*** Applied Machine Learning Fundamentals *** Evaluation of ML Models

Daniel Wehner, M.Sc.

SAP SE / DHBW Mannheim

Winter term 2023/2024

Find all slides on GitHub (DaWe1992/Applied_ML_Fundamentals)

Lecture Overview

Agenda for this Unit

Section:

Evaluation Methods and Data Splits

Evaluation of trained Models

1. **Validation through experts:** A domain expert checks the plausibility
   - Subjective, time-intensive, and costly
   - Often the only option
2. **Validation on data:** Evaluate the performance on a **separate (!)** test set
   - Labeled data is scarce and could be better used for training
   - Fast and simple, no domain knowledge needed
3. **On-line validation:** Test the model in a field test
   - Bad models may be costly (e. g. autonomous driving)
   - Gives the best estimate for the overall utility

Out-of-Sample Testing

- The performance cannot be measured on the training data (why?)
- Usually, a portion of the available data is reserved for testing
    - $^2/_3$ for training, $^1/_3$ for testing (evaluation)
    - The model is trained on the training set and evaluated on the test set
- **Problems**:
    - Waste of data
    - Labeling may be expensive
- **Solution**: Cross-Validation (X-Val)

Important

Three Splits: Train, Dev/Validation, Test

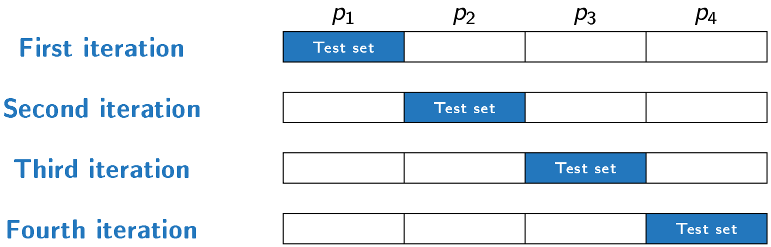
In practice it is also common to split the data into three portions:

Stratified splits have the same class distribution as the entire dataset

1. **Training set** (used for training as before)
2. **Dev/Validation set**
   - Used for hyper-parameter tuning of the model
   - Using the test set for that would be cheating
3. **Test set**
   - The final model is tested on the test set
   - The test set is used to estimate the **generalization error**

Cross-Validation (k-fold X-Val)

- Split the dataset into k equally sized partitions $P = \{p_1, p_2, \ldots, p_k\}$
- For each partition p_i: Use $P \setminus \{p_i\}$ for training and p_i for testing
- Average the results



Leave-One-Out Cross-Validation (LOO X-Val)

- *n*-fold X-Val
  - *n* is the number of examples
  - Use $n-1$ examples for training, one example for testing
- **Properties:**
  - Makes best use of the data
  - Very expensive for large datasets (large *n*)

Note: We get *k* trained models when using *k*-fold X-Val!

- **Which of these models is used in production?**
- **Answer:** None of them. X-Val is only used for error estimation. The final model is trained on the entire dataset

Section:
Evaluation Metrics for Classifiers

Confusion Matrices
Drawback of Accuracy
Precision, Recall and F1-Score
ROC and AUC

Types of Errors

- **Type I Error**: False negatives
  - An instance which is labeled $\oplus$ is classified as $\ominus$
  - E. g. a spam e-mail is not detected
- **Type II Error**: False positives
  - An instance which is labeled $\ominus$ is classified as $\oplus$
  - E. g. a non-spam (ham) e-mail is classified as spam

a. k. a. α/β error

Depending on the context the costs of false negatives and false positives can be different!

Confusion Matrices (two Classes)

- How often is class $\mathcal{C}_i$ confused with class $\mathcal{C}_j$?
- Calculate **accuracy**:

		Classified	
		$\oplus$	$\ominus$
Is	$\oplus$	**#tp**	#fn
	$\ominus$	#fp	**#tn**

$$accuracy = \frac{\#\mathbf{tp} + \#\mathbf{tn}}{\#\mathbf{tp} + \#\mathbf{tn} + \#\mathrm{fp} + \#\mathrm{fn}}$$

$$error = 1 - accuracy$$

tp	true positive
fn	false negative
fp	false positive
tn	true negative

Confusion Matrices (multiple Classes)

		Classified			
		A	B	C	Σ
Is	A	$n_{A,A}$	$n_{B,A}$	$n_{C,A}$	n_A
	B	$n_{A,B}$	$n_{B,B}$	$n_{C,B}$	n_B
	C	$n_{A,C}$	$n_{B,C}$	$n_{C,C}$	n_C
	Σ	$\overline{n_A}$	$\overline{n_B}$	$\overline{n_C}$	n

$$accuracy = \frac{n_{A,A} + n_{B,B} + n_{C,C}}{n}$$

Drawback of Accuracy

- Real-world datasets are usually **imbalanced**, i. e. some classes appear more frequently than others
- **Example**:
  - A dataset $\mathcal{D}$ contains two classes $\mathcal{C}_1$ and $\mathcal{C}_2$
  - $\mathcal{C}_1$ appears 99 % of the time, $\mathcal{C}_2$ only 1 % of the time
  - It is easy to reach 99 % accuracy by always predicting the majority class
  - **Is this useful?** *Probably not...*

We need some more sophisticated evaluation metrics!

Precision and Recall

Precision: Ratio of #tp to all instances predicted as $\oplus$

$$P := \frac{\#\text{tp}}{\#\text{tp} + \#\text{fp}} \tag{1}$$

Recall (Sensitivity): Ratio of #tp to all instances actually labeled as $\oplus$

$$R := \frac{\#\text{tp}}{\#\text{tp} + \#\text{fn}} \tag{2}$$

Precision-Recall Trade-Off

There is a trade-off between precision and recall:

It is very easy to get 100 % precision:

- Simply classify one instance as $\oplus$ where you are absolutely sure
- But recall is bad... *(many $\oplus$-instances are not detected)*

It is also quite easy to achieve 100 % recall:

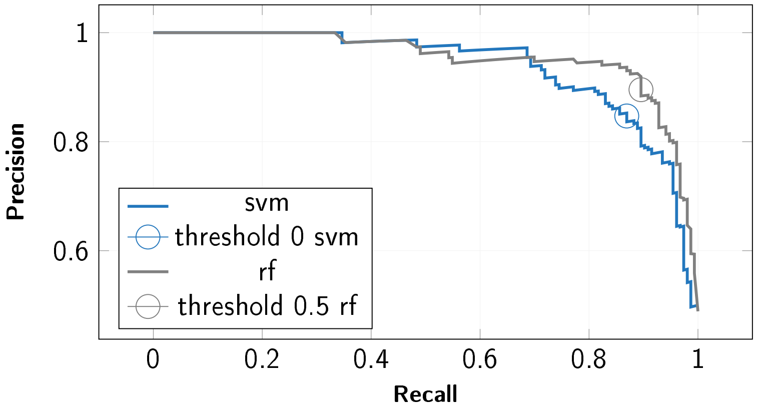
- Classify all instances as $\oplus$
- But precision is bad... *(many $\ominus$-instances are detected)*

Precision-Recall Curves / P-R-Curves

- Visualization of the precision-recall trade-off
- Influence precision and recall by changing thresholds
- **Example**:
  - Consider a ranker, e. g. a logistic regression classifier
  - It outputs probabilities for each class
  - The threshold when to predict $\oplus$ can be changed
  - This has an influence on precision and recall

A P-R-curve plots precision and recall for all possible thresholds.

Precision-Recall Curves / P-R-Curves (Ctd.)



Combining Precision and Recall: F1-Score

- When to use precision, when recall?
- This depends on the cost of fp and fn
  - If fp are expensive ⇒ **use precision!**
  - If fn are expensive ⇒ **use recall!**
- **F1-score** *(harmonic mean of precision and recall)*

Why the harmonic mean?

$$F_1 := \frac{2 \cdot P \cdot R}{P + R} \qquad F_\beta := (1 + \beta^2) \cdot \frac{P \cdot R}{(\beta^2 \cdot P) + R} \quad (\beta \in \mathbb{R}^+) \tag{3}$$

- Large β emphasizes recall

Calculation for multiple Classes (Example Precision)

- Precision must be calculated for each class separately
- For K classes we get K results. **How to combine the results?**
  - **Macro average:** Calculate P for each class and average the result

$$P_{macro} := \frac{1}{K} \sum_{k=1}^{K} P_k \tag{4}$$

  - **Micro average:** Sum #tp and #fp for all classes and calculate P

$$P_{micro} := \sum_{k=1}^{K} \#\mathrm{tp}_k / \sum_{k=1}^{K} (\#\mathrm{tp}_k + \#\mathrm{fp}_k) \tag{5}$$

Calculation for multiple Classes (Example Precision, Ctd.)

		Classified				
		A	B	C	D	Σ
Is	A	40	12	4	8	64
	B	7	51	2	0	60
	C	2	17	27	11	57
	D	39	4	15	8	66
	Σ	88	84	48	27	247

Calculation for multiple Classes (Example Precision, Ctd.)

$$P_A = \frac{40}{40 + 48} = 0.45 \qquad P_B = \frac{51}{51 + 33} = 0.61$$

$$P_C = \frac{27}{27 + 21} = 0.56 \qquad P_D = \frac{8}{19} = 0.30$$

$$P_{macro} = \frac{0.45 + 0.61 + 0.56 + 0.30}{4} = 0.48$$

$$P_{micro} = \frac{40 + 51 + 27 + 8}{(40 + 51 + 27 + 8) + (48 + 33 + 21 + 19)} = 0.51$$

ROC-Curves

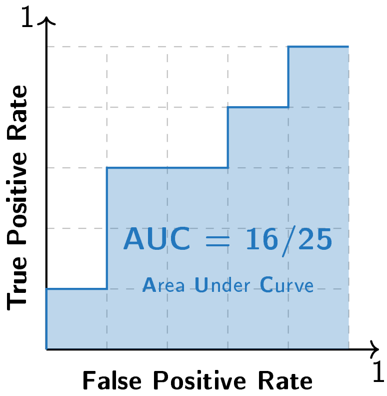
- ROC is short for **R**eceiver **O**perating **C**haracteristic
- Borrowed from signal theory *(hence the name)*
- Uses *true positive rate* (recall) and *false positive rate* $= \frac{\#\text{fp}}{\#\text{fp}+\#\text{tn}}$

General procedure:

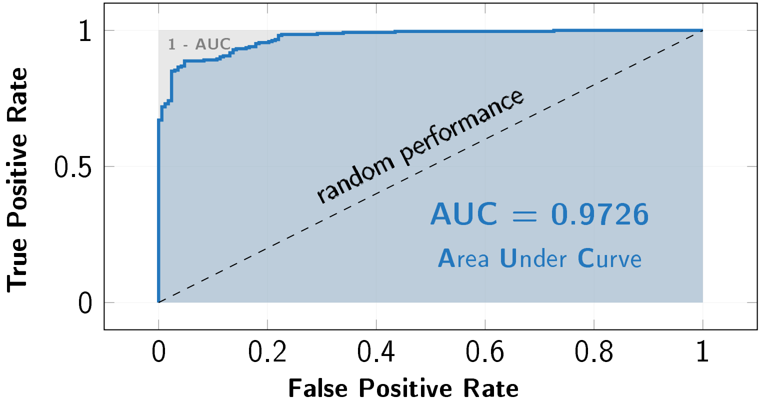
- Rank test instances by decreasing certainty of class $\oplus$
- Start at the origin $(0, 0)$
- If the next instance in the ranking is $\oplus$: move $^1/_{|\oplus|}$ up
- If the next instance in the ranking is $\ominus$: move $^1/_{|\ominus|}$ right

Sample ROC-Curve (Example I)

Rank	Prob.	True class
1	0.95	$\oplus$
2	0.85	$\ominus$
3	0.78	$\oplus$
4	0.75	$\oplus$
5	0.62	$\ominus$
6	0.41	$\ominus$
7	0.37	$\oplus$
8	0.22	$\ominus$
9	0.15	$\oplus$
10	0.05	$\ominus$



Sample ROC-Curve (Example II)



ROC-Curve Interpretation

- AUC can be interpreted as the probability of a positive example always being listed before a negative example
- A high AUC value entails a good class separation:

AUC = 1.0:	All $\oplus$ listed before all $\ominus$ (desiderata)
AUC = 0.5:	Random ordering (worst case)
AUC = 0.0:	All $\ominus$ listed before all $\oplus$ (not the worst case $\Rightarrow$ Invert classification)

Analogy: In a quiz you are allowed to answer those questions first where you feel the most confident (ranking). If you answer the first questions wrong, you won't perform very well overall $\Rightarrow$ **small AUC**.

Section:
Evaluation Metrics for Regressors

R^2, RMSE and MAE
An Example

R^2, RMSE and MAE

- **Coefficient of determination R^2:**

$$R^2 := \frac{\sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - \overline{y}\right)^2}{\sum_{i=1}^{n} \left(y^{(i)} - \overline{y}\right)^2} = \frac{\text{Variance explained by model}}{\text{Total variance}} \qquad R^2 \in [0, 1] \tag{6}$$

- **Root mean square error (RMSE):**

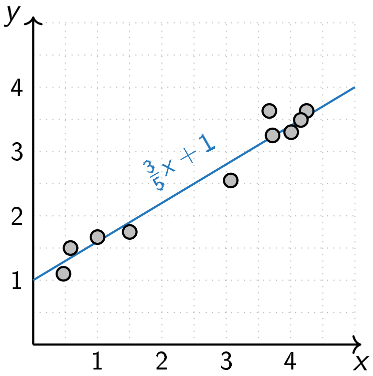
$$RMSE := \left(\frac{1}{n} \cdot \sum_{i=1}^{n} \left(h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}\right)^2\right)^{1/2} \tag{7}$$

- **Mean absolute error (MAE):**

$$MAE := \frac{1}{n} \cdot \sum_{i=1}^{n} \left|h_{\boldsymbol{\theta}}(\boldsymbol{x}^{(i)}) - y^{(i)}\right| \tag{8}$$

Evaluation of Regressors (Ctd.)

$x^{(i)}$	$y^{(i)}$	$h_\theta(x^{(i)})$
0.47	1.10	1.28
0.58	1.50	1.35
1.00	1.67	1.60
1.50	1.75	1.90
3.07	2.55	2.84
3.67	3.63	3.20
3.72	3.25	3.23
4.01	3.30	3.41
4.16	3.49	3.50
4.25	3.63	3.55
$\overline{y} = 2.59$		



Evaluation of Regressors (Ctd.)

- Coefficient of determination:

$$R^2 = \frac{(1.28 - 2.59)^2 + \cdots + (3.55 - 2.59)^2}{(1.10 - 2.59)^2 + \cdots + (3.63 - 2.59)^2} = \frac{7.97}{8.89} = 0.90 \tag{9}$$

- Root mean square error:

$$RMSE = \left(\frac{1}{10} \cdot [(1.28 - 1.10)^2 + \cdots + (3.55 - 3.63)^2]\right)^{1/2} = 0.19 \tag{10}$$

- Mean absolute error:

$$MAE = \frac{1}{10} \cdot (|1.28 - 1.10| + \cdots + |3.55 - 3.63|) = 0.15 \tag{11}$$

Section:

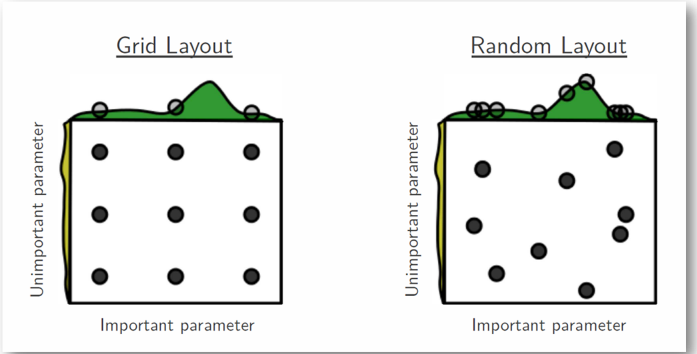
Model Selection and Model Complexity

Hyper-Parameter Tuning: Grid Search and Random Search
Bias and Variance

Hyper-Parameter Tuning with Grid Search

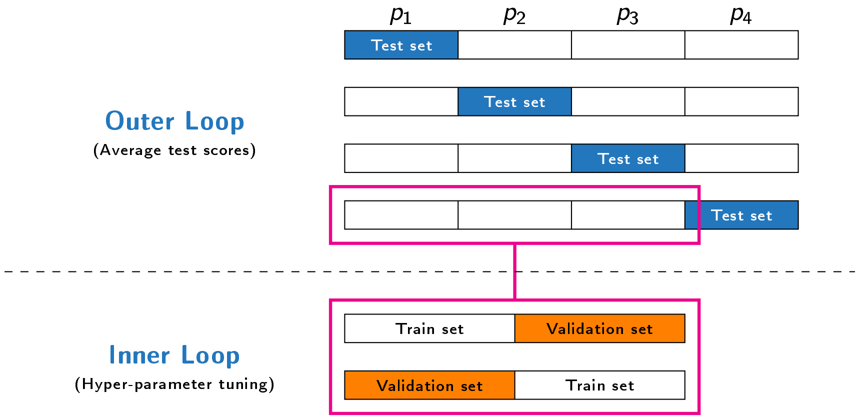
- **Grid search** is applied to find **optimal hyper-parameter settings**
- Hyper-parameter tuning should be done on the **dev** set
- We have to specify the search space / ranges of hyper-parameter values
- Grid search will try **all combinations** to find the best model
  - **Computationally very expensive**
  - Scikit-learn provides parameters to parallelize the search (`n_jobs=-1` $\Rightarrow$ use all cores available)
  - May not find the optimal setting $\Rightarrow$ **Random search**

Grid Search vs. random Search



cf. Bergstra/Bengio.2012, `https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf`, page 284

Nested Cross-Validation



Bias-Variance Decomposition for MSE

- Suppose that we have a training set $\mathcal{D} = \left\{(x^{(1)}, y^{(1)}), \dots, (x^{(n)}, y^{(n)})\right\}$
- There is an **unknown** function $f(x)$ such that

$$y^{(i)} = f\left(x^{(i)}\right) + \varepsilon \tag{12}$$

- The noise term ε has zero mean and variance σ_ε^2: $\mathbb{E}\{\varepsilon\} = 0, \mathbb{V}\{\varepsilon\} = \sigma_\varepsilon^2$
- Based on the dataset $\mathcal{D}$ we want to find a function $\widehat{f}(x; \mathcal{D})$ which approximates $f(x)$ 'as well as possible'
- For this we try to minimize the mean squared error $\mathbb{E}\left\{\left(y^{(i)} - \widehat{f}(x^{(i)})\right)^2\right\}$

Bias-Variance Decomposition for MSE (Ctd.)

- We can decompose the **expected error** of $\widehat{f}$ on an unseen sample x:

Bias-Variance decomposition of the mean squared error:

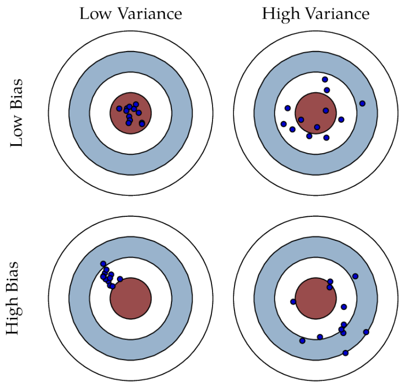
$$\mathbb{E}_{\mathcal{D},\varepsilon}\left\{\left(y-\widehat{f}(x;\mathcal{D})\right)^2\right\} = \mathbb{B}^2_{\mathcal{D}}\{\widehat{f}(x;\mathcal{D})\} + \mathbb{V}_{\mathcal{D}}\{\widehat{f}(x,\mathcal{D})\} + \sigma^2_\varepsilon \quad (13)$$

- $\mathbb{B}_{\mathcal{D}}$ is the bias, and $\mathbb{V}_{\mathcal{D}}$ the variance of the model $\widehat{f}$
- σ^2_ε is **irreducible**

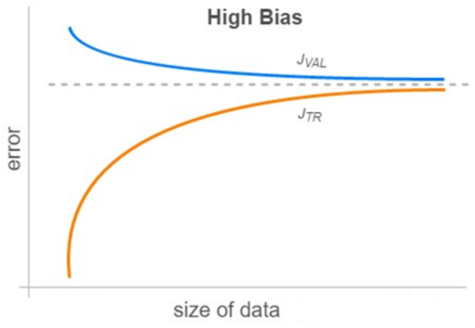
Bias and Variance

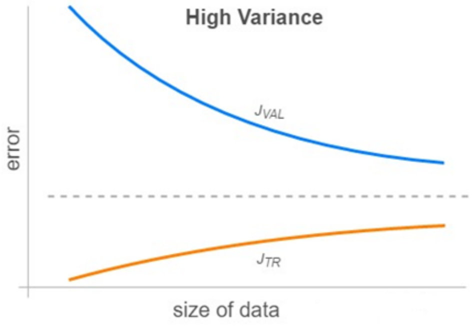
The **bias** results from erroneous assumptions in the learning algorithm. High bias can cause an algorithm to miss the relevant relations between features and target outputs **(underfitting)**.

The **variance** is an error from sensitivity to small fluctuations in the training set. High variance may result from an algorithm modeling the random noise in the training data **(overfitting)**.



Bias and Variance (Ctd.)





Recall: Expectation and Variance

- Let $\mathcal{X}$ be a random variable. $\Omega(\mathcal{X})$ is the domain of $\mathcal{X}$ (set of possible values $\mathcal{X}$ can take)
- Definition of **expectation** (discrete case):

$$\mathbb{E}\{\mathcal{X}\} := \sum_{k \in \Omega(\mathcal{X})} k \cdot p(\mathcal{X} = k) \tag{14}$$

- Definition of **variance** (discrete case):

$$\mathbb{V}\{\mathcal{X}\} := \sum_{k \in \Omega(\mathcal{X})} (k - \mathbb{E}\{\mathcal{X}\})^2 \cdot p(\mathcal{X} = k) \tag{15}$$

Recall: Expectation and Variance (Ctd.)

Let $\mathcal{X}$ and $\mathcal{Y}$ be random variables and $a, b \in \mathbb{R}$:

- **Linearity** of $\mathbb{E}$ (very important!):

$$\mathbb{E}\{a\mathcal{X} + b\mathcal{Y}\} = a\mathbb{E}\{\mathcal{X}\} + b\mathbb{E}\{\mathcal{Y}\} \tag{16}$$

- If $\mathcal{X}$ and $\mathcal{Y}$ are independent: $\mathbb{E}\{\mathcal{X}\mathcal{Y}\} = \mathbb{E}\{\mathcal{X}\}\mathbb{E}\{\mathcal{Y}\}$
- $\mathbb{V}\{\mathcal{X}\} = \mathbb{E}\{\mathcal{X} - \mathbb{E}^2\{\mathcal{X}\}\} = \mathbb{E}\{\mathcal{X}^2\} - \mathbb{E}^2\{\mathcal{X}\}$
- $\mathbb{V}$ is **not** linear: $\mathbb{V}\{a + b\mathcal{X}\} = b^2\mathbb{V}\{\mathcal{X}\}$
- However, if $\mathcal{X}$ and $\mathcal{Y}$ are uncorrelated: $\mathbb{V}\{\mathcal{X} + \mathcal{Y}\} = \mathbb{V}\{\mathcal{X}\} + \mathbb{V}\{\mathcal{Y}\}$

Derivation of the Bias-Variance Decomposition for MSE

- The MSE is given by

$$\begin{aligned}\text{MSE} := \mathbb{E}\{(y-\widehat{f})^2\} &= \mathbb{E}\{y^2 - 2y\widehat{f} + \widehat{f}^2\} \\ &= \underbrace{\mathbb{E}\{y^2\}}_{❶} - 2\,\underbrace{\mathbb{E}\{y\widehat{f}\}}_{❷} + \underbrace{\mathbb{E}\{\widehat{f}^2\}}_{❸} \qquad (17)\end{aligned}$$

- Term ❸ is straight-forward:

$$\begin{aligned}\mathbb{E}\{\widehat{f}^2\} &= \mathbb{E}\{\widehat{f}^2\} - \mathbb{E}^2\{\widehat{f}\} + \mathbb{E}^2\{\widehat{f}\} \\ &= \mathbb{V}\{\widehat{f}\} + \mathbb{E}^2\{\widehat{f}\} \qquad (18)\end{aligned}$$

Derivation of the Bias-Variance Decomposition for MSE (Ctd.)

- We rewrite term ➊:

$$\begin{aligned}
\mathbb{E}\{y^2\} &= \mathbb{E}\{(f+\varepsilon)^2\} && \text{Definition of } y \\
&= \mathbb{E}\{f^2\} + 2\mathbb{E}\{f\varepsilon\} + \mathbb{E}\{\varepsilon^2\} && \text{Linearity of } \mathbb{E} \\
&= f^2 + 2f\mathbb{E}\{\varepsilon\} + \mathbb{E}\{\varepsilon^2\} && f \text{ does not depend on } \mathcal{D} \\
&= f^2 + 2f \cdot 0 + \sigma_\varepsilon^2 && \mathbb{E}\{\varepsilon\} = 0 \qquad (19)
\end{aligned}$$

Derivation of the Bias-Variance Decomposition for MSE (Ctd.)

- We rewrite term ❷:

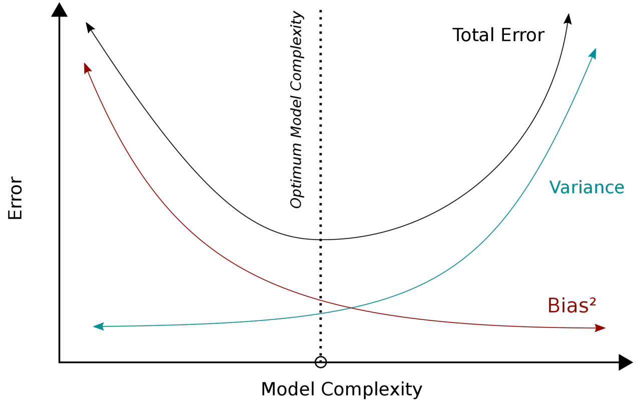
$$\begin{aligned}
\mathbb{E}\{y\widehat{f}\} &= \mathbb{E}\{(f+\varepsilon)\widehat{f}\} && \text{Definition of } y \\
&= f\mathbb{E}\{\widehat{f}\} + \mathbb{E}\{\varepsilon\widehat{f}\} && \text{Linearity of } \mathbb{E} \\
&= f\mathbb{E}\{\widehat{f}\} + \mathbb{E}\{\varepsilon\}\mathbb{E}\{\widehat{f}\} && \widehat{f} \text{ and } \varepsilon \text{ are independent} \\
&= f\mathbb{E}\{\widehat{f}\} && \mathbb{E}\{\varepsilon\} = 0 \qquad (20)
\end{aligned}$$

Derivation of the Bias-Variance Decomposition for MSE (Ctd.)

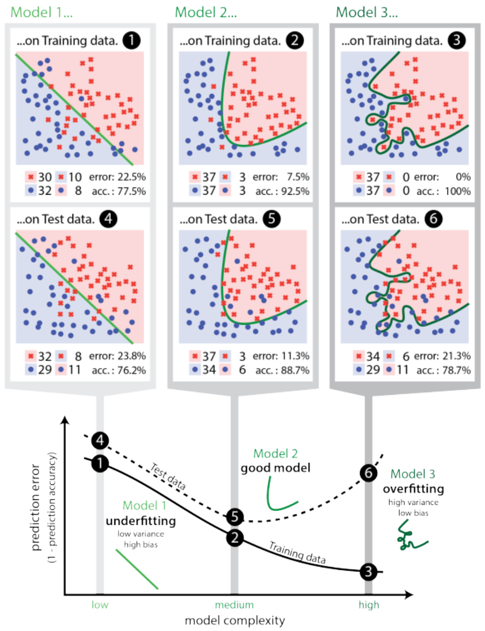
- Let us now plug these results into our MSE definition:

$$\begin{aligned} \text{MSE} &= \mathbb{E}\{y^2\} - 2\mathbb{E}\{y\widehat{f}\} + \mathbb{E}\{\widehat{f}^2\} = f^2 + \sigma_\varepsilon^2 - 2f\mathbb{E}\{\widehat{f}\} + \mathbb{V}\{\widehat{f}\} + \mathbb{E}^2\{\widehat{f}\} \\ &= (f - \mathbb{E}\{\widehat{f}\})^2 + \mathbb{V}\{\widehat{f}\} + \sigma_\varepsilon^2 = \mathbb{E}^2\left\{(f - \widehat{f})\right\} + \mathbb{V}\{\widehat{f}\} + \sigma_\varepsilon^2 \\ &= \mathbb{B}^2\{\widehat{f}\} + \mathbb{V}\{\widehat{f}\} + \sigma_\varepsilon^2 \end{aligned} \tag{21}$$

Bias, Variance, and Model Complexity



Use early stopping!



Section:
Wrap-Up

Summary
Self-Test Questions
Lecture Outlook

Summary

- **Out-of-sample testing:** Split data into `train`, `dev` and `test` sets
- Cross-validation makes **maximum use of the data**
- Confusion matrices reveal **which classes are frequently confused**
- Precision, recall and F1 are **more robust w. r. t. imbalanced datasets**
- ROC curves are used for the evaluation of rankers
- Hyper-parameters are optimized using **grid search** or **random search**
- Keep the **bias-variance trade-off** in mind! We can decompose the error into bias and variance

Self-Test Questions

1. Why should you split the data into `train`, `dev` and `test` sets?
2. You perform 10-fold cross validation. How many models do you have to learn? Which one do you use in production?
3. What is the problem with accuracy?
4. Why do we apply the harmonic mean to compute the F1 score?
5. Your model gets an AUC value of 0. What does this mean?
6. Random search is usually preferred to optimize hyper-parameters. Why?
7. Your model does not perform well due to its high bias. Your boss suggests adding more training examples. How would you respond?

What's next...?

Unit I	Machine Learning Introduction
Unit II	Mathematical Foundations
Unit III	Bayesian Decision Theory
Unit IV	Regression
Unit V	Classification I
Unit VI	Evaluation
Unit VII	**Classification II**
Unit VIII	Clustering
Unit IX	Dimensionality Reduction

Thank you very much for the attention!

Topic: *** Applied Machine Learning Fundamentals *** Evaluation of ML Models
Term: Winter term 2023/2024

Contact:
Daniel Wehner, M.Sc.
SAP SE / DHBW Mannheim
daniel.wehner@sap.com

Do you have any questions?