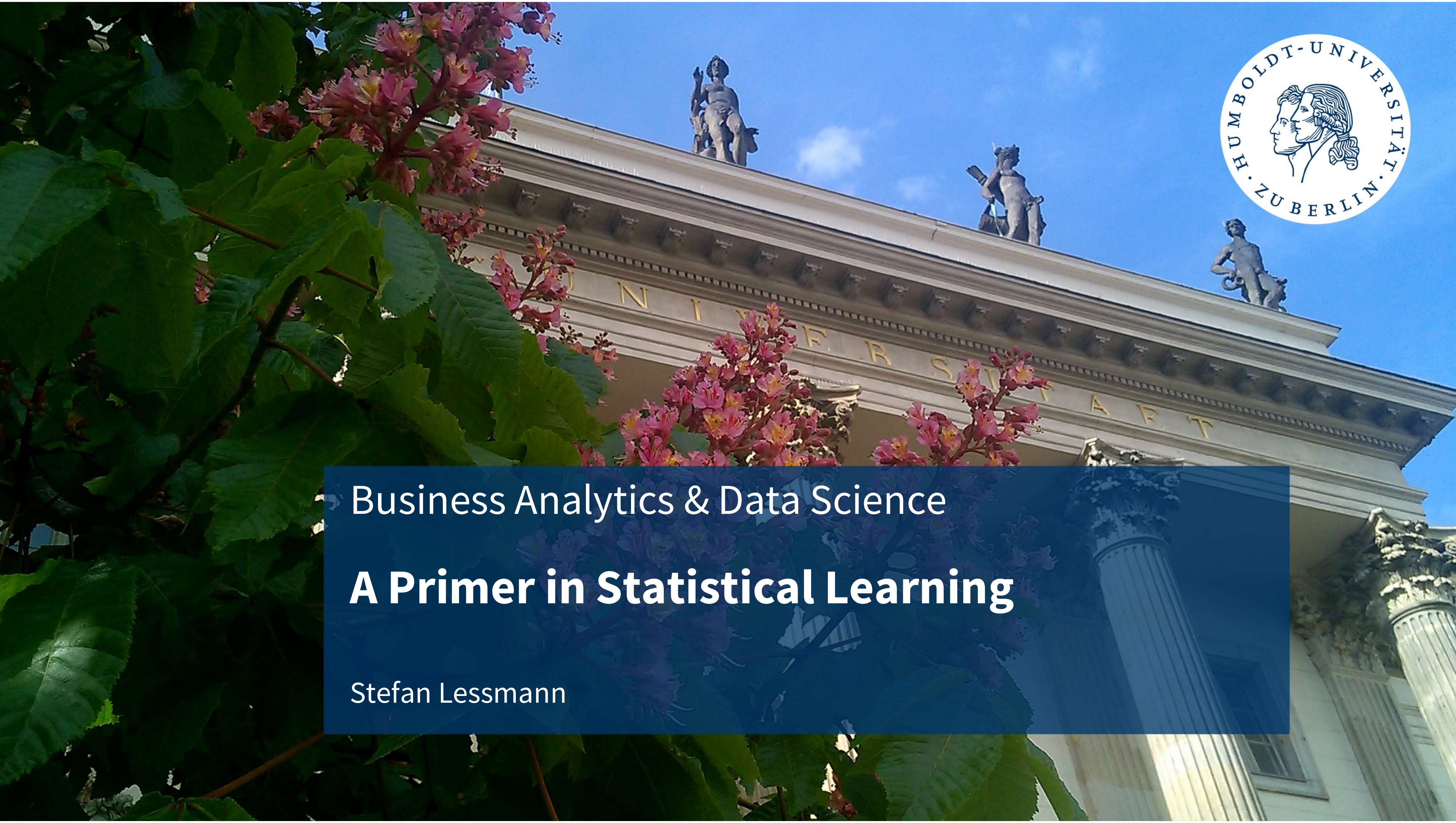


Business Analytics & Data Science

A Primer in Statistical Learning

Stefan Lessmann

Agenda

The Bias-Variance Trade-Off
Bias, variance, and the fundamental problem of overfitting

Regularization
Idea, connection to bias-variance trade-off, implementation

Model Selection
Search strategies and process perspective

Summary

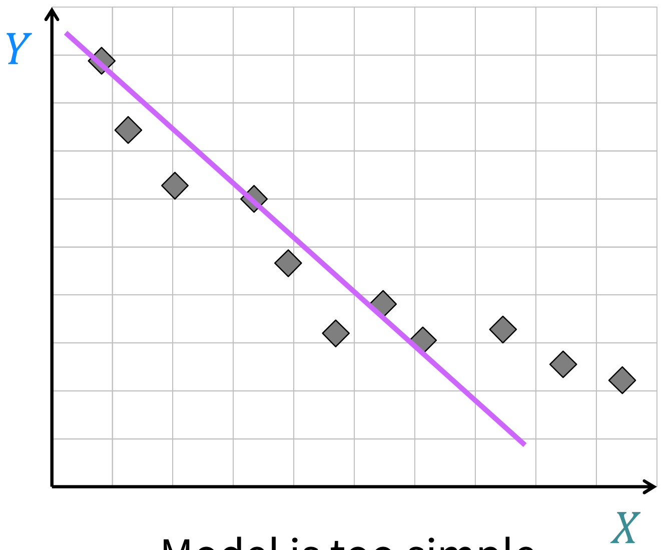
Bias-Variance Trade-Off

Bias, variance, and the fundamental problem of overfitting

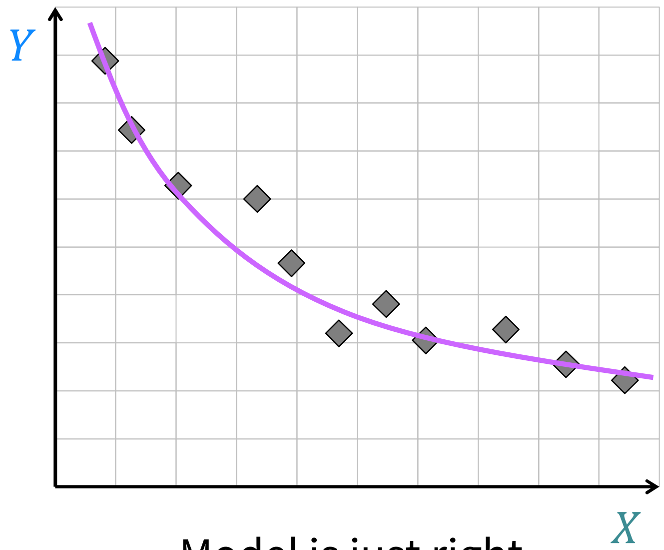
The Fundamental Problem of Overfitting

Advanced ML algorithms may overfit the training data

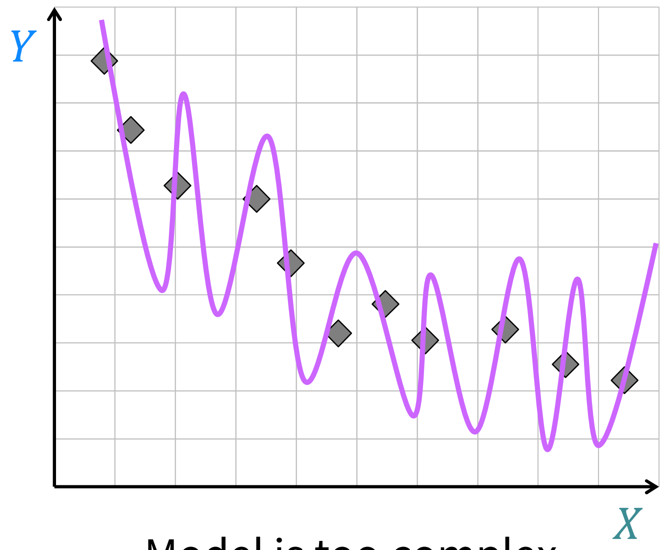
- **Model training is about loss minimization function. Mathematically optimal solution has loss equal to zero. Does this mathematically optimal solution imply an ‘optimal’ model?**



Model is too simple
(**underfitting**)



Model is just right
(**good solution**)

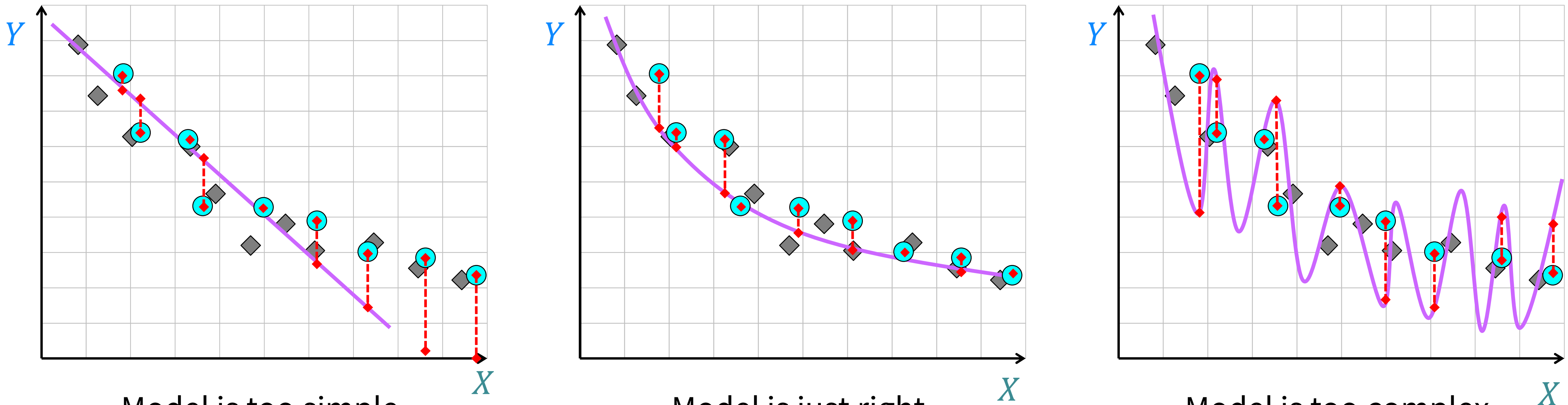


Model is too complex
(**overfitting**)

The Fundamental Problem of Overfitting

Advanced ML algorithms may overfit the training data

- **Model training minimizes a loss function. Mathematically optimal solution has loss equal to zero. Does this mathematically optimal solution imply an 'optimal' model?**
- **No! A model with zero training loss is too specific. It has picked up random noise that only exists in the training set and will show high forecast error on novel data**



Model is too simple (**underfitting**)

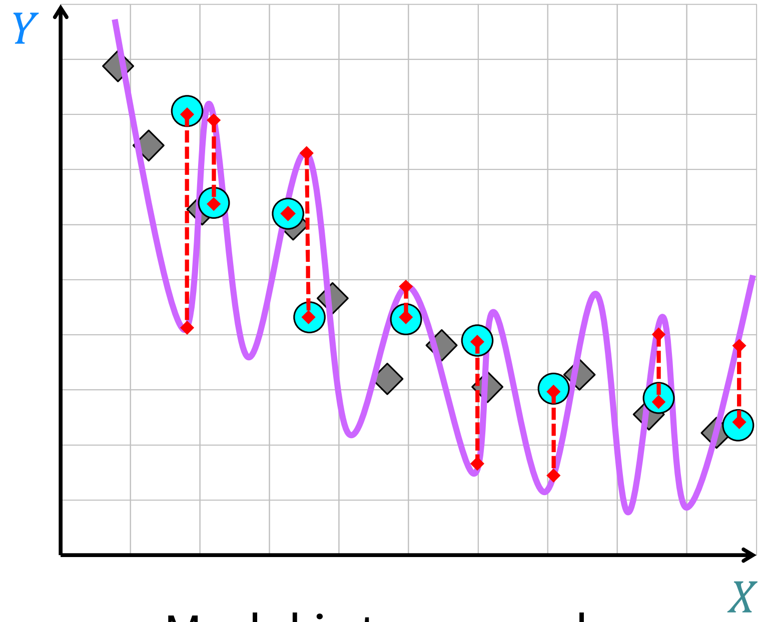
Model is just right (**good solution**)

Model is too complex (**overfitting**)

The Fundamental Problem of Overfitting

Advanced ML algorithms may overfit the training data

- **Training data is a sample**
  - We assume the same is representative of the population
  - Sample comprises actual structure
    - How inputs and outputs related to another
    - Feature-to-target relationship
  - Sample also comprises random variation
- **Zero loss during model training**
  - A perfect solution only if the inputs facilitate perfect prediction of the target
  - More likely scenario: the learning algorithm was fooled by the random variation in the training sample
  - Learnt model will then also embody that randomness
  - The model will perform poorly when applied to novel data

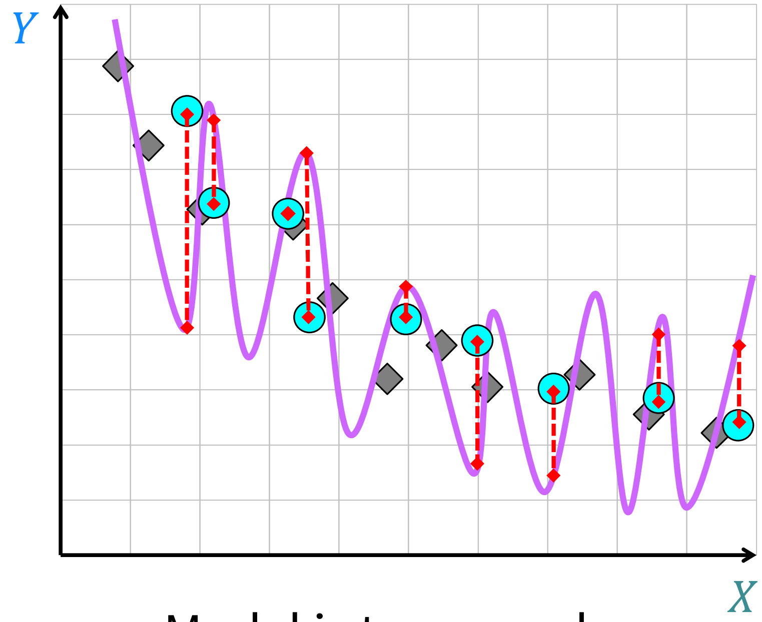


Model is too complex (**overfitting**)

The Fundamental Problem of Overfitting

Detecting overfitting issues by split-sampling / cross-validation

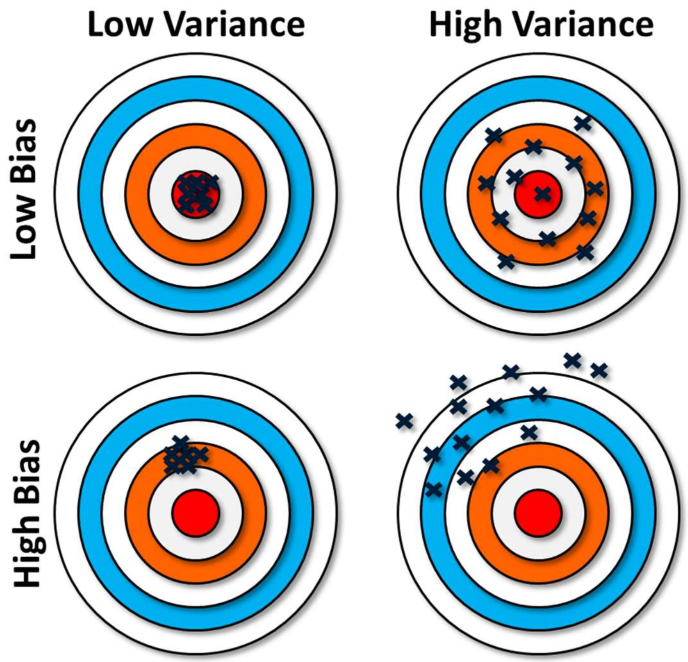
- **Recall idea of hold-out validation from last session**
  - Split data randomly into training and test set
  - Estimate model using training data
  - Assess model using test data
  - Perhaps repeat random splitting / use cross-validation
- **Overfitting implies a large difference in model performance on training versus test data**
- **Note that detecting overfitting, while crucial, does not tell you how to improve the model**
  - Change learning algorithm or its configuration
  - Regularization, early-stopping, ensembling, …
  - We learn about these approaches later



Model is too complex (**overfitting**)

The Trade-Off Between Bias and Variance

- **We can show that the generalization error of a model is a function to two 'evils'**
  - Generalization error means the error on data in general
  - Not the error you can measure on the training set
- **Bias**
  - Can the model approximate the true relationship between features and the target?
  - Refers to the expressive power of a learning algorithm
  - The more complex a model the lower its bias
- **Variance**
  - Think of it as sensitivity of a model to data
  - How much will forecasts vary with small changes in features?
  - How much will the model change with small changes of the training data?



The Trade-Off Between Bias and Variance

Generalization error is a function to two 'evils'

- **Let** $Y = f(X) + \epsilon$**, with** $\epsilon \sim \mathcal{N}(0, \sigma^2)$
- **Generalization error**

  $E_{\mathcal{D},\epsilon}\left[\left(Y - \hat{f}(X,\mathcal{D})\right)^2\right]$
- **Bias**

  $\text{Bias}_{\mathcal{D}}[\hat{f}(X,\mathcal{D})] = E_{\mathcal{D}}[\hat{f}(X,\mathcal{D})] - f(X)$
- **Variance**

  $\text{Var}_{\mathcal{D}}[\hat{f}(X,\mathcal{D})] = E_{\mathcal{D}}\left[\left(E_{\mathcal{D}}[\hat{f}(X,\mathcal{D})] - \hat{f}(X,\mathcal{D})\right)^2\right]$
- **Expectation taken over different training sets** $\mathcal{D} = \{(X_1, Y_1), \dots, (X_n, Y_n)\}$ **sampled from the same joint distribution** $P(X, Y)$
- **Bias-variance decomposition of the mean-squared error**

$$E_{\mathcal{D},\epsilon}\left[\left(Y - \hat{f}(X,\mathcal{D})\right)^2\right] = \left(\text{Bias}_{\mathcal{D}}[\hat{f}(X,\mathcal{D})]\right)^2 + \text{Var}_{\mathcal{D}}[\hat{f}(X,\mathcal{D})] + \epsilon$$

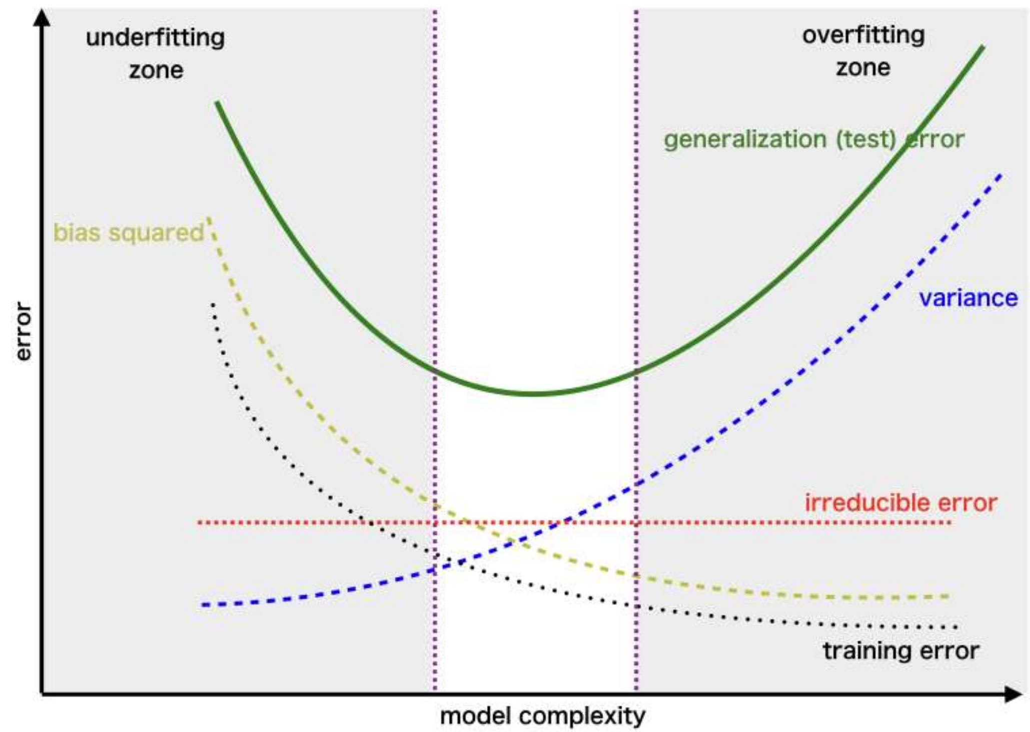


Image source: G. Papachristoudis (2019): https://towardsdatascience.com/the-bias-variance-tradeoff-8818f41e39e9

Bias-Variance Trade-Off and Overfitting

- **Simple classifiers**
  - High bias
  - Low variance
- **Complex classifiers**
  - Low bias
  - High variance
- **Much of supervised ML is about finding a good compromise**
- **Common paradigm**
  - Use an advanced, complex model
  - Manage / control complexity somehow

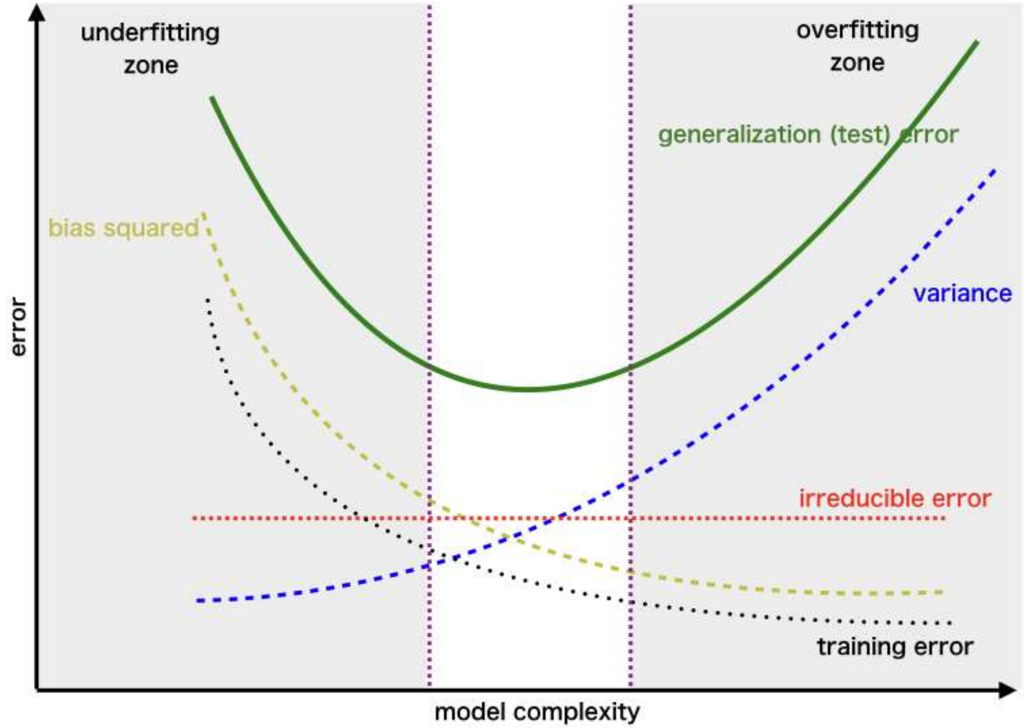


Image source: G. Papachristoudis (2019): https://towardsdatascience.com/the-bias-variance-tradeoff-8818f41e39e9

Regularization

Idea, connection to bias-variance trade-off, implementation

Regularization

- **Regularization revises practices to estimate models**
  - Do not focus on training error alone rather balance between two conflicting objectives
  - Goal 1: **low training error (i.e., low bias)**
  - Goal 2: **low complexity (i.e., low variance)**
- **Complex prediction models …**
  - Display low bias but high variance
  - Are prone to overfit the training set
- **Introducing bias can, therefore, …**
  - Help prevent overfitting
  - Reduce generalization error
- **Regularization involves …**
  - Penalizing model complexity
  - Introducing bias to decrease variance, and the generalization error

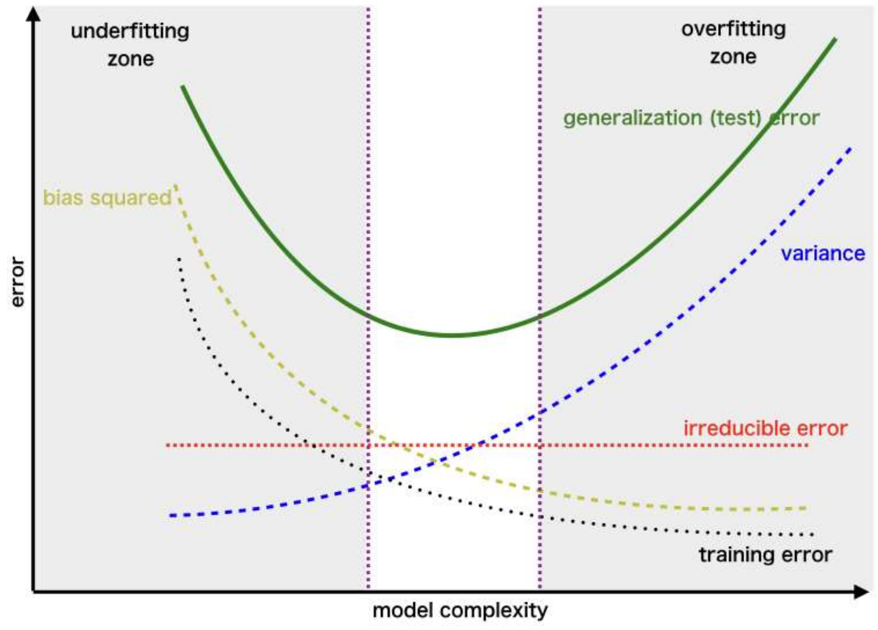
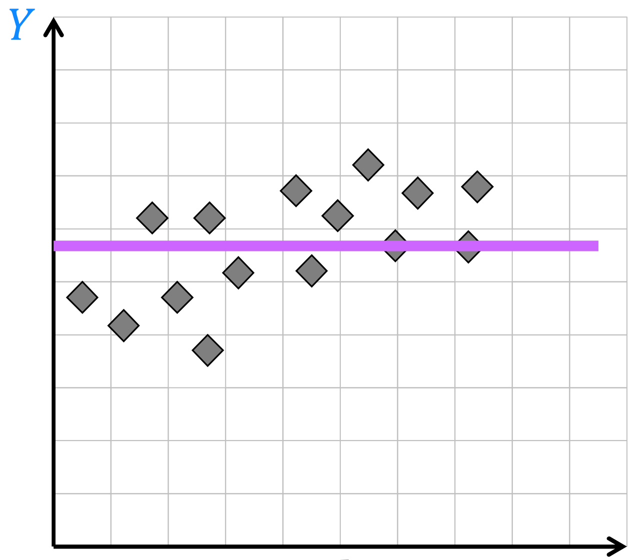


Image source: G. Papachristoudis (2019): https://towardsdatascience.com/the-bias-variance-tradeoff-8818f41e39e9

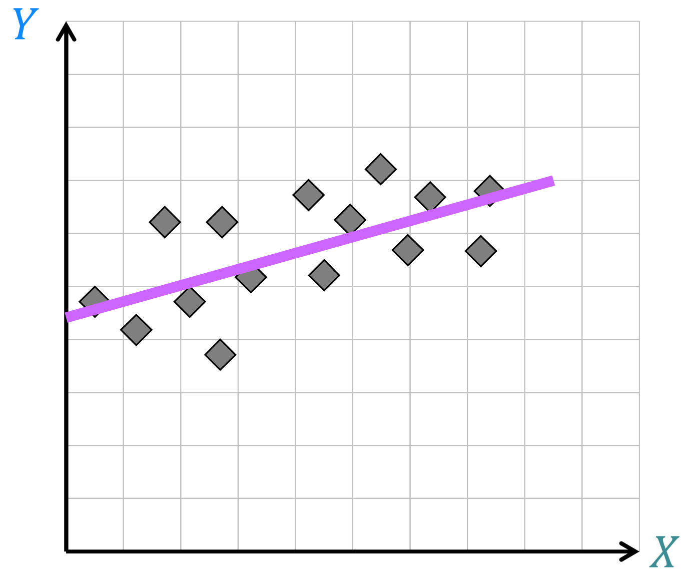
Measuring Model Complexity

Motivating example for regression models

- **Approaches toward measuring complexity vary across prediction models**
- **Consider for example univariate linear regression**



$Y = b$



$Y = b + w_1 X$

Which model is simpler?

Implementing Regularization

Two common complexity penalties for regression-type models

- **LASSO penalty** $L_1(\boldsymbol{w}) = \sum_{j=1}^{m} |w_j|$
- **Ridge penalty** $L_2(\boldsymbol{w}) = \sum_{j=1}^{m} w_j^2$
- **Considerations on penalty choice**
  - LASSO complicates model estimation but gives sparser models
  - Ridge imposes stronger penalty on (very) large coefficients
- **Elastic net penalty** $L_{enet}(\boldsymbol{w}) = \frac{1-\alpha}{2} \sum_{j=1}^{m} w_j^2 + \alpha \sum_{j=1}^{m} |w_j|$
  - With α a mixing parameter between ridge ($\alpha = 0$) and LASSO ($\alpha = 1$)
  - Needs additional tuning by the modeler

Implementing Regularization
Logistic regression revisited

- **Model formulation: model log-odds ratio as linear function of the features**

$$\log\left(\frac{p(Y=1|\boldsymbol{X})}{1-p(Y=1|\boldsymbol{X})}\right) = b + \sum_{j=1}^{m} w_j X_j$$

- **Loss function: negative of the log-likelihood function**

$$\mathcal{L}(\boldsymbol{w}) = -\left(\sum_{i=1}^{n} Y_i \log\big(p(Y_i = 1|\boldsymbol{X}_i)\big) + (1 - Y_i)\log\big(1 - p(Y_i = 1|\boldsymbol{X}_i)\big)\right)$$

- **Model estimation: minimize the loss function with respect to coefficients**

$$\widehat{\boldsymbol{w}} \leftarrow \min_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w})$$

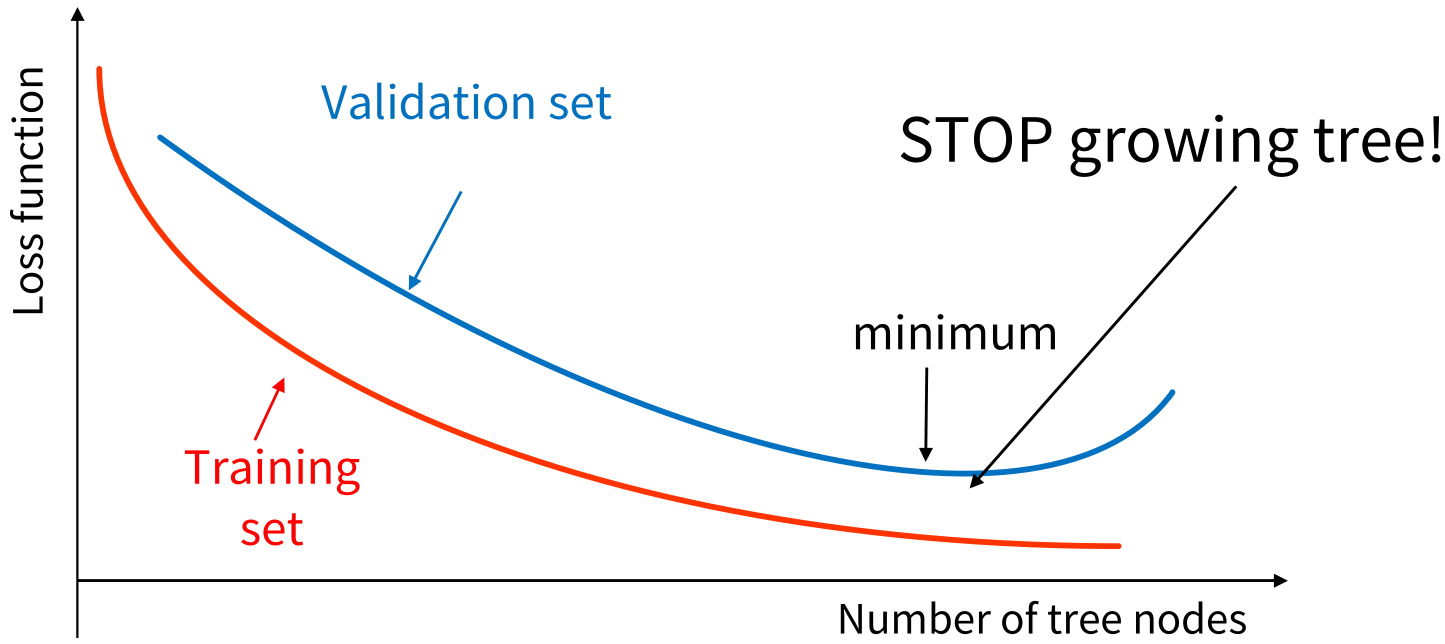
Regularized logistic regression
Extension of the loss function through adding a penalty term

- **Regularized logistic regression with ridge penalty**
  - Loss function with ridge penalty $\mathcal{L}^{ridge}(\boldsymbol{w}) = \mathcal{L}(\boldsymbol{w}) + \lambda\, L_2(\boldsymbol{w})$
  - $\widehat{\boldsymbol{\beta}}^{ridge} \leftarrow \min\left\{-\left(\sum_{i=1}^{n} y_i \log\left(p(y_i = 1|\boldsymbol{x}_i)\right) + (1 - y_i)\log\left(1 - p(y_i = 1|\boldsymbol{x}_i)\right)\right) + \lambda \sum_{j=1}^{m} w_j^2\right\}$
- **Regularized logistic regression with LASSO penalty**
  - Loss function with LASSO penalty $\mathcal{L}^{lasso}(\boldsymbol{w}) = \mathcal{L}(\boldsymbol{w}) + \lambda\, L_1(\boldsymbol{w})$
  - $\widehat{\boldsymbol{\beta}}^{lasso} \leftarrow \min\left\{-\left(Y_i \log\left(p(Y_i = 1|\boldsymbol{X}_i)\right) + (1 - Y_i)\log\left(1 - p(Y_i = 1|\boldsymbol{X}_i)\right)\right) + \lambda \sum_{j=1}^{m} |w_j|\right\}$
- **Additional (meta-)parameter λ controls the degree of regularization**
  - $\lambda \to \infty$ ➔ all elements of $\boldsymbol{w}$ will be zero
  - $\lambda \to 0$ ➔ recovers original logistic regression
- **Finding suitable settings for λ requires tuning (see model selection)**

Decision Tree Pruning Revisited

Pruning as a form of regularization

- **Complexity of a tree-based model often measured as number of terminal nodes**
- **Tree pruning is, therefore, also a form of regularization**



Model Selection

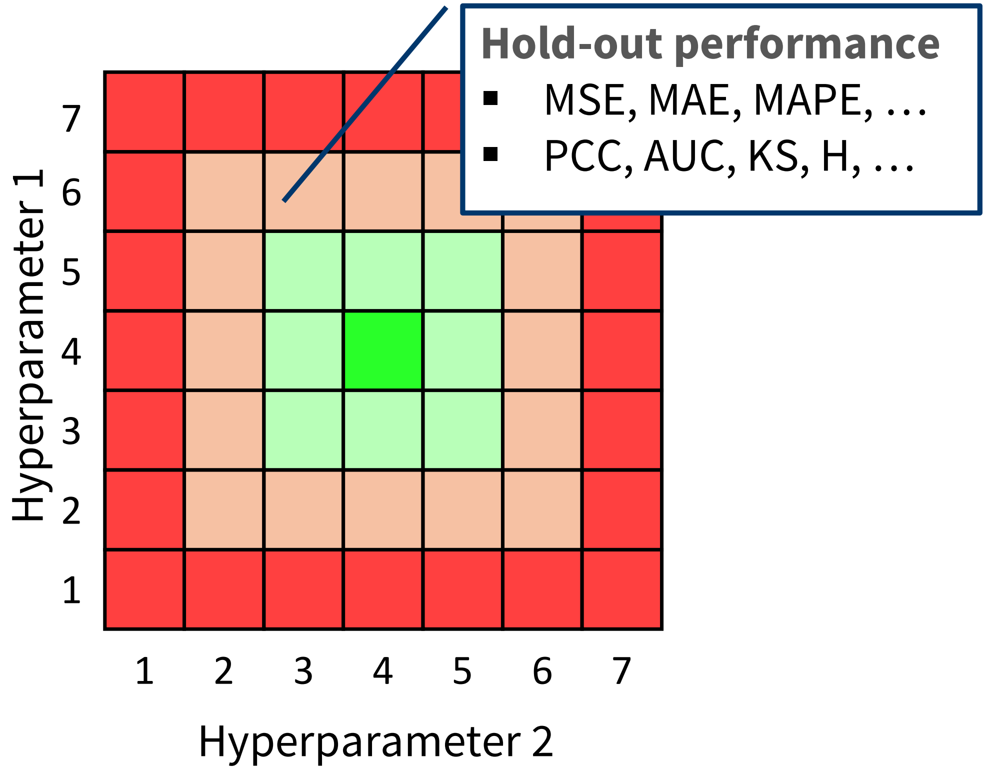
Search strategies and process perspective

