the best variable for the next split

- Advantages
  - Reduces overfitting by using the most statistically powerful variables first
  - Provides a natural stopping criterion (no more splits after no variables are significant)
    - This also makes pruning unnecessary
  - Does not favour variables with many missing values

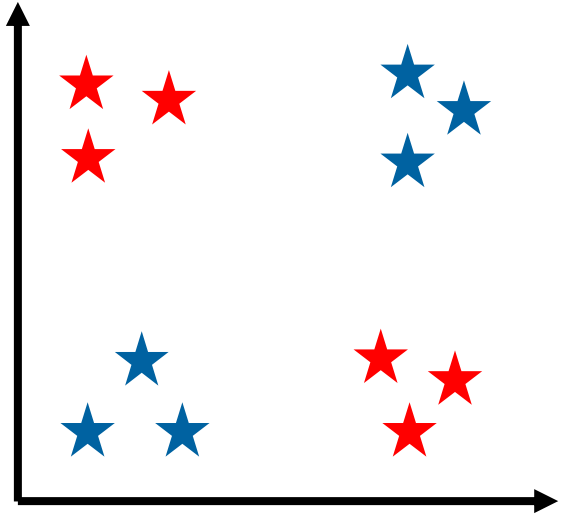
- Disadvantages
  - More computationally intensive, without necessarily providing better performance
  - Feature significance is not directly related to predictive performance of the tree model

Conditional Inference Trees
Situations where they struggle

− Simple linear relationships



− Strong crossover interaction



Introduction to Ensemble Methods
Example: Loan Default Prediction – Confusion Matrix Comparison

		Logistic Regression		Decision Tree (CART)		Decision Tree (CTREE)	
		Validation Data		Validation Data		Validation Data	
		No Default	Default	No Default	Default	No Default	Default
Predictions	No Default	84,951	2,134	84,889	13,740	85,032	1,672
	Default	155	18,855	217	7,249	74	19,317
Accuracy		97.8%		*86.8%*		***98.4%***	
Precision		99.8%		*99.7%*		***99.9%***	
Recall		89.8%		*34.5%*		***92.0%***	
F1-Score		94.2%		*50.9%*		***95.7%***	

- Remember: **No information rate** in the validation data is **80.2%**.

Hands-on Exercises

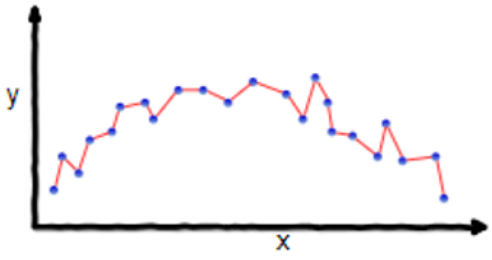
Detour: Bias-Variance Tradeoff

Bias-Variance Tradeoff
Expected Prediction Error

- Expected prediction error for a new observation with $\boldsymbol{X = x_0}$?
  - Consider $Y = f(X) + \varepsilon,\ \text{s.t.}\ \mathrm{E}(\varepsilon) = 0\ and\ Var(\varepsilon) = \sigma_\varepsilon^2$
- Using a quadratic loss function (squared error)
  - $Error(x_0) = \mathrm{E}[\mathrm{Y} - \hat{f}(x_0)]^2 = \mathrm{E}[\mathrm{Y} - f(x_0)]^2 +$ ⇒ $Var(\varepsilon) = \sigma_\varepsilon^2$ — Irreducible unless σ_ε^2=0!
    - $\mathrm{E}[\hat{f}(x_0) - f(x_0)]^2 +$ ⇒ $\boldsymbol{Bias}^2$ — Result of misspecifying $f(X)$
    - $\mathrm{E}[\hat{f}(x_0) - E[\hat{f}(x_0)]]^2$ ⇒ $\boldsymbol{Var(\hat{f}(x_0))}$ — Result of using a sample to estimate f
- Goal in Machine Learning: minimize this combination
- Complex models: usually low bias but **high variance**

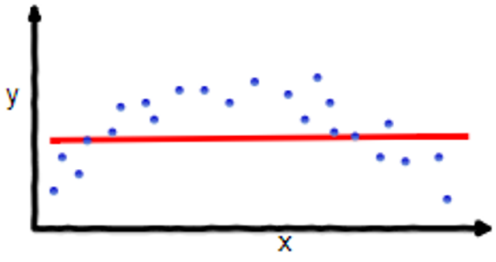
Bias-Variance Tradeoff Examples

Low Bias
High Variance



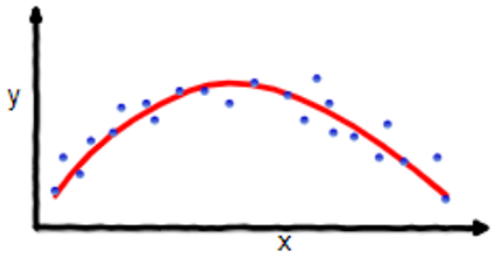
Overfitting

High Bias
Low Variance



Underfitting

Low Variance
Low Bias

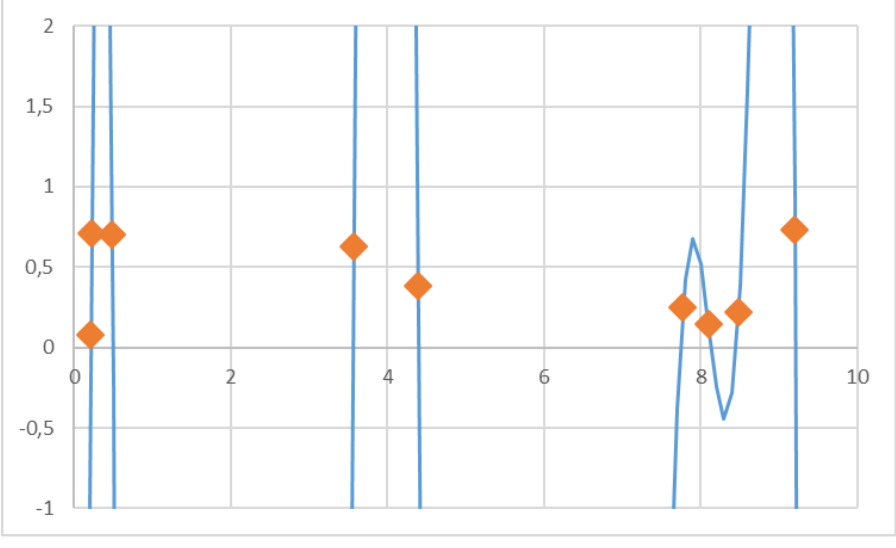


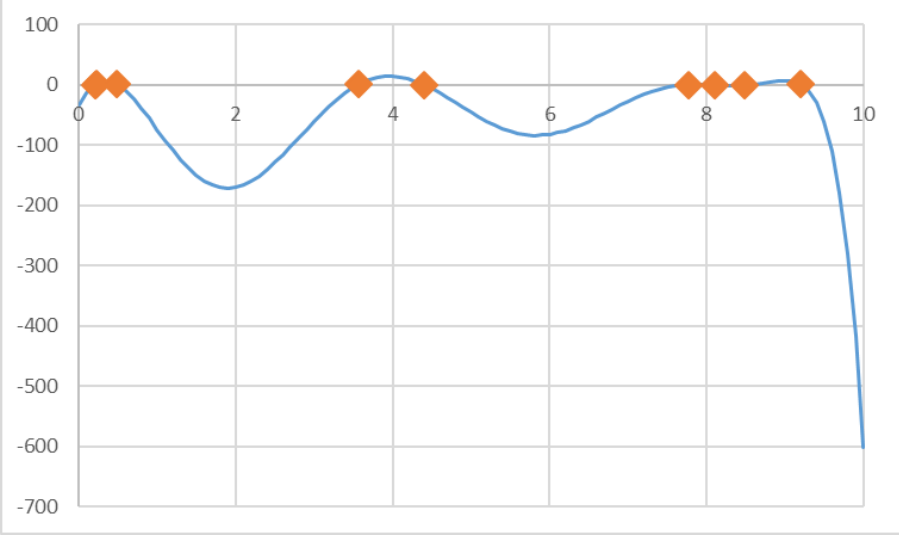
Good Balance

Source: towardsdatascience.com/understanding-the-bias-variance-tradeoff-165e6942b229

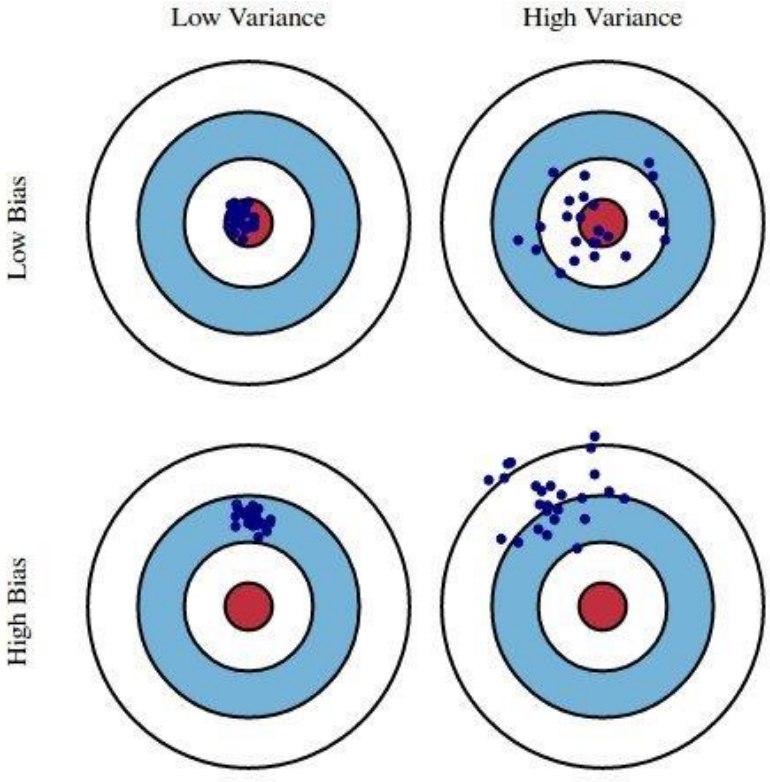
Bias-Variance Tradeoff
Examples

- Extreme Overfitting: Polynomial regression of degree 9 through 9 datapoints
- Every known observation is hit perfectly, but in between…



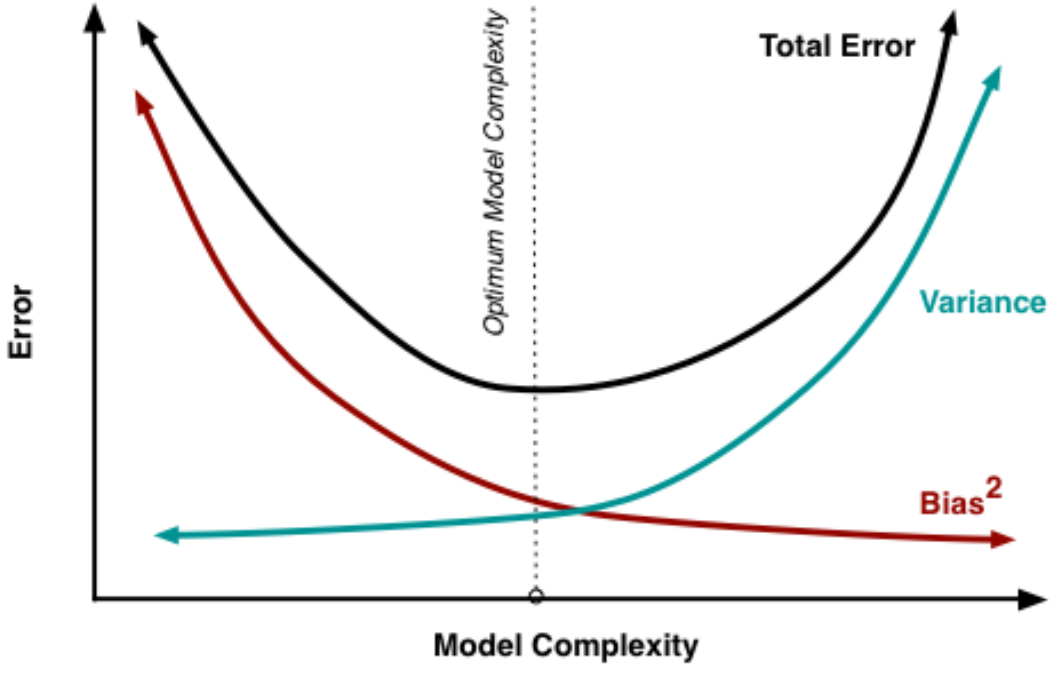


Bias-Variance Tradeoff
Examples



Source: https://www.kdnuggets.com/2016/08/bias-variance-tradeoff-overview.html

Bias-Variance Tradeoff
Changes in Error with Model Complexity



Source: scott.fortmann-roe.com/docs/BiasVariance.html

Bias-Variance Tradeoff
What Are Your Options?

- Use cross-validation to estimate the validation error
- Test and validate your model on new observations (set part of the data aside for that)
- Combine multiple models
  - This combination is known as **ensemble learning**
  - **Most influential development in Machine Learning** in the past decade (Seni & Elder, 2010)
  - Main idea: average **predictions of several models**

Introduction to Ensemble Methods

Introduction to Ensemble Methods
Overview

- Again (!) not really a “new” development
  - Bagging (Breiman,**1996**)
  - Boosting (Freund & Schapire, 1997)
  - Random Forests (Breiman, 2001)
  - Gradient Boosting (Friedman, 2001)

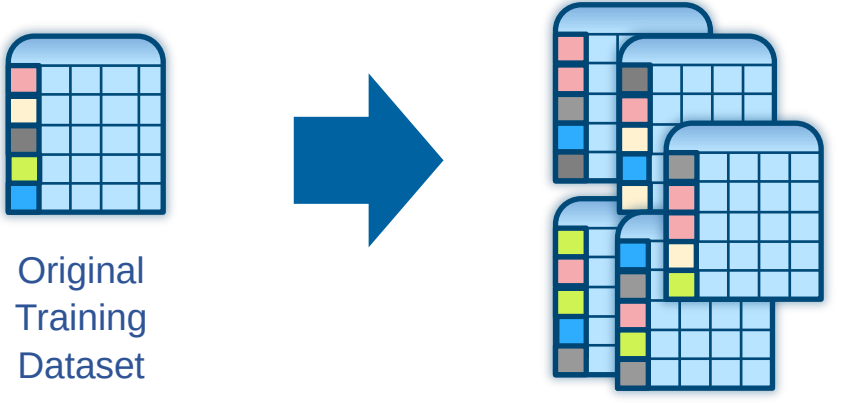
- Group of models that together deliver **one aggregate prediction**
  - Idea: combine a **large number of “weak” models** to produce a **stronger, more stable** model
  - Use **average** of predictions (numerical dep. var.) or **majority voting** (categorical dep. var.)

Introduction to Ensemble Methods
Overview

- Ensemble methods retain many advantages of Machine Learning methods
  - Model **higher order interactions** automatically
  - Helpful in **p >>n** problems
  - Robust to **outliers**
  - **Multicollinearity** is not a problem
- But…
  - More **difficult to visualize and interpret** results
  - Usually known as a “**black box**”

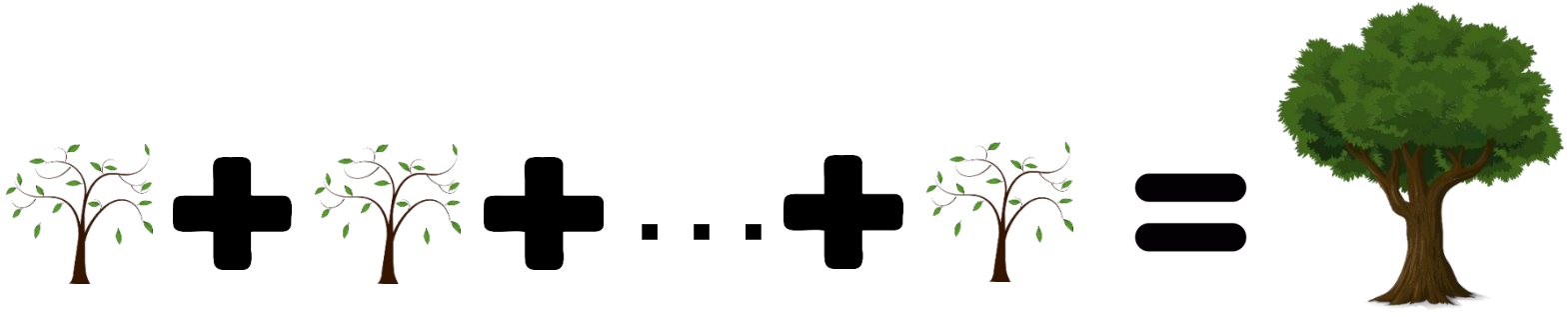
Introduction to Ensemble Methods
Bagging Weak Models

- Bagging: bootstrap aggregating (Breiman, 1996)
- Idea: use your dataset to artificially **generate “new” datasets**
  - **Sample with replacement** from training data (bootstrapping)



Introduction to Ensemble Methods
Bagging Weak Models

- **Estimate** a model on **each "new" bootstrapped dataset**
- **Aggregate predictions** (average or majority voting)
  - Improves **accuracy** and model **stability**
  - Often used for decision trees, but you can use it with **any weak supervised model**



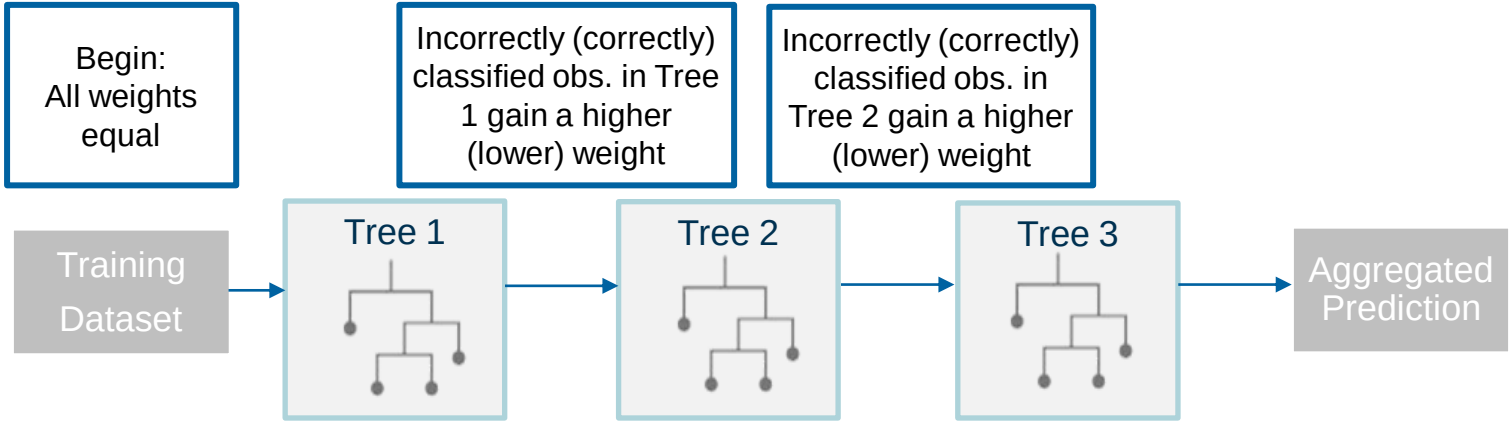
Introduction to Ensemble Methods
Boosting Weak Models

- Ensemble method proposed by Freund & Schapire (1997)
- Main idea
  - **Sequentially** train new models, giving **more importance** to observations **“difficult to predict”**
  - **Weights or residuals** reflect **how “difficult”** it is to **predict** the outcome for a specific **observation**
  - **Dataset** is the **same**, only the **weights change** at each iteration
- **Often** applied in the context of **decision trees, but** applicable to **any “weak” model**
- Trees are usually smaller than in Random Forests (often “stumps”: only one split)

Introduction to Ensemble Methods
Boosting Weak Models (Classification)

- **Objective: minimize classification error**
- **Weights** are **adjusted** and **renormalized** at each **iteration**
- Aggregated prediction: **Sum** the **“votes”** of all models
- **“Voting power”** of each model is a **function of its accuracy**
- Models that **accurately predict** many observations **change weights** more significantly and have a **higher influence** in the aggregated prediction

Introduction to Ensemble Methods
Boosting Weak Models (Classification)



Subsequent tree focuses on predicting observations with higher weights

Voting power: the vote of "good models" matters more

Introduction to Ensemble Methods
Example: Loan Default Prediction – Confusion Matrix Comparison

		Logistic Regression		Decision Tree (CTREE)		Bagged Trees		Boosted Trees	
		Validation Data		Validation Data		Validation Data		Validation Data	
		No Default	Default	No Default	Default	No Default	Default	No Default	Default
Predictions	No Default	84,915	2,151	85,032	1,672	82,750	15,904	84,792	5,306
	Default	161	18,868	74	19,317	2,356	5,085	314	15,683
Accuracy		97.8%		***98.4%***		*82.8%*		*94.7%*	
Precision		99.2%		***99.9%***		*97.2%*		*99.6%*	
Recall		89.8%		***92.0%***		*24.2%*		*74.7%*	
F1-Score		94.2%		***95.7%***		*35.8%*		*84.8%*	

- Remember: **No information rate** in the validation data is **80.2%**.

Hands-on Exercise

References

Alfaro, E., Gamez, M. & Garcia, N. (2013), adabag: An R Package for Classification with Boosting and Bagging, *Journal of Statistical Software*, 54 (2), 1-35.

Alfaro, E., Garcia, N., Gamez, M. & Elizondo, D. (2008), Bankruptcy forecasting: An empirical comparison of AdaBoost and neural networks'. *Decision Support Systems*, 45, 110-122.

Breiman, L. (1996), Bagging predictors, *Machine Learning*, 24(2), 123-140.

Breiman, L. (1998), Arcing classifiers, *The Annals of Statistics*, 26(3), 801-849.

Breiman, L. (2001), Statistical Modeling: The Two Cultures, *Statistical Science*, 16(3), 199-231.

Breiman, L., Friedman, J. H., Stone, C. J. & Olshen, R. A. (1984), Classification and Regression Trees. Belmont, CA, Wadsworth.

Hastie, T., Tibshirani, R. & Friedman, J. (2009), The Elements of Statistical Learning. Data Mining, Inference, and Prediction, Springer.

Hothorn, T., Hornik, K. & Zeileis, A. (2006), Unbiased Recursive Partitioning: A Conditional Inference Framework, Journal of Computational and Graphical Statistics, 15(3), 651-674.

References

Loh, W.-Y. (2014), "Fifty Years of Classification and Regression Trees, International Statistical Review, 82(3), 329-348.

Loh, W.-Y. & Shih, Y.-S. (1997), "Split Selection Methods for Classification Trees, Statistica Sinica, 7(4), 815-840.

Quinlan, J. R. (1986), Induction of Decision Trees, Machine Learning, 1(1), 81-106.

Quinlan, J. R. (1993), C4.5: Programs for Machine Learning, San Mateo, CA, Morgan Kaufmann.

Shmueli, G. (2010), To Explain or to Predict?, *Statistical Science*, 25(3), 289-310.

Therneau, T. M. & Atkinson, E. J. (1997), An Introduction to Recursive Partitioning Using the RPART Routines, Technical Report 61, Mayo Clinic, Section of Statistics, Rochester, Minnesota.

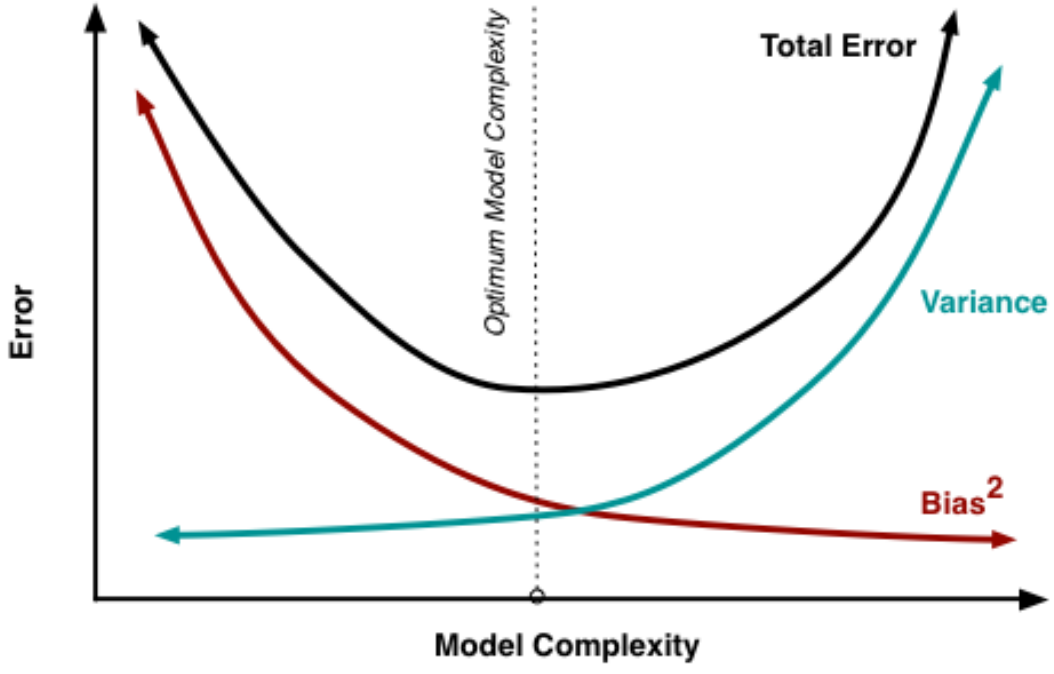
Day 3
Ensemble Methods

Agenda

- Random Forests
- Causal Random Forests
- Hands-on Exercise
- Break
- Gradient Boosting
- Hands-on Exercise

Recap

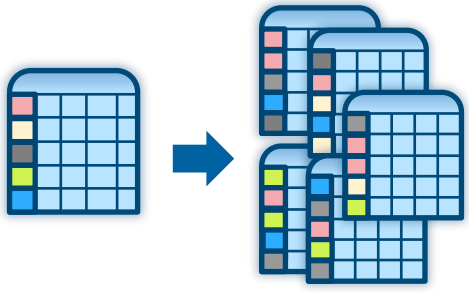
Bias-Variance Tradeoff
Changes in Error with Model Complexity

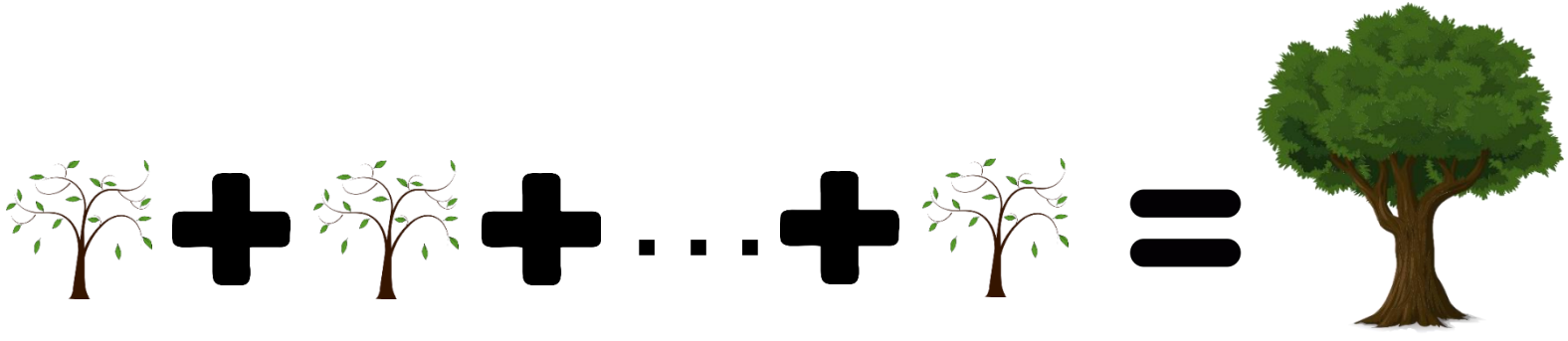


Source: scott.fortmann-roe.com/docs/BiasVariance.html

Ensemble Methods
Bagging Weak Models

- **Estimate** a model on **each "new" bootstrapped dataset**
- **Aggregate predictions** (average or majority voting)
  - Improves **accuracy** and model **stability**
  - Often used for decision trees, but you can use it with **any weak supervised model**





Random Forests

Random Forests
Motivation

- **Bagging** helps to **reduce model variance**

- But…
  - Trees often have **many splits in common**
  - Consequence: **similar predictions**

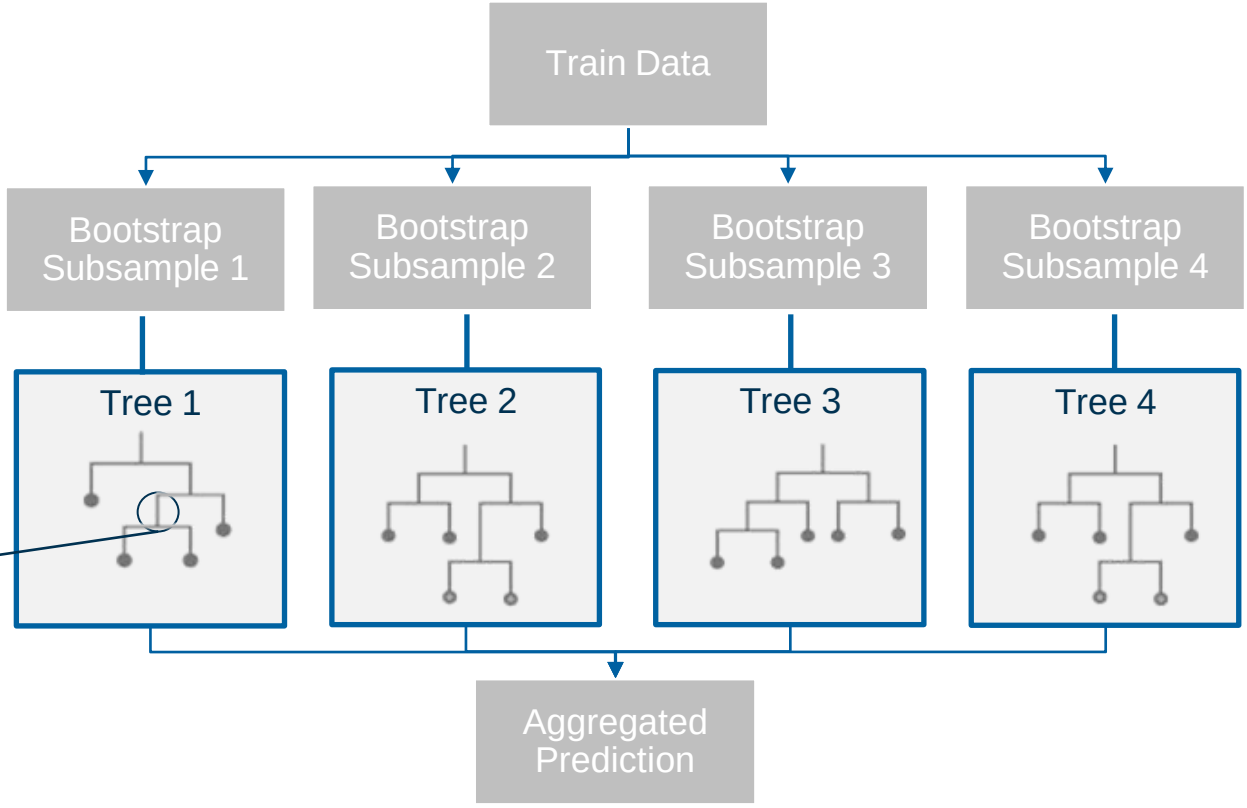
- Idea: why not also introduce **randomness** in the selection of the candidate **variables** for a split?

- This idea is implemented in Random Forests (Breiman, 2001)

Random Forests
Overview

- Use bootstrap to generate N **"new" datasets**
- For **each tree and split**:
  - Only give the algorithm a **random set of m independent variables** to choose from (out of the **total M independent variables** in the dataset)
- Estimate N **trees**, each with a **different dataset**
- **Aggregate** predictions with **averaging** or **majority voting**

Random Forests
Overview



At each split the tree considers only **a fraction** of all the available **independent variables** (**m variables** out of the total M variables)

Random Forests
How to Choose m?

- How to choose $\boldsymbol{m}$?
- Breiman's suggested heuristic for choosing $\boldsymbol{m}$
  - $m = \sqrt{n}$
  - n: total number of independent variables
- Use cross-validation to find the "best" value of $\boldsymbol{m}$ for the data / problem at hand
  - This is called (hyper)**parameter tuning**
- **When $\boldsymbol{m} = \boldsymbol{n}$, you're back to bagging**
- **$\boldsymbol{m}$ in the R implementation: mtry parameter**

Random Forests
Out-of-Bag Observations

- Another nice feature of Random Forests: OOB (Out-Of-Bag)
  - Refers to **observations** in the **original dataset** that were **not part of** the respective **bootstrap replicate**
  - So, **OOB observations** were **not used** to **construct the model**
  - Therefore, **great** to **estimate the generalization error**!

- By default, OOB ~ 1/3 of observations
  - But you can **adjust** it as you wish

- OOB observations are also used to compute variable importance measures

Original Train Dataset

Bootstrap Replicate

In this bootstrap replicate, the gray observation is OOB

Random Forests
Parameters

- **ntree:** Number of trees
- **mtry:** m parameter: number of randomly selected candidate variables at each split
- **nodesize:** minimum number of observations in each leaf node

Introduction to Ensemble Methods
Example: Loan Default Prediction – Confusion Matrix Comparison

		Logistic Regression		Decision Tree (CTREE)		Boosted Trees		Random Forest	
		Validation Data		Validation Data		Validation Data		Validation Data	
		No Default	Default	No Default	Default	No Default	Default	No Default	Default
Predictions	No Default	84,915	2,151	85,032	1,672	84,792	5,306	85,069	2,396
	Default	161	18,868	74	19,317	314	15,683	7	18,623
Accuracy		97.8%		***98.4%***		*94.7%*		*97.7%*	
Precision		99.2%		***99.9%***		*99.6%*		***99.9%***	
Recall		89.8%		***92.0%***		*74.7%*		*88.7%*	
F1-Score		94.2%		***95.7%***		*84.8%*		*94.0%*	

- Remember: **No information rate** in the validation data is **80.2%**.

Random Forests
Discussion

- Random Forests reduce variance by
  - **Training** trees on **different subsets** of the training dataset
  - **Limiting the subspace** of independent **variables** considered for a **split**

- But, there's no free lunch…
  - **Bias** may slightly increase (remember the **potentially biased splits** in each tree)
  - More difficult to **interpret**

Random Forests
Discussion

- Bias can be reduced with tuning
  - Find **optimal parameters** for your model / data (e.g. tree depth, number of trees, number of observations in leaf nodes)
  - Usually done with a **grid search** (test a **finite combination** of parameters)

- Interpretability can be tackled with interpretability methods
  - E.g., variable importance measures, partial dependence plots, etc.

Causal Forests

Causal Inference and Potential Outcomes

- Correlation does not imply causation
- Understanding cause and effect is highly relevant for decision-making
  - Medical treatments
  - impact of marketing
  - Targeting of financial aid
- The *fundamental problem of causal inference* makes direct observation impossible
  - Additional assumptions allow for estimation of causal effects

Causal Inference and Potential Outcomes

- Causal effect: the magnitude by which an outcome is changed by an interventional change, between the real world and the counterfactual world
  - $E\left[Y_{real,A=1}\right] - E\left[Y_{counterfactual,A=0}\right]$

Causal Inference and Potential Outcomes
Rubin causal model

1. Conditional independence (or conditional ignorability/exogeneity or conditional unconfoundedness):

$$\left(Y_i^1, Y_i^0\right) \perp\!\!\!\perp D_i | X_i.$$

2. Stable Unit Treatment Value Assumption (SUTVA) (or counterfactual consistency):

$$Y_i = Y_i^0 + D_i(Y_i^1 - Y_i^0).$$

3. Overlap Assumption (or common support or positivity):

$$\forall x \in supp(X_i), \quad 0 < P(D_i = 1|X_i = x) < 1,$$
$$P(D_i = 1|X_i = x) \stackrel{\text{def}}{=} e(x). \tag{1}$$

4. Exogeneity of covariates:

$$X_i^1 = X_i^0.$$

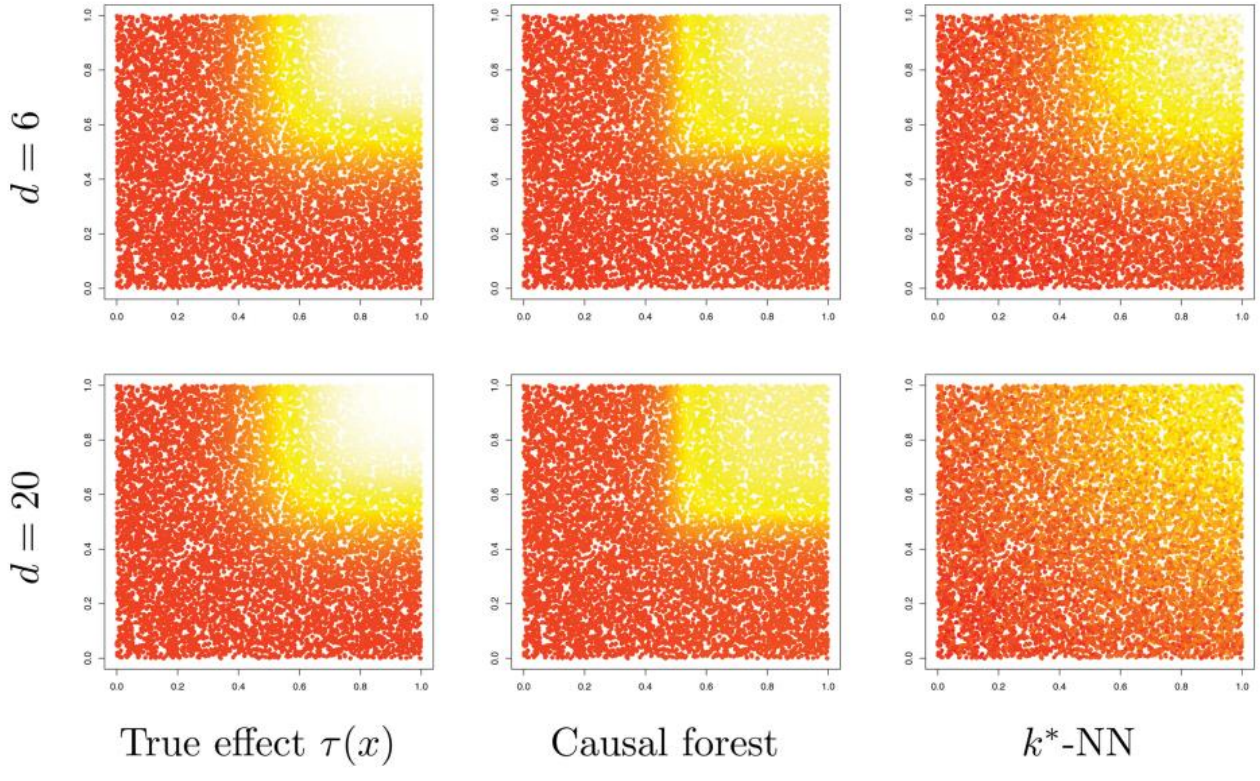
- The treatment assignment is independent of the potential outcomes (no “perfect doctor”)
- No hidden variation between the treated and control observations
- No subgroup is located entirely within either the treatment or control group
- The treatment has no effect on the other variables

Causal Forests

- Random forests are made of random trees, causal forests are made of causal trees
- To avoid overfitting the data is split into sets: for **splitting** and for **estimating**

- The splitting criterion optimizes for finding splits associated with treatment effect heterogeneity
  - Find leaves where the treatment effect is constant, but different from other leaves
  - Only split when the nodes will still contain enough samples for estimation
- Using the leaves makes it possible to estimate the conditional average treatment effect (CATE)

Causal Forests



Causal Tree
Build Procedure: double-sample trees

- Input: n training samples with features X_i, response Y_i and treatment assignment W_i

1. Divide the samples into a splitting subsample and an estimation subsample
2. Grow a tree via recursive partitioning (with the splitting subsample)
   - For each possible split: estimate treatment effect by comparing samples with and without treatment
   - Use the split that best optimizes the treatment effect in the two new nodes
3. Estimate conditional treatment effect on samples within leaf nodes (with the estimation subsample)

Hands-on Exercise

Gradient Boosting

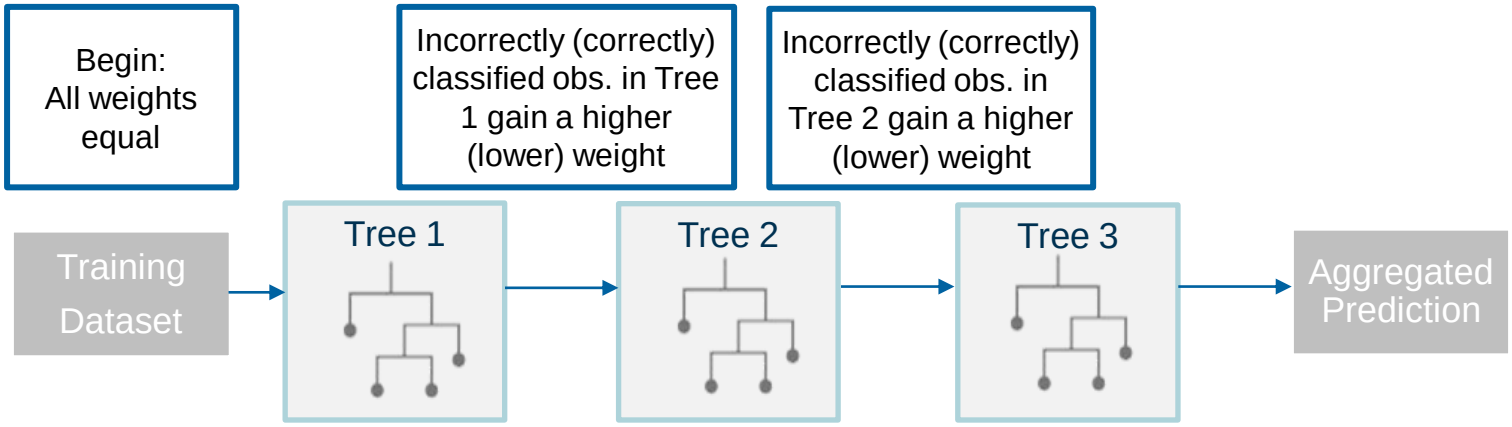
Gradient Boosting
First Idea of Boosting

- Ensemble method proposed by Freund & Schapire (1997)
- Main idea
  - **Sequentially** train new models, giving **more importance** to observations **“difficult to predict”**
  - **Weights or residuals** reflect **how “difficult”** it is to **predict** the outcome for a specific **observation**
  - **Dataset** is the **same**, only the **weights change** at each iteration
- **Often** applied in the context of **decision trees, but** applicable to **any “weak” model**
- Trees are usually smaller than in Random Forests (often “stumps”: only one split)

Gradient Boosting
Boosting for Classification

- **Objective: minimize classification error**
- **Weights** are **adjusted** and **renormalized** at each **iteration**
- Aggregated prediction: **Sum** the **“votes”** of all models
- **“Voting power”** of each model is a **function of its accuracy**
- Models that **accurately predict** many observations **change weights** more significantly and have a **higher influence** in the aggregated prediction

Gradient Boosting
Boosting for Classification



Subsequent tree focuses on predicting observations with higher weights

Voting power: the vote of "good models" matters more

Gradient Boosting
Other Views of Boosting

- Friedman, Hastie & Tibshirani (2000)
  - Adaboost ~ **optimization method** to **minimize** a particular exponential **loss function** (for classification)
  - Find that exponential loss ~ Bernoulli likelihood
  - So, maximize Bernoulli likelihood instead (for classification)

- Breiman (1999)
  - Boosting ~ **gradient descent** with a **special loss function**

- Friedman (2000)
  - Generalize: from **Adaboost** to **Gradient Boosting**
  - Handle many **other loss functions**
  - Very important development: link **"obscure" computational learning** to standard **statistics** (likelihood) and **function optimization**

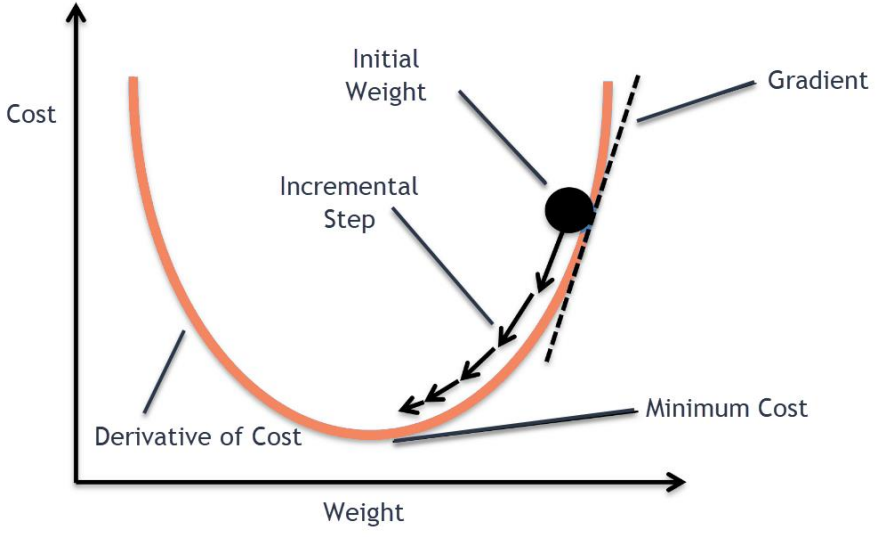
Gradient Boosting
Slightly Change the Target Function

- Instead of training $\boldsymbol{h}(\boldsymbol{X})$ on residuals of $F(X), \boldsymbol{y} - \widehat{\boldsymbol{F}}(\boldsymbol{X})$….
- Train $\boldsymbol{h}(\boldsymbol{X})$ on gradient of the loss function: $L\big(y, F(X)\big) = \frac{\big(y - F(X)\big)^2}{2}$
- Goal: minimize J $= \sum_i L\big(y_i, F(X_i)\big)$ for all obs. i by finding appropriate $F(X_i)$
- Yields an optimization problem: $\frac{\partial J}{\partial F(X_i)} = F(X_i) - y_i$
- Residuals are negative gradients of L!

$$y_i - F(X_i) = -\frac{\partial J}{\partial F(X_i)}$$

Gradient Boosting
Slightly Change the Target Function

- $y_i - F(X_i) = -\frac{\partial J}{\partial F(X_i)}$ - Does this look familiar?
- Hint: Gradient Descent: $\theta_i := \theta_i - \rho \frac{\partial J}{\partial \theta_i}$
- **Gradient Boosting = Boosting + Gradient Descent**



Source: https://www.oreilly.com/library/view/learn-arcore-/9781788830409/e24a657a-a5c6-4ff2-b9ea-9418a7a5d24c.xhtml

Gradient Boosting
Generalization Using Gradient Descent

- Changing from residuals to gradient descent → Gradient Boosting
  - Generalize the method to **other loss functions** (e.g. log loss, absolute loss)
- So, in our new framework:
  - Start with conservative estimate for $F(X)$ (e.g. mean of y)
  - Compute negative gradient (i.e. residual): $-g(X) = -\frac{L(y,F(X))}{\partial F(X)} = y - F(X)$
  - Fit a new model $h(X)$ to $-g(X)$
  - Update $F(X)$: $F(X) \coloneqq F(X) + \rho h(X)$
- ρ is the **size of the step**

Gradient Boosting
Generalization Using Gradient Descent

- By restricting the **tree depth** we can control the **order of approximation** of $F(X)$
  - E.g., Gradient Boosting with **stumps** (only one split) are a **first order (linear) approximation**
- But, if we fit the training data "too closely": **overfitting**!
- Solution: **regularization** (or shrinkage)
  - Introduce a **parameter** to **slow down** the **incorporation of new results** to the aggregate model
  - Each update is "scaled" or "shrunk" by the **"learning rate"** parameter ν.
  - $F(X) \coloneqq F(X) + \boldsymbol{\nu} \cdot \rho h(X)$

Gradient Boosting
Generalization Using Gradient Descent

- How to choose the learning rate?
  - **Tradeoff** between **learning rate** and **number of trees**
  - Best to **optimally choose** parameters (e.g. using cross-validation)
- **Stochastic Gradient Boosting** (Friedman, 2002)
  - Gradient Boosting + **randomness** in the train data seen by each model (as in **Bagging**)

Gradient Boosting Parameters

- Parameters to choose (names in italic: inputs to gbm function in R)
  - **Loss function** (*distribution*)
  - **Number of trees** (*n.trees*)
  - **Learning rate** (*shrinkage*)
  - **Tree depth** - shorter trees usually give better results! (*interaction.depth*)
  - **Minimum** number of **observations in leaf nodes** (*n.minobsinnode*)
  - **Fraction of observations** in training data **randomly selected** to grow the next tree (*bag.fraction*)

Introduction to Ensemble Methods
Example: Loan Default Prediction – Confusion Matrix Comparison

		Logistic Regression		Decision Tree (CTREE)		Random Forest		Gradient Boosting	
		Validation Data		Validation Data		Validation Data		Validation Data	
		No Default	Default	No Default	Default	No Default	Default	No Default	Default
Predictions	No Default	84,915	2,151	85,032	1,672	85,069	2,396	85,008	1,394
	Default	161	18,868	74	19,317	7	18,623	98	19,595
Accuracy		97.8%		*98.4%*		*97.7%*		***98.6%***	
Precision		99.2%		***99.9%***		***99.9%***		***99.9%***	
Recall		89.8%		*92.0%*		*88.7%*		***93.4%***	
F1-Score		94.2%		*95.7%*		*94.0%*		***96.3%***	

- Remember: **No information rate** in the validation data is **80.2%**.

Gradient Boosting
Discussion

- Gradient Boosting reduces bias by
  - **Training** trees **sequentially** on the training dataset
- Stochastic Gradient Boosting reduces variance by
  - **Training** trees on **different subsets** of the training **dataset**
- What kind of problems can it solve?
  - **Any (!) loss function** for which you can **compute the gradient**

Gradient Boosting
Discussion

- But:
  - Usual shortcomings of ensembles: **bias, interpretability**
  - Gradient Boosting requires **more computing time**
- Remedies:
  - Tuning
  - Tools and measures for interpretability
  - A few other tricks possible (see *xgboost*)

Hands-on Exercise

References

Athey, S., Tibshirani, J., and Wager, S. (2019), Generalized Random Forests, *Annals of Statistics*, *47*(2), 1148-1178.

Breiman, L. (1999), Prediction Games and Arcing Algorithms, *Neural Computation*, 11(7), 1493-1517.

Breiman, L. (2001), Random Forests, *Machine Learning,* 45(1), 5-32.

Breiman, L. (2001), Statistical Modeling: The Two Cultures, *Statistical Science*, 16(3), 199-231.

Friedman, J., Hastie, T. & Tibshirani, R. (2000), Special Invited Paper. Additive Logistic Regression: A Statistical View of Boosting, *Annals of Statistics*, 28 (2), 337-374.

Friedman, J.H. (2001), Greedy Function Approximation: A Gradient Boosting Machine, *Annals of Statistics*, 29 (5), 1189-1232.

Friedman, J.H. (2002), Stochastic Gradient Boosting, *Computational Statistics & Data Analysis*, 38 (4), 367-378.

Hastie, T., Tibshirani, R. & Friedman, J. (2009), *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*, Springer.

References

Ridgeway, G. (1999), The State of Boosting, *Computing Science and Statistics*, 31, 172-181.

Shmueli, G. (2010), To Explain or to Predict?, *Statistical Science*, 25(3), 289-310.

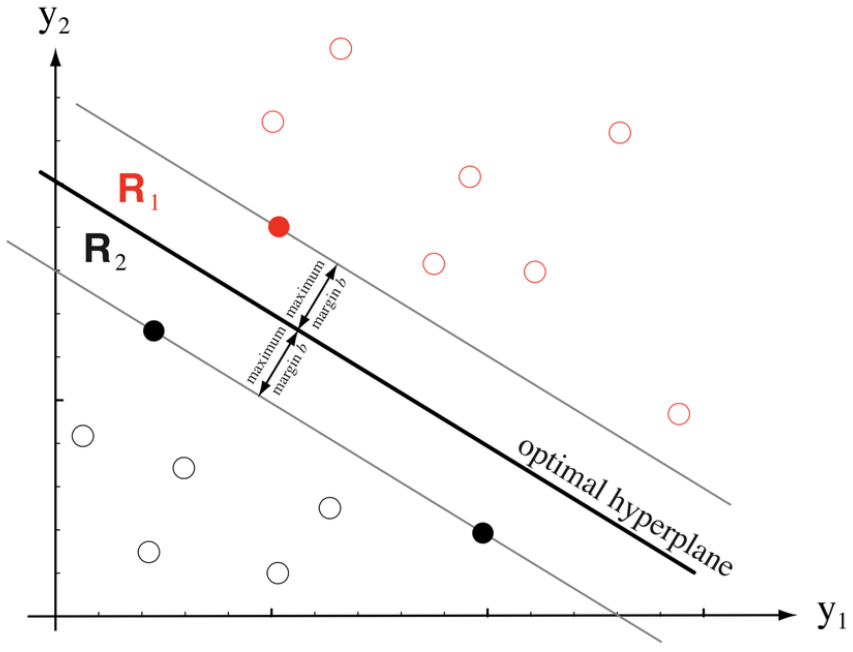
Wager, S., and Athey, S. (2018), Estimation and Inference of Heterogeneous Treatment Effects Using Random Forests, *Journal of the American Statistical Association*, *113*(523), 1228-1242.

Day 4
Support Vector Machines & Outlook

Support Vector Machines (SVM)
Introduction

- Supervised model for binary classification
  - Extensions to multi-class or regression problems exist

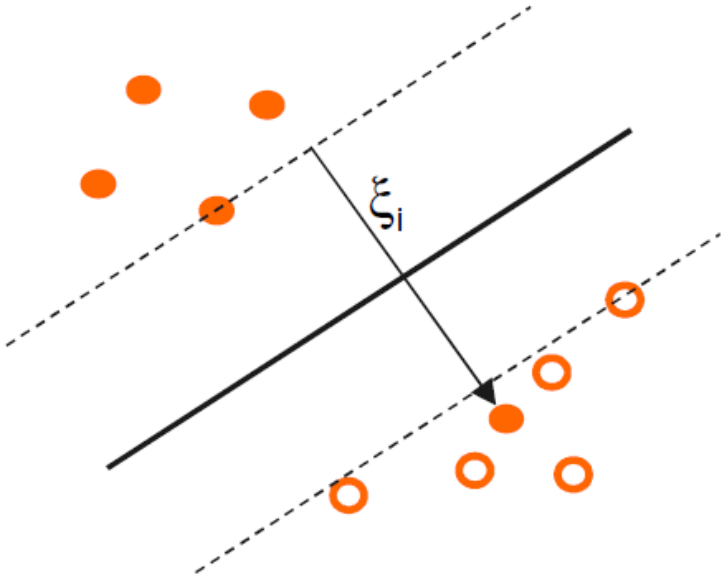
- Empirically a very robust prediction method
- Operate by finding a linearly separating hyperplane
- Through the kernel trick, can be applied widely, including to text and image data



