Die Dispositionstabelle - Leicht Gemacht

Endri Endri
ProXpress Clinical Research GmbH
Huttenstraße 2
D-10553 Berlin
endri@proxpress-clinical.com

Benedikt Trenggono
ProXpress Clinical Research GmbH
Huttenstraße 2
D-10553 Berlin
benedikt.trenggono@proxpress-clinical.com

Zusammenfassung

Die Dispositionstabelle gehört innerhalb der statistischen Programmierung zu den komplexesten Tabellen, da sie eine Übersicht mehrerer klinischer Daten aus verschiedenen Quellen hergibt. Diese kann eine Übersicht von Patienten sein, die diverse Epochen der Studie abgeschlossen haben und die Anzahl an unerwünschten Ereignissen (Adverse Events) oder unerwünschten ernsten Ereignissen (serious AE) bezogen auf die Medikation umfassen. Einige Dispositionstabellen benötigen die Zusammenführung verschiedener Datensätze bei der weitere mögliche Problemfelder in Betracht gezogen werden müssen, zu der auch die Bedingungen der Imputationsregel gehören. Diese Komplexitäten steigern den Arbeitsaufwand und führen zeitlich zu einem höheren Programmierungsprozess. Diese Präsentation soll die benötigten Schritte erklären, um den Arbeits- und Zeitaufwand für die Erzeugung einer Dispositionstabelle unter Verwendung von SAS als Werkzeug zu vermindern. Hierbei soll die finale Tabelle nicht nur ausgegeben, sondern auch ein Tabellengenerierungscode an Hand der Datenspezifikationen und den Daten selbst automatisiert erzeugt werden. Die Grundlage dieser Idee basiert auf der Abfrage von Inhalten und Strukturen von SAS Datensätzen, die einfache Text-Vergleich-Algorithmen verwenden. Um ein besseres Verständnis für die Algorithmen zu bekommen, wird diese Präsentation einfache Beispiele mit Abbildungen und Erklärungen beinhalten und die einzelnen Schritte der automatischen Programmierung der Dispositionstabelle näher erläutern.
Ziel dieses Papers ist es, Ideen aufzuzeigen, wie die manuelle Programmierung durch die Verwendung von Künstlicher Intelligenz vereinfacht werden kann, um so die Arbeit eines Statistikers zu vereinfachen und mehr Zeit für die Validierung und Nachprüfung zu haben.

Schlüsselwörter: Künstliche Intelligenz, Dispositionstabelle, Code Generator

1 Einführung

Die Dispositionstabelle (oder sogenannte „Übersichtstabelle“) stellt eine Übersicht einer klinischen Studie dar. Um die Dispositionstabelle (mit SAS) programmieren zu können, muss man in einigen Fällen zuerst die klinischen Daten verstehen, welches in manchen Fällen auf Grund der Komplexität zu einem erhöhten Arbeits- und Zeitaufwand führen kann.
Obwohl CDSIC SDTM als Datenstandard in der klinischen Forschung definiert ist, sind klinische Daten (besonders die Strukturen) in vielen Studien sehr unterschiedlich. Das Design der klinischen Studie und die Definition der SDTM Strukturen spielen hier die ausschlaggebende Rolle

2 Dispositionstabelle

Der Aufwand solcher Tabellen ist abhängig vom Inhalt und umfasst zum Beispiel Datensammlungen von mehreren Source – Datensätzen, oder auch Datentransformationen. Als klassisches Beispiel beschreibt die Dispositionstabelle, wie viele Patienten in einer Studie eingeschlossen (screened) und randomisiertwerden und wieviele drop-out (mit den Drop-out Gründen) sind.

Abbildung 1 zeigt eine solche Dispositionstabelle und in Abbildung 2 ist eine Tabelle der abgeleiteten Analyse-Datensätze gezeigt.

