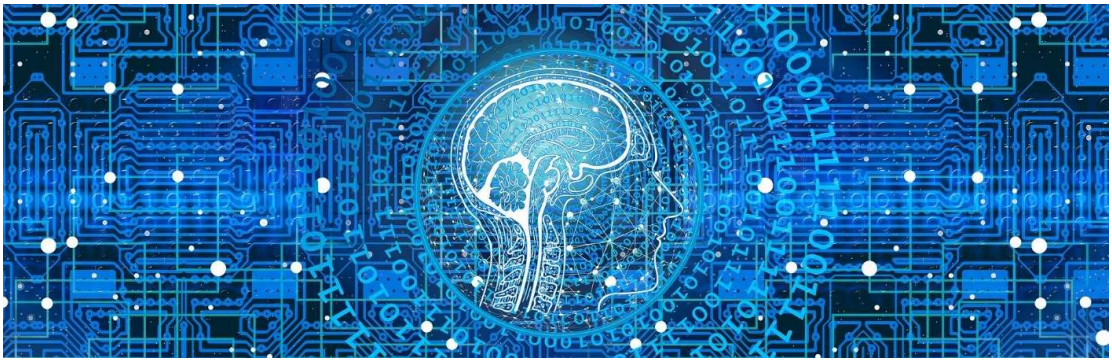
Data Science
7. Teil – Bewertung und Abschluss

Vorlesung an der DHBW Stuttgart, Prof. Dr. Monika Kochanowski



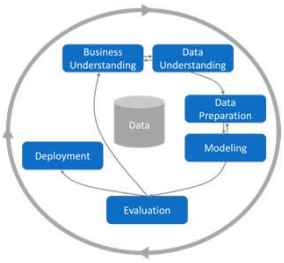
1

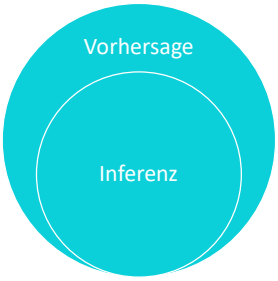
Inhalte der heutigen Vorlesung

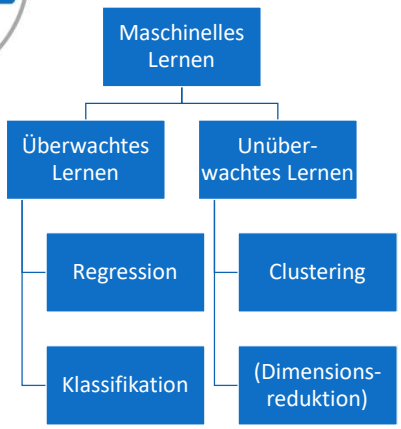
- Lernverfahren für Klassifikation
  - Logistische Regression
  - LDA (Eigene Folien und Jankos)
  - QDA
  - Naive Bayes (für Texte)
- Bewertung von Klassifikation
  - Accuracy
  - Sehr viele andere Metriken
  - Übung Statistische Weisheit des Tages
  - ROC-Curve
  - Custom Metriken
- Abschluss Modellierung und Evaluation
  - Bootstrap & Varianz
  - Feature Importance
  - Grid Search
  - Interpretierbarkeit und Qualität
  - Abhaken der Algorithmen
  - Round-Trip zurück zum Business Understanding

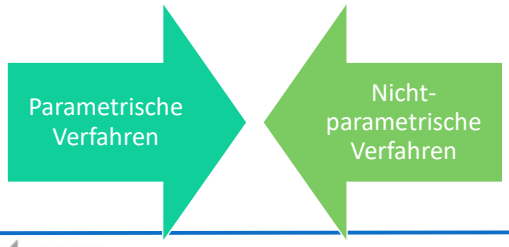
2

Wiederholung



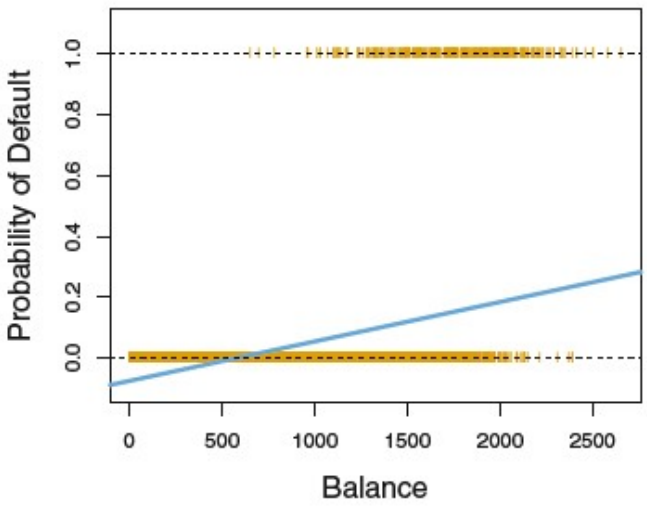






3

Logistische Regression – Motivation

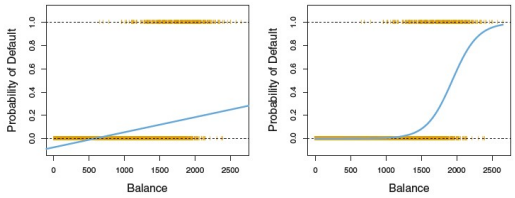


4

Logistische Regression
Ein Klassifikationsalgorithmus

- Analog zu linearer Regression: Parameter finden
  - Parameter werden nicht mit „Least Squares“ (kleinsten Quadrate) bestimmt, sondern mit „Maximum Liklihood“
  - Ziel: Für alle Werte, bei denen „Ja“ erscheinen soll, soll das Ergebnis nahe 1 sein, sonst nahe 0
- Vorteil: Es gibt eine Wahrscheinlichkeit aus.

$$p(X) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}.$$

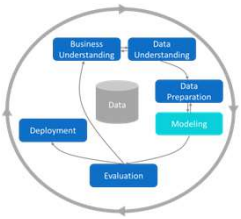


- Weitere Methoden für Klassifikation
  - LDA – Linear Discriminant Analysis
  - QDA – Quadratic Discriminant Analysis
  - https://scikit-learn.org/stable/auto_examples/linear_model/plot_iris_logistic.html
  - https://scikit-learn.org/stable/modules/lda_qda.html

Logistische Regression

Kontinuierliche und kategoriale Variablen können verwendet werden	Annahmen über Form der Separierbarkeit sind getroffen
Parameter sind interpretierbar	Annahmen über Zusammenhang der Variablen und Zielvariable (linear!)
Variablen müssen nicht normalverteilt sein	

Naive-Bayes-Verfahren
Ein bekanntes Klassifikationsverfahren



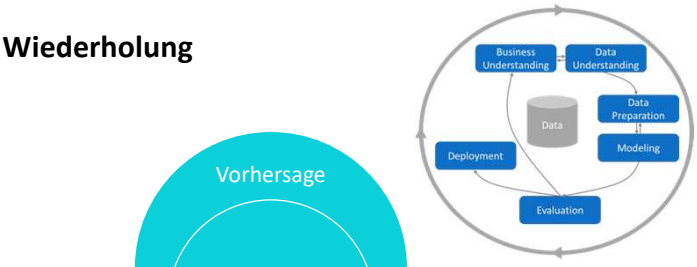
- Spamfilter auf Basis einer »großen« Sammlung von E-Mails
- **Art des Problems? Mögliche Verfahren? Später: Vorteile des Verfahrens?**
- Annahme: Alle Wörter in E-Mails sind **unabhängig** (!)
- Anwendung des Satzes von Bayes (in der Übung bereits angewendet)
  - $P(S)$ = Wahrscheinlichkeit, dass eine Mail Spam ist
  - $P(W)$ = Wahrscheinlichkeit, dass eine Mail ein bestimmtes Wort enthält
  - $P(S|W) = \frac{P(W|S)P(S)}{P(W)}$
  - Vereinfachung der Vorhersage in ein simples »Zählproblem«
- Mit der Annahme der Unabhängigkeit von Wörtern kann man über Multiplikation bzw. Addition (Umformung über Logarithmus) erstmal berechnen, wie wahrscheinlich eine bestimmte »Wortkombination« ist, unter der Bedingung dass eine Spam vorliegt
- Mit Bayes kann man dann daraus die Wahrscheinlichkeit einer »Wortkombination« berechnen, eine Spam zu sein (was eigentlich das Ziel ist)
- Mehr in [Grues 2016] [O'Neil and Schutt 2013]

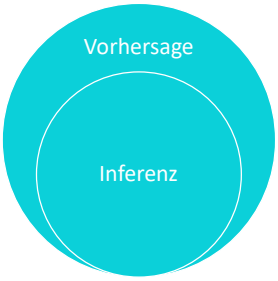
Pseudocode: Naive-Bayes-Verfahren
Eine sehr simple Implementierung sehr grob