Support Vector Machines (SVM)
Primal Problem

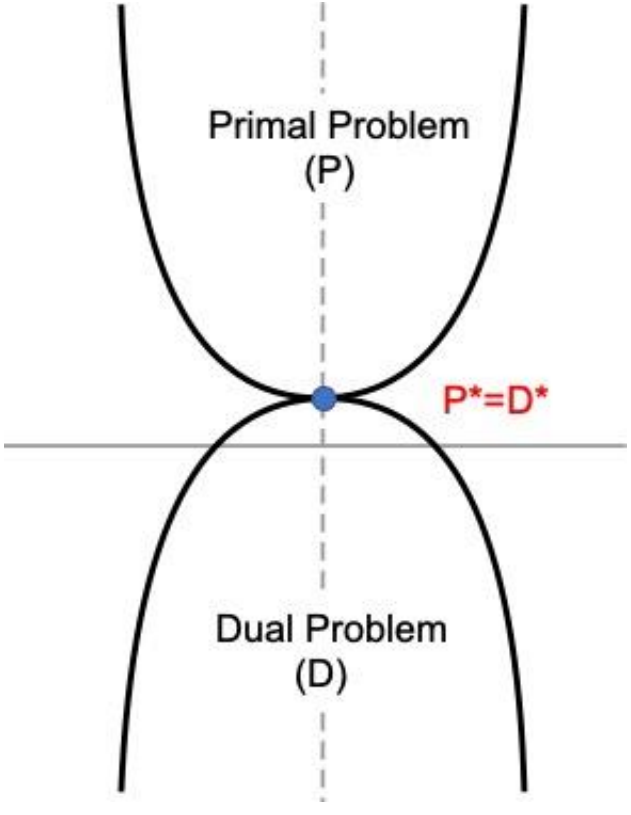
- Try to find the plane that yields the largest margin between the 2 classes
- If classes cannot be perfectly split, minimize the distance to the plane ζ_i (hinge loss)
- Only some data points actually influence the model, these are the *support vectors*
- The parameter λ can tune between the margin size and the error
- $minimize\ \frac{1}{n}\sum_{i=1}^{n}\zeta_i + \lambda\|w\|^2$
- $subject\ to\ y_i(w^t x_i - b) \geq 1 - \zeta_i$

$$\zeta_i \geq 0$$



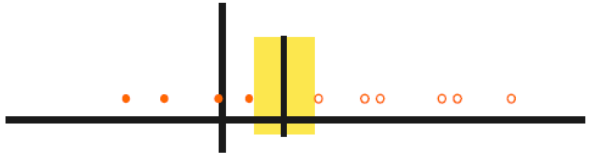
Support Vector Machines (SVM)
Dual Problem

- An equivalent formulation, which can be computed more efficiently
- The weights w can also be defined as a linear combination each support vector with strength c_i
- $w = \sum_{i=1}^{n} c_i y_i x_i$
- Then $c_i = 0$ exactly when x_i lies on the correct side of the margin
- $maximise \ \sum_{i=1}^{n} c_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} y_i c_i (x_i^T x_j) y_j c_j$
- $subject\ to \ \sum_{i=1}^{n} c_i y_i = 0 \ and \ 0 \leq c_i \leq \frac{1}{2n\lambda}$

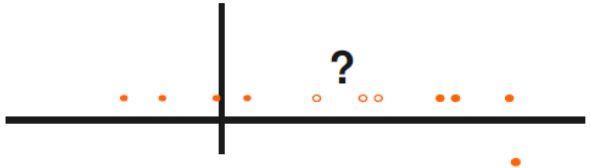


Support Vector Machines (SVM)
Non-Linearly Separable Data

The linear case:



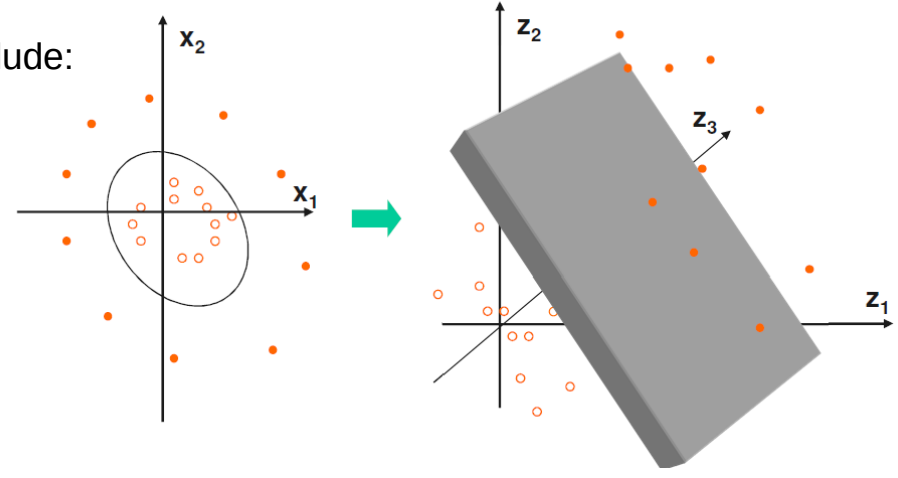
The non-linear case:



The trick:

Support Vector Machines (SVM)
Nonlinear Classification Through Transformation Kernels

- In the dual problem, the data points were only used in a dot product
  - This makes it possible to replace it with a dot product in a transformed feature space where linear separation is easier
- Common kernel transformations from x_i, x_j include:
  - Linear: $x_i \cdot x_j$
  - Polynomial: $(x_i \cdot x_j + r)^d$
  - Radial basis: $\exp(-\gamma \|x_i - x_j\|^2)$ for $\gamma > 0$
  - Sigmoid: $\tanh(\kappa x_i \cdot x_j + c)$



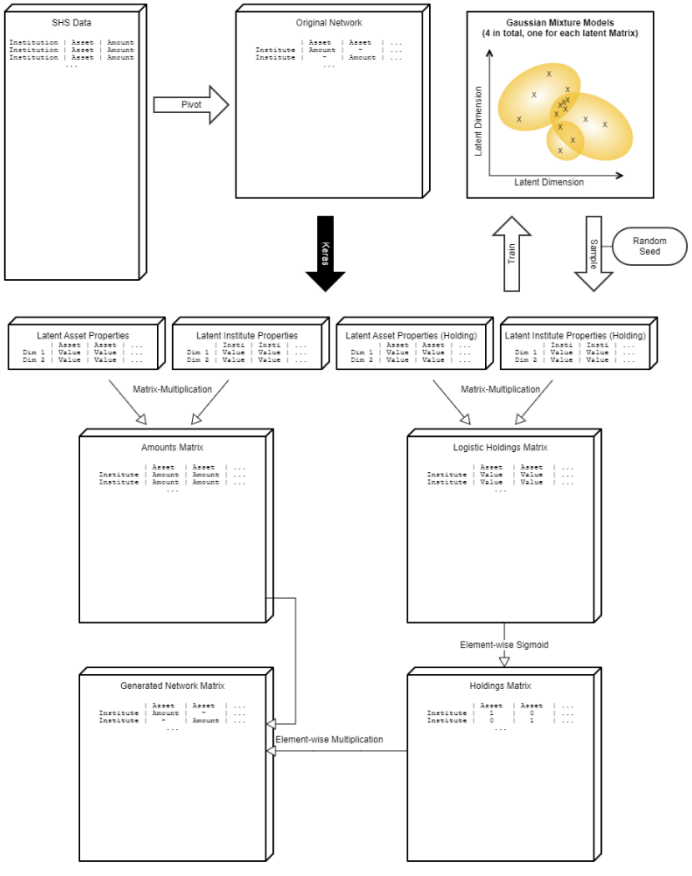
Support Vector Machines (SVM)
Extensions

- Multiclass SVM
  - One-versus-rest strategy
    - Train as many models as there are classes, the model with the highest output confidence wins
  - One-vs-one strategy
    - Train one model for each pair of classes, the class with the most votes wins
- Regression (SVR)
  - $minimize\ \frac{1}{2}\|w\|^2\ subject\ to\ |y_i - w \cdot x_i - b| \leq \varepsilon$
  - This cost function ignores any training data close to the model prediction

Hands-on Exercise

Machine Learning in Practice

Machine Learning for Synthetic Data Generation



Application

- Generate Synthetic Data that sufficiently looks like real network, while still maintaining confidentiality
- Variational Autoencoder + Differential Privacy

Advantages

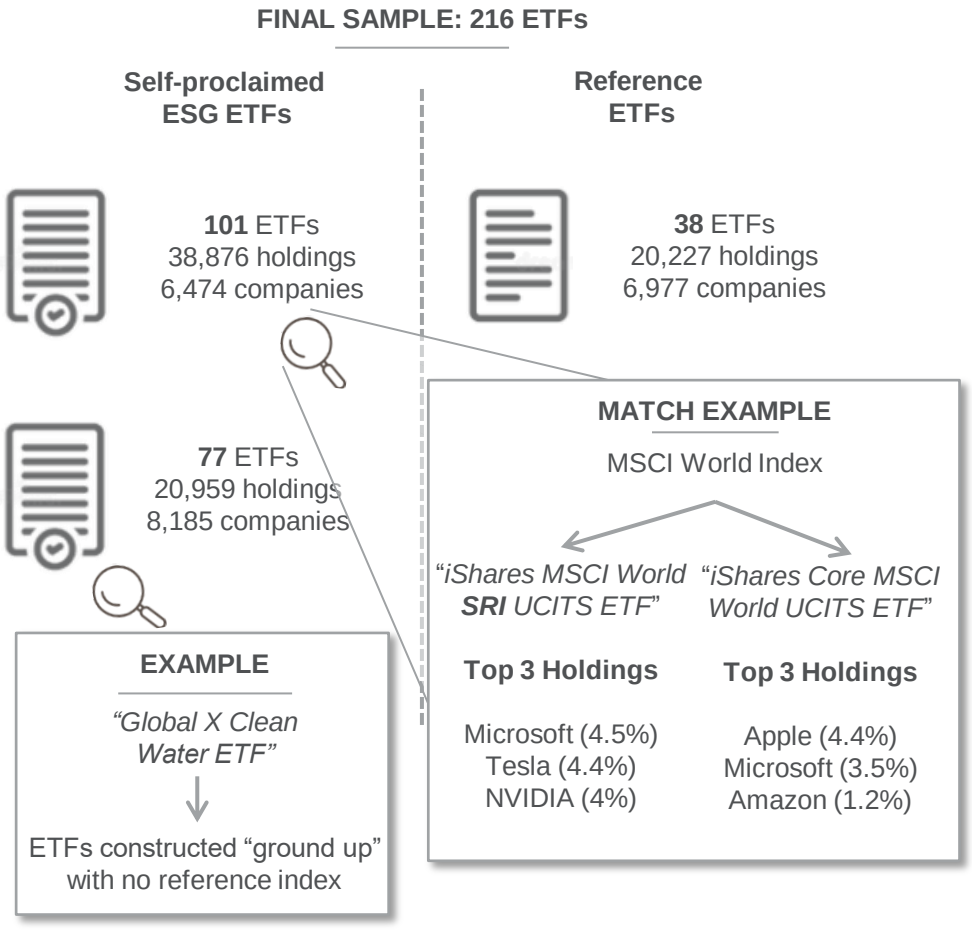
- Could send synthetic data home to researchers, to allow initial investigation without rigid controls

Challenges

- Data in matrix form is extremely sparse and cannot be reduced or decomposed without violating differential privacy

Source: Sebastian Seltmann (Research Data Service Centre)

Machine Learning for Classification of Self-Proclaimed Sustainable Funds



Application

- How do self-proclaimed "ESG" ETFs differ in terms of **sustainability strategy** and how does their strategy reflect on the portfolio's **emission intensity**?
- **Unsupervised learning** to detect clusters of self-proclaimed "ESG" ETFs and reference ETFs

Advantages

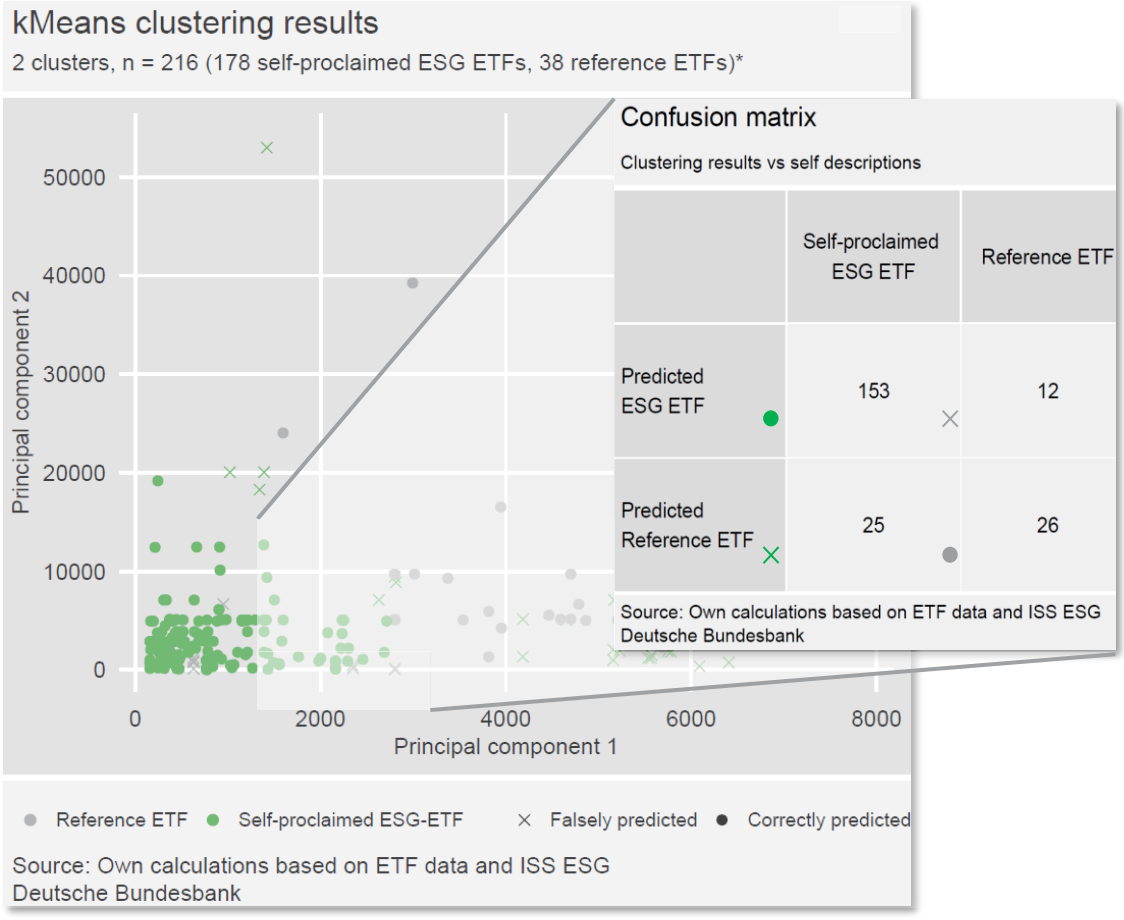
- Critical information for informed policy-making

Challenges

- No central public data source for self-proclaimed ESG ETFs (webscraping from each fund issuer)
- Available information is limited, heterogeneous, and in different formats

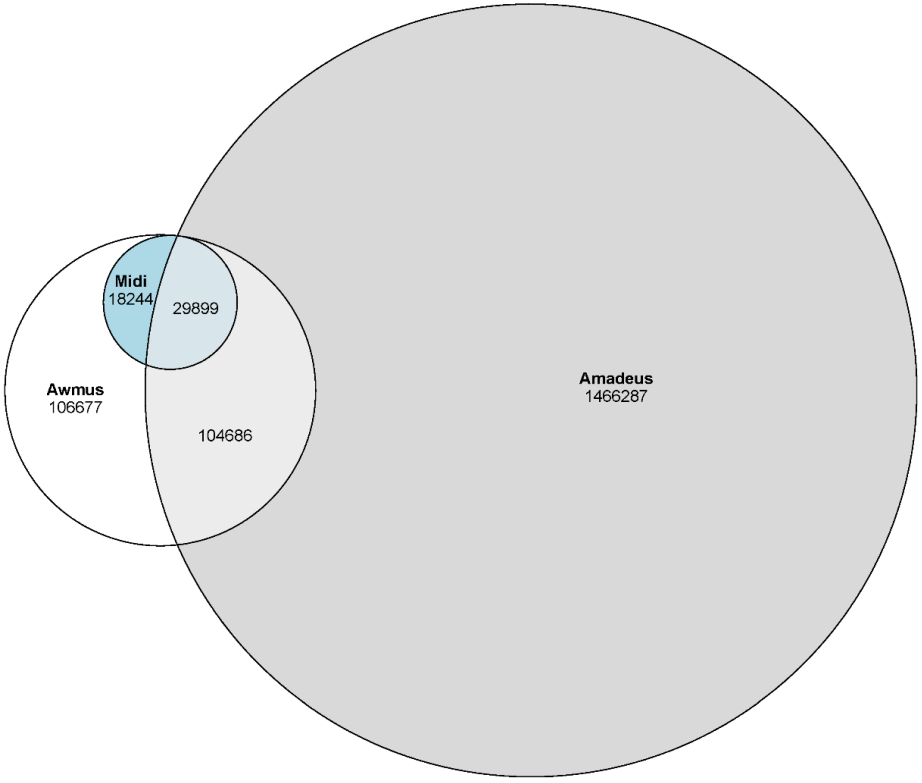
Source: Prof. Gabriela Alves Werb Ph.D., Hendrik Doll, Maurice Fehr, Ece Yalcin-Roder (Sustainable Finance Data Hub, Research Data Service Centre)

Machine Learning for Classification of Self-Proclaimed Sustainable Funds



- Clustering with a range of features based on **emissions**, **sectors**, **best-in-class proxies**, and **fund-level variables**
- Most informative features are **maximum scope 1 emission intensities** (not weighted) and the **ETF's age**
- Sector composition doesn't seem to add much predictive value (neither single nor aggregated using PCA)
- Issues with class imbalance

Machine Learning for Record Linkage



Application

- Matching / duplicates detection for record linkage in company data
- Multiple sources, no common unique identifier
- Probabilistic matching scores (Random Forest, Gradient Boosting)

Advantages

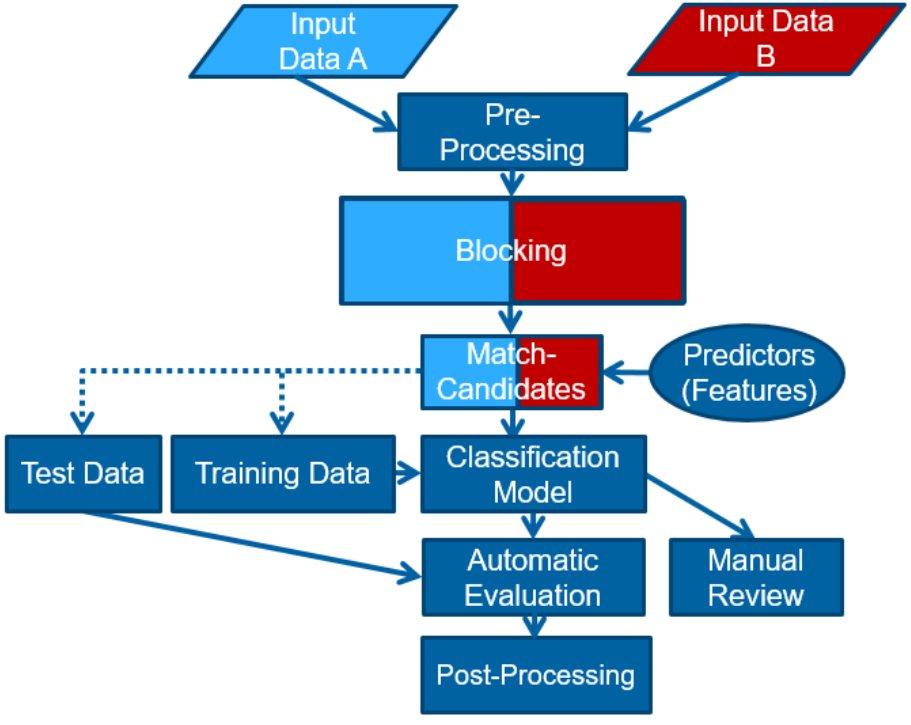
- Improve data matching quality
- Historicized matching tables for research projects using multiple data sets

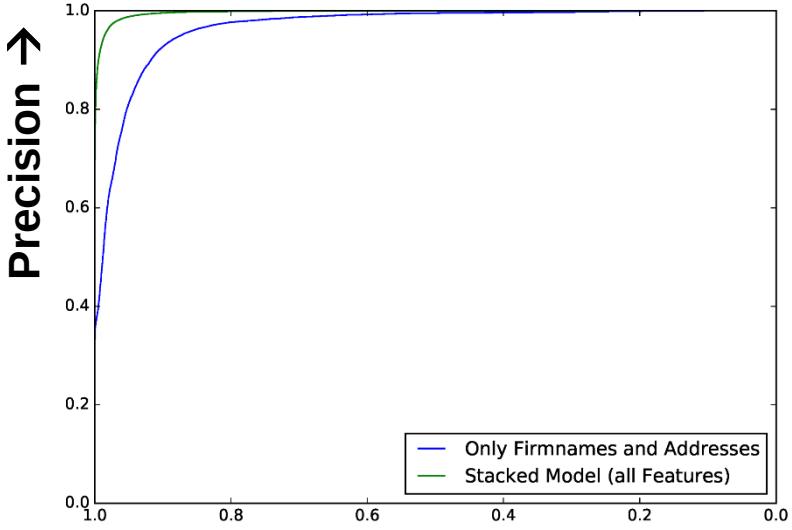
Challenges

- Data pre-processing (standardization of strings, feature engineering)

Source: Dr. Christopher-Johannes Schild (Research Data Service Centre)

Machine Learning for Record Linkage

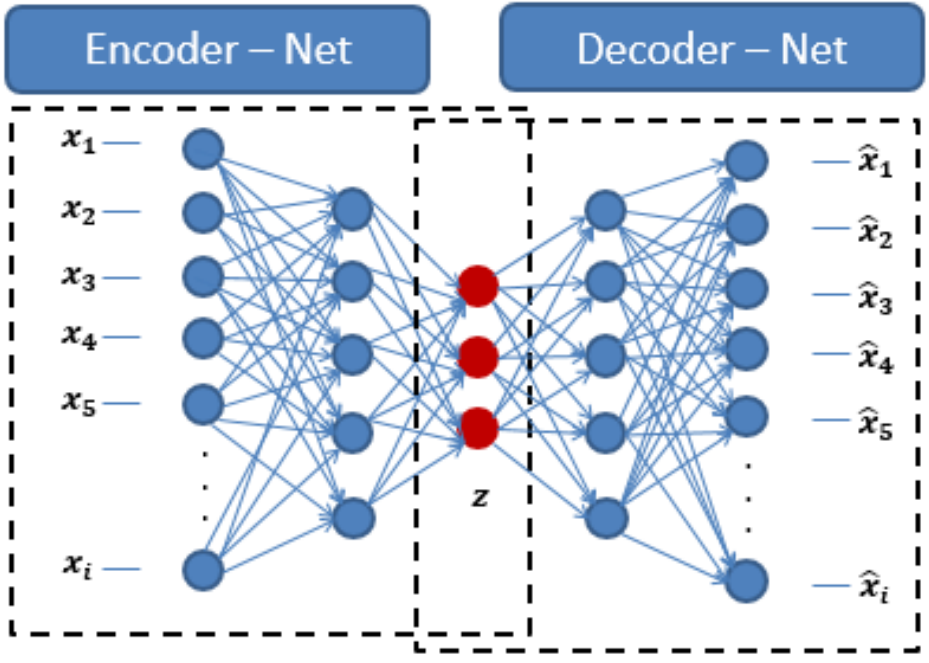




← **Recall**

- Precision = 98,0%
- Recall = 93,3%

Explaining Anomalies using Denoising Autoencoders (DAE)



Application

- Use Denoising Autoencoder Neural Networks to detect and explain anomalies in financial data
- Identify attributes associated with high anomaly scores using attribute reconstruction error

Advantages

- Shed light into binary prediction from anomaly detection (e.g., why is it an anomaly? which value was expected?)
- Faster and more efficient screening

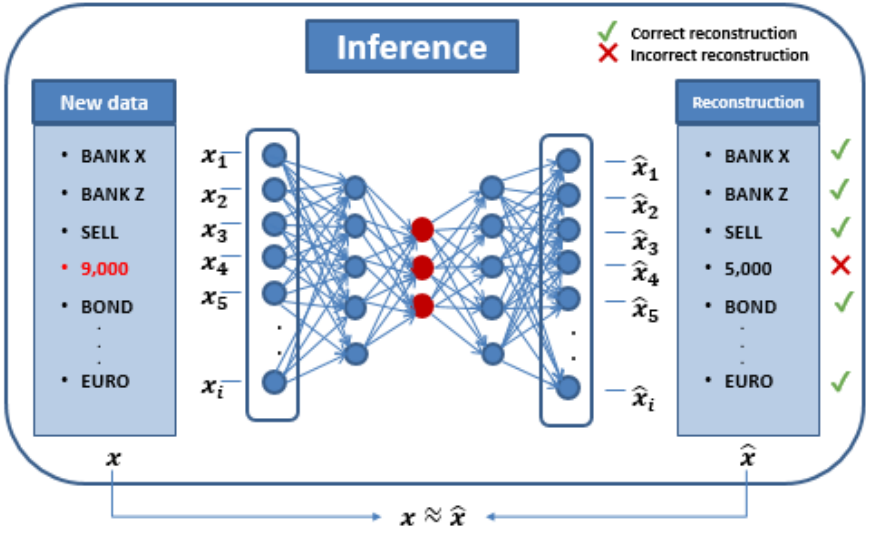
Challenges

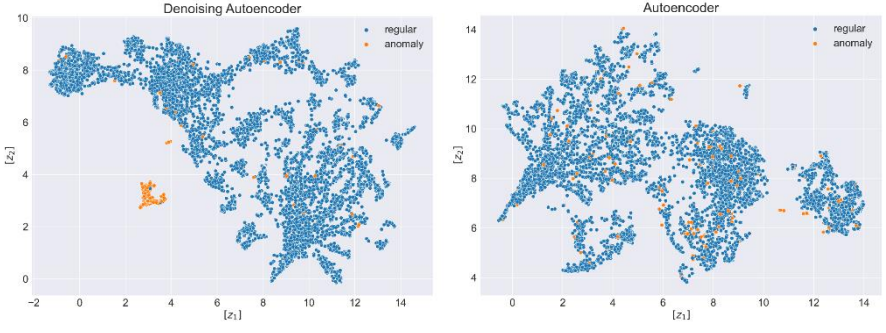
- Robustness against uninformative attributes

Source: Timur Sattarov (Data Service Center)*

* Sattarov, Herurkar, and Hees (2022), Explaining Anomalies using Denoising Autoencoders for Financial Tabular Data.

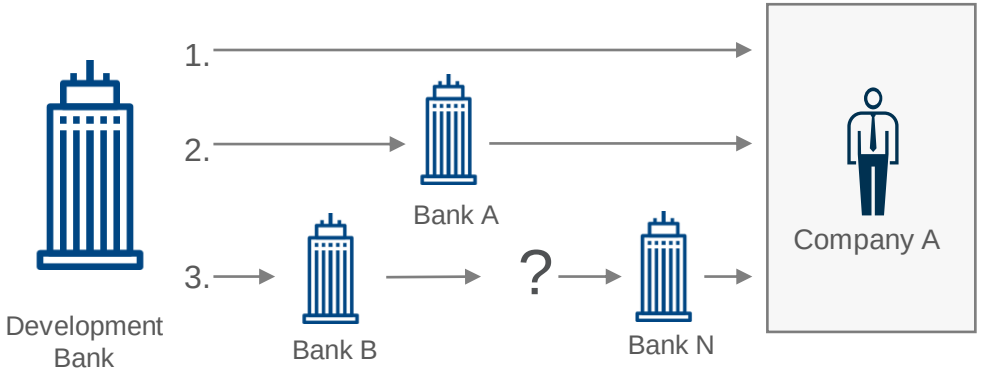
Explaining Anomalies using Denoising Autoencoders (DAE)

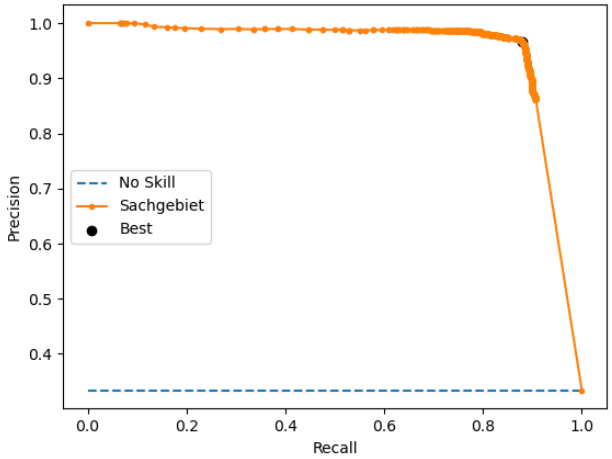




- Data sets
  - Publicly available: Credit Default, IEEE Fraud, Adult
  - Internal: Investment Funds Statistics Base
- Performance metrics
  - Precision at K (number of anomalies in the test data)
  - Mean average precision (mAP)
  - Mean expected value (mEV)
- The DAE **outperforms** its counterparts on average by **5%-30%**, depending on the metric.

Matching of Pass-Through Coronavirus Aid Loans





Application

- Difficult to “follow” a loan until the end of the pass-through chain
- Use supervised learning to identify coronavirus aid loans (state-backed) based on loan characteristics
- Combination of rule-based matching with an iterative matching algorithm

Advantages

- Amplify manual and rule-based matching
- Provide insights on the reach and effects of fiscal policy

Challenges

- Uneven reporting quality from different MFIs
- Small set of labelled data available (risk of overtfitting)

Source: Florian Hoffmann (Data Service Centre)

Machine Learning for Validating Company Data

ID	Company	Street	City	Postal Code	Economic Sector	Employees	Parent Company
1	Company 1	Street X	City A	1234	Car manufacturing	500	X
2	Company 2	Street Y	City B	5678	Car manufacturing	400	X

Application

- Multimodal learning to validate secondary company data (not collected for statistical purposes)
- Leverage external, publicly available data (e.g., satellite images, street view, textual data from companies' websites)

Advantages

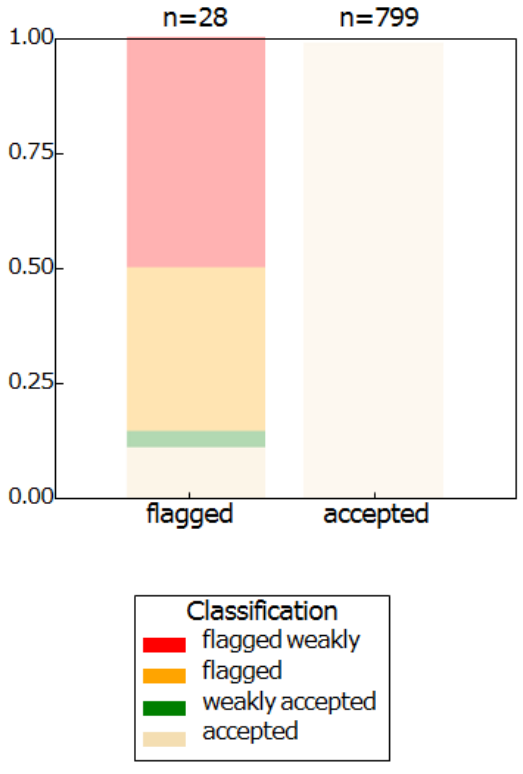
- Reduce effort with manual validations and quality checks (millions of entities)

Challenges

- High effort to generate annotate training data
- Handling special cases for companies with multiple offices or activities

Source: Prof. Gabriela Alves Werb Ph.D., Dr. Susanne Walter, Ece Yalcin-Roder (Research Data Service Centre)

Machine Learning for Data Quality Management (Securities Statistics)



Application

- Check data reported by German banks
- Securities Holdings Statistics

Advantages

- All of the 28 out of 827 securities in Top 35
- 50% reduction in time for check and increased effectiveness of evaluations

Challenges

- Incorporation to the data production process

Source: Dr. Tobias Cagala (Statistics)

Wrap Up & Q&A

Where Are We Headed?
Current Challenges and Opportunities

- Internal stakeholders: “what do we gain from using AI / machine learning?”
- Systems, talent, regulation (e.g. privacy)
- Markets are **complex**: regimes change, trends come and go
- **Unstable** models (asset management: “no longer than 3 weeks”)
- Use **networks** to better assess **systemic risk**
- Need for **explainable machine learning**

Where Are We Headed?
Explainable Machine Learning

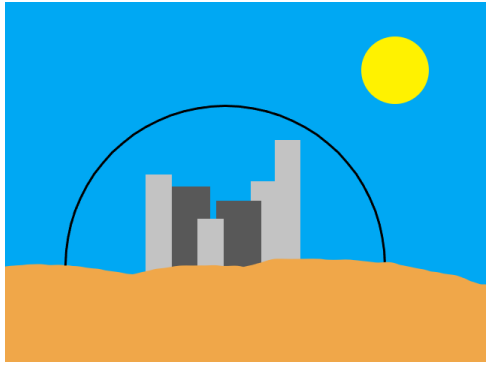
- **Avoid poor decisions**
  - Text mining at an investment fund – Berkshire Hathaway x Anne Hathaway
  - IBM “Watson for Oncology”
  - Tyndaris Investments automated trading: ~ US$20 M daily losses
- **Prevent algorithm bias**: is the model doing what it is supposed to?
  - Bias in recruiting tools (e.g., Amazon AI recruiting tool)
  - Predicting the likelihood of a criminal reoffending (e.g., US COMPAS)

Cutting Edge (publicly visible) Developments

- 2020 GPT-3: generates human-like text (175 billion parameters requiring 800GB of storage)
  - Trained on Common Crawl database (monthly ~300 Terabyte crawl of the web)
- 2021 DALL-E: generates images from natural-language descriptions, based on GPT-3 network
  - *“Teddy bears working on new AI research underwater with 1990s technology”*

Cutting Edge (publicly visible) Developments

- 2022 Stable Diffusion: besides generating images from text-prompts, can also perform image-to-image translations
  - Model, including weights (4 GB), is public since 22 August 2022
  - *“A distant futuristic city full of tall buildings inside a huge transparent glass dome, In the middle of a barren desert full of large dunes, Sun rays, Artstation, Dark sky full of stars with a shiny sun, Massive scale, Fog, Highly detailed, Cinematic, Colorful”*



A very big thank you to…

- All participants for the lively discussions and insightful comments
- Eréndira, Laura, Gerardo and all other colleagues we did not have a chance to meet for the outstanding organization
- The wonderful team of translators and IT support for a seamless experience

References

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992), A Training Algorithm for Optimal Margin Classifiers, *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, 144-152.

Chang, C. C. and Lin, C. J. (2011), LIBSVM: a Library for Support Vector Machines, *ACM Transactions on Intelligent Systems and Technology*, *2*(3), 1-27.

Cortes, C. and Vapnik, V. (1995), Support-Vector Networks, *Machine Learning,* 20, 273–297

Vapnik V. and Chervonenkis A. (1974), *Theory of Pattern Recognition* [in Russian], Nauka, Moscow.