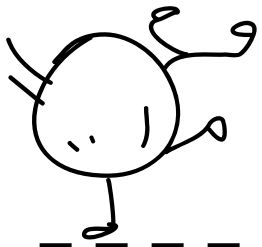


Evaluierung

Vorlesung 186.844

14.01.2016

TECHNISCHE
UNIVERSITÄT
WIEN
Vienna University of Technology

Überblick

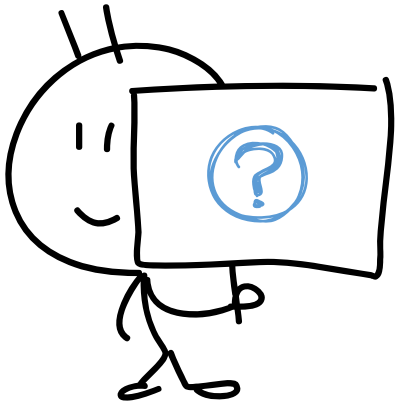
I. No Free Lunch Theorem
II. Bias und Varianz
III. Datensätze
IV. Quantitative Evaluierung
V. Infos zur Prüfung

I. No Free Lunch Theorem

Einleitung

… eine Frage die sich wahrscheinlich jeder von euch ein oder mehrmals gestellt hat:

> Welcher Klassifikator, Trainingsalgorithmus oder welche Methode ist **DIE BESTE**?



Einleitung

Methode **bevorzugt wegen**:

- ihrer geringen Komplexität oder
- ihrer Fähigkeit Vorwissen (Priors) zu berücksichtigen

ABER: Es gibt Mustererkennungsprobleme bei denen die obigen oder ähnliche Eigenschaften **nicht relevant** sind oder gleich für alle zu vergleichenden Methoden sind.

No Free Lunch Theorem

Für das **allgemeine Problem** der Mustererkennung, ohne Annahmen über Art der Muster, Verteilungen, Vorwissen, etc., beantwortet das „*No Free Lunch Theorem*“ folgende Fragen:

Gibt es irgendwelche Gründe einen Klassifikator oder Trainings-algorithmus einem anderen vorzuziehen?

Gibt es einen Algorithmus oder eine Methode die generell besser ist als der Zufall?

Nein!

No Free Lunch Theorem

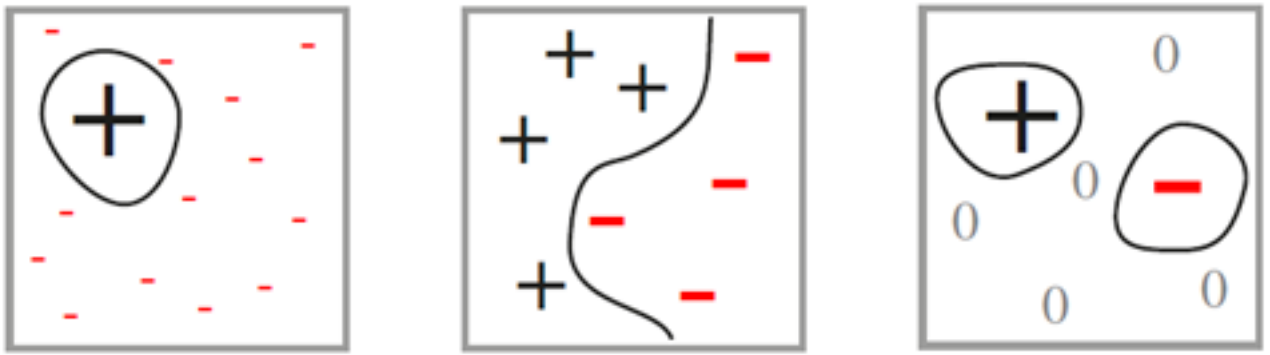
„No Free Lunch Theorem“ anders formuliert: Es gibt **KEINE kontextunabhängige** oder **anwendungsunabhängige** Gründe einen bestimmten Klassifikator, Trainingsalgorithmus oder eine Methode zu bevorzugen.

Dieses Theorem erinnert uns beim Design eines Musterer-kennungssystems auf die wesentlichen Dinge zu achten:

- ✓Vorwissen
- ✓Datenverteilung
- ✓Größe des Trainingsdatensatzes
- ✓Kosten oder Gewinn der jeweiligen Entscheidung

No Free Lunch Theorem

… mögliche Musterkennungssysteme



Legende (Problemräume)

\+ … Generalisierungsfähigkeit höher als Durchschnitt

\- … Generalisierungsfähigkeit niedriger als Durchschnitt

0 … durchnittliche Generalisierugnsfähigkeit

Größe der Symbole … beschreibt die Größe der Abweichung vom Durchschnitt

No Free Lunch Theorem

… unmögliche Musterkennungssysteme

Legende (Problemräume)

\+ … Generalisierungsfähigkeit höher als Durchschnitt

\- … Generalisierungsfähigkeit niedriger als Durchschnitt

0 … durchnittliche Generalisierugnsfähigkeit

Größe der Symbole … beschreibt die Größe der Abweichung vom Durchschnitt

Schlussfolgerung

Es gibt **kein** universell einsetzbares, bestes Mustererkennungsssystem.

Was wir tun können:

Evaluieren wie gut ein Mustererkennungssystem für ein bestimmtes Musterkennungsproblem geeignet ist.

II. Bias und Varianz

Bias und Varianz

Bias und **Varianz** ermöglichen zu messen wie sehr ein Mustererkennungssystem für ein bestimmtes Mustererkennungsproblem „**geeignet**“ ist.

Ein **hoher Bias** bedeutet, dass das **System zu einfach oder unflexibel** und somit **ungeeignet** ist.

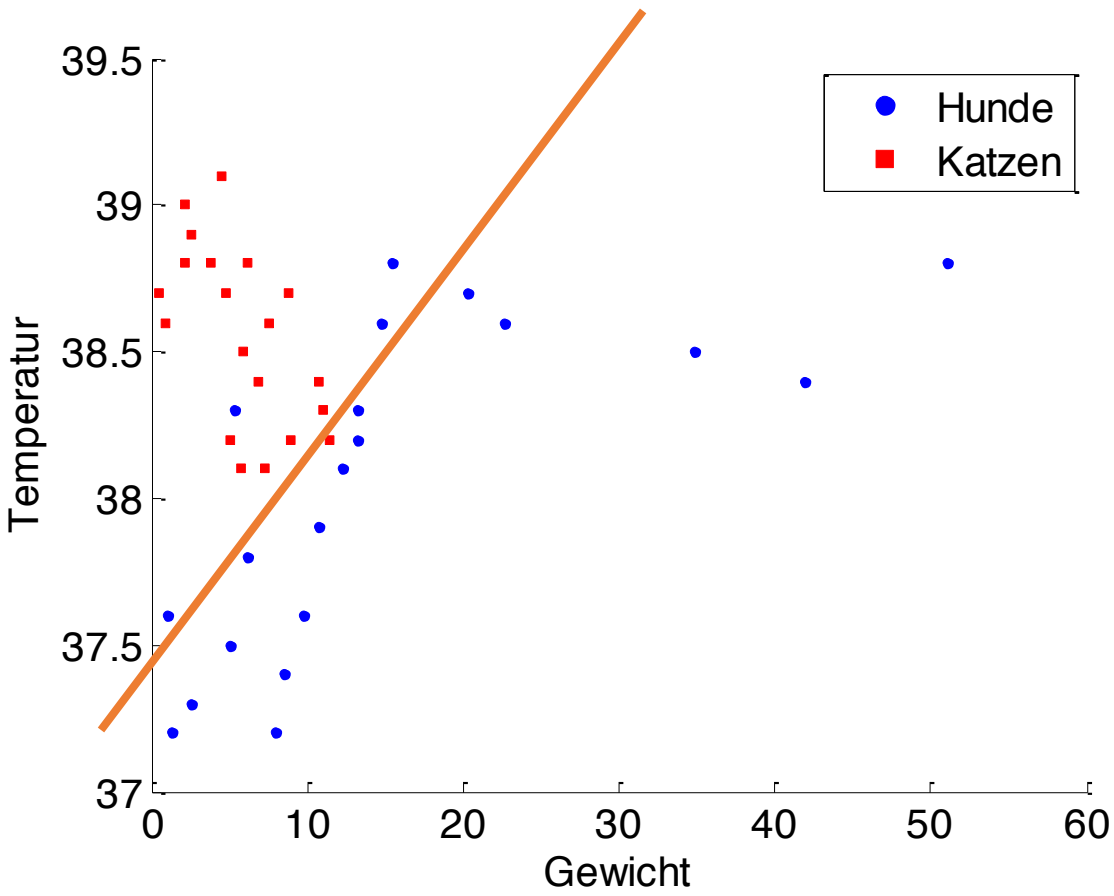
Eine **hohe Varianz** bedeutet, dass das **System zu sensibel** auf die Trainingsdaten reagiert und somit ebenfalls **ungeeignet** ist.

… Bias und Varianz sind abhängig von einander

Beispiel: Hund oder Katze?

lineares Modell

- Klassifikator ist nicht in der Lage das Trainingsset fehler-frei zu unterscheiden
- Lineare Entscheidungs-grenze ist ein zu einfach-es Modell
- gute Generalisierungs-fähigkeit

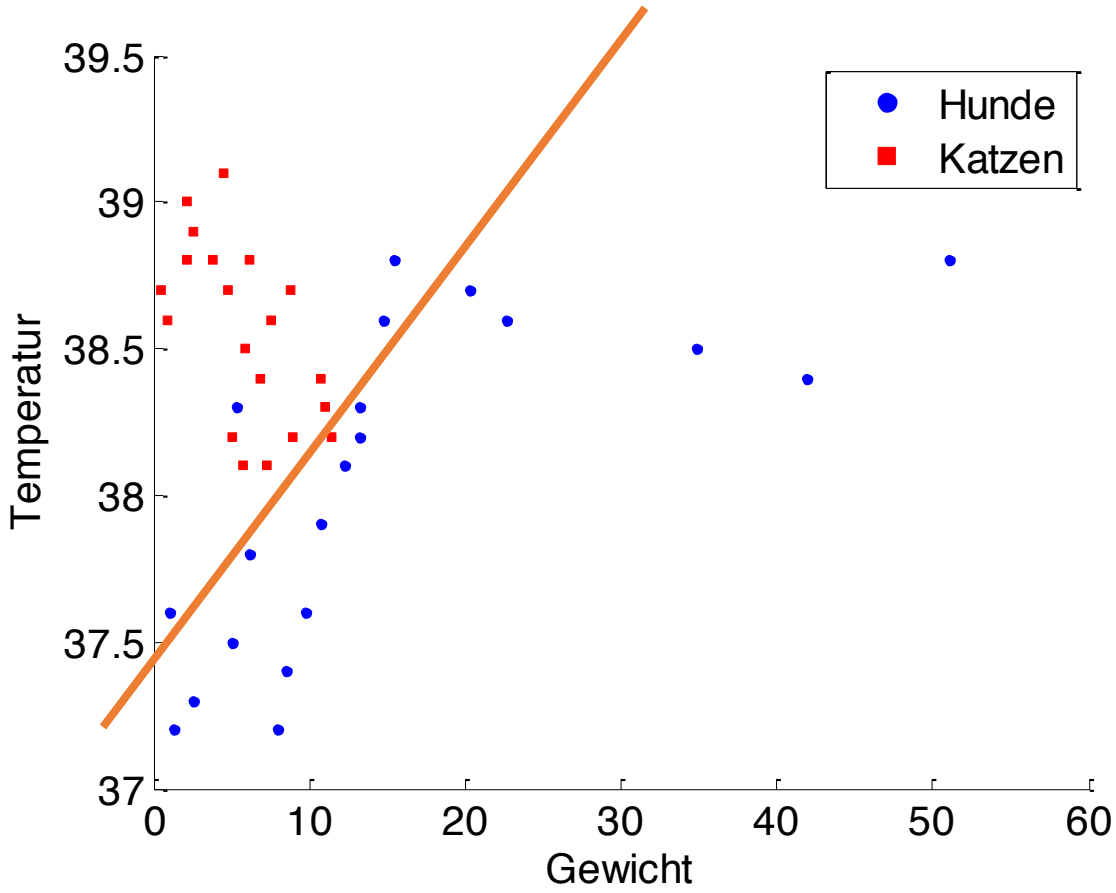


Beispiel: Hund oder Katze?

„underfitting“

→hoher Bias

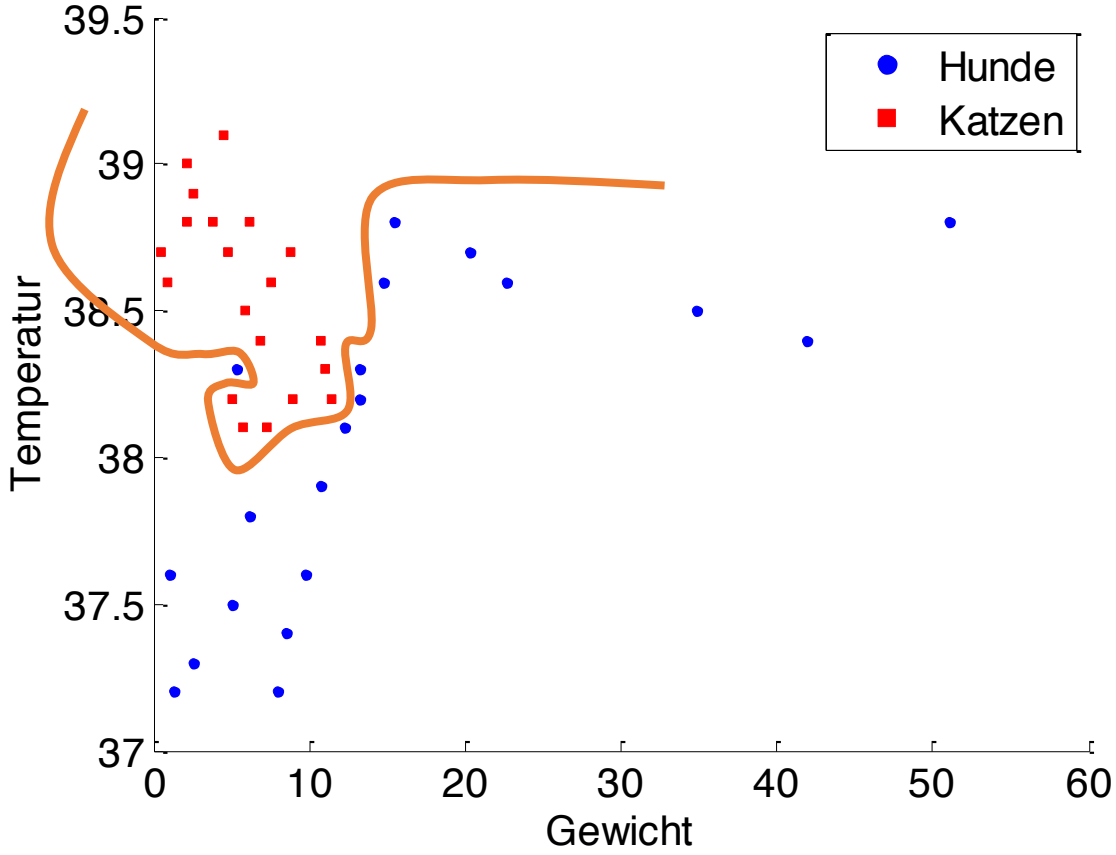
→niedrige Varianz



Beispiel: Hund oder Katze?

komplexes Modell

- fehlerfreie Klassifikation auf Trainingsset
- komplexe Entscheid-ungsgrenze
- schlechte Generali-sierungsfähigkeit

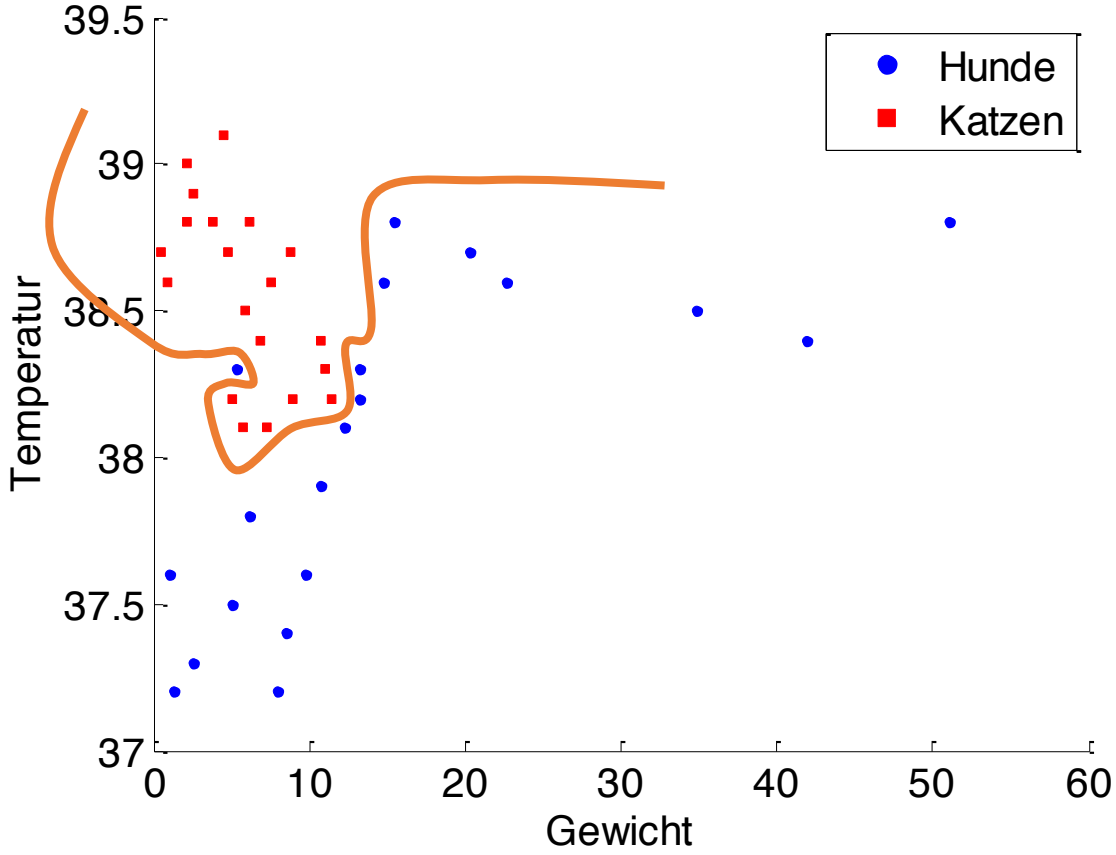


Beispiel: Hund oder Katze?

„overfitting“

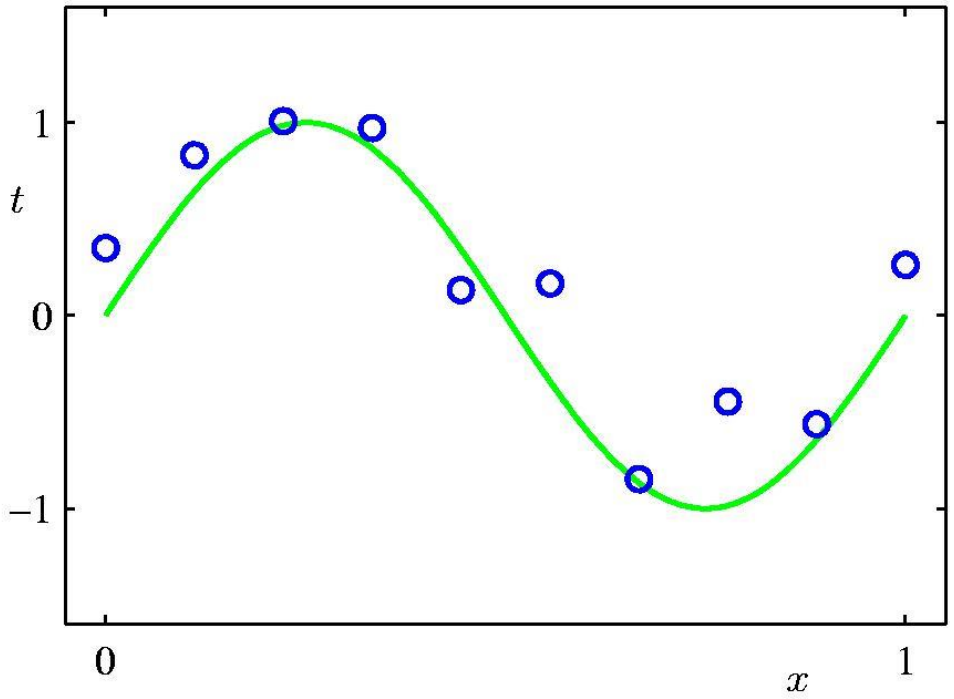
→hohe Varianz

→niedriger Bias



WH: Regressionsanalyse

- Schätzung der Zusammenhänge zwischen Variablen
  → Suche nach Modell
- Trainingsdaten: $N = 10$, $X \equiv (x_1, \dots, x_N)^T$ und $\mathrm{T} \equiv (t_1, \dots, t_N)^T$
- Tatsächliches Modell: $\sin(2\pi x)$, Trainingsdaten beinhalten Störungen (noise)



WH: Polynomial Curve Fitting

… Kurvenanpassung mit Polynomen (freie Übersetzung)

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

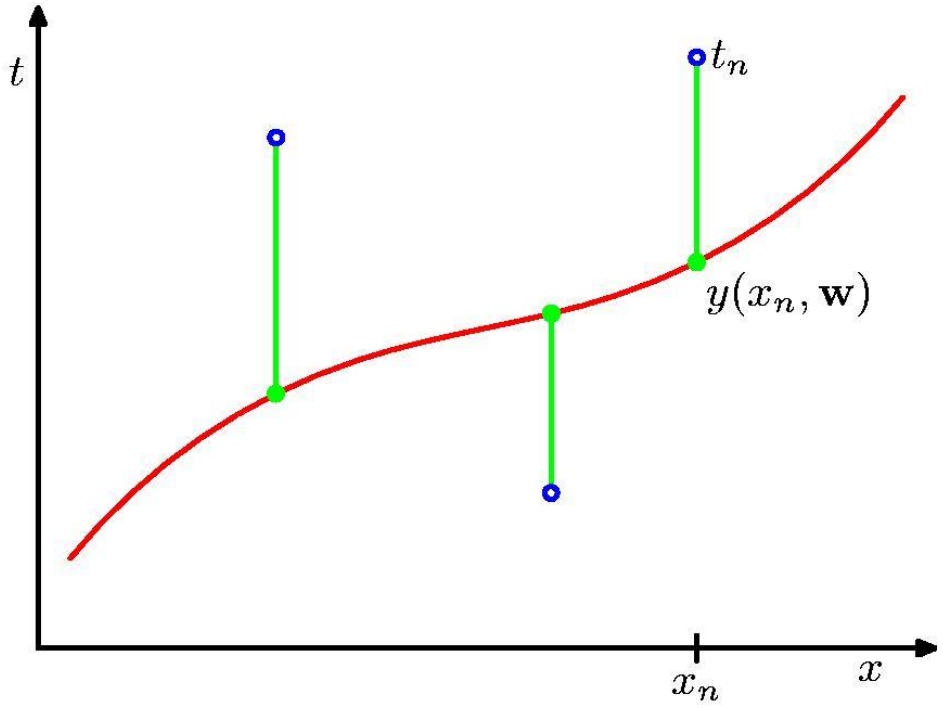
Vektor mit Koeffizienten

Grad des Polynoms

- Werte der Koeffizienten werden durch Anpassung des Polynoms an die Trainingsdaten ermittelt

WH: Fehlerfunktion

- Fehlerfunktion misst Abweichung der Beobachtung t_n von Schätzung $y(x_n, \boldsymbol{w})$



Sum-of-Squares Fehlerfunktion:

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2$$

für verschiedene **w**

WH: Lösung

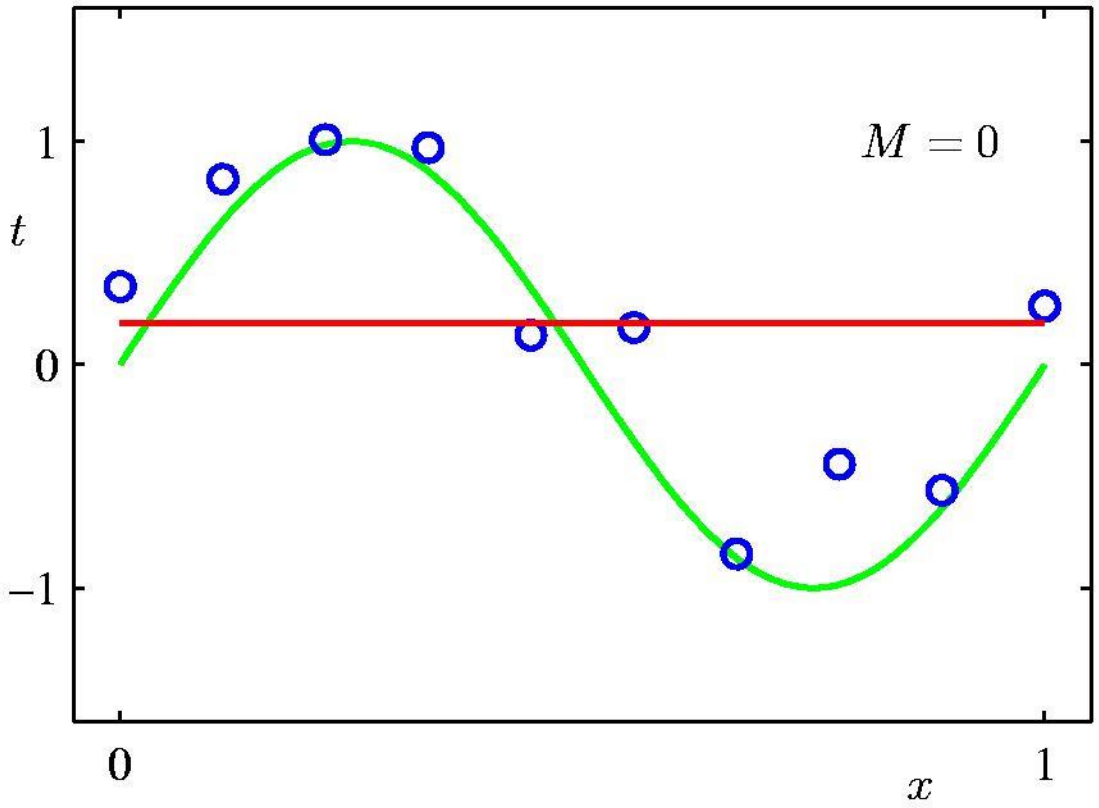
- Kurvenanpassung durch Minimierung der Fehlerfunktion
- Eindeutige Lösung mit Methode der kleinsten Quadrate (Least-Squares Estimation)
- Ergebnis: Koeffizienten $\boldsymbol{w}^*$ und Polynom $\mathrm{y}(\mathrm{x}, \boldsymbol{w}^*)$

Problem: Wahl des Grades des Polynoms …

WH: Polynom 0. Grades

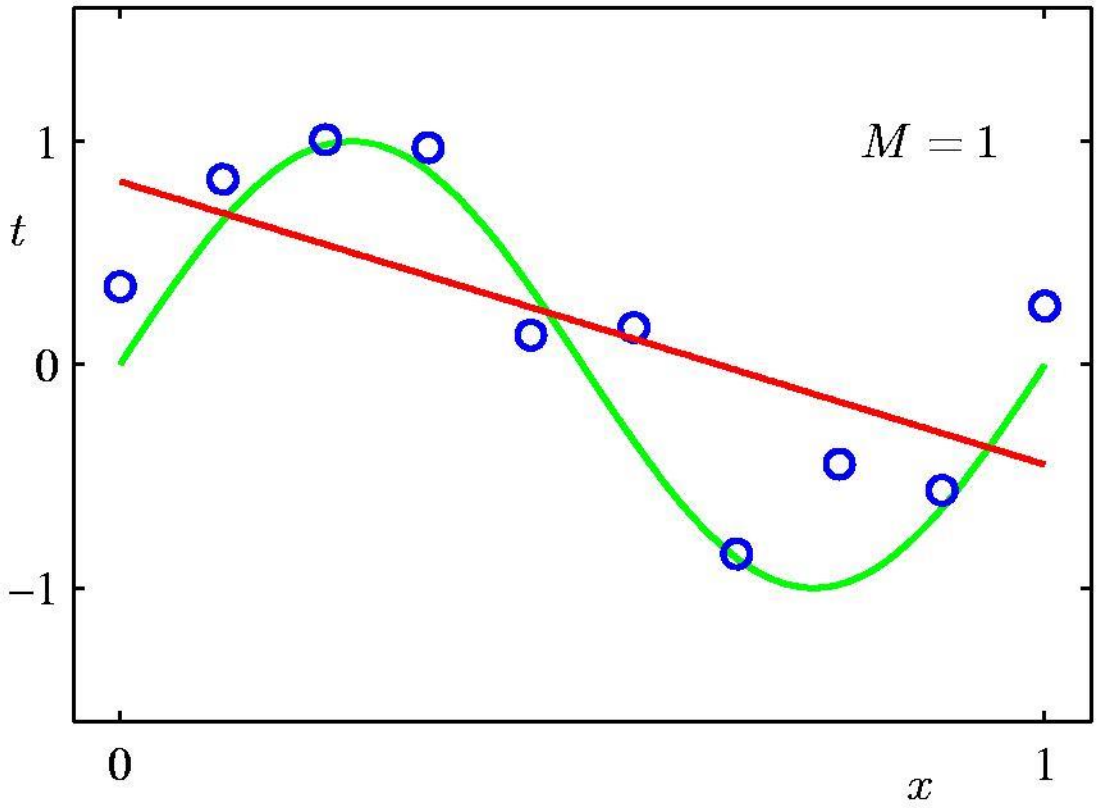
$$y(x, \boldsymbol{w}) = w_0$$

Grad des Polynoms
=> Freiheitsgrade = M+1



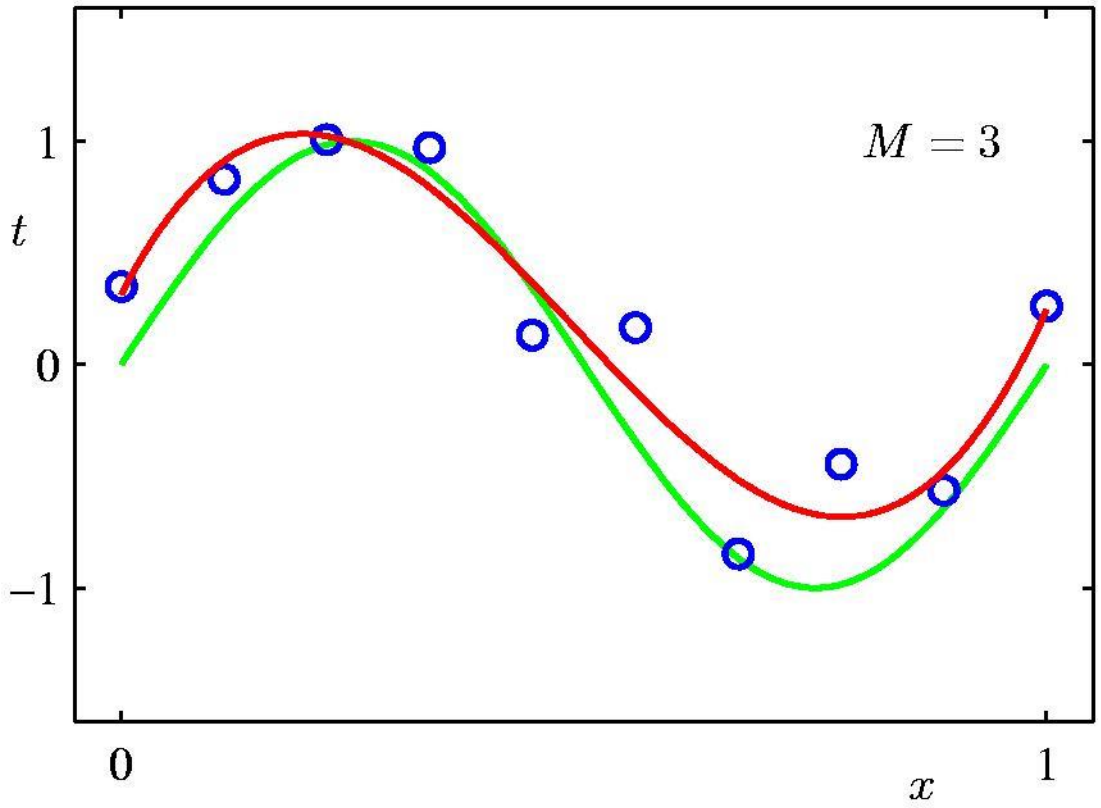
WH: Polynom 1. Grades

$$y(x, \boldsymbol{w}) = w_0 + w_1 x$$



WH: Polynom 3. Grades

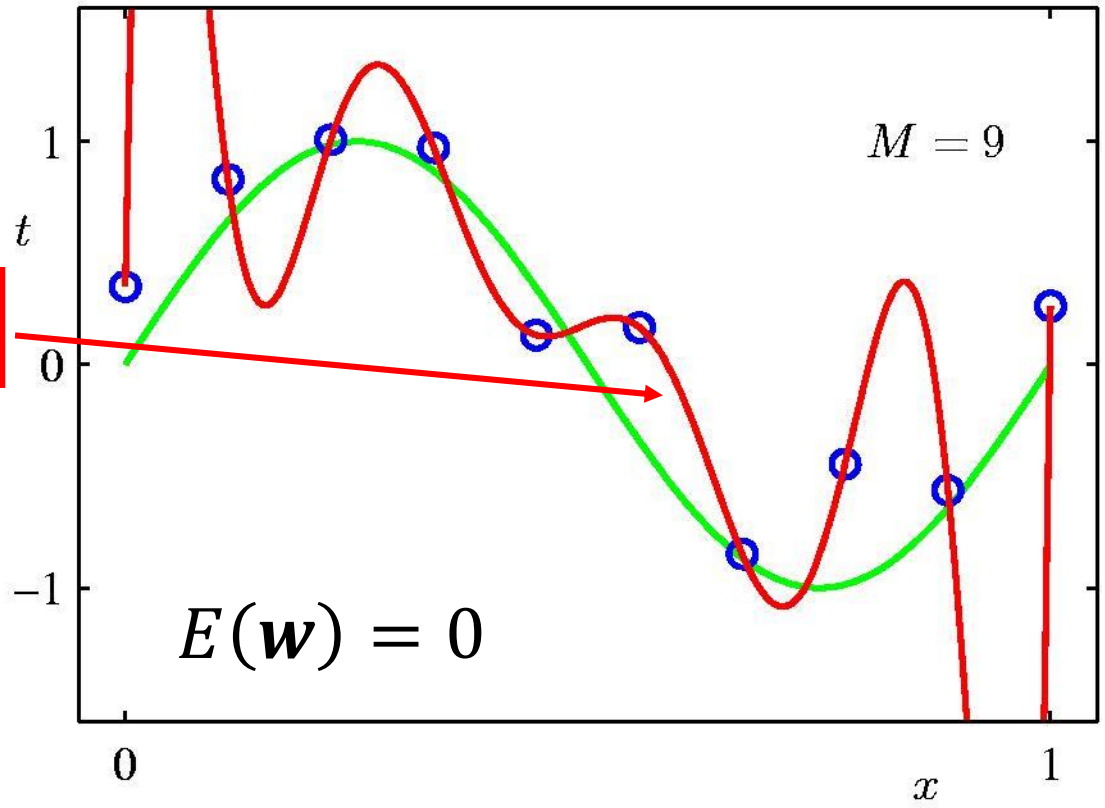
$$y(x, \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2 + w_3 x^3$$