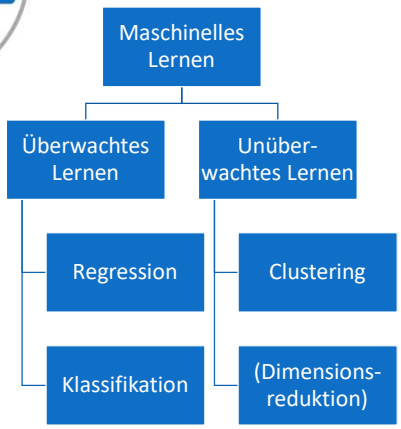
```
create_dictionary #datenstruktur: wort, anzahl in nicht_spams, anzahl in spams)
    for (all mails) #Mail als spam / nicht_spam markiert
        for (all words in mail) #Jedes Wort nur einmal zählen
                if (mail.is_spam)
                        update_dictionary(word, not_spam_count, spam_count++)
                else    # analog
    return dictionary
word_probabilities (dictionary, total_spams, total_not_spams, k) #k als Glättungsparameter Exkurs
        return [(word, (spam_count+k)/(total_spams + 2k), (not_spam_count+k)/(total_non_spams+2k)]
spam_probability(prob, message)
        for (all words in all mails ever = vocabulary) #analog log_prob_if_not_spam
                if (message.contains(word)) log_prob_spam += math.log(prob_if_spam), #analog else
prob_if_spam = math.exp(log_prob_if_spam) #analog: log_prob_if_not_spam
spam_probability = prob_if_spam / (prob_if_spam + prob_if_not_spam) #Satz von Bayes, p_spam = 50 %
```

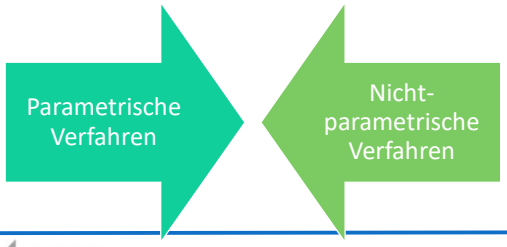
Algorithmus grob nach[Grues 2016]

Wiederholung









DHBW Duale Hochschule Baden-Württemberg

12

Gängige Bewertungsmetriken

Für Klassifikationsprobleme – Wahrheitsmatrix (binäre Klassifikation)

- **Accuracy** (Korrektklassifikationsrate): Anteil korrekt klassifizierter Objekte
- **Recall** (Richtig-positiv-Rate, Sensitivität): Anteil korrekt positiv klassifizierter Objekte an der Gesamtheit aller positiven Objekten
- **Precision** (Genauigkeit): Anteil korrekt positiv klassifizierter Objekte an der Gesamtheit pos. klassifizierten Objekte
- **Specificity (**Richtig-negativ-Rate, Spezifität**)** Analog Recall (für negativ!), also Anteil korrekt negativ .. an..
- **F1-Maß:** harmonisches Mittel aus Precision und Recall

		Wirklichkeit	**Wirklichkeit**
	Alle (n = 1.000.000)	Ist wirklich erkrankt (100)	Ist wirklich gesund (999.900)
Vorhersage	Test sagt erkrankt	Richtig-positiv (99 %)	Falsch-positiv (1 %)
Vorhersage	Test sagt gesund	Falsch-negativ (1 %)	Richtig-negativ (99 %)

DHBW Duale Hochschule Baden-Württemberg

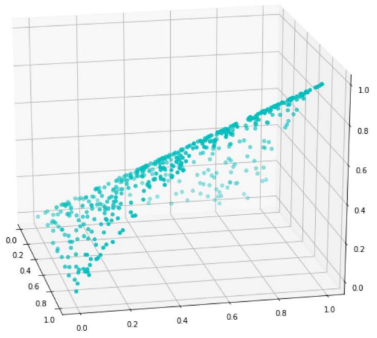
13

F1-Score

- D.h. die Sensitivität/Recall gibt mehr Auskunft über Performance bezüglich der falsch negativen Ergebnisse (Wie viele „Relevante“ gehen uns verloren?) und Precision gibt mehr Auskunft über die falsch positiven Ergebnisse (Wie viele „Irrelevante verwässern“ mein Ergebnis?)
- F1-Score (F1- oder F-Score) als harmonisches Mittel aus Precision und Recall:

$$F_1 = \frac{2}{Recall^{-1} + Precision^{-1}} = 2\,\frac{Recall * Precision}{Recall + Precision}$$

- Je nachdem was „teurer“ zu bewerten ist lässt sich allerdings auch eine Kostenfunktion aus den einzelnen Teilen „zusammenbauen“ und entsprechend gewichten



Wiederholung: Klassifikationsbewertung

Wahrheitsmatrix (binäre Klassifikation)

Übungsaufgabe:

Ein Coronatest hat eine Sensitivität von 97% und eine Spezifität von 99,5%.

1. **Berechnen Sie die Chance, wirklich infiziert zu sein, wenn der Test sagt „positiv“**
   1. **0.01 % infiziert sind (bei sehr niedrigen Inzidenzen)**
   2. **0.1 % infiziert sind (0,23% in LE bei Inzidenz von 177, Stand 4.11.21)**
   3. **1 % infiziert sind (1,6% in LE bei Inzidenz von ca. 1500, Stand 14..3.22)**
   4. **10 % infiziert sind**
2. **Berechnen Sie Accuracy und F1-Score des Tests (in allen gegebenen Fällen), siehe xls.**
3. **Sie wollen reich werden und entwickeln einen „Test“, der immer sagt „negativ“. Was sind die o.g. Metriken für Ihren Test?**
4. **Was schließen Sie daraus?**

Statistische Weisheit des Tages: Prävalenzfehler (Base Rate Fellacy)
Beispiel: Medikamente

- **Annahmen**
  - 100 Medikamente, davon sind 10 wirksam
  - Teststärke: 0,8 Schwellwert für Signifikanz: 0,05
- **Ergebnis**
  - 5 Stück falsch Positiv (wirkungslos, aber scheinen wirksam) (Signifikanzwert)
  - 8 von den wirklich Wirksamen werden erkannt (Teststärke)
  - => von 13 Mitteln sind nur 8 wirksam, also 62% „Trefferquote“. Fehlerrate: 38%.
- **Grund**
  - Basisrate ist niedrig (10 %)
  - => Analogie zu Klassifikationsproblem?
  - => Was passiert wenn die Basisrate extrem niedrig ist?

Teststärke: Die Teststärke eines Hypothesentests ist die Wahrscheinlichkeit, dass er ein statistisch signifikantes Ergebnis liefert (z. B. mit p <= 0,05).

- Dieses hängt ab von:
  - Der Größe der gesuchten Abweichung
  - Der Größe der Stichprobe
  - Möglichen Messfehlern

Quelle und Empfehlung: Reinhart 2015

18

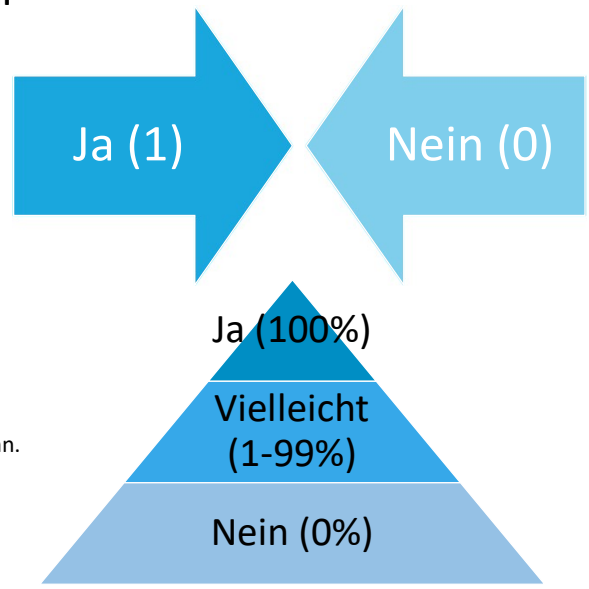
Fehlerarten und Bezeichnungen
Eine Übersicht

- **Verständlich: Richtig positiv, Falsch negativ, Falsch positiv, Richtig negativ**
- **Korrektklassifikationsrate**: Vertrauenswahrscheinlichkeit, Treffergenauigkeit, **Accuracy**
- Richtig-positiv-Rate: Sensitivität, Empfindlichkeit, Trefferquote, **Recall**
- Falsch-negativ-Rate: Miss Rate
- Richtig-negativ-Rate: Spezifität
- Falsch-positiv-Rate: Ausfallrate
- Positiver Vorhersagewert (Relevanz, Wirksamkeit, Genauigkeit, **Precision**) (korrekt als positiv vs. alle positiv)
- Negativer Vorhersagewert (Segreganz) (korrekt als negativ vs. alle negativ)
- Klassifikationsfehler: 1 – Korrektklassifikationsrate (relevant für Loss-Funktionen)
- F1-Maß: Harmonisches Mittel aus Precision und Recall (Genauigkeit + Trefferquote)
- https://de.wikipedia.org/wiki/Beurteilung_eines_bin%C3%A4ren_Klassifikators
- https://en.wikipedia.org/wiki/Evaluation_of_binary_classifiers

19

Neu: Rechnen mit Wahrscheinlichkeiten

- Bisher: binäre Antwort
  - Ja
  - Nein
- Jetzt: Wahrscheinlichkeit
  - Ist zu 99% infiziert
  - Ist zu 50% infiziert
  - Ist zu 12% infiziert
- Viele Tests / Vorhersagemethoden liefern Wahrscheinlichkeiten, mit denen man arbeiten kann.
  - Logistische Regression
  - Neuronale Netze zur Bildklassifikation
- => Nutzbar dafür, einen **Schwellwert** festzulegen



20

Gängige Bewertungsmetriken
Für Klassifikationsprobleme – ROC-Kurve (Receiver Operator Characteristic)

- Sortierung nach Absteigender Wahrscheinlichkeit für einen Treffer von allen Trainingsdaten (bzw. Testdaten)
- T-P: True Positives
- F-P: False Positives
- False Positive Rate: Anteil Falscher Positive an allen Negativen
- Recall: Anteil True Positives an allen Positiven

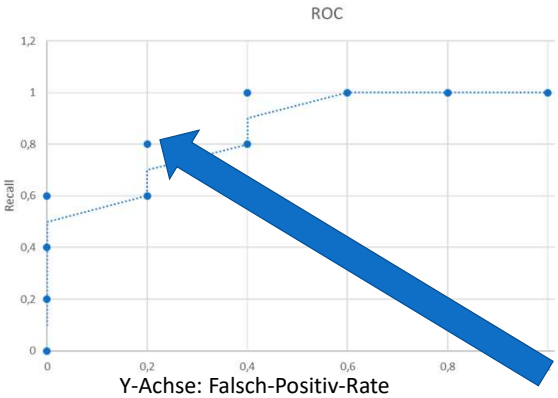
Fall	p(krank)	Krank?	T-P	F-P	F-P-R	Rec. (T-P-R)
			0	0	0	0
Fall 1	0,99	Ja	1	0	0	0,2
Fall 2	0,9	Ja	2	0	0	0,4
Fall 3	0,8	Ja	3	0	0	0,6
Fall 4	0,65	Nein	3	1	0,2	0,6
Fall 5	0,6	Ja	4	1	0,2	0,8
Fall 6	0,5	Nein	4	2	0,4	0,8
Fall 7	0,4	Ja	5	2	0,4	1,0
Fall 8	0,1	Nein	5	3	0,6	1,0
Fall 9	0,02	Nein	5	4	0,8	1,0
Fall 10	0,01	Nein	5	5	1,0	1,0

21

Receiver Operator Characteristic

ROC-Kurve – oder: Wie viel Verwässerung nehme ich hin, um mehr Relevante zu finden?

Y-Achse:
Richtig-Positiv-Rate
= Recall
= Sensitivität



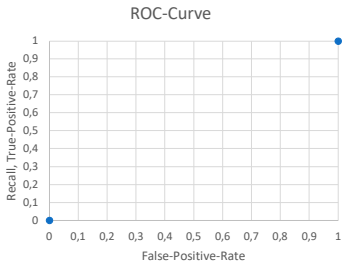
Y-Achse: Falsch-Positiv-Rate
= 1 – Spezifität = 1 – Richtig-negativ-Rate

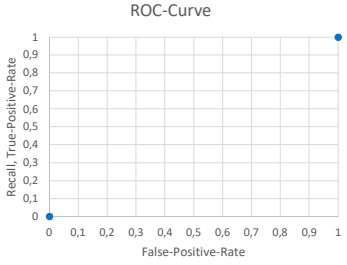
https://en.wikipedia.org/wiki/Receiver_operating_characteristic
Sehr gute Zusammenfassung der Themen (Alle Raten, ROC, etc.)

Wenn insgesamt 6 Fälle von 10 aussortiert werden, dann werden insgesamt 4 kranke (von 5) erkannt mit 80% Recall und auch 2 Gesunde angezeigt (20% Falsch Positiv Rate).

22

Übung: Roc-Curve



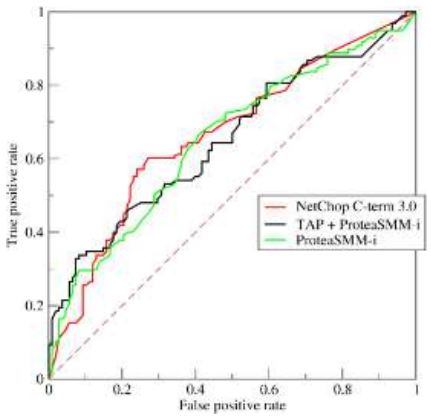


Fall	p(krank)	Krank?	T-P	F-P	F-P-R	Rec. (T-P-R)
Fall 1	0,96	Ja				
Fall 2	0,95	Ja				
Fall 3	0,87	Ja				
Fall 4	0,77	Nein				
Fall 5	0,72	Ja				
Fall 6	0,69	Nein				
Fall 7	0,56	Ja				
Fall 8	0,55	Nein				
Fall 9	0,50	Nein				
Fall 10	0,45	Ja				
Fall 11	0,32	Nein				
Fall 12	0,22	Nein				
Fall 13	0,02	Nein				
Fall 14	0,01	Nein				
Fall 15	0,001	Nein				

