Statistisches Consulting

Master Studiengang Statistik
Institut für Statistik
Ludwig-Maximilians-Universität München

Statistisches Consulting für die Interhyp AG

Marketing im Internet

Daniel Fuckner d.fuckner@gmx.de
Markus Vogler markus@vogler-lindau.de

Projektpartner:
Interhyp AG

Betreuer:
Dr. Fabian Scheipl

München, 11.11.2011

Abstract:

Background: In patients with

Inhaltsverzeichnis

1 Einleitung

Die Interhyp AG ist Vermittler für private Baufinanzierungen. Das heißt, sie wählt aus einem Angebot von verschiedenen Darlehensgebern die optimale Finanzierungsstruktur für einen Kunden aus. Das Unternehmen wurde 1999 basierend auf der Idee, die Baufinanzierungsbranche zu revolutionieren, von den ehemaligen Goldman-Sachs-Bankern Robert Haselsteiner und Marcus Wolsdorf gegründet. Sechs Jahre später eröffnete die Interhyp AG erste Niederlassungen und konnte gleichzeitig den erfolgreichsten deutschen Börsengang des Jahres verzeichnen. Nach weiteren drei Jahren erfolgte die Übernahme durch ING DIRECT, der weltweilt größten und erfolgreichsten Direktbanken-Gruppe. Heute ist die Interhyp AG der größte Vermittler für private Baufinanzierungen in Deutschland, wurde acht mal in Folge als "Bester Baufinananzierer"(Zeitschrift €, Ausgabe 08/2013) ausgezeichnet und verfügt über mehr als 60 Beratungsstandorte mit über 1.000 Mitarbeitern.
Das primäre Ziel des Marketing der Interhyp AG ist die Kundenakquise. Da etwa 80% aller Kundenanträge online abgeschickt werden, liegt der Fokus der Marketing-Abteilung auf dem Online-Marketing, das über verschiedene Kanäle verfügt. Beispiele sind die Kooperationen (z.B. mit Immobilienscout24), Suchmaschinen (bezahlte Anzeigen und unbezahlte Ergebnisse), Affiliate Marketing (Netzwerk kleinerer Partnerseiten), Display Advertising (diverse Bannerschaltungen), Newsletter und Social Media (vorrangig Facebook und gutefrage.net). Durch Online-Tracking können die Werbekontakte eines potentiellen Kunden mit der Interhyp AG zusammengefasst werden. So entsteht ein Customer Journey (siehe Abbildung ??), dass zum Abbruch oder im Idealfall zum Ausfüllen eines Onlineantrages führt. An dieser Stelle kommt die Refined Labs GmbH ins Spiel.
Customer Journey Grafik aus Folien

Die Refined Labs GmbH ist auf dem Gebiet des Online-Marketing spezialisiert und führender Anbieter für Performance-Marketing-Software. Zum Kundenportfolio zählt unter anderem auch die Interhyp AG, das heißt das Online-Tracking wird von der Refined Labs GmbH durchgeführt und verwaltet. Ein Customer Journey beginnt mit dem ersten Online-Werbekontakt eines potentiellen Kunden mit der Interhyp AG und der damit einhergehenden Erstellung eines Cookies. So können alle weiteren Werbekontakte dem potentiellen Kunden eindeutig zugewiesen werden. Das Tracking endet sobald der potentielle Kunde einen Onlineantrag versendet und damit zum Kunden wird. Wird innerhalb von 90 Tagen kein Onlineantrag versendet, so wird das Cookie automatisch gelöscht und man spricht von einem Abbruch.
Die Interhyp AG ist daran interessiert, ob ein Abbruch von Customer Journeys verhindert werden kann. Hier dann die Zielsetzungen einfügen, die wir gelöst haben.....
Überblick über alle folgenden Kapitel.
Quellen: Projektausschreibung und Folien von Frau Gries; wie angeben?

2 Datenlage

3 Zeitdiskretes Proportional-Hazards-Modell von Cox

3.1 Lebensdauer-Modell

Aufgrund der in Kapitel 2 beschriebenen Datenlage erscheint die Anwendung eines Modells aus dem Feld der Lebensdaueranalyse intuitiv. Es wird die Zeit T bis zu einem Ereignis betrachtet, welches in diesem Fall das Ausfüllen eines Online-Antrages ist. Der Kunde befindet sich während der Beobachtungsspanne im transienten Zustand bis er durch die Konvertierung in den absorbierenden Zustand wechselt, an dem die Beobachtung endet. Wenn nach 255 Kontaktpunkten noch keine Konvertierung eingetreten ist, so endet die Beobachtung. In diesem Fall spricht man von Rechtszensierung.
Die Verteilung der nicht-negativen, stetigen Zufallsvariable T lässt sich durch die Dichte $f_T(t)$, die Verteilungsfunktion $F_T(t)$, die Survivorfunktion $S_T(t)$, die Hazardrate $\lambda_T(t)$ und die kumulierte Hazardrate $\Lambda_T(t)$ beschreiben.

$$S_T(t) = P(T \geq t) = 1 - F_T(t) \tag{1}$$

$$\lambda_T(t) = \lim_{\Delta t \to 0} \frac{1}{\Delta t} P(t \leq T < t + \Delta t | T \geq t \tag{2}$$

$$\Lambda_T(t) = \int_0^t \lambda(u)\, \mathrm{d}u. \tag{3}$$

Die Hazardrate $\lambda_T(t)$ ist also das infinitesimale Risiko im nächsten Moment zu konvertieren, vorausgesetzt man ist bis zum Zeitpunkt t noch nicht konvertiert.
Das Proportional-Hazards-Modell von Cox [5] ist ein multiplikatives Hazardraten-Modell.

$$\lambda(t, x(t)) = \lambda_0(t) \exp(x(t)'\beta) = \lambda_0(t) \exp(\beta_1 x_1(t)) ... \exp(\beta_p x_n(t)) \tag{4}$$

Der Nuisance-Parameter $\lambda_0(t)$, auch Baseline-Hazardrate genannt, ist individuenunabhängig. $x(t)'\beta$ enthält die zeitabhängigen Features $x_i(t)$ und die Effekte β_i.
Die Positionen bilden die Zeitachse $T \in \{1, ..., P = 250\}$. Es handelt sich somit um ein zeitdiskretes Modell. Folglich ist auch die Hazardfunktion nicht stetig, sondern diskret.

$$\lambda(t|x(t)) = P(T = t | T \geq t, x(t)), t = 1, ..., 250 \tag{5}$$

Sie gibt in diesem Fall das Risiko für die Konvertierung zwischen der letzten und der aktuellen Position an, gegeben die aktuelle Position wurde erreicht. Die rechtzensierten Daten müssen in der Form $(t, \delta_i, x_i(t))$, $i = 1, .., n$, $t = 1, ..., t_i$ vorliegen, wobei δ_i der Zensierungsindikator ist. Zudem wird der Ereignisindikator y_{it} definiert, der angibt, ob im Zeitpunkt t_i eine Konvertierung stattfindet. Die Risikomenge R_t ist die Anzahl der

Individuen, die in den vorherigen Positionen noch nicht konvertiert oder zensiert sind.

$$\delta_i = \begin{cases} 1 & \text{Konvertierung in } t_i \\ 0 & \text{Zensierung oder noch keine Konvertierung in } t_i \end{cases} \tag{6}$$

$$y_{it} = \begin{cases} 1 & \text{für } t = t_i \text{ und } \delta_i = 1 \\ 0 & \text{sonst} \end{cases} \quad \text{für } i \in R_t \tag{7}$$

Damit lässt sich die diskrete Hazarfunktion zu

$$\lambda_i(t, x_i(t)) = P(y_{it} = 1 | x_i(t)), t = 1, ..., 250 \tag{8}$$

umschreiben. Für (8) wird ein Logit-Modell angenommen. Es wird also für jede Position die Wahrscheinlichkeit für die Konvertierung geschätzt.

$$\pi_{it} = P(y_{it} = 1 | x_i(t)) = \frac{\exp(\eta_{it})}{1 + \exp(\eta_{it})} \tag{9}$$

$$\eta_{it} = \beta_{0t} + x'_{it}\beta \tag{10}$$

$$\Leftrightarrow \frac{\pi_{it}}{1 - \pi_{it}} = \exp(\beta_{0t}) \exp(x'_{it}\beta) \tag{11}$$

η_{it} ist der Prädiktor und $\exp(\beta_{0t})$ die Baseline-Hazardfunktion. Da für jede Position der Parametervektor β berechnet werden muss, ist die parametrische Maximum Likelihood-Inferenz instabil. **Treppenfunktionen, blablabla, siehe Bericht von Felix und Sebastian**

3.2 Stochastic Gradient Boosting

Stochastic Gradient Boosting ist eine Ensemble-Methoden, die durch mehrfache Anwendung des sogenannten base learners ein Ensemble von Schätzern für eine Prognosefunktion liefert. Durch Aggregation der Schätzer erhält man die endgültige Prognosefunktion. In diesem Fall ist der base learner das zeitdiskrete Cox-Modell.
Gegeben sind die Zielvariable y und die erklärenden Variablen $x = (x_1, ..., x_n)$. Anhand der Trainingsdaten soll eine Prognosefunktion $F^*(x)$ gefunden werden, die x auf y abbildet, so dass der Erwartungswert einer Verlustfunktion $L(y, F(x))$ minimiert wird.

$$F^*(x) = \arg\min_{F(x)} E_{y,x}(y, F(x)) \tag{12}$$

$F^*(x)$ wird mit Hilfe eines additiven Modells der Form

$$F(x) = \sum_{m=0}^{M} \beta_m h(x, a_m) \tag{13}$$

geschätzt, wobei $h(x, a_m)$ der base learner ist und die Koeffizienten β_m und die Parameter a_m anhand des Prinzips des forward stagewise additive modeling an die Trainingsdaten

gefittet werden. Ausgehend von einem Startwert $F_0(x)$ wird für $m = 1, ..., M$ (14) und (15) iteriert.

$$(\beta_m, a_m) = \arg\min_{\beta,a} \sum_{i=1}^{N} L(y_i, F_{m-1}(x_i) + \beta h(x_i, a)) \tag{14}$$

$$F_m(x) = F_{m-1}(x) + \beta_m h(x, a_m) \tag{15}$$

Die Optimierung in (14) erfolgt durch fitten des base learners $h(x, a)$ an die Pseudo-Residuen $\tilde{y}$ mittels der Methode der kleinsten Quadrate in (16) und der Optimierung des leicht lösbaren Problems in (18) mit nur einem reellen Parameter.

$$a_m = \arg\min_{a,\rho} \sum_{i=1}^{N} (\tilde{y}_{im} - \rho h(x_i, a))^2 \tag{16}$$

$$\tilde{y}_{im} = -\left(\frac{\delta L(y_i, F(x_i))}{\delta F(x_i)} \right)_{F(x)=F_{m-1}(x)} \tag{17}$$

$$\beta_m = \arg\min_{\beta} \sum_{i=1}^{N} L(y_i, F_{m-1}(x_i) + \beta h(x_i, a_m)) \tag{18}$$

Zudem wird der base learner in jeder Iteration nur auf einer Stichprobe der Trainingsdaten angewendet [8].

4 Sequential Pattern Mining

4.1 Überblick

Überarbeiten!!!
Sequential pattern mining entdeckt häufige *subsequences* (dt. Teilfolgen) in Datenbanken. Sogenannte *sequence databases* bestehen aus Transaktionen, die jeweils *items* enthalten, welche der Zeit nach geordnet sind. Die Daten lassen sich also mit dem Schema [Transaction/ID, <Ordered Sequence Items>] darstellen.
Ein Anwendungsfeld ist die Warenkorbanalyse. Angenommen es wird das Kaufverhalten in einem Supermarkt einen Monat lang beobachtet, dann könnte [Kunde 1, <(Brot, Milch), (Brot, Milch, Tee), (Zucker), (Milch, Salz)>]; [Kunde 2, <(Brot), (Milch, Tee)>] eine Beispiel-Datenbank sein. Kunde 1 war vier mal im beobachteten Monat im Supermarkt einkaufen, wobei Kunde 2 nur zweimal einkaufen war. Der Kunde kann nur eins oder auch mehrere *items* pro Besuch einkaufen. Im Falle von mehreren *items* spricht man von *itemsets*.
Web usage mining ist das am weitesten verbreitete Anwendungsfeld von *sequential pattern mining* in der Literatur ([11, 18, 9]). Unter der Annahme, dass ein Internetnutzer nur eine Webseite an einem Zeitpunkt aufrufen kann, besteht die Folge von geordneten *items* nur aus einzelnen *items* und nicht aus *itemsets*. Ist also eine Menge von *items* $I = \{a, b, c, d, e\}$ gegeben, die beispielsweise verschiedene Webseiten repräsentieren, so

könnte eine Datenbank mit zwei Nutzern folgendermaßen aussehen: [Nutzer 1, <abedcab>]; [Nutzer 2, <edcaa>] ([12, 3:1-3:2]).

4.2 Problemstellung

Das Problem wurde erstmals im Jahre 1995 in der Arbeit *Mining Sequential Patterns* [1] von Agrawal & Srikant vorgestellt. Es ist eine Datenbank D von Transaktionen gegeben, wobei jede Transaktion eine ID, den Zeitpunkt der Transaktion und die dazugehörigen items enthält. Für jede ID können mehrere Transaktionen existieren allerdings nie zwei oder mehr Transaktionen mit dem selben Zeitpunkt.
Ein itemset $i = (i_1 i_2 ... i_m)$ ist eine nichtleere Menge von items i_j und eine Sequenz $s =< s_1 s_2 ... s_n >$ eine geordnete Liste von itemsets s_j. Ohne Beschränkung der Allgemeinheit wird angenommen, dass die Menge der items auf eine Menge von fortlaufenden ganzen Zahlen abgebildet wird.
Eine Sequenz $< a_1 a_2 ... a_n >$ heißt Subsequenz einer anderen Sequenz $< b_1 b_2 ... b_m >$ wenn ganze Zahlen $i_1 < i_2 < ... < i_n$ existieren, so dass $a_1 \subseteq b_{i_1}$, $a_2 \subseteq b_{i_2}$, ..., $a_n \subseteq b_{i_n}$ gilt. Beispielsweise ist $< (1)(58)(27) >$ Subsequenz von $< (15)(248)(358)(27) >$, da $(1) \subseteq (15)$, $(58) \subseteq (358)$ und $(27) \subseteq (27)$ aber $< (27) >$ keine Subsequenz von $< (2)(7) >$ und umgekehrt. Eine Sequenz heißt maximal in einer Menge von Sequenzen, wenn sie von keiner Sequenz in dieser Menge Subsequenz ist.
Die Transaktionen einer ID können als Sequenz verstanden werden. Jede Transaktion entspricht dabei einem itemset und die nach den Zeitpunkten $T_1, T_2, ..., T_n$ geordneten Transaktionen entsprechen einer Sequenz $< itemset(T_1) itemset(T_2) ... itemset(T_n) >$.
Der support einer Sequenz s ist der Anteil der IDs, die s unterstützen, das heißt deren Sequenz Subsequenz von s sind. Eine Sequenz mit k items wird auch k-Sequenz genannt.
Das Problem des sequential pattern mining besteht darin, in einer Datenbank D unter allen existierenden Sequenzen die maximalen Sequenzen zu finden, deren support größer oder gleich einem festgelegten minimalen support min_sup ξ ist.

4.3 Auswahl geeigneter Algorithmen

In den letzten zwei Jahrzehnten wurden im Forschungsfeld des sequential pattern mining eine Vielzahl von Algorithmen entwickelt ([16, 19, 1, 17, 13, 2, 10, 15, 14, 6, 4, 20, 7]; **die die an anderer stelle zitiert werden, hier später löschen**). Im Folgenden werden verschiedene Arten von Algorithmen sowie Merkmale anhand derer deren Qualität bewertet werden kann vorgestellt. Anschließend wird ausgearbeitet, welche dieser Algorithmen für die vorliegenden Daten am besten geeignet sind. Die Algorithmen lassen sich in die Kategorien *apriori-based*, *pattern-growth* und *early-pruning* einteilen, wobei es auch hybride Algorithmen gibt.

4.3.1 Apriori-Based

Apriori-basierte Algorithmen waren Mitter der neunziger Jahre die ersten auf dem Feld des sequential pattern mining und hängen vor allem von der apriori Eigenschaft ab. Diese besagt, dass alle nichtleeren Teilmengen eines häufigen itemsets ebenfalls häufig sein müssen. Sie können als monoton fallend verstanden werden da für jede Sequenz, die den minimalen support $min\_sup\ \xi$ nicht erfüllt auch all deren Übermengen diesen nicht erfüllen. Das bedeutet, es gilt Gleichung (19).

$$X \subseteq Y \Rightarrow support(Y) \leq support(X) \text{ für } X, Y \subseteq J, J: \text{ Menge aller items in } D \quad (19)$$

Dmit sind für eine nicht häufige Sequenz auch alle Übermengen nicht häufig und für eine häufige Sequenz alle Untermengen ebenfalls häufig. Apriori-Algorithmen sind durch drei Hauptmerkmale gekennzeichnet. Ein Nachteil ist die *generate-and-test*-Eigenschaft. Sie bringt mit sich, dass ausgiebige join-Operatoren verwendet werden, die die Sequenzen in jedem Schritt um nur ein item vergrößern und gegen den minimalen support testen. Dadurch ensteht eine enorme Anzahl von Kandidat-Sequenzen, die bereits zu Beginn des Algorithmus viel Speicher aufbrauchen. Sind beispielsweise n häufige 1-Sequenzen und $min\_sup = 1$ gegeben. Dann werden $n^2 + \binom{n}{2}$ Kandidat-2-Sequenzen und $\binom{n}{m}$ Kandidat-m-Sequenzen erzeugt.

4.4 SPADE

Im Folgenden wird der SPADE Algorithmus [20] vorgestellt, der in dem R-Paket *arules-Sequences* [3] implementiert ist. Dieser sucht häufige Sequenzen von itemsets, das heißt er kann auch mit Daten umgehen, die mehrere items pro Zeitpunkt enthalten. Da ein Benutzer nur eine Webseite zum exakt selben Zeitpunkt besucht werden kann, ist diese Eigenschaft in diesem Fall nicht relevant. Die Daten werden vertikal bezüglich der IDs angeordnet. Die zweite Spalte enthält die Zeit des Kontaktes, die dritte Spalte die Anzahl der items zu diesem Zeitpunkt und die vierte Spalte die eigentlichen items. Bei den gegebenen Daten sind die Zeitpunkte die Positionen und damit pro ID von eins bis zur Anzahl der Positionen für die ID durchnummeriert. Die Anzahl der items in der dritten Spalte ist immer gleich eins und die vierte Spalte enthält pro Zeile nur ein item (Tabelle 4.4).

4.4.1 Berechnung des Supports

In diesem Abschnitt wird beschrieben, wie der Support einer Sequenz berechnet wird.

ID	Zeitpunkt	Anzahl Items	Items
1	1	1	A
1	2	1	C
1	3	1	A
1	4	1	B
2	1	1	C
3	1	1	D
3	2	1	A

Tabelle 1: Beispiel für eine Input-Datenbank für den SPADE-Algorithmus

A		B		C		D	
ID	Zeitpunkt	ID	Zeitpunkt	ID	Zeitpunkt	ID	Zeitpunkt
1	1	1	4	1	2	3	1
1	3			2	1		
3	2						

Tabelle 2: ID-Listen für die items

Literatur

[1] R. Agrawal & R. Srikant. „Mining sequential patterns“. In: *Proceedings of the 11th Conference on Data Engineering (ICDE'95)* (1995), S. 3–14.

[2] J. Ayres et al. „Sequential pattern mining using a bitmap representation“. In: *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2002), S. 429–435.

[3] Christian Buchta, Michael Hahsler & with contributions from Daniel Diaz. *arulesSequences: Mining frequent sequences*. R package version 0.2-6. 2014. URL: `http://CRAN.R-project.org/package=arulesSequences`.

[4] D.-Y. Chiu, Y.-H. Wu & A. L. P. Chen. „An efficient algorithm for mining frequent sequences by a new strategy without support counting“. In: *Proceedings of the 20th International Conference on Data Engineering* (2004), S. 375–386.

[5] D. R. Cox. „Regression Models and Life-Tables“. In: *Journals of the Royal Statistical Society* 34.2 (1972), S. 187–220.

[6] M. El-Sayed, C. Ruiz & E. A. Rundensteiner. „FS-Miner: Efficient and incremental mining of frequent sequence patterns in web logs“. In: *Proceedings of the 6th Annual ACM International Workshop on Web Information and Data Management* (2004), S. 128–135.

[7] C. I. Ezeife, Y. Lu & Y. Liu. „PLWAP sequential mining: Open source code“. In: *Proceedings of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementation* (2005), S. 26–35.

[8] J. H. Friedman. „Stochastic Gradient Boosting“. In: *Computational Statistics and Data Analysis* 38.4 (2002), S. 367–378.

[9] B. Goethals. „Frequent set mining“. In: *The Data Mining and Knowledge Discovery Handbook* (2005), S. 377–397.

[10] J. Han et al. „Freespan: Frequent pattern projected sequential pattern mining“. In: *Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2000), S. 355–359.

[11] Y. Lu & C. I. Ezeife. „Position coded pre-order linked WAP-tree for web log sequential pattern minings“. In: *Proceedings of the 7th Pacific-Asia Conference on Knowledge Discovery and Data Mining* (2003), S. 337–349.

[12] Nizar R. Mabroukeh & C. I. Ezeife. „A Taxonomy of Sequential Pattern Mining Algorithms“. In: *ACM Computing Surveys* 43.1 (2010), 3:1–3:41.

[13] F. Masseglia, O. Poncelet & R. Cicchetti. „An efficient algorithm for web usage mining“. In: *Network Inform. Syst. J. 2* (1999), S. 571–603.

[14] J. Pei, J. Hani, B. Mortazavi-Asl & H. Zhu. „Mining access patterns efficiently from web logs“. In: *Knowledge Discovery and Data Mining. Current Issues and New Applications. Lecture Notes Computer Science* 1805 (2000), S. 396–407.

[15] J. Pei, J. Hani, B. Mortazavi-Asl & H. PINTO. „PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth“. In: *Proceedings of the International Conference on Data Engineering* (2001), S. 215–224.

[16] S. Song, H. Hu & S. Jin. „HVSM: A new sequential pattern mining algorithm using bitmap representation“. In: *Advanced Data Mining and Applications* 3584 (2005), S. 455–463.

[17] R. Srikant & R. Agrawal. „Mining sequential patterns: Generalizations and performance improvements“. In: *Proceedings of the 5th International Conference on Extending Database Technology: Advances in Database Technology* 1057 (1996), S. 3–17.

[18] J. Wang & J. Han. „BIDE: Efficient mining of frequent closed sequences“. In: *Proceedings of the 20th International Conference on Data Engineering* (2004), S. 79–90.

[19] Z. Yang, Y. Wang & M. Kitsuregawa. „LAPIN: Effective sequential pattern mining algorithms by last position induction for dense databases“. In: *Advances in Databases: Concepts, Systems and Applications* 4443 (2007), S. 1020–1023.

[20] M. J. Zaki. „SPADE: An efficient algorithm for mining frequent sequences“. In: *Mach. Learn.* 42 (2001), S. 31–60.