WH: Polynom 9. Grades

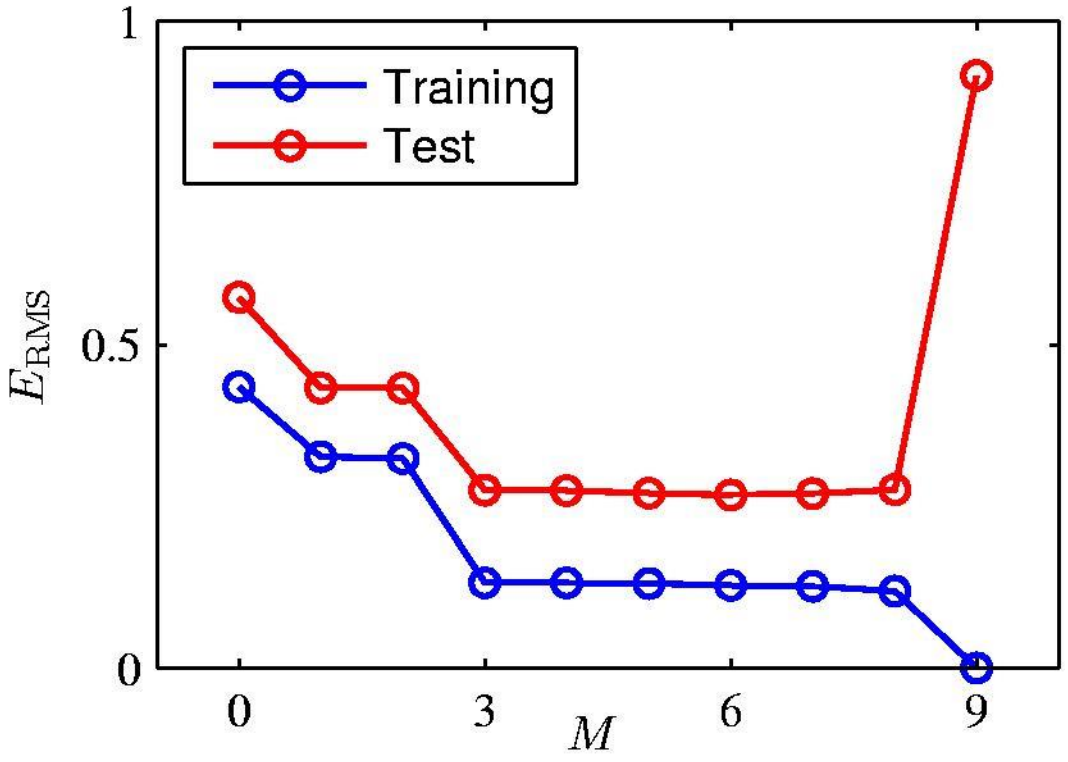
$$y(x, \boldsymbol{w}) = w_0 + w_1 x + w_2 x^2 + \cdots + w_8 x^8 + w_9 x^9$$

Starke „Oszillation“



WH: Overfitting

Trainingsdaten N = 10, Testdaten N = 100



Root-Mean-Square (RMS) Error:

$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N}$$

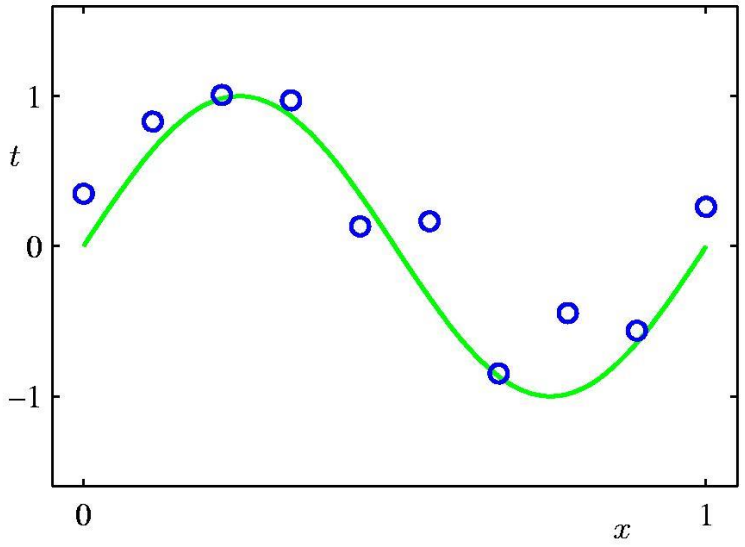
Normalisierung

Bias und Varianz

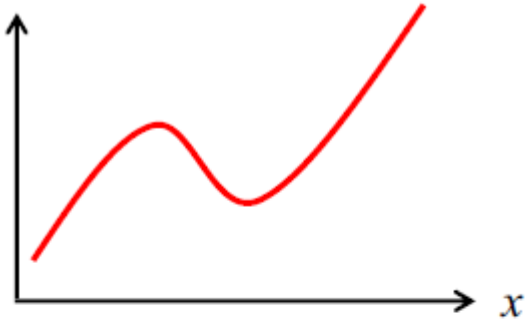
… für Regressionsanalyse

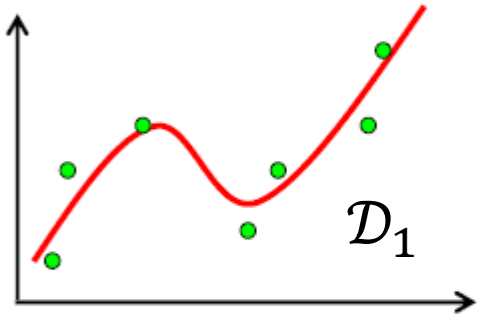
$F(\boldsymbol{x})$ … tatsächliche Funktion

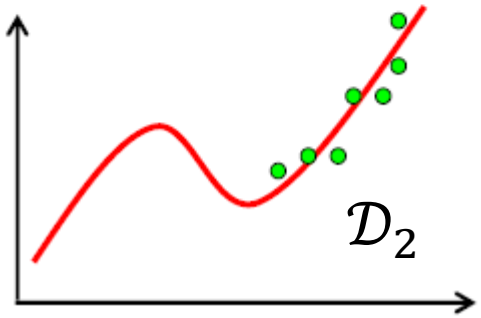
$g(\boldsymbol{x}; \mathcal{D})$ … geschätzte Funktion basierend auf Trainingsdatensatz $\mathcal{D}$ mit n Elementen



Abhängigkeit von Trainingsdaten







Zwei zufällig generierte Trainingsdatensätze $\mathcal{D}_1$ und $\mathcal{D}_2$ bestehen aus der **gleichen Anzahl an Elementen**, aber

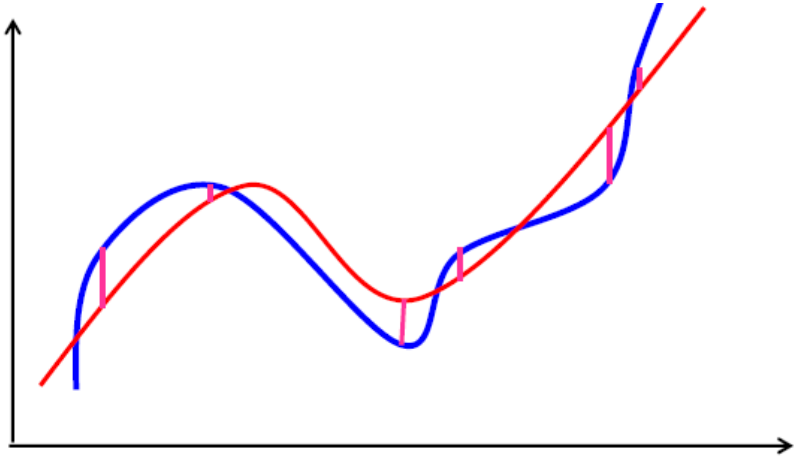
$\mathcal{D}_1$ → ermöglicht gute Schätzung

$\mathcal{D}_2$ → führt zu schlechter Schätzung

Bias und Varianz

Ein Maß für die Effektivität einer geschätzten Funktion $g(\boldsymbol{x}; \mathcal{D})$ ist ihre quadratische Abweichung von der tatsächlichen Funktion $F(\boldsymbol{x})$. Um die Abhängigkeit vom Trainingsdatensatz zu berücksichtigen mittelt man über alle Datensätze $\mathcal{D}$ einer fixen Größe n

$$\mathcal{E}_{\mathcal{D}}[(g(\boldsymbol{x}; \mathcal{D}) - F(\boldsymbol{x}))^2]$$



Bias und Varianz

… man kann

$$\mathcal{E}_{\mathcal{D}}[(g(\boldsymbol{x};\mathcal{D}) - F(\boldsymbol{x}))^2]$$

in zwei Teile spalten:

$$= \underbrace{(\mathcal{E}_{\mathcal{D}}[g(\boldsymbol{x};\mathcal{D})] - F(\boldsymbol{x}))^2}_{\textbf{Bias}^2}$$

$$+\underbrace{\mathcal{E}_{\mathcal{D}}[(g(\boldsymbol{x};\mathcal{D}) - \mathcal{E}_{\mathcal{D}}[g(\boldsymbol{x};\mathcal{D})])^2]}_{\textbf{Varianz}}$$

Bias

$$(\mathcal{E}_{\mathcal{D}}[g(\boldsymbol{x};\mathcal{D})] - F(\boldsymbol{x}))^2$$

Erwartung der Schätzfunktion über alle $\mathcal{D}$

Unterschied zwischen der Erwartung und F

Beantwortet die Frage: Wie geeignet ist die Schätzfunktion für die Daten?

Varianz

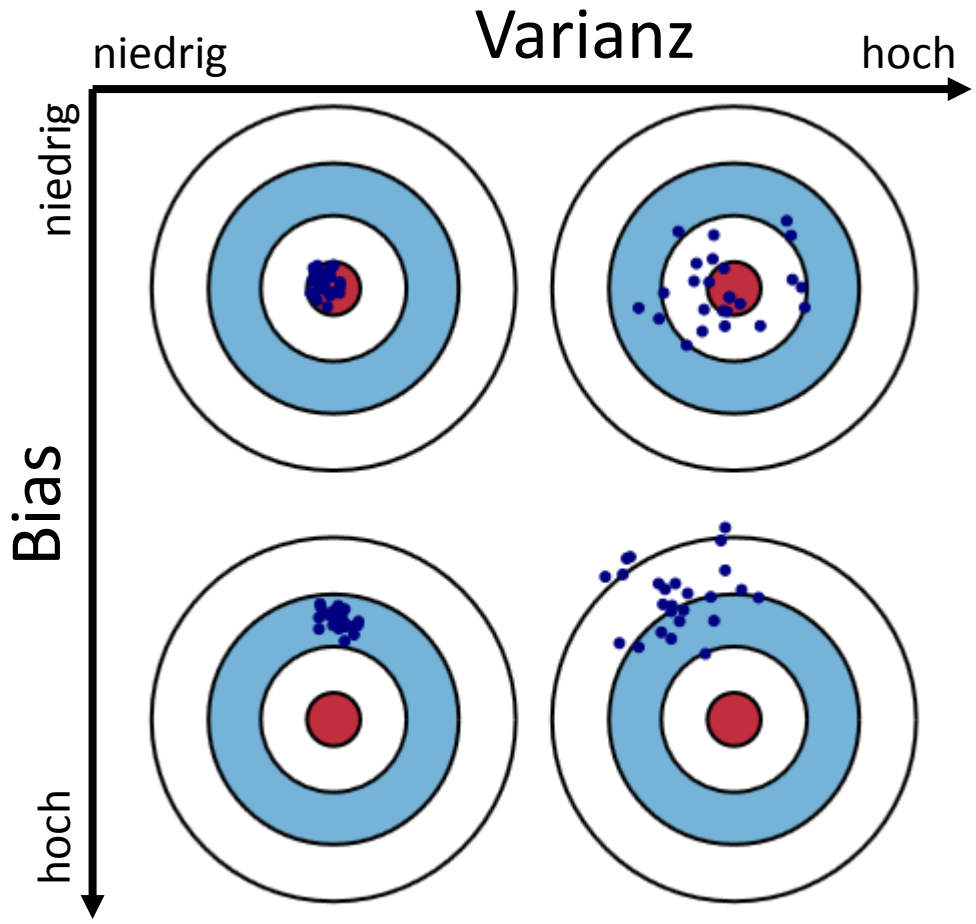
$$\mathcal{E}_{\mathcal{D}}[(g(\boldsymbol{x};\mathcal{D}) - \mathcal{E}_{\mathcal{D}}[g(\boldsymbol{x};\mathcal{D})])^2]$$

Erwartung der Schätzfunktion über alle $\mathcal{D}$

Quadratische Abweichung zwischen der Schätzfunktion und ihrer Erwartung

Beantwortet die Frage: Wie stark hängt die Schätzfunktion von der Wahl des Trainingsdatensatzes ab?

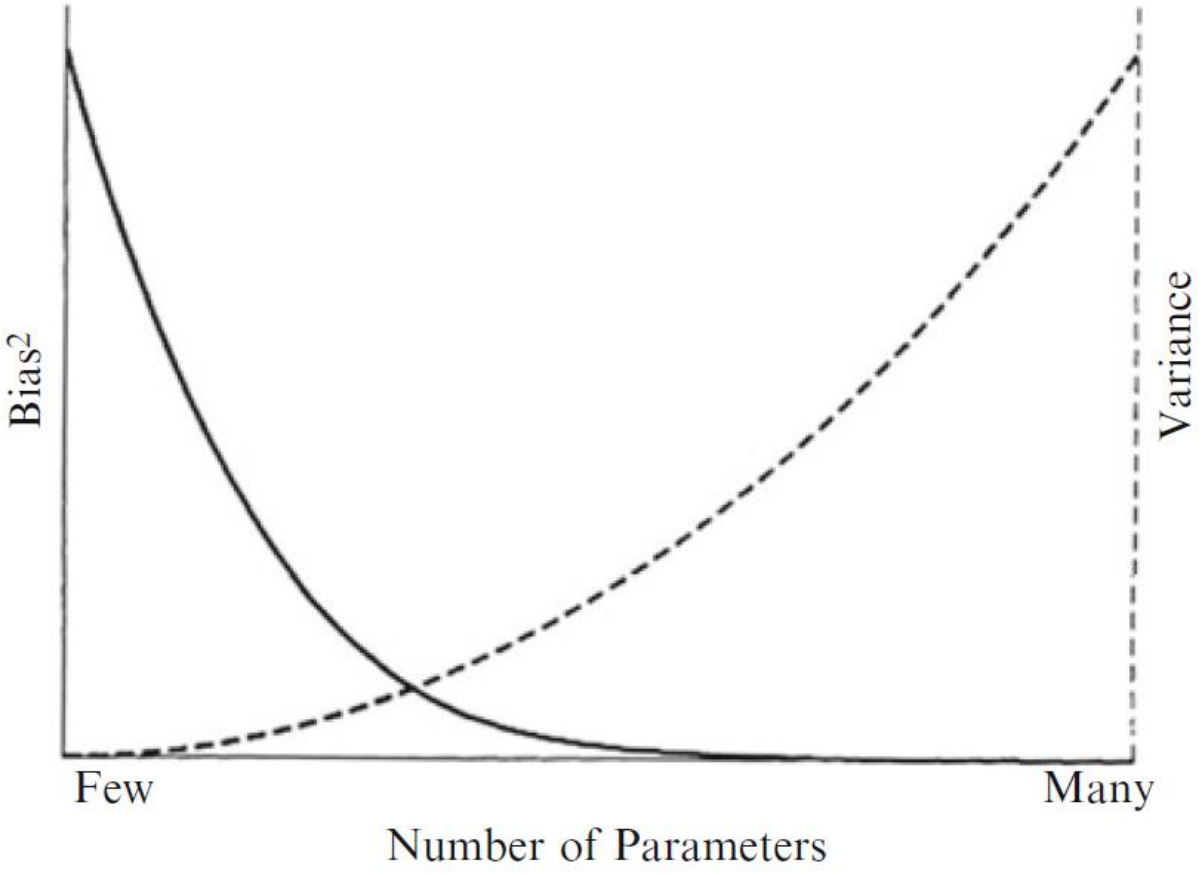
Zusammenhang zwischen …



Quelle: http://scott.fortmann-roe.com/docs/BiasVariance.html

Bias-Varianz-Dilemma

[Quelle: Dougherty 2013]



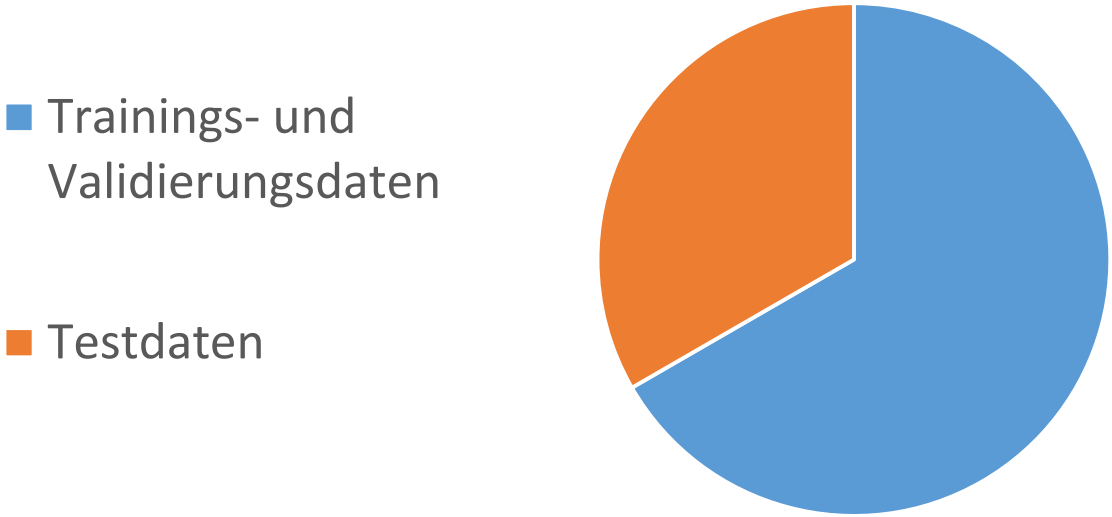
Schlussfolgerungen

- Modelle mit vielen Parametern:
  - niedriger Bias (passen sich gut an die Daten an)
  - hohe Varianz
- Modelle mit wenigen Parametern:
  - hoher Bias
  - niedrige Varianz
- Niedriger Bias und niedrige Varianz → durch Vorwissen über das gesuchte Modell
- Praktisch kann Bias = 0 und Varianz = 0 nicht erreicht werden

III. Datensätze

Arten

Wenn man im **Designprozess eines Mustererkennungssystems** gleichzeitig ein **Klassifikationsmodell auswählen** und den **Fehler** schätzen will, sollte man die Daten in drei Sätze unterteilen: Trainingsdaten, Validierungsdaten & Testdaten.



Arten

mehrere Trainings- und Validierungsdatensätze werden erzeugt → „Cross-Validation“

Trainingsdaten: zum Lernen, optimieren der Parameter eines Klassifikators mit Hilfe eines Trainingsalgorithmus und eines Modells

Validierungsdaten: zum Optimieren der „Hyperparameter“ des Algorithmus oder Modells

Beispiel: Neurales Netzwerk

- Trainingsdatensatz wird verwendet, um Gewichte zu optimieren
- Validierungsdatensatz wird gebraucht, um Anzahl der verborgenen Einheiten, Länge des Trainings, etc. festzulegen

Warum Trennung in ...

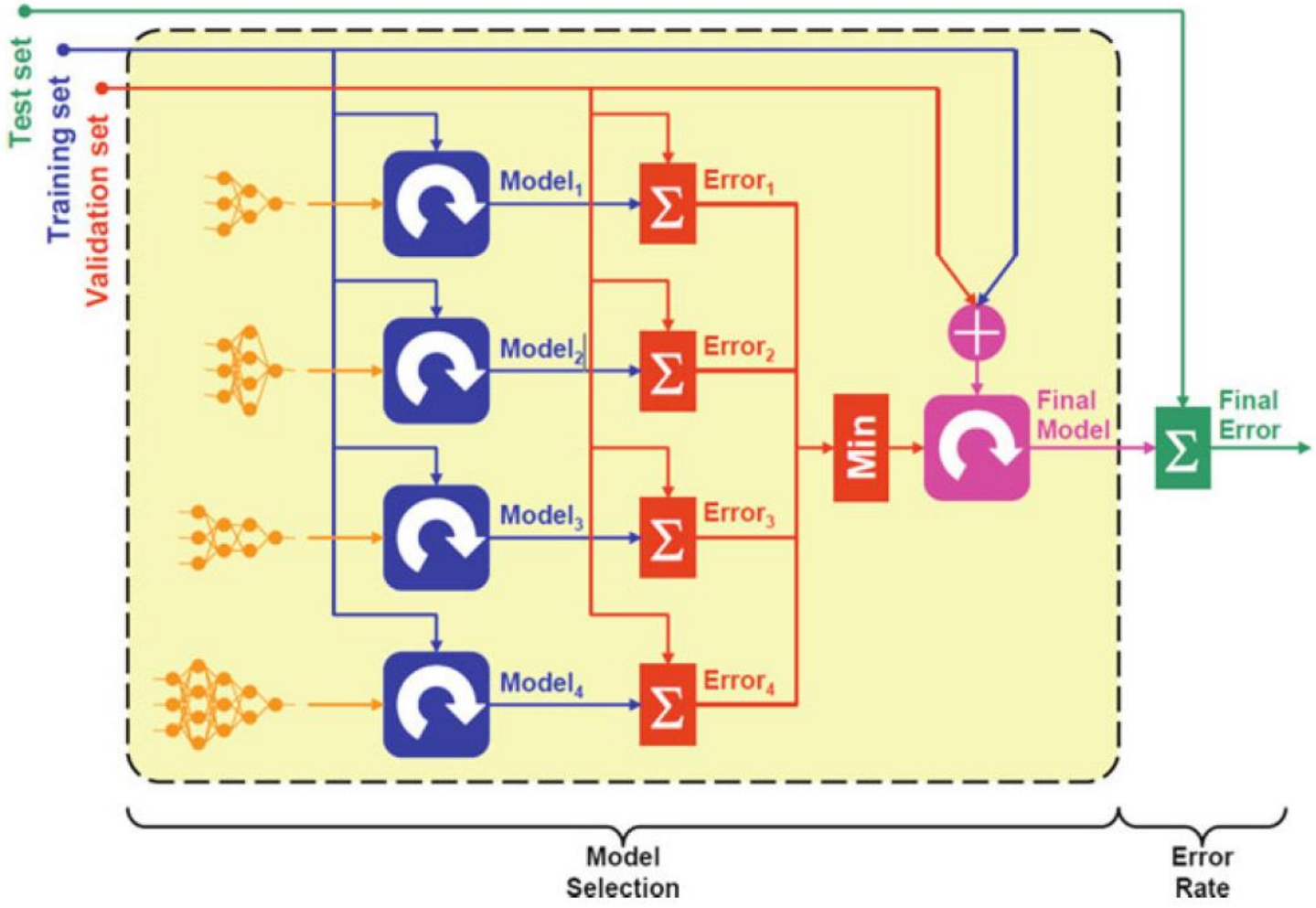
Test- und Validierungsdatensatz?