Model Selection
Tuning of algorithmic hyperparameters

- **Advanced classifiers offer hyperparameters (also called meta-parameters)**
  - Facilitate adapting the classifier to a given data set
  - Need to be set by the data scientist
- **Similar to feature selection (in regression modeling)**
  - Manually decide which features to use in a model
  - Try out candidate settings using heuristic search (forward/backward, stagewise regression)
- **How to take corresponding decisions?**
  - Default settings / rules of thumb (not a good idea!)
  - Experience (may work, may fail as well)
  - Empirically, in a model selection process (common practice)

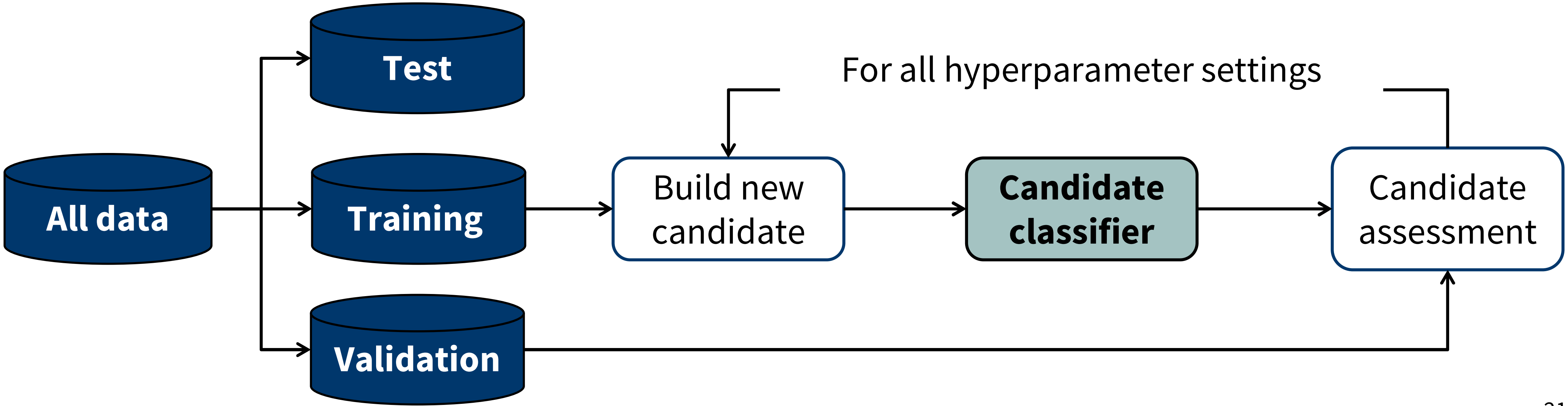
Grid Search
A versatile approach toward model selection

- **Fully enumerative search through all possible combinations of candidate hyperparameter settings**
- **Algorithm**
  - Define candidate range for each hyperparameter
  - Enumerate combinations of candidate values
  - Train model with given configuration
  - Assess model performance on hold-out data
  - Repeat with next configuration
- **Magnify grid resolution in promising regions of the search space**



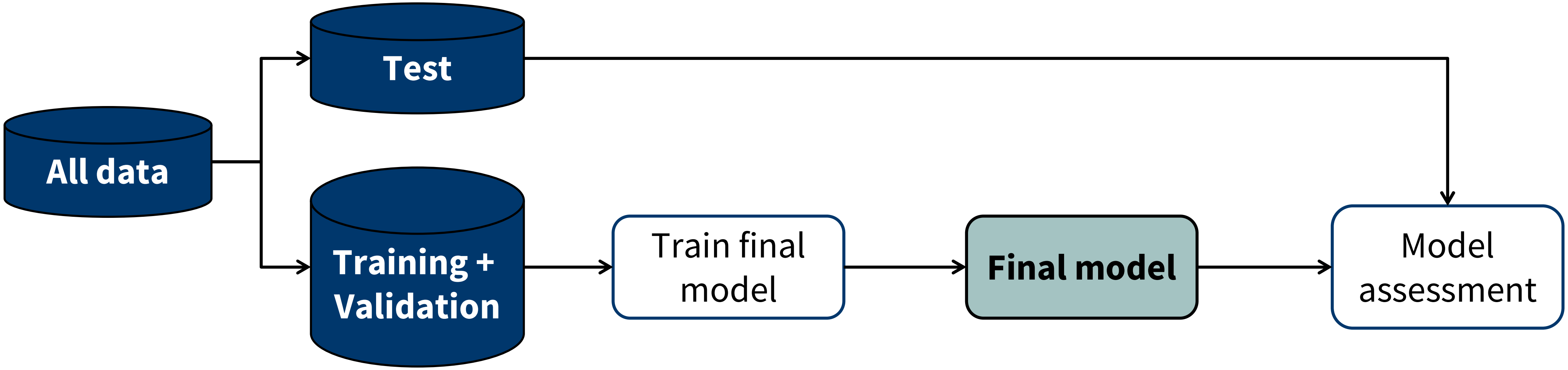
Model Selection Process

- **Additional modeling step to tune hyperparameters**
  - Rules of accuracy assessment apply to model selection
  - Need 'fresh' set of hold-out data to assess candidate models with different hyperparameters
- **Generalization of the split-sample approach**
- **Can also involve cross-validation**



Model Selection Process (cont.)

- **Identify best hyperparameter values**
- **Build final classifier with best hyperparameters**
  - No need for auxiliary validation data anymore
  - Can train on the union of training and validation sample



Model Selection Process (cont.)
A note on computational efficiency

- **Model selection is costly**
  - Iterative estimation of different candidate models
  - As many as candidate hyperparameter values in grid-search
  - Potentially more if using cross-validation
  - Careful exploration of parameter space computationally challenging
- **Practical recommendation**
  - Check whether you reduce the among of data during model selection
  - Does the best hyperparameters depend on the size of the training sample?
  - If not (aggressively) down-sample the training set, determine best hyperparameters, and build a model with best hyperparameters on the full training set can give a major speed-up
  - Can start from a learning curve analysis (Perlich et al., 2003) to determine how much down-sampling is possible

Summary

Summary

Learning goals	▪ Understand overfitting problem ▪ and its connection to bias and variance
Findings	▪ Detect overfitting by comparing training to test error ▪ Complex models display low bias and high variance ▪ Regularization introduces bias to decrease variance ▪ Implementing regularization through penalties ▪ Model selection for tuning algorithmic hyperparameters including regularization penalty
What next	▪ Demo notebook on model selection ▪ XMAS Break

Literature

- Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization Journal of Machine Learning Research, 13, 281-305.
- Perlich, C., Provost, F., Simonoff, J. S., & Cohen, W. W. (2003). Tree induction vs. logistic regression: A learning-curve analysis. Journal of Machine Learning Research, 4(2), 211-255.