Table xx.x.xxxx Disposition of patients

	Treat A	Treat B	Total
Number of patients	X (xxx.x)	X (xxx.x)	X (xxx.x)
Enrolled / signed informed consent [N (%)]			
N	X (xxx.x)	X (xxx.x)	X (xxx.x)
No	X (xxx.x)	X (xxx.x)	X (xxx.x)
Yes	X (xxx.x)	X (xxx.x)	X (xxx.x)
Entered / randomised [N (%)]			
N	X (xxx.x)	X (xxx.x)	X (xxx.x)
No	X (xxx.x)	X (xxx.x)	X (xxx.x)
Yes	X (xxx.x)	X (xxx.x)	X (xxx.x)
Termination of trial medication [N (%)]			
N	X (xxx.x)	X (xxx.x)	X (xxx.x)
Progressive disease	X (xxx.x)	X (xxx.x)	X (xxx.x)
Adverse Event	X (xxx.x)	X (xxx.x)	X (xxx.x)
Protocol Deviation	X (xxx.x)	X (xxx.x)	X (xxx.x)
Lost to follow-up	X (xxx.x)	X (xxx.x)	X (xxx.x)
Refused cont. medicat	X (xxx.x)	X (xxx.x)	X (xxx.x)
Other	X (xxx.x)	X (xxx.x)	X (xxx.x)
Missing	X (xxx.x)	X (xxx.x)	X (xxx.x)

Abbildung 1: Übersicht der Patienten

Table xx.x.xxxx Analysis Sets (All Randomized Subjects)

Analysis Set	Treat A (N=xx)	Treat B (N=xx)	Total (N=xx)
Randomized	xx (xx.x%)	xx (xx.x%)	xx (xx.x%)
Safety Analysis Set[1]	xx (xx.x%)	xx (xx.x%)	xx (xx.x%)
Full Analysis Set[2]	xx (xx.x%)	xx (xx.x%)	xx (xx.x%)
Per Protocol Set[3]	xx (xx.x%)	xx (xx.x%)	xx (xx.x%)
Pharmacokinetics Analysis Set[4]	xx (xx.x%)	xx (xx.x%)	xx (xx.x%)
Pharmacodynamic Analysis Set[5]	xx (xx.x%)	xx (xx.x%)	xx (xx.x%)

[1] All randomized subjects who took at least one dose of study drug.
[2] All randomized subjects who took at least one dose of study drug and had at least one efficacy endpoint evaluation.
[3] xxxx
[4] xxxx
[5] xxxx

Abbildung 2: Analyse-Datensätze

Ein anderes klassisches Beispiel zeigt die Übersicht der unerwünschten Ereignisse (Adverse Events):

Table xx.x.xxxx Overall Summary of Treatment-Emergent Adverse Events (Safety Population)

	Treat A (N = x) n(%)
Treatment emergent adverse event (TEAEs)	x (xx.x%)
TEAEs Related to Treat A	x (xx.x%)
TEAEs, Grade >= 3	x (xx.x%)
TEAEs Related to Treat A, Grade >= 3	x (xx.x%)
Serious TEAEs	x (xx.x%)
Serious TEAEs with fatal outcome	x (xx.x%)
TEAEs leading to an action regarding the dose (dose reduced/increased/interrupted)	x (xx.x%)
TEAEs Related to Treat A leading to an action regarding the dose (dose reduced/increased/interrupted)	x (xx.x%)
TEAEs leading to drug withdrawn/study termination	x (xx.x%)

Abbildung 3: Gesamtzusammenfassung von Adverse Events

3 Bisherige Programmierung

Die Qualität der klinischen Daten sowie die Qualität statistischer Tabellen und der Zeitaufwand sind in der klinischen Forschung sehr wichtig. Deshalb werden bisher verschiedene Programmierungsmethoden in der klinischen Forschung angewendet. Oft wird die „Code Template“ - Methode benutzt, um die Programmierung zu beschleunigen.

Die Verwendung dieser Methode hat viele Vorteile, sie ist jedoch sehr fehleranfällig. Der Programmierer muss sich erst mit dem Programmcode auseinandersetzen, um genau verstehen zu können, wie dieser anzuwenden oder ggfs. zu modifizieren ist.

