

Seminar Statistische Lernverfahren

Klassifikation von Rezensionstypen

Till Gräfenberg, Matthias Häußler, Alexander Kohlscheen, Michael Lau,
Tanja Niklas, Jonathan Schmitz

12. Dezember 2019

Inhaltsverzeichnis

1. Problemstellung
2. Erstellen von Prädiktoren
3. Analysemethoden
 - 3.1 Naive Bayes
 - 3.2 Entscheidungsbaum
 - 3.3 Random Forest
 - 3.4 Support Vector Machine
 - 3.5 weitere Anpassungen und Modelle

Problemstellung

- Ziel: Klassifizierung von Reviews in folgende Typen:

Texttyp	introvertiert	extrovertiert
emotional	stetig	initiativ
rational	gewissenhaft	dominant

- Gegeben: 439 bereits klassifizierte Reviews

Erstellen von Prädiktoren

- ▶ Klassifikation sollte durch verwendete Wörter geschehen
- ▶ Zurückführung auf Grundwörter notwendig
- ▶ Benutzung verschiedener Packages in R bzw. Python ermöglichte verschiedene Verfahren

Erstellen von Prädiktoren

Was ist Stemming?

- ▶ Verfahren, mithilfe dessen man verschiedene Varianten eines Wortes auf ihren gemeinsamen Wortstamm zurückführt
- ▶ Durch Abschneiden von Prä-/In- und Suffixen und Ersetzen von Umlauten, Diphthongen etc. erzeugen von Wortstämmen
- ▶ Beispiele: gelernt → lernen; Wohnungen → Wohnung

Erstellen von Prädiktoren

Stemming

- ▶ Eigene Implementierung nach Vorgabe von COMPEON in R
- ▶ Für Englische Sprache bereits vorgefertigte Tools z.B.
 - ▶ `porterstemmer` von `nltk` in Python
 - ▶ `snowballstemmer` von `nltk` in Python

Erstellen von Prädiktoren

Was ist Lemmatisierung?

- ▶ Das Lemma ist im Bereich der Linguistik die Grundform eines Wortes
→ Wortform z.B. in einem Nachschlagewerk
- ▶ Zurückführung auf grammatikalische Grundformen
- ▶ Lemmatisierung als ein lexikonbasiertes Stemmingverfahren
- ▶ auftretende Probleme des Vorgangs:
 - ▶ Ambiguitäten
 - ▶ Wahl des Lemma eines Wortes (Verbinfinitiv vs. Nomen)
 - ▶ Kompositazerlegung nicht eindeutig (Beispiel: Wachstube)
 - ▶ Simplizia (Beispiel: Kreuzer, Tangente)
 - ▶ Unregelmäßigkeit von Verben im Deutschen

Erstellen von Prädiktoren

Lemmatisierung

- ▶ Erfordert vorgefertigte Packages z.B.
 - ▶ Spacy in Python
 - ▶ nltk in Python
- ▶ Diese lieferten zusätzlich Informationen über die Wortart
- ▶ Auch hier für Englische Sprache ausgereifter als die deutsche Alternative

Erstellen von Prädiktoren

Wortliste

- ▶ Ausgangssituation: 439 Reviews über die Firma COMPEON
- ▶ 8792 Wörter in reviews_preprocessed (mitunter mehrfach)

cleaned_text	preprocessed_text
richtigen	richtig
darlehen	darleh
gewünschte	wunsch
taggenuae	taggenua
gegenüber	genub

Tabelle: Wortliste Beispiele

Erstellen von Prädiktoren

Filterung der Prädiktoren, weitere

- ▶ Nach Erstellung der Grundwörter konnte gefiltert werden, welche Wörter häufig auftraten
- ▶ Denkbare Filtermethoden:
 - ▶ Nur Wörter, die mind. n Mal aufgetaucht sind
 - ▶ Nur Wörter, die in mind. $p\%$ der Reviews verwendet wurden
- ▶ Anschließend Erstellung einer binären Document-Term-Matrix, die kodiert, welche Grundwörter in welchen Reviews auftauchten
- ▶ Alternative: PCA um aussagekräftige „Wörterachsen“ zu bestimmen.

Erstellen von Prädiktoren

PCA - Principal Component Analysis

Ziel: Dimensionsreduktion

Idee: Suche die Datenachsen, auf denen die Varianz am größten ist

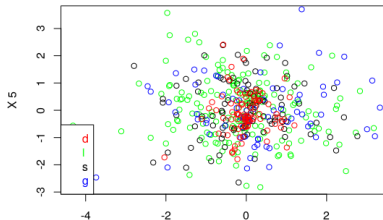
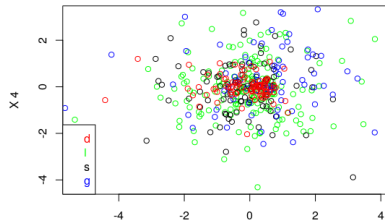
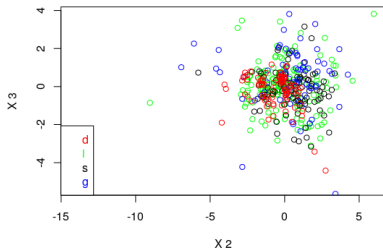
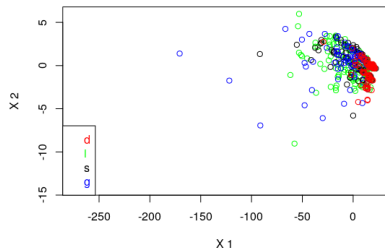
Verfahren:

- ▶ Sei X die DT-Matrix (Spaltenmittelwerte = 0)
- ▶ Bestimme die Kovarianzmatrix $Cov = X^T X$
- ▶ Bestimme die Eigenwerte λ_i und Eigenvektoren v_i von Cov
- ▶ Sei $V = (v_1 | v_2 | \dots)$
- ▶ Transformiere die Daten zu $\hat{X} = XV$

Problem: Die Resultate verlieren an Interpretierbarkeit

Erstellen von Prädiktoren

PCA - Principal Component Analysis



Erstellen von Prädiktoren

PCA - Principal Component Analysis

Fazit:

- ▶ Die Dominanten Reviews haben eine geringere Varianz
- ▶ keine erkennbaren Gruppen
- ▶ Mittelwerte der Gruppen sind ähnlich

Das Verfahren liefert keine besseren Ergebnisse und benötigt nicht wesentlich weniger Variablen.

- ▶ Das Naive Bayes Verfahren fußt auf dem Bayes Theorem

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}$$

bzw. für unabhängige Prädiktoren x_1, \dots, x_n als

$$p(y|x_1, \dots, x_n) = \frac{p(x_1|y) \cdots p(x_n|y)p(y)}{p(x_1, \dots, x_n)} \propto p(x_1|y) \cdots p(x_n|y)p(y).$$

- ▶ Durch Schätzen von $p(y)$ und $p(x_i|y)$ (für die Reviewtypen y) durch die relativen Häufigkeiten, können wir dann Klassifikationen durchführen als

$$\hat{y} = \operatorname{argmax}_y p(y) \prod_{i=1}^n p(x_i|y).$$

Analysemethoden

Naive Bayes

- ▶ Durchführung war in R mit dem Package `caret`, über Python mit `sklearn` möglich. Mit letzterem haben wir jeweils die deutschen und englischen Reviews klassifiziert.
- ▶ Dieses Vorgehen zeigte nur wenig bessere Ergebnisse als eine einheitliche Zuweisung.

(Bester) Naive Bayes, R, Wortaufkommen > 20

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	14	2	8	1		0,412	0,778	0,538
Gewissenhaft	0	0	0	0		n.d.	0	n.d.
Initiativ	4	11	28	17		0,467	0,778	0,583
Stetig	0	1	0	0		n.d.	0	n.d.
Total					0,494	n.d.	0,389	n.d.

Naive Bayes, Python, Wortvorkommen in mind. 1% der Texte,
Lemmatisierung mit spacy, Englisch

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	13	4	12	4		0,433	0,722	0,542
Gewissenhaft	0	3	3	0		0,5	0,214	0,3
Initiativ	4	5	16	11		0,444	0,444	0,444
Stetig	1	2	5	3		0,273	0,167	0,207
Total					0,407	0,413	0,387	0,373

Naive Bayes, Python, Wortvorkommen in mind. 1% der Texte,
Lemmatisierung mit spacy, Deutsch

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	16	2	13	2		0,444	0,889	0,593
Gewissenhaft	0	5	4	1		0,5	0,357	0,417
Initiativ	2	5	16	13		0,444	0,444	0,444
Stetig	0	2	3	2		0,286	0,111	0,16
Total					0,453	0,419	0,45	0,403

Analysemethoden

weitere Anpassungen und Modelle

Mit Naive Bayes und den Wortarten als Prädiktoren lässt sich zuverlässig voraussagen, ob eine Person extrovertiert ist:

Naive Bayes, R, Wortarten, Wortvorkommen in mind. 10 Texten

	extro	intro	Acc.	Prec.	Recall	F1
extrovertiert	30	5		0,714	0,556	0,625
introvertiert	24	27		0,529	0,843	0,65
Total			0,663	0,622	0,7	0,638

Idee:
Nutze die Vorhersage dieses Modells um ein neues Modell anzupassen.

Analysemethoden

weitere Anpassungen und Modelle

Random Forest, R, Wortvorkommen in mind. 10 Texten

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	14	2	8	1		0,56	0,778	0,561
Gewissenhaft	1	4	1	0		0,667	0,286	0,4
Initiativ	3	7	25	15		0,5	0,694	0,581
Stetig	0	1	2	2		0,4	0,111	0,174
Total					0,523	0,532	0,467	0,452

Random Forest mit Naive Bayes, R, Wortvorkommen in mind. 10 Texten

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	15	2	9	1		0,577	0,833	0,682
Gewissenhaft	0	4	1	1		0,667	0,286	0,4
Initiativ	3	8	24	13		0,5	0,667	0,571
Stetig	0	0	2	3		0,6	0,167	0,261
Total					0,535	0,586	0,488	0,479

Resultate Entscheidungsbaum, R, mind. 20 mal Wörter

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	14	3	9	1		51,8%	77,7%	62,1%
Gewissenhaft	0	3	5	5		23,0%	21,4%	22,1%
Initiativ	2	4	19	5		63,3%	52,7%	57,5%
Stetig	2	4	3	7		43,7%	38,8%	41,1%
Total					50,0%	45,4%	47,6%	45,4%

Resultate Entscheidungsbaum, Python, mind. in 1% der Texte, englisch

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	9	4	12	5		30,0%	50,0%	37,5%
Gewissenhaft	2	3	4	2		27,2%	21,4%	23,9%
Initiativ	5	5	12	8		40,0%	33,3%	36,3%
Stetig	2	2	8	3		20,0%	16,6%	18,1%
Total					31,3%	29,3%	30,3%	28,9%

Resultate Entscheidungsbaum, Python, mind. in 1% der Texte, deutsch

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	13	4	16	1		38,2%	72,2%	49,9%
Gewissenhaft	2	4	1	3		40,0%	28,5%	33,2%
Initiativ	3	5	13	7		46,4%	36,1%	40,6%
Stetig	0	1	6	7		50,0%	38,8%	43,6%
Total					43,0%	43,6%	43,9%	41,8%

Analysemethoden

Random Forest

- ▶ Entscheidungsbaum nicht beste Option
 - ▶ gut für Trainingsdaten
 - ▶ nicht flexibel
 - ▶ Probleme mit neuen Datensätzen
- ▶ Generiere neue Testdaten durch Wählen mit Zurücklegen
- ▶ Erzeuge Entscheidungsbaum
- ▶ Generiere so viele Entscheidungsbäume
- ▶ Entscheidung durch Mehrheitsentscheidung
- ▶ R `randomForest` 2000 Bäume, analog in Python

Resultate Random Forest, R, mind. 20 mal Wörter

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	14	2	10	0		53,8%	77,7%	63,5%
Gewissenhaft	2	4	0	1		57,1%	28,5%	28,0%
Initiativ	2	7	25	14		52,0%	69,4%	59,4%
Stetig	0	1	1	3		60,0%	16,6%	26,0%
Total					53,4%	55,7%	48,0%	44,2%

Resultate Random Forest, Python, mind. in 1% der Texte, englisch

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	16	4	14	4		42,1%	88,8%	57,1%
Gewissenhaft	0	6	2	1		66,6%	42,8%	52,1%
Initiativ	1	4	16	9		53,3%	44,4%	48,4%
Stetig	1	0	4	4		44,4%	22,2%	29,6%
Total					48,8%	51,6%	49,5%	46,8%

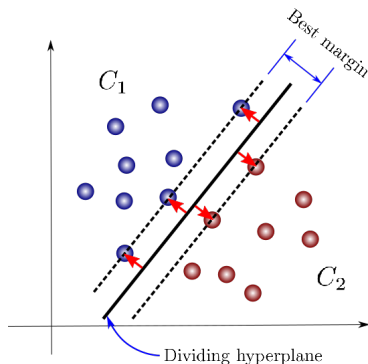
Resultate Random Forest, Python, mind. in 1% der Texte, deutsch

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	15	4	15	2		41,6%	83,3%	55,4%
Gewissenhaft	0	3	4	3		30,0%	21,4%	24,9%
Initiativ	2	4	15	9		50,0%	41,6%	45,4%
Stetig	1	3	2	4		40,0%	22,2%	28,5%
Total					43,0%	40,4%	42,1%	38,5%

Analysemethoden

Support Vector Machine

- ▶ Versucht Entscheidungsgrenze (Hyperebene) zu finden, die die Distanz der nächsten Datenpunkte jeder Klasse zu ihr maximiert
- ▶ Diese nächsten Datenpunkte sind die *Support Vectors*



Quelle: <https://towardsdatascience.com/support-vector-machines-for-classification-fc7c1565e3>

Analysemethoden

Support Vector Machine

- ▶ Verschiedene Kerne (Funktionen) um dem Separierungsproblem gerecht zu werden
- ▶ Kerne projizieren nicht-linear separierbare Daten niedrigerer Dimensionen auf linear-separierbare Daten höherer Dimensionen
- ▶ Vier häufig verwendete Kerne:
 - ▶ Linear $\langle u, v \rangle$
 - ▶ Polynomiell $(\gamma \langle u, v \rangle + r)^d$
 - ▶ Radial $\exp(-\gamma \|u - v\|^2)$
 - ▶ Sigmoidal $\tanh(\gamma \langle u, v \rangle + r)$
- ▶ In R mit `e1071` und in Python mit `sklearn`

Support Vector Machine, R, Wortvorkommen mind. 10 mal,
Radialer Kern, Modifizierter CISTEM-Stemmer, Deutsch

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	14	3	8	1		0,538	0,778	0,636
Gewissenhaft	0	1	1	0		0,500	0,071	0,163
Initiativ	4	10	27	14		0,570	0,750	0,419
Stetig	0	0	0	3		1,000	0,167	0,209
Total					0,5233	0,632	0,441	0,410

Support Vector Machine, Python, Wortvorkommen mind. 20 mal,
Sigmoid Kern, Porter-Stemmer aus nltk, Englisch

	D	G	I	S	Acc.	Prec.	Recall	F1
Dominant	14	2	9	1		0,54	0,78	0,64
Gewissenhaft	0	5	4	1		0,50	0,36	0,42
Initiativ	3	5	19	13		0,47	0,53	0,50
Stetig	1	2	4	3		0,30	0,17	0,21
Total					0,477	0,45	0,46	0,44

Schwierigkeiten

- ▶ Keine eindeutige Klassifikation
 - ▶ Auch für Menschen nicht eindeutig
 - ▶ Teilweise sehr geringe Unterschiede zwischen den Typen
- ▶ Stemming nicht unbedingt eindeutig
 - ▶ Unregelmäßigkeit von Verben im Deutschen
 - ▶ Komposita
- ▶ Geringe Zahl an Trainingsdaten
- ▶ Unbalanciertes Studiendesign
- ▶ Repräsentativität
 - ▶ Introvertierte Kunden schreiben weniger häufig Reviews
 - ▶ Nur positive Bewertungen lagen vor