Thank you for your attention!

Stefan Lessmann

Chair of Information Systems
School of Business and Economics
Humboldt-University of Berlin, Germany

Tel. +49.30.2093.5742
Fax. +49.30.2093.5741

stefan.lessmann@hu-berlin.de
http://bit.ly/hu-wi

www.hu-berlin.de

Appendix

Further considerations related to model selection

Model Selection

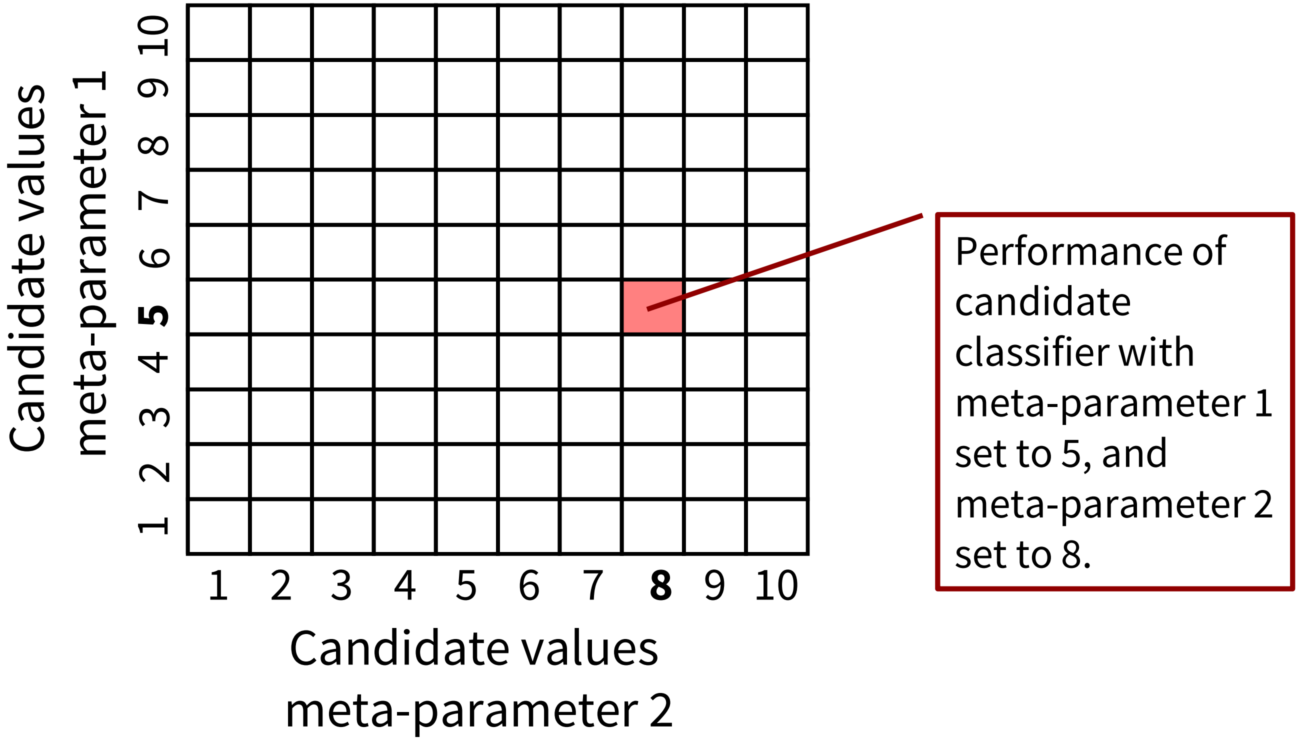
Learning algorithms may exhibit many hyperparameters

- **Regularized logistic regression**
  - Regularization coefficient
  - Two coefficients for elastic net penalty
- **Decision trees**
  - Splitting criterion
  - Max depth
  - Min observations per leaf
  - Magnitude of IG to continue splitting
  - Post-pruning
- **Tree based ensembles**
  - Hyperparameters of the base learner
  - Size of the ensemble
  - …
- **Neural networks**
  - Regularization coefficient, dropout rate
  - No. hidden layers
  - No. hidden nodes
  - Activation function
  - Learning rate, decay schedule
  - Solver
- **Support vector machines**
  - Regularization coefficient
  - Kernel function
  - Parameters of kernel function

Model Selection

Grid search: a versatile approach toward model selection

- **Three step approach**
  - For each meta-parameter,
  - define candidate settings
  - test combinations empirically
- **Example**
  - Two parameters
  - 10 candidate settings each
  - Grid search explores 10*10=100 value combinations



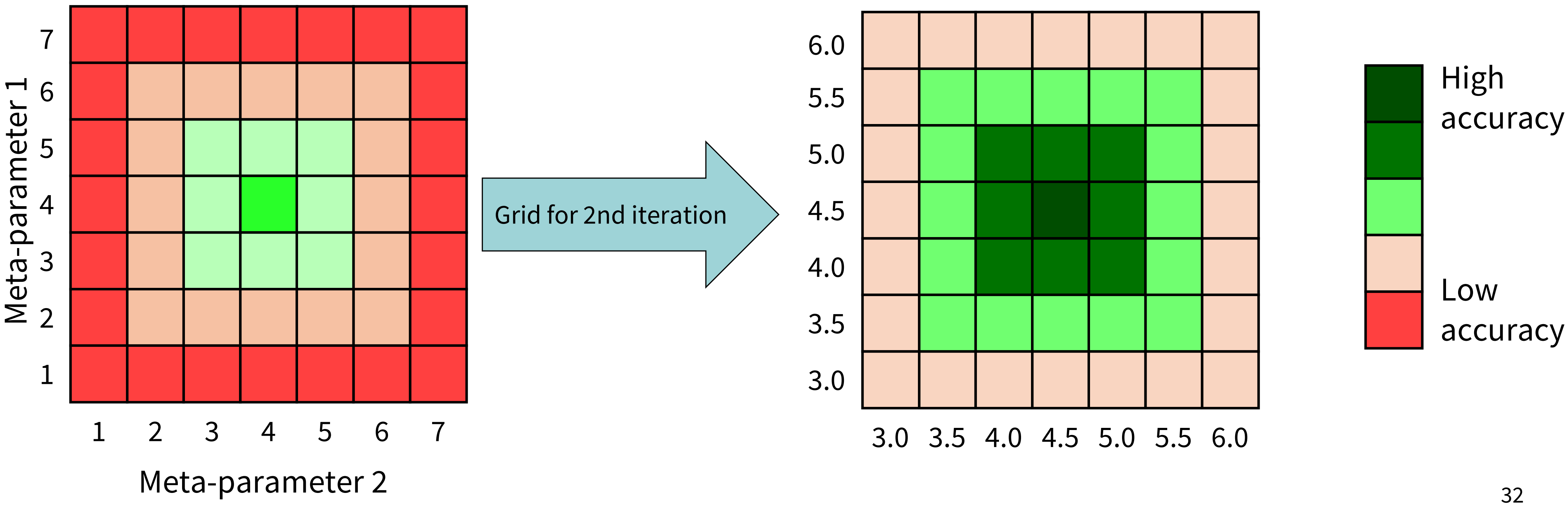
Model Selection
Some paper to learn about candidate parameter settings