Der Validierungsdatensatz wird verwendet, um das endgültige Modell zu wählen. Er hat also einen Bias und sollte deshalb nicht zur Schätzung der Fehlerrate verwendet werden.

Nachdem das endgültige Modell am Testdatensatz evaluiert wurde, sollte es nicht mehr verändert werden.

Verwendung der Datensätze

[Quelle: Dougherty 2013]



Cross-Validation

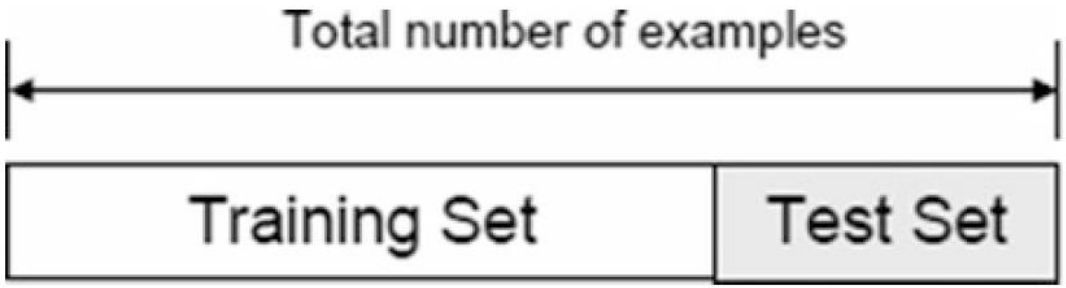
… Vergleichsprüfung (im Deutschen)

- meistens stehen für das Design und Evaluierung eines ME-Systems **zu wenig Daten** zur Verfügung
- man ist verleitet **die gesamten Daten für das Training** zu verwenden
  - Gefahr: Overfitting, hohe Varianz
  - Unterschätzung der Fehlerrate
  - nicht selten 100% richtige Klassifizierung auf Trainingsdaten

Cross-Validation ist eine Methode zur Evaluierung von Klassifikatoren, bei der ein Teil der Daten vor dem Training entfernt wird. Dieser separate Teil der Daten wird verwendet, um die Performance des erlernten Modells zu testen.

Holdout-Methode

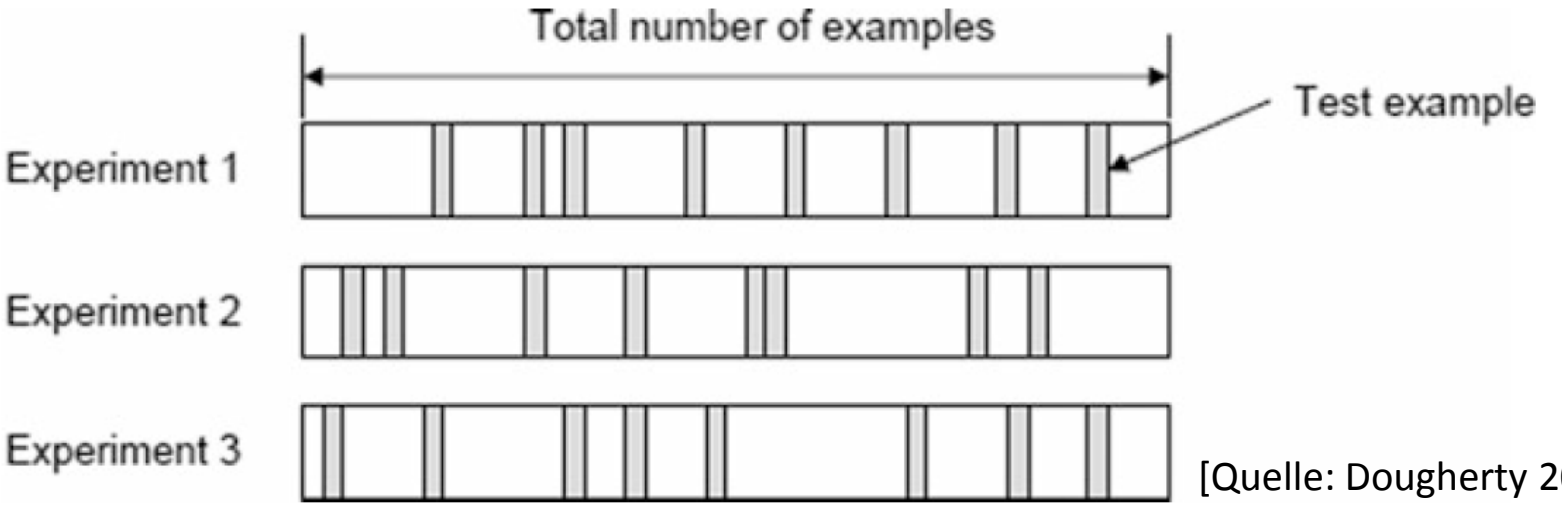
- **einfachste** Methode des Cross-Validation
- zwei Datensätze: Training und Test
- Aufteilung der Daten nach einer Regel
  - z.B.: Trainingdatensatz die Hälfte oder zwei Drittel der Daten



[Quelle: Dougherty 2013]

Random Subsampling

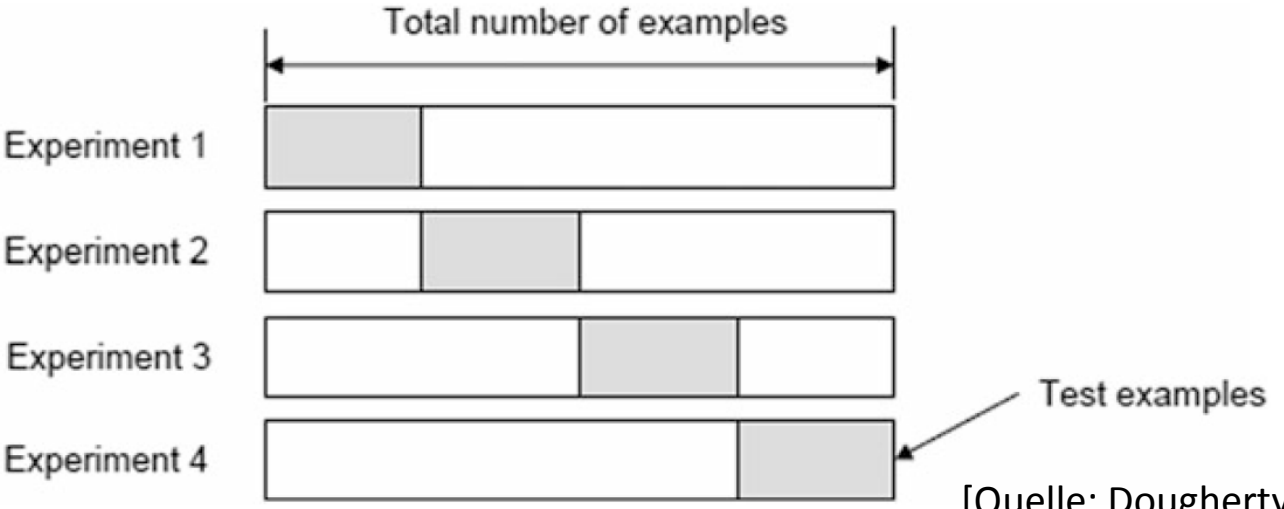
- **mehrmalige Wiederholung** der Holdout-Methode
- zufällige Auswahl der Testdaten
- für jede Aufteilung k:
  - Training des Klassifikators
  - Ermittlung der Fehlerrate am Testdatensatz



[Quelle: Dougherty 2013]

k-Fold Cross-Validation

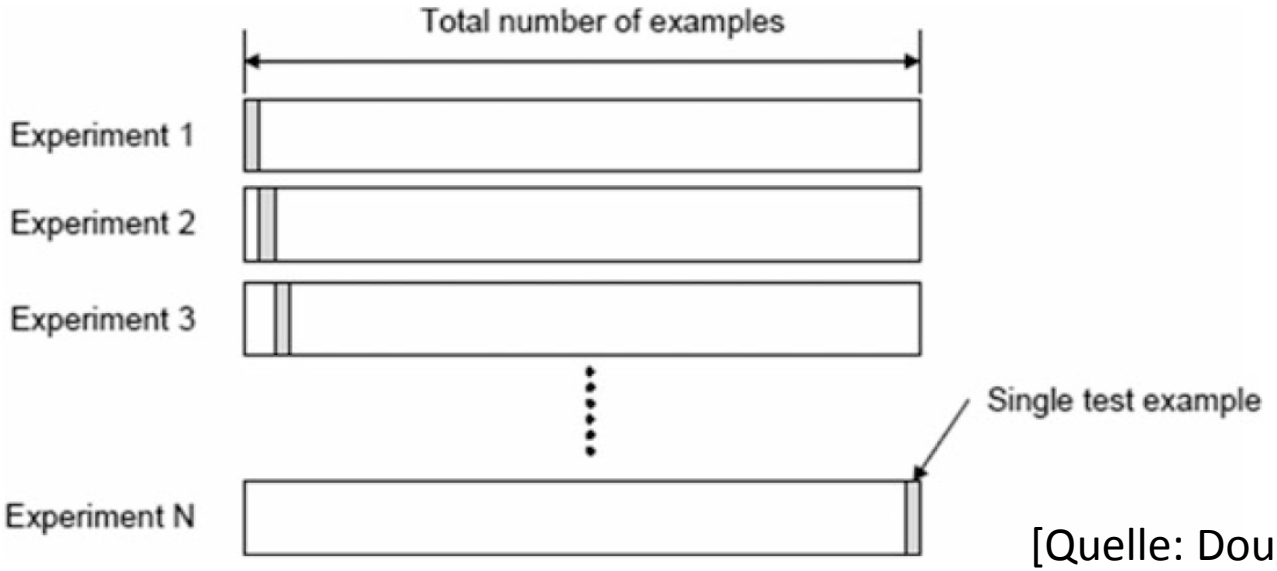
- Daten werden in k **gleich große Datensätze** unterteilt
  - ein Teil: Testdatensatz
  - restliche Teile: Trainingsdatensatz
- k Wiederholungen:
  - Training des Klassifikators
  - Ermittlung der Fehlerrate am Testdatensatz



[Quelle: Dougherty 2013]

Leave-one-out Cross-Validation

- **spezieller Fall** des k-Fold Cross-Validation
- $k = N$ … Größe des Datensatzes
- Testdatensatz besteht nur aus einem **einzigen Element**
- nutzt den **größt möglichen Trainingsdatensatz**

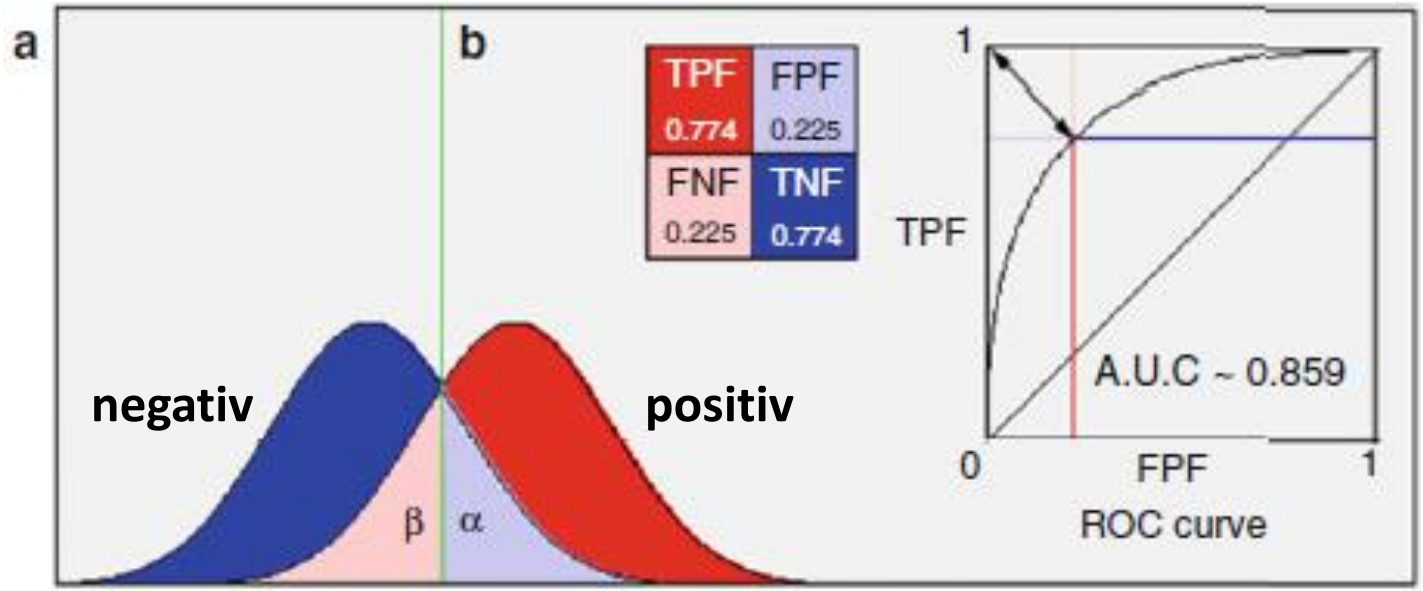


[Quelle: Dougherty 2013]