4 Künstliche Intelligenz

Eine Lösungsmöglichkeit wäre die Entwicklung einer Art „Künstlicher Intelligenz“, die die Programmierarbeit erleichtert. Unter Künstlicher Intelligenz ist zu verstehen, dass das System so intelligent ist, Texte (wie zum Beispiel Mock-Up Tables, Statistical Analysis Plan, usw.) zu analysieren und daraus SAS Programme, basierend auf den klinischen Daten, für die Auswertung zu schreiben.
Die Komplexität eines solchen Systems ist abhängig von den Herausforderungen, vor denen es gestellt wird. Die zwei folgenden Bedingungen werden zunächst am Anfang definiert:

- Unabhängig von Strukturen sowie Inhalten der klinischen Daten
- Freie Texte in den Mock-Up Tables / Statistical Analysis Plan

Diese vervielfachen die Komplexität eines solchen Systems mit Künstlicher Intelligenz. Aus diesem Grund soll das System in drei Prozess-Gruppen eingeteilt werden:

- Extract Information (EI)
  In diesem Prozess werden die klinischen Daten unabhängig ihrer Struktur und der Inhalte „extrahiert“ und in einem „Metadata“- Datensatz gespeichert, um später die Suchfunktionen zu ermöglichen.
- Text Analysis (TA)
- Hierbei werden die Texte aus dem Statistical Analysis Plan (sowie Mock-Up Tables) „gescannt“ und hinterher mit den extrahierten „Metadaten“ aus dem EI Prozess verglichen. Die Details über die Methode des Vergleichsprozesses werden im nächsten Abschnitt genauer beschrieben
- Code Generator
- Die Ergebnisse dieses Prozesses sind SAS-Programme, die vom System vorgeschrieben werden, damit der Programmierer die erstellten Programme für weitere nachfolgende Prozesse (Validierung, usw.) verwenden kann.

4.1 Extract Information (EI)

Wie bereits vorher erläutert, soll das System die klinischen Daten „selbst extrahieren“ können. Um diese Informationen zu extrahieren, werden von SAS einige Funktionen / Prozeduren zur Verfügung gestellt, die für diese Problemstellung angewendet werden können. Folgende Ansätze sollen dieses Problem lösen.

- SASHELP.VCOLUMN / SASHELP.VTABLE
  Mit Hilfe von SASHELP.VCOLUMN können folgende Informationen herausgelesen werden:

- Alle Datensätze innerhalb der SAS-Bibliothek
- Alle Variablennamen von jedem Datensatz innerhalb der SAS-Bibliothek
- Attribute einer Variable wie Variablenlänge, -Typ, -Format und –Label

➢ PROC FORMAT
Studienspezifische Formate können mit Hilfe der Prozedur FORMAT in einem Datensatz gespeichert werden

➢ PROC SORT NODUPKEY
Für eine bessere und komplette Erfassung von Daten können auch die eindeutigen Textvariablen gespeichert werden. Dies ist besonders für größere Datensätze wichtig, um später im weiteren Prozess eine schnellere Suche von bestimmten Informationen zu ermöglichen.

➢ PROC FREQ
Als letzten Schritt des Extract Information (EI) Prozesses wird ein Entity Relationship Modell (ERM) automatisch mit Hilfe der FREQ Prozedur erstellt. Das Entity Relationship Modell (ERM) gibt Informationen wieder, wie die Datensätze miteinander verknüpft sind. Dies kann durch die Suche der eindeutigen Variablen innerhalb eines Datensatzes erleichtert werden.

Am Ende des Extract Information (EI) Prozesses werden alle Informationen in einem „Metadata“- Datensatz gespeichert.

	Member Name	Column Name	Column Label	Column Type	Column Length	Column Format	Column Number in Table
102	CONC	LLQ	LLQ of conc1 or conc2	num	8	-	12
103	CONC	LLQ_DPM	LLQ in dpm	num	8	-	10
104	CONC	MATRIX	matrix	char	20	-	2
105	CONC	PCTPTNUM	Planned Time Point Number	num	8	-	6
106	CONC	SAMCOM		char	42	-	14
107	CONC	SAMPNO	sample number	num	8	-	5
108	CONC	STUDYID	Study Identifier	char	40	-	23
109	CONC	SUBJECT	subject (N)	num	8	-	4
110	CONC	TRTMNT		char	2	-	1
111	CONC	UNIT	unit (cold)	char	8	-	9
113	CONC	WEIGHT	sample weight (g)	num	8	-	17
114	CONC	WEIGHT_C	sample weight (g)	char	8	-	16
115	CY	CYDTC	Date/Time of Collection	char	19	-	7
116	CY	CYSEQ	Sequence Number	num	8	-	4
117	CY	CYSPECIFY	Specification of Reason	char	40	-	8
119	CY	STUDYID	Study Identifier	char	40	-	1
121	CY	VISIT	Visit Name	char	20	-	6

Abbildung 4: Modifizierung von SASHELP.VCOLUMN

	Memb Name	Column Name	value	Column Label
1	AE	AEACN	DOSE INTERRUPTED	Action Taken with Study Treatment
2	AE	AEACN	DOSE NOT CHANGED	Action Taken with Study Treatment
3	AE	AEACN	DOSE REDUCED	Action Taken with Study Treatment
4	AE	AEACN	DRUG WITHDRAWN	Action Taken with Study Treatment
5	AE	AEACN	NOT APPLICABLE	Action Taken with Study Treatment
6	AE	AEACNOTH	CONCOMITANT MEDICATION	Other Action Taken
7	AE	AEACNOTH	CONCOMITANT PROCEDURE	Other Action Taken
8	AE	AEACNOTH	NONE	Other Action Taken
9	AE	AECONTRT	N	Concomitant or Additional Trtmnt Given
10	AE	AECONTRT	Y	Concomitant or Additional Trtmnt Given
11	AE	AEENDTC		End Date/Time of Adverse Event
12	AE	AEENDTC	2012-12-02	End Date/Time of Adverse Event
13	AE	AEENDTC	2012-12-05	End Date/Time of Adverse Event

Abbildung 5: Ausgabe von PROC SORT NODUPKEY

Durch den Vergleich der Abbildungen lassen sich nun alle Variablen und Texte innerhalb des klinischen Datensatzes erfassen.
Es existieren noch weitere Methoden, mit denen klinische Daten unabhängig von deren Strukturen korrekt erkannt werden können. Die bisher genannten Methoden sollen aufzeigen, dass eine Erfassung von klinischen Datensätzen sowie deren Inhalte unabhängig von der Datenbankstruktur kein großes Problem darstellen.

4.2 Text-Analyse (TA)

4.2.1 Ähnlichkeits-Bewertung

Um die Texte aus den gegebenen Mock-Up Tabellen in der klinischen Datenbank zu finden, müssen die Texte nach Ähnlichkeiten untersucht werden. Im Folgenden werden die gängigen Methoden zur Ähnlichkeitsbewertung von zwei Wörtern beschrieben.

- Fuzzy String Methode
  Die unscharfe Suche, auch Fuzzy-Suche oder Fuzzy-String-Suche genannt, umfasst in der Informatik eine Klasse von String-Matching-Algorithmen, die eine bestimmte Zeichenkette (engl. string) in einer längeren Zeichenkette oder einem Text suchen bzw. finden sollen.
- Levenshtein-Distanz Methode
  Die Levenshtein-Distanz (auch Editierdistanz) zwischen zwei Zeichenketten ist die minimale Anzahl von Einfüge-, Lösch- und Ersetz-Operationen, um die erste Zeichenkette in die Zweite umzuwandeln. Benannt ist die Distanz nach dem russischen Wissenschaftler Wladimir Levenshtein, der sie 1965 einführte. Mathematisch ist die Levenshtein-Distanz eine Metrik auf dem Raum der Symbolsequenzen.
- Cosine Similarity Methode
  Cosinus-Ähnlichkeit ist ein Maß für die Ähnlichkeit zweier Vektoren. Dabei wird der Kosinus des Winkels zwischen beiden Vektoren bestimmt. Der Kosinus des eingeschlossenen Winkels Null ist eins; für jeden anderen Winkel ist der Kosinus

des eingeschlossenen Winkels kleiner als eins. Er ist daher ein Maß dafür, ob zwei Vektoren ungefähr in die gleiche Richtung zeigen.

Die drei genannten Methoden sind die gängigsten Methoden zur Ähnlichkeitsbewertung von zwei Wörtern und drücken die Ähnlichkeit von Wörtern in Prozentwerten aus. Da diese klassischen Methoden für allgemeine Fälle entwickelt worden sind, können diese nicht direkt in der klinischen Forschung angewendet werden. Beispielsweise werden „**Normal**“ und „**Abnormal**“ mit einer Ähnlichkeit **von über 70%** bewertet.

Basierend auf dieser Tatsache wurden eigene Berechnungen zur Auswertung der Ähnlichkeit von Wörtern mit Hilfe von SAS-Macros entwickelt. Dabei wurde lediglich eine Zusammensetzung aus Schleifen, dem Buchstabenweisen-Vergleich der Zeichenketten und Gewichtung für die Ähnlichkeit in die Berechnung eingebaut. Da die Berechnung selbst entwickelt ist, kann die Gewichtung der gleichen Strings variiert/proportioniert werden. Somit wird unser Trivialbeispiel, nämlich der Vergleich zwischen „**Normal**“ und „**Abnormal**“ mit einer Ähnlichkeit **von unter 50%** bewertet.

4.2.2 Eigenwörterbuch

Es wird ein Eigenwörterbuch im System entwickelt, um zum einen Synonyme zu erkennen, zum anderen aber auch Studien-spezifische Definitionen übersetzen zu können. Beispielsweise haben „Subject“ und „Patient“ eine gleiche Bedeutung.

In diesem Wörterbuch werden Schlüsselwörter in zwei Typen unterteilt:

- System-Schlüsselwörter
  Zu System-Schlüsselwörtern gehören zum einen allgemeine SAS Prozeduren, Funktionen und Syntax, zum anderen Definitionen von Standardmakros der Abteilung / des Unternehmens, so dass alle Standardmakros auch in diesem System eingebettet werden. Andere Schlüsselwörter sowie Synonyme, welche sehr oft im SAP verwendet werden, wie zum Beispiel: „only“, „at least“, können als Schlüsselwörter definiert werden.
- Inhalts-Schlüsselwörter
  Zu den Inhalts-Schlüsselwörtern gehören alle anderen, die nicht System-Schlüsselwörter sind, wie zum Beispiel studienspezifische Inhalte in den klinischen Daten.

Das Eigenwörterbuch ist eine intelligente Methode, um Definitionen sowie Schlüsselwörter schneller festlegen zu können. Je mehr Studien das System bereits ausgewertet hat, desto intelligenter und schneller wird das ganze System, da eine Memory-, Erinnerungsmethode eingebaut ist.
Diese Memory-, Erinnerungsmethode speichert zum einen alle Wörter, zum anderen auch die Definitionen sowie deren Lösungsmethode in seinem System. Mit diesem Eigenwörterbuch kann gleichzeitig auch die eigene Ähnlichkeits-Berechnung zum großen Teil eingespart werden.

4.2.3 Layout-Analyse

Um Programme für die Berechnung der Dispositionstabelle oder anderer statistischer Tabellen richtig zu schreiben, müssen folgende Punkte durch die Text-Analyse berücksichtigt werden.

- Eingrenzung der Daten Selektierung
  Für bestimmte Tabellen werden nur ein Teil der Daten ausgewertet, wie zum Beispiel: Populationseingrenzung.
- PAGE Variable / BY Variable
  Die statistischen Tabellen können, wie im „Statistical Analysis Plan“ beschrieben, auf mehreren Seiten (bzgl. Variablen) dargestellt werden, zum Beispiel: nach bestimmten Untergruppen.
- Frequence / Incidence / Descriptive Statistics
  Die Tabellen in drei verschiedenen Arten unterteilt:
  - o Frequence: beschreibt die Häufigkeit von Events
  - o Incidence: beschreibt die Anzahl der Patienten mit Events
  - o Deskriptive Statistics berechnet die N, MIN, MEDIAN, MAX, usw. von numerischen Variablen.

4.2.4 Beispiel der Text-Analyse

Eine „safety population“ ist wie folgt im SAP definiert:
“. . . if he/she is *randomized* to a treatment group and has taken at least one unit of the study medication and has post-treatment safety data available”

In diesem Beispiel werden die CDISC SDTM Datensätze verwendet. Somit ist die Lösung wie folgt:

- „randomized” : DS Datensatz
- “treatment group” : DM Datensatz (ARMCD variable)
- “at least” :System-Schlüsselwort als „HAVING MIN (##var##)>= ##“ definiert.
- “study medication” : EX Datensatz
- “post-treatment safety data” : VS Datensatz

Analog werden die Dispositionstabellen, wie in Abbildung 1, Abbildung 2 und Abbildung 3 dargestellt, in SAS Programme „übersetzt“.

In der folgenden Abbildung ist das Ergebnis einer Text-Analyse dargestellt, die den erstellten Programmcode (geschrieben im SQL-Modus) einer Zusammenfassung der unerwünschten Ereignisse zeigt.

```
/************************************************************************
* Project           : xxxxxxxxxxxxxxxx
* Program name      :
*/     %progstart(program = t_15030101_adae_sum.sas)
/* Author             : Endri Endri (EE) - ProXpress Clinical Research GmbH
* Date/version      : 2015-03-26 18:44:02 /v 1.0
* Environment       : SAS 9.2 Windows
* Purpose           : Table 15.3.1.1 Overall Summary of Treatment-Emergent Adverse Events
(Safety population)
* Note              :
************************************************************************/


DATA temp_select;
SET test.ae;
RUN;


/*************************************************************************
* Counting
*************************************************************************/
PROC SQL;
CREATE TABLE adae_100_calc (WHERE =(NOT MISSING(trt01a))) AS

/* 1. Treatment emergent adverse event (TEAEs) */
SELECT DISTINCT 1 AS var1_id
, 'Treatment emergent adverse event (TEAEs)' AS var1
, SUM(temp) AS count
, trt01a
FROM (SELECT DISTINCT usubjid, trt01a, MAX(IFN(aetrtfl = 'Y', 1, 0)) AS temp
FROM  temp_select
GROUP BY usubjid
)
GROUP BY trt01a

/* 2. TEAEs Related to Treat A*/
OUTER UNION CORR
SELECT DISTINCT 2 AS var1_id
, 'TEAEs Related to Treat A' AS var1
, SUM(temp) AS count
, trt01a
FROM (SELECT DISTINCT usubjid, trt01a, MAX(IFN(aetrtfl = 'Y' AND aerel IN ('RELATED' ''),
1, 0)) AS temp
FROM temp_select
GROUP BY usubjid
)
GROUP BY trt01a

/* 3. TEAEs, Grade >= 3*/
OUTER UNION CORR
SELECT DISTINCT 3 AS var1_id
, 'TEAEs, Grade >= 3' AS var1
, SUM(temp) AS count
, trt01a
FROM (SELECT DISTINCT usubjid, trt01a, MAX(IFN(aetrtfl = 'Y' AND INPUT(aetoxgr, best.) >=
3, 1, 0)) AS temp
FROM temp_select
GROUP BY usubjid
)
GROUP BY trt01a

/* 4. TEAEs Related to Treat A, Grade >= 3 */
OUTER UNION CORR
SELECT DISTINCT 4 AS var1_id
, 'TEAEs Related to Treat A, Grade >= 3' AS var1
, SUM(temp) AS count
, trt01a
FROM (SELECT DISTINCT usubjid, trt01a, MAX(IFN(aetrtfl = 'Y' AND INPUT(aetoxgr, best.) >=
3 AND aerel IN ('RELATED' ''), 1, 0)) AS temp
FROM temp_select
```

```
GROUP BY usubjid
)
GROUP BY trt01a

/* 5. Serious TEAEs */
OUTER UNION CORR
SELECT DISTINCT 5 AS var1_id
, 'Serious TEAEs' AS var1
, SUM(temp) AS count
, trt01a
FROM (SELECT DISTINCT usubjid, trt01a, MAX(IFN(aetrtfl = 'Y' AND aeser = 'Y', 1, 0)) AS
temp
FROM temp_select
GROUP BY usubjid
)
GROUP BY trt01a
;
QUIT;
PROC SORT; BY trt01a; RUN;
```

Abbildung 6: Code generiert vom Code Generator

4.3 Code Generator

Der SAS-Code wird nach den zu haltenden Standards und SOPs generiert und nutzt SAS Prozeduren / Funktionen sowie vorhandene Standardmakros optimal aus, welche die notwendigen Berechnungen und Transformationen umsetzen (frequencies, summaries und incidences). Der Code muss außerdem gut strukturiert, klar definiert und kommentiert sein. Um so nah wie möglich einem Code zu ähneln, der validiert werden kann, sollten zudem alle notwendigen Informationen im Kopfteil etc. nicht fehlen.
Durch die Anwendung der Künstlichen Intelligenz wurde der Code in Abbildung 6 innerhalb weniger Minuten produziert.
Die Methode des Code-Generators wurde 2011 auf der PhUSE Konferenz in Brighton, UK [1] präsentiert. Weitere Informationen bezüglich des SAS Programmgenerators können in diesem Paper nachgelesen werden. Es zeigt wie ADaM Datensätze durch Microsoft Excel programmiert werden können.
Mit der Auto-Text Funktion (##text##) kann der Code-Generator zum Beispiel den vordefinierten Kopfteil eines Programms durch die Anwendung von SAS Standard Makros, Funktionen und Prozeduren etc. generieren.

5 Fazit

Durch die Einführung von Standards der Regulierungsbehörden mit der Verwendung von CDISC Strukturen haben sich die Aufgabenfelder für Programmierer in der klinischen Forschung vergrößert. Gleichzeitig ist es deshalb notwendig, dass die Qualität der Programmierung hierbei stetig verbessert wird.
Das Ziel war die vollautomatische Studienanalyse– von SAP, TLF Spezifikationen und Mock-Up Tabellen bis hin zur finalen Analyse – durch die automatisierte Abfrage der benötigten Informationen dieser Dokumente und dessen Anwendung auf eine SDTM Datenbank durch einen komplexen Algorithmus.

Die Vorteile bei der Anwendung der Künstlichen Intelligenz sind:

- WeitereKapazitäten von “Programmierern“
- Weniger menschliche Fehler
- Zeitvorteil
- “lernendes System”
- Standardisierter Aufbau eines Programms
- Verschiedene Methoden in der Text-Analyse und in der Ausgabe der SAS-Programme
- Vorgeschriebene Programme lassen sich bei Fehlern bearbeiten

Weiterhin kann das System auf Grund der Zusammensetzung aus individuellen Programmschritten dabei helfen, einzelne oder mehrere Aufgaben wie folgt zu erfüllen:

- EinfacheAnweisungen
- Mapping und Datenableitung
- Query und Datenvalidierung
- Tabellen-, Abbildungen- und Listenprogrammierung

Die grundlegende Fragestellung ist also, ob es möglich ist, die klinischen Daten und Anforderungen sowie einen Überblick für die klinischen Daten vollständig automatisiert zu generieren, um unabhängig von der Datenstruktur eine komplette automatisierte Studienanalyse mit den gegebenen Informationen durch den SAP umgesetzt werden kann?

Und die Antwort lautet: „**Ja, es geht!**“

Literatur

[1] PhUSE Paper 2011, DH07 – Endri Endri; Rowland Hale: Much ADaM about Nothing - a PROC Away in a Day