- Caruana, R., Niculescu-Mizil, A., Crew, G., & Ksikes, A. (2004). Ensemble Selection from Libraries of Models. In C. E. Brodley (Ed.), *Proc. of the 21st Intern. Conf. on Machine Learning* (pp. 18-25). Banff, Alberta, Canada: ACM.
- Caruana, R., Munson, A., & Niculescu-Mizil, A. (2006). Getting the Most Out of Ensemble Selection. In *Proc. of the 6th Intern. Conf. on Data Mining* (pp. 828-833). Hong Kong, China: IEEE Computer Society.
- Dejaeger, K., Verbeke, W., Martens, D., & Baesens, B. (2011). Data mining techniques for software effort estimation: A comparative study. *IEEE Transactions on Software Engineering, 38*, 375-397.
- Hsu, C.-W., Chang, C.-C., & Lin, C.-J. (2003). A Practical Guide to Support Vector Classification. In. Taiwan: Department of Computer Science and Information Engineering, National Taiwan University.
- S. Lessmann, M.C. Sung, J.E. Johnson, T. Ma, A new methodology for generating and combining statistical forecasting models to enhance competitive event prediction, *European Journal of Operational Research*, 218(1) (2012) 163-174.
- S. Lessmann, S. Voss, Customer-centric decision support: A benchmarking study of novel versus established classification models, *Business and Information System Engineering*, 2(2) (2010) 79-93.
- S. Lessmann, B. Baesens, C. Mues, S. Pietsch, Benchmarking classification models for software defect prediction: A proposed framework and novel findings, *IEEE Transactions on Software Engineering*, 34(4) (2008) 485-496.
- Lessmann, S., Baesens, B., Seow, H.-V., & Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research, 247*, 124-136.
- Lessmann, S., Haupt, J., Coussement, K., & De Bock, K. W. (2019). Targeting customers for profit: An ensemble learning framework to support marketing decision-making. Information Sciences, (doi:10.1016/j.ins.2019.05.027).
- Loterman, G., Brown, I., Martens, D., Mues, C., & Baesens, B. (2012). Benchmarking regression algorithms for loss given default modeling. *International Journal of Forecasting, 28*, 161-170.
- Partalas, I., Tsoumakas, G., & Vlahavas, I. (2010). An ensemble uncertainty aware measure for directed hill climbing ensemble pruning. *Machine Learning, 81*, 257-282.
- Van Gestel, T., Suykens, J. A. K., Baesens, B., Viaene, S., Vanthienen, J., Dedene, G., De Moor, B., & Vandewalle, J. (2004). Benchmarking least squares support vector machine classifiers. *Machine Learning, 54*, 5-32
- Verbeke, W., Dejaeger, K., Martens, D., Hur, J., & Baesens, B. (2012). New insights into churn prediction in the telecommunication sector: A profit driven data mining approach. *European Journal of Operational Research, 218*, 211-229.

Repeated Grid Search

Repeat grid-search zooming in on promising search areas

- **Magnify resolution of candidate settings in promising areas**
- **Consider two to three iterations and/or trace degree of improvement**



Model Selection Approaches Beyond Grid Search

- **Other search strategies**
  - Random search (popular for deep neural networks, see Bergstra & Bengio, 2012)
  - Meta-heuristics and evolutionary algorithms (genetic algorithms, evolution strategies, particle swarm optimization, harmony search, etc.)
- **Promise autonomous, self-adaptive hyperparameter tuning**
  - Do not require candidate settings for prediction model hyperparameters to be defined
  - But what about the parameters of the search strategy ???
- **Practical recommendation**
  - Using an advanced search strategy, you trade one tuning problem for another
  - Availability in software packages might also be an issue
  - In most cases, grid search will work well

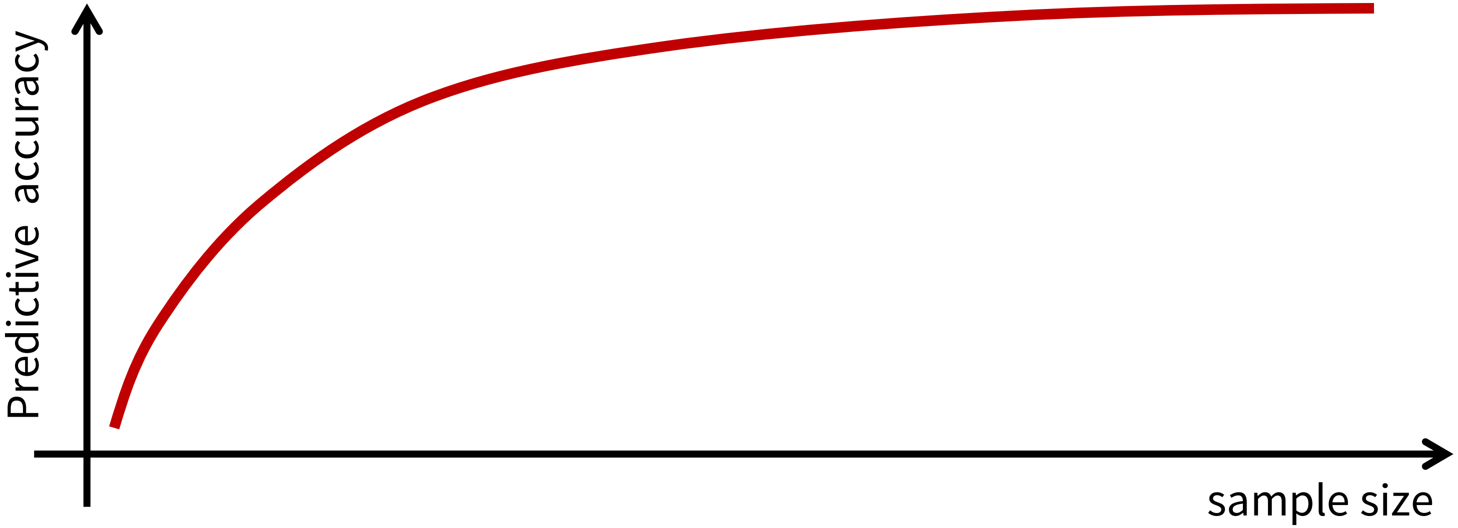
Model Selection Efficiency

- **Model selection is costly**
  - Iterative estimation of different candidate models
  - As many as candidate hyperparameter values in grid-search
  - Careful exploration of parameter space challenging
- **Approaches to increase efficiency**
  - Algorithmic specific strategies (much work on support vector machines; Lessmann & Voß, 2009)
  - Generic approaches
- **Practical recommendation**
  - Consider learning curve-based heuristic
  - Learning curve analysis tells you how much data is needed
  - Carry out model selection with this amount of data might give substantial speed-up
  - Following slides detail this idea

Learning Curve Analysis

Examines the sensitivity of a model regarding training data size

- **How much data is needed or what is the marginal value of more data**
- **Three-step approach**
  - Draw small sample from your data
  - Estimate and assess model (e.g., split-sample method)
  - Increase samples size and repeat
- **Perlich et al. (2003)**
- **Learning curve will often display a degressive trend**
  - Marginal value of data diminishes
  - Curve offers insight when training has stabilized



Learning Curve Analysis & Model Selection
A heuristic to increase the efficiency of model selection

- **Assumption**
  - Hyperparameter efficacy does not depend on sample size
  - Relaxation: Moderate dependence is still ok
- **Perform learning curve analysis with default hyperparameter values**
- **Find sample size where classifier training stabilizes**
- **Perform model selection using that sample size**
- **Can give substantial speed-up**
  - For many classifiers, training time increases exponentially with sample size
  - Small reduction in sample size facilitates notable speed-up

Learning Curve Analysis & Model Selection

A heuristic to increase the efficiency of model selection