IV. Quantitative Evaluierung

Für zwei Klassen …

- Klasse w_1: negativ (links von der Entscheidungsgrenze)
- Klasse w_2: positive (rechts von der Entscheidungsgrenze)

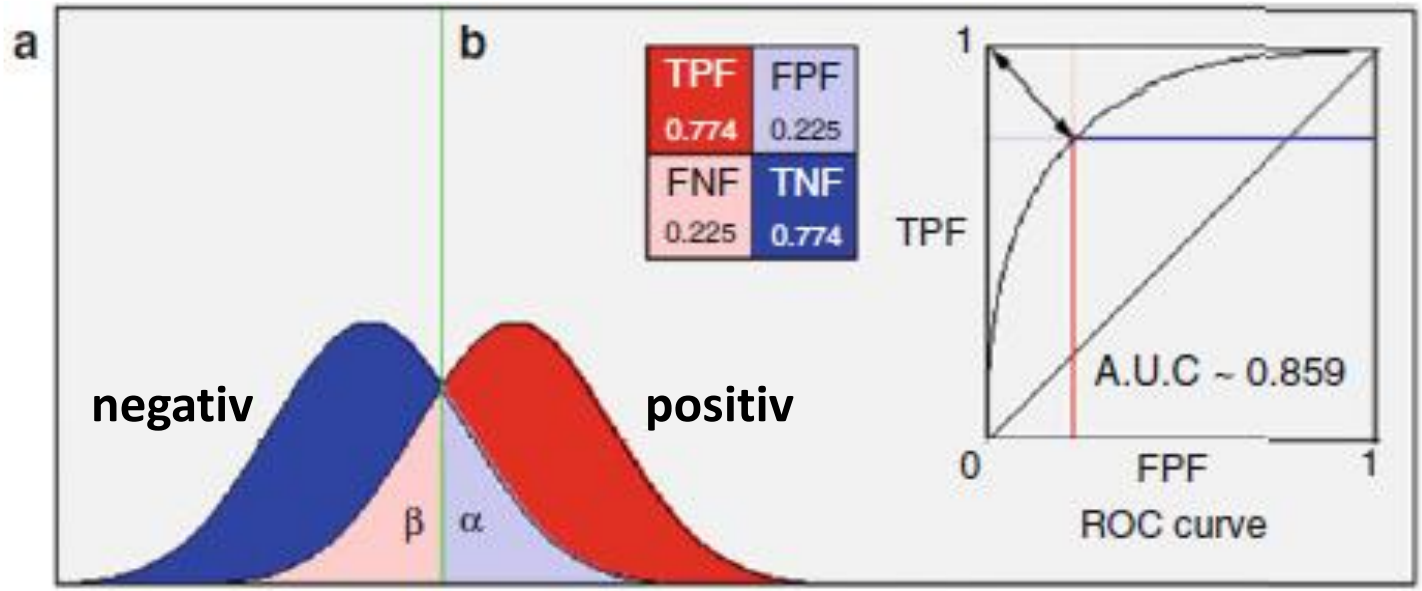


[Quelle: Dougherty 2013]

Für zwei Klassen …

Mögliche Ergebnisse:

- **TP** = True Positive
- **FN** = False Negative
- **TN** = True Negativ
- **FP** = False Positiv



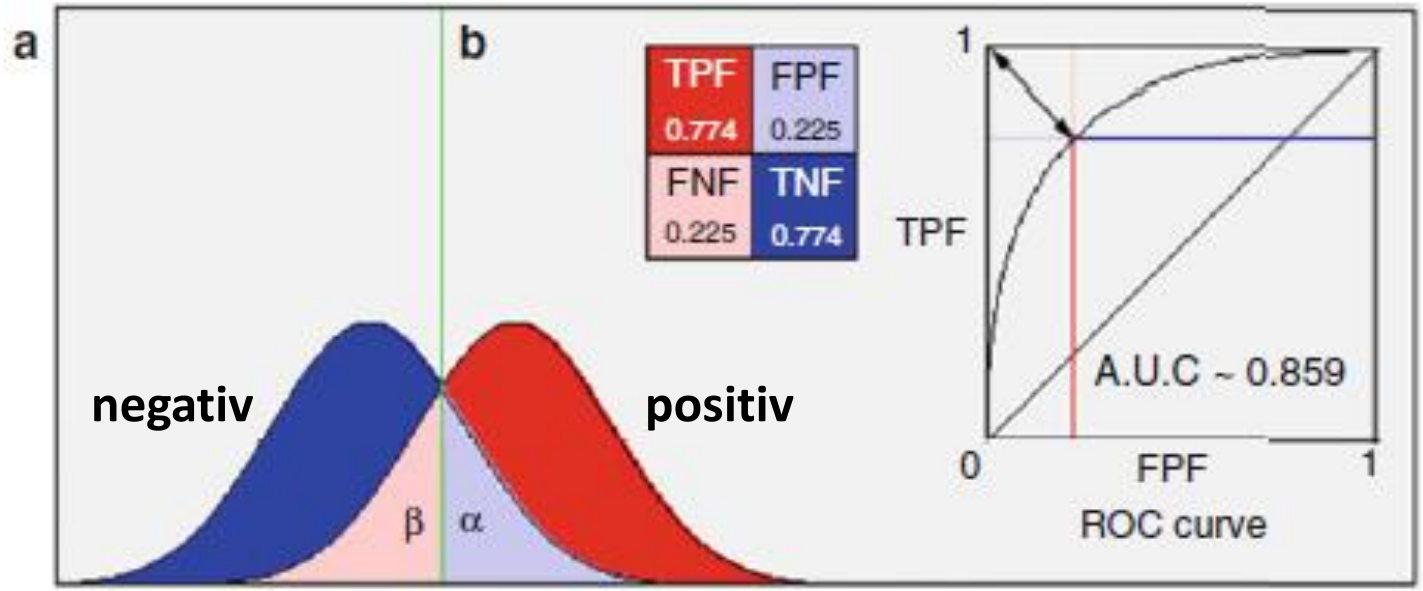
[Quelle: Dougherty 2013]

WH: ROC-Kurve

ROC = Receiver Operating Characteristic

TPF = True Positive Fraction

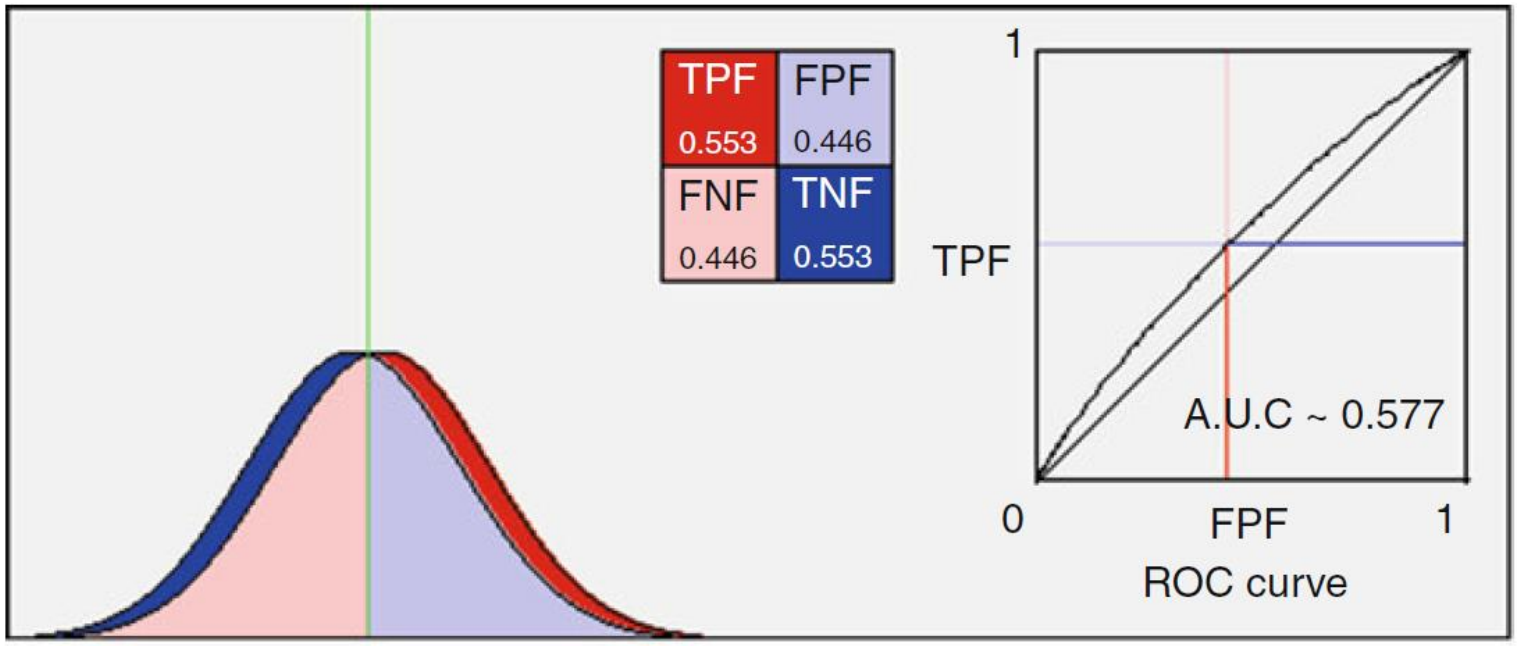
FPF = False Positive Fraction



[Quelle: Dougherty 2013]

Area Under Curve

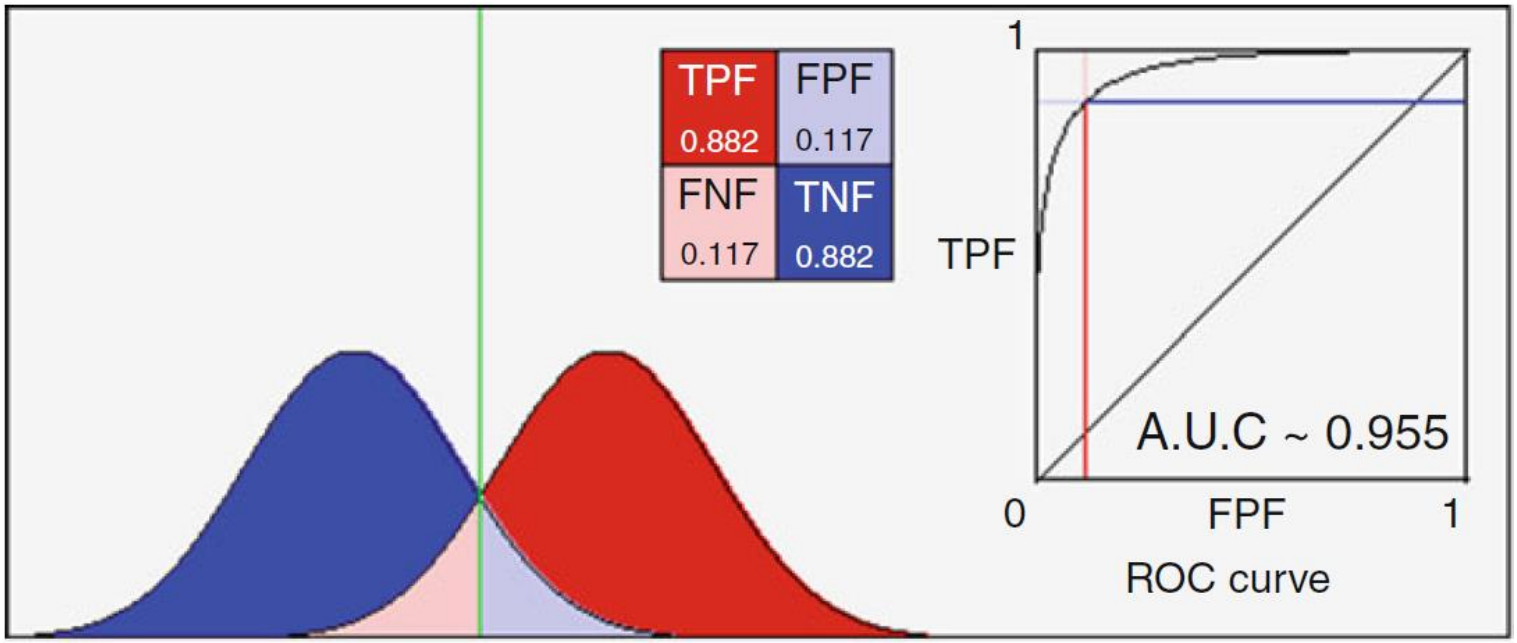
- AUC $\approx 0{,}5$ → (fast) vollständiger Überlapp der beiden Klassen (Kurven)



[Quelle: Dougherty 2013]

Area Under Curve

- AUC nahe an 1 → Klassen überlappen kaum



[Quelle: Dougherty 2013]

Confusion-Matrix

- für zwei Klassen

Predicted			
Actual	Positive	Negative	Total
Positive	TP	FN	p
Negative	FP	TN	n
Total	p'	n'	N

Ground Truth

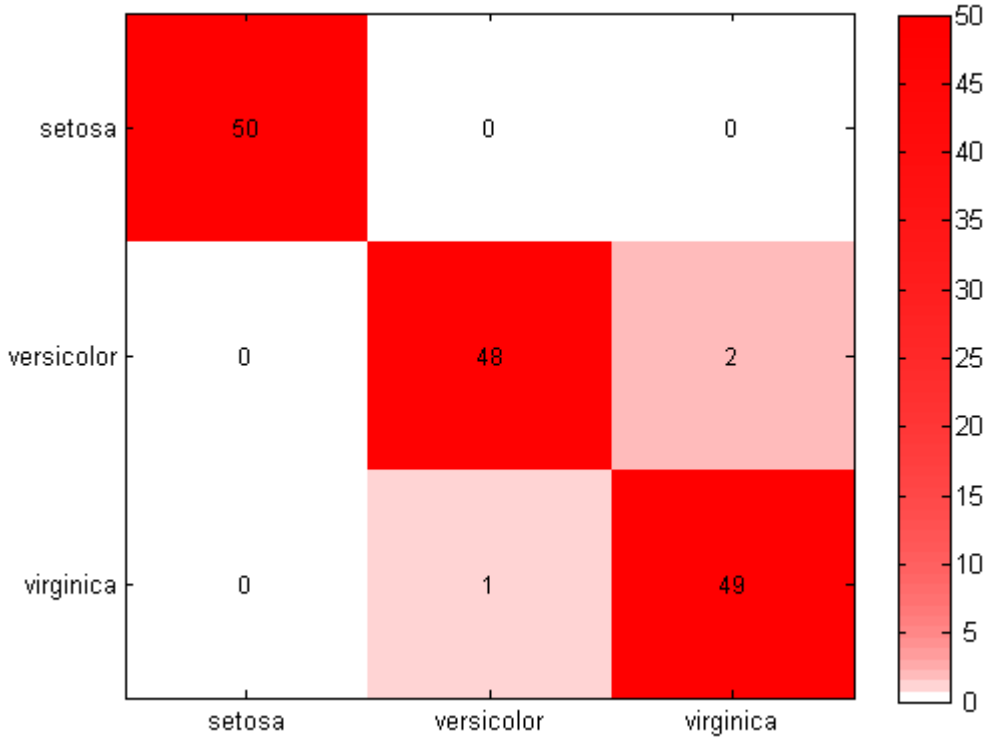
Anzahl der positiv klassifizierten Elemente

Anzahl der negativ klassifizierten Elemente

[Quelle: Dougherty 2013]

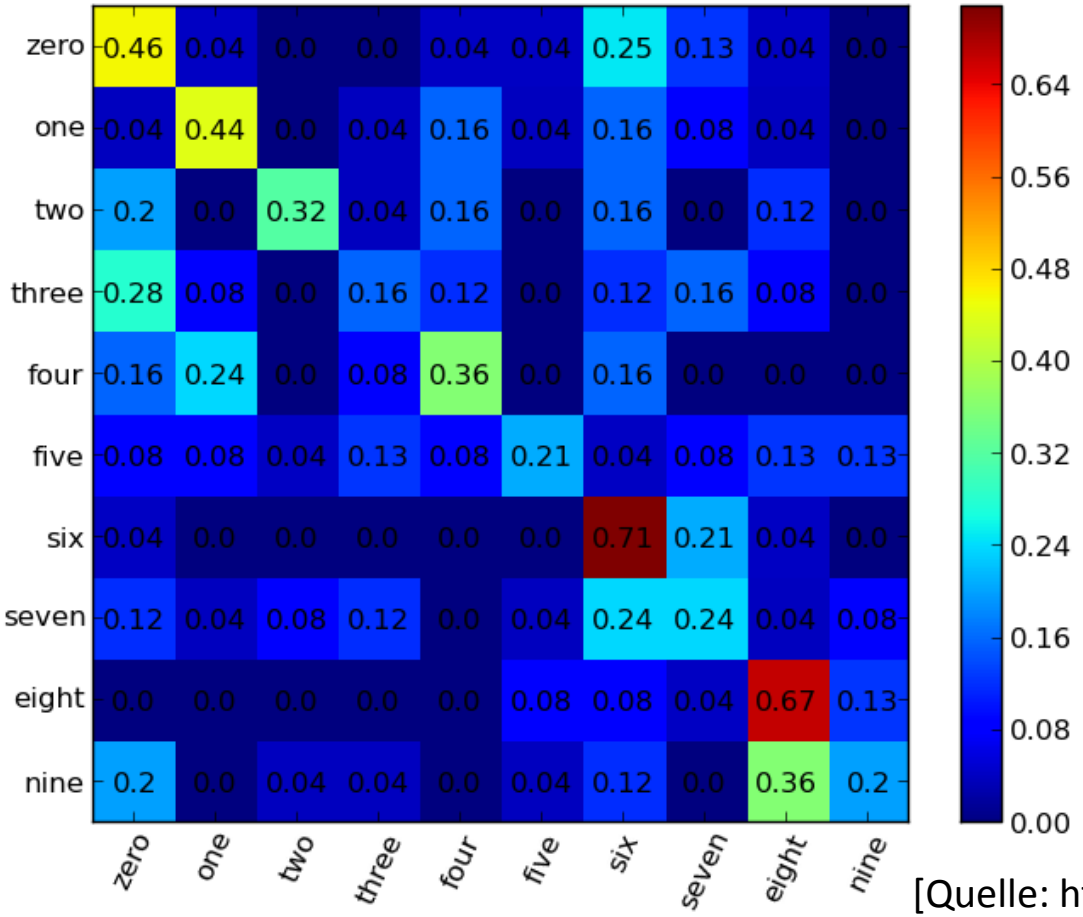
Confusion-Matrix

- für mehr als zwei Klassen
- Idealfall: alle Werte außer jene in der Diagonale sind „0“



Confusion-Matrix

- für mehr als zwei Klassen



[Quelle: http://stackoverflow.com]

Precision

Die **Precision** P_i gibt an wieviel Prozent der Muster die als Klasse i klassifiziert wurden auch tatsächlich der Klasse i angehören (Ground-Truth). Die „Precision“ der 1. Klasse eines binären Klassifikations-problems wird folgendermaßen bestimmt:

$$P_1 = \frac{C(1,1)}{C(1,1)+C(2,1)} \text{ wobei}$$

C die Confusion-Matrix ist und ihre Elemente $C(i,j)$ entsprechen der Anzahl an Mustern welche das Klassenlabel i haben (Ground-Truth) und als j klassifiziert wurden.

Anmerkung: Falls die 1. Klasse die „positive“ bzw. die „negative“ Klasse wäre, könnte man die „Precision“ auch wie folgt anschreiben: $\frac{TP}{TP+FP} = \frac{TN}{TN+FN}$

Recall

Recall R_i gibt an wieviel Prozent der Klasse i auch als Klasse i klassifiziert. In einem binären Klassifikationsproblem wird „Recall“ folgendermaßen bestimmt:

$$R_1 = \frac{C(1,1)}{C(1,1)+C(1,2)} \text{ wobei}$$

C die Confusion-Matrix ist und ihre Elemente $C(i,j)$ entsprechen der Anzahl an Mustern welche das Klassenlabel i haben (Ground-Truth) und als j klassifiziert wurden.

Anmerkung: Falls die 1. Klasse die „positive“ bzw. die „negative“ Klasse wäre, könnte man „Recall“ auch wie folgt anschreiben: $\frac{TP}{TP+FN} = \frac{TN}{TN+FP}$

Overall Accuracy

Overall Accuracy A gibt (klassenübergreifend) an wieviel Prozent der Muster richtig klassifiziert wurden. Für M Klassen und N Muster berechnet sich die „Overall Accurarcy“ wie folgt:

$$A = \frac{1}{N}\sum_{i=1}^{M} C(i,i) \text{ wobei}$$

C die Confusion-Matrix ist und die Elemente $C(i,i)$ liegen auf ihrer Diagonale.

Anmerkung: Der Rest auf 100% entspricht der Fehlklassifikation in Prozent.

V. Infos zur Prüfung

Was kommt nicht …

… zur Prüfung

reine Theoriefragen:

- Zähle die x auf …
- Was ist x …?
- etc.

Prüfungsaufgaben

1. Rechenbeispiele

- Bayes-Theorem
- Perceptron
- PCA
- Neuronale Netze
- Kennzahlen
- Diskriminantenfunktionen
- etc.

2. Verständnisbeispiele

- überprüfen Verständnis der Theorie anhand konkreter Beispiele
- Fragen zur Theorie können im Zusammenhang mit den Beispielen gestellt werden

Erlaubte Hilfsmittel

Taschenrechner

- **Rechner die Gleichungssysteme lösen können, o.ä., sind nicht erlaubt!**
- erlaubter (und nötiger) Funktionsumfang:
  - Grundrechenarten
  - Wurzel ziehen
  - Logarithmen

- die Prüfung ist so aufgebaut, dass ein Taschenrechner kaum oder gar nicht notwendig ist
- manchmal ist es ratsam ohne Taschenrechner zu arbeiten (mit Brüchen rechnen, …)

Erlaubte Hilfsmittel

Offizieller Schummler

- Regeln zum Erstellen des Schummlers im TUWEL (in der Vorlage)
- Upload des Schummlers im TUWEL
- **Achtung: Deadline muss eingehalten werden!**
  - **sonst gibt es keinen Schummler …**
- Ausdruck durch LVA-Leitung falls Regeln befolgt wurden
  - falls nicht: kein Schummler
- Schummler werden vor Beginn der Prüfung ausgeteilt

Danke