23

Gängige Bewertungsmetriken
Für Klassifikationsprobleme – ROC-Kurve (Receiver Operator Characteristic)

Richtig-
Positiv-Rate
= Recall
= Sensitivität

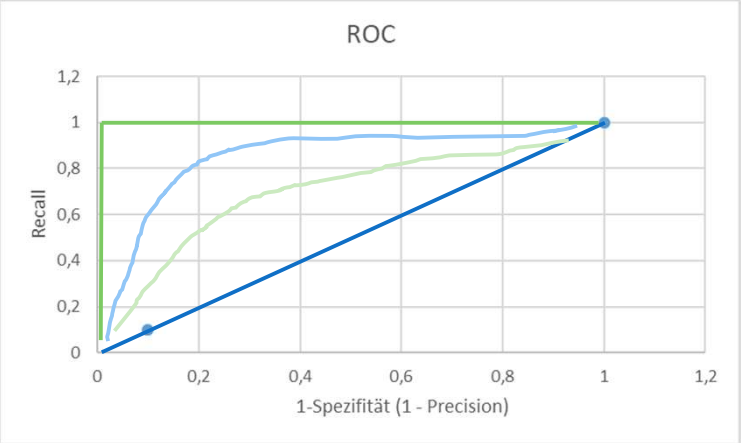


Falsch-Positiv-Rate, 1 – Spezifität = 1 – Richtig-negativ-Rate

Quelle: https://en.wikipedia.org/wiki/Receiver_operating_characteristic Creative Commons

Gängige Bewertungsmetriken
Für Klassifikationsprobleme – ROC-Kurve (Receiver Operator Characteristic)

- Perfekter Klassifikator: Erkennen aller true positives ohne Fehler
- Zufälliger Klassifikator: Linie von 0,0 zu 1,1
- **AUC** (area under curve) als Qualitätsmaß beim Vergleich von Klassifikationsverfahren / Parametern
- **»Sweet Spot«?**



Gängige Bewertungsmetriken
Für Klassifikationsprobleme – mit mehreren Klassen

- **Accuracy** (Korrektklassfikationsrate)
  - Anteil korrekt klassifizierter Objekte über alle Objekte
- **Confusion Matrix** (Konfusionsmatrix, Klassifikationsmatrix)
  - Hilft, bestimmte systematische Probleme bei Mehrklassenproblemen zu finden
- Für einen eigenen Anwendungsfall (egal ob binär oder mit mehreren Klassen)
  - Muss man sich eine **eigene Metrik** erstellen (welche Art Fehler sind wie teuer?)

		Wirklichkeit	Wirklichkeit	Wirklichkeit
		Kategorie 1	Kategorie 2	Kategorie 3
Vor-hersage	Kategorie 1	**Richtig**	?	?
	Kategorie 2	?	**Richtig**	?
	Kategorie 3	?	?	**Richtig**

Beispiel Bildklassifikation

Custom Scoring Functions
Eigene Metriken erstellen

- Wie erwähnt ist es manchmal notwendig und sinnvoll, eigene Bewertungsmetriken zu erstellen.
- In scikit-learn: `make_scorer`
  - Erstellt einen Scorer, den man nutzen kann, um Ergebnisse zu bewerten
  - Für Score (Erfolg): `greater_is_better = True`
  - Achtung: Direkt der sog. Loss (Fehler) während der Optimierung kann nicht einfach ausgetauscht werden (aus Performance Gründen!)
    - Das heißt, es wird immer noch nach den „Standardbewertungswerten“ optimiert im Algorithmus (also Accuracy, oder R^2, oder ..)
    - Wenn man das ändern will, muss man tief in den Code.
- Für einen Score bei der Vergleich von Modellen oder für eine GridSearch: Hier kann man sehr leicht eine eigene Funktion zu definieren, die eine Vorhersage nach eigenen Metriken bewerten kann

```
def my_example_function (y, y_pred) ..
make_scorer (my_example_function, greater_is_better=True)
```

Inhalte der heutigen Vorlesung

- Statistische Weisheit des Tages
- Lernverfahren für Klassifikation
  - Logistische Regression
  - LDA (Eigene Folien und Jankos)
  - QDA
- Bewertung von Klassifikation
  - Accuracy
  - Sehr viele andere Metriken
  - ROC-Curve
  - Custom Metriken
- **Abschluss Modellierung und Evaluation**
  - **Bootstrap & Varianz**
  - **Feature Importance**
  - **Grid Search**
  - **Interpretierbarkeit und Qualität**
  - **Abhaken der Algorithmen**
  - **Round-Trip**

28

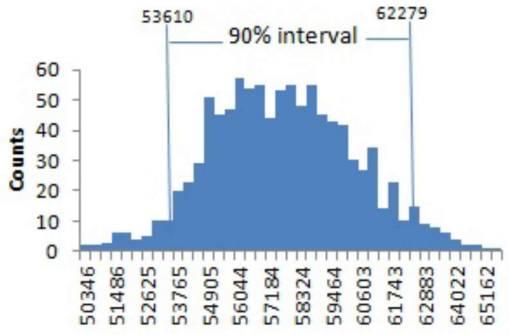
Performance-Explainability-Trade-Off

An Introduction to Statistical Learning, J. Gareth Et. Al., Springer, 2017

29

Abschätzung der Konfidenzintervalle
mit Cross Validation und Bootstrap

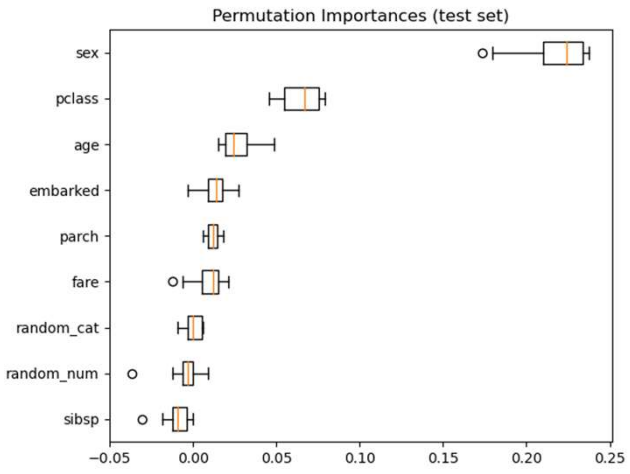
- Wie schätzt man die Varianz einer Vorhersage?
  - Bei linearer Regression: Mathematisch
  - Was ist mit nicht-parametrischen Verfahren?
- Wir haben sehr viel Rechenleistung!
  - Bootstrap – generiere neue Trainingsdatensets aus dem Trainingsdatenset
  - Bestimme die Schwankung
  - Mache das 100 mal
  - Verwende das 95% Intervall
- Alternativ: k-Fold Cross Validation



Literaturempfehlung:
Bildquelle: Bruce, Bruce & Gedeck: Practical Statistics for Data Science, 2017 (https://math2510.coltongrainger.com/books/2017-bruce-and-bruce-pratical-statistics-for-data-scientists.pdf)
https://www.researchgate.net/profile/Ron_Kohavi/publication/2352264_A_Study_of_Cross-Validation_and_Bootstrap_for_Accuracy_Estimation_and_Model_Selection/links/02e7e51bcc14c5e91c000000.pdf

Feature Importance
Hilfe bei der Interpretation nicht interpretierbarer Modelle

- Einige Algorithmen liefern bereits „built-in“ Möglichkeiten, die Wichtigkeit von Features für die Vorhersage darzustellen
  - **Welche?**
- The **permutation feature importance** is defined to be the decrease in a model score when a single feature value is randomly shuffled.
- **Further Reading (hier werden auch Pipelines erklärt):**
  - https://scikit-learn.org/stable/auto_examples/inspection/plot_permutation_importance.html

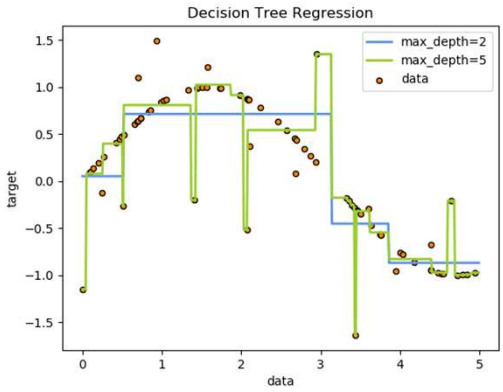


Fortgeschrittene Methoden zur Optimierung
GridSearch

- Wie finde ich die richtigen **Hyperparameter** für ein Algorithmus?
- Beispiel unten: Decision Tree – 13 Parameter
- GridSearch kann verschiedene davon systematisch testen
- **Vorteile / Nachteile? Was braucht man dazu?**

http://scikit-learn.org/stable/auto_examples/model_selection/plot_grid_search_digits.html

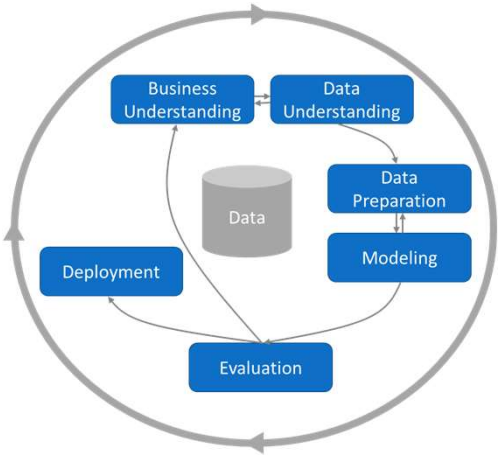
```
class
sklearn.tree.DecisionTreeClassifier(criterion='gini',
splitter='best', max_depth=None, min_samples_split=2,
min_samples_leaf=1, min_weight_fraction_leaf=0.0,
max_features=None, random_state=None,
max_leaf_nodes=None, min_impurity_decrease=0.0,
min_impurity_split=None, class_weight=None,
presort=False)
```

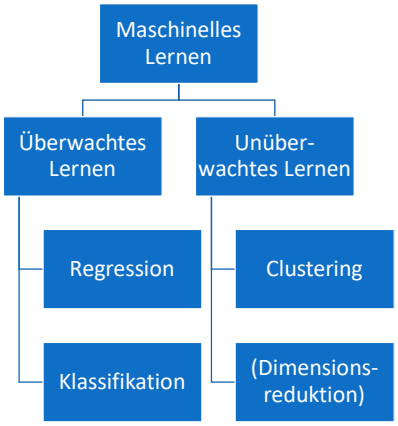


http://scikit-learn.org/stable/modules/tree.html

32

Wiederholung





33

Literaturliste

- [James et al. 2013] Gareth James, Daniela Witten, Trevor Hastie, Robert Tibshirani: An introduction to statistical learning
  - Favorit: Sehr gut gemachte Einführung, jedoch Beispiele in R, verständlich mit Mathematik, als pdf frei erhältlich
- [Hastie et al. 2008] Trevor Hastie, Robert Tibshirani, Jerome Friedman: The elements of statistical learning
  - DIE Referenz, für Mathematiker geschrieben, als pdf frei erhältlich
- [O'Neil and Schutt 2013] Cathy O'Neil and Rachel Schutt: Doing Data Science
  - Spannend zu lesen, teilweise Erfahrungsberichte (durch Drittautoren)
- [Mueller and Guido 2017] Andreas C. Müller & Sasha Guido: An Introduction to Machine Learning with Python
  - Interessant da Python 3 tatsächlich genutzt wird für die Einführung inklusive der üblichen Bibliotheken
- [Grues 2016] Joel Grues (übersetzt von Kristian Rother): Einführung in Data Science
  - Auf deutsch gut übersetzt, nutzt Python für grundlegendes Verständnis ohne die üblichen Bibliotheken, extrem leicht lesbar
- [Alpaydin 2008]: Ethem Alpaydin (übersetzt von Simone linke): Maschinelles Lernen
  - Auf deutsch gut übersetzt, relativ viel Mathematik, in Deutschland scheint das weit verbreitet zu sein
- [Bruce et al. 2020]: Peter Bruce, Andrew Bruce, Peter Gedeck: Practical Statistics for Data Scientists
  - Das einzig wahre Statistikbuch was keines ist
- [Reinhart 2016]: Alex Reinhart (übersetzt von Knut Lorenzen): Statistics done wrong
  - Bevor man wirklich Konfidenzintervalle oder p-Werte angibt und über „Signifikanz" spricht, sollte man das gelesen haben

Literaturliste contd.

- Online-Ressource zu Visualisierung
  - https://www.visualisingdata.com/
- Storytelling with Data [Buch]: Klassiker für Überzeugungsarbeit in Präsentationen von Ergebnissen
  - http://www.bdbanalytics.ir/media/1123/storytelling-with-data-cole-nussbaumer-knaflic.pdf
- Show Me the Numbers [Buch]: Ganz konkrete Tipps für die Praxis
  - https://courses.washington.edu/info424/2007/readings/Show_Me_the_Numbers_v2.pdf
- Now you see it [Buch]: Ebenfalls ganz konkrete Inhalte