Automatisierung der statistischen Auswertung und Notenberechnung für die Prüfung zur Vorlesung *Informatik für Ökonomen I*

Facharbeit im Nebenfach Informatik 30

vorgelegt

von

Florentin Liebmann

Winterthur, Zürich, Schweiz

Matrikelnummer 10-750-180

Angefertigt am
Institut für Informatik
der Universität Zürich
Prof. A. Bernstein, Ph.D.

Betreuer: Patrick M. de Boer

Abgabe der Arbeit: 23. Dezember 2014

Inhaltsverzeichnis

1 Einleitung

Im Rahmen dieser Facharbeit sollte ein Skript oder Programm (im Folgenden: Programm/Applikation) erstellt werden, welches dem Lehrstuhl *Dynamic and Distributed Information Systems Group* am Institut für Informatik der Universität Zürich die statistische Auswertung und Notenberechnung für die Prüfung zur Vorlesung *Informatik für Ökonomen I* erleichtert.

Die statistische Auswertung sollte aus einer Analyse der Prüfungsfragen bestehen, welche es möglich macht, Fragen mit zu tiefem oder zu hohem Schwierigkeitsgrad oder nicht hinreichender Eindeutigkeit zu identifizieren, zu überprüfen und gegebenenfalls auszuschliessen.

Die Hauptanforderung an den Programmteil der Notengebung war, dass die benötigten Mindestpunktzahlen für die Noten 4 und 6 bestimmt (und variiert) und die damit verbundenen Auswirkungen auf die Verteilung der Noten und die Anzahl bestandener und nicht bestandener Prüfungen überwacht werden können.

2 Die Applikation *inf4oec*

In diesem Abschnitt wird kurz und übersichtlich beschrieben, wie die Implementierung der Applikation abgelaufen ist und was daraus aus Benutzersicht resultierte.

2.1 Aufgabenstellung

Ausgangslage für den kompletten Prozess der statistischen Analyse und Notenberechnung ist eine automatisch generierte CSV-Datei, welche pro Frage und Student/in jeweils eine Zeile mit relevanten Informationen (insbesondere Matrikelnummer und Anzahl Punkte) enthält. Davon ausgehend sollten die unter Abschnitt 1 beschriebenen Kernanforderungen erfüllt werden. Für die Entwicklungsphase standen die Daten aus dem Vorjahr[1] zur Verfügung.

Als wichtigste und unverzichtbare Ausgabe am Schluss eines Programmdurchlaufs wurde eine Tabelle (im CSV-Format) mit je einer Spalte für Matrikelnummer, Anzahl Punkte und Note definiert. Eine als wünschenswert aber nicht zwingend eingestufte Anforderung war ein kurzer Bericht zur Auswertung (im PDF-Format), welcher am Schluss des Programms erstellt und gespeichert werden kann.

[1] In der ausgehändigten Datei waren die Matrikelnummern mit einer kryptologischen Hashfunktion des Typs *MD5* versehen. Rückschlüsse auf Personen waren somit ausgeschlossen.

2.2 mögliche Lösungen

Für die Implementierung wurde von Anfang an die freie Programmiersprache *R* favorisiert, welche auf statistische Fragestellungen ausgelegt ist. Dies änderte sich auch im Verlauf der Arbeit nicht. Als IDE fand die Open-Source-Software *RStudio* Anwendung.

Erste Gehversuche und Berechnungen wurden in Form eines Skripts, welches innerhalb der IDE ausgeführt werden kann, getätigt. Zu diesem Zeitpunkt kristallisierte sich bereits eine Idee über die Struktur des Programms und dessen Ablauf heraus. Diese blieb in ihren Grundzügen bis zum Schluss bestehen und wird unter Abschnitt 2.4 genauer beschrieben. Auch einige Itemkennwerte[2] wurden berechnet und ihr Nutzen in Bezug auf die Kernanforderungen des Programms beurteilt.

Zusammenfassend kann gesagt werden, dass einige wichtige Eckpunkte schon bald nach Beginn der Arbeit festgelegt werden konnten. Einzig die Form, in welcher das Programm ausgeführt wurde, war unbefriedigend (z.B. mussten Variablen im eignetlichen Skript - dem Quelltext - gesetzt werden). Es wurden einige Versuche getätigt, mithilfe des *R*-Pakets *tcltk* eine einfache grafische Benutzeroberfläche zu generieren. Diese Versuche waren nicht erfolglos, entsprachen aber noch nicht dem Gewünschten.

2.3 gewählter Lösungsweg

Die im vorherigen Abschnitt beschriebenen Probleme betreffend einer grafischen Benutzeroberfläche konnten durch den Einsatz des *R*-Pakets *shiny* gelöst werden. Eine *shiny*-Applikation besteht in ihrer einfachsten Form aus zwei *R*-Dateien: *ui.r* und *server.r*. Erstere erlaubt es, intuitiv verständlich eine grafische Benutzeroberfläche zu kreieren. Die zweite Datei kommuniziert mit *ui.r* und kann sowohl Daten empfangen als auch senden. Dabei können diese, im Rahmen der Möglichkeiten, welche *R* bietet, auf beliebige Art und Weise manipuliert werden. Damit waren alle Voraussetzungen für die definitive Implementierung gegeben.

2.4 Ergebnisse

Im Folgenden werden die drei Hauptteile der Applikation beschrieben. Dabei steht die Sicht eines Benutzers, welcher die Applikation verwenden möchte, im Vordergrund.

[2] Definition: „Item. Frage oder Aussage in einem Fragebogen bzw. Aufgabe in einem Text.“ (Bortz & Döring, 2006, S. 730)

2.4.1 Applikation starten

Der gesamte Quelltext von *inf4oec* befindet sich auf *GitHub* und kann direkt aus *RStudio* gestartet werden, welches die Applikation herunterlädt und lokal startet. Eine Anleitung dazu ist als Video (*screencast.mov*) im gleichen repository zu finden, wie die Applikation (URL: https://github.com/fliebm/inf4oec.git).

2.4.2 Daten

Im ersten Hauptteil werden die Rohdaten so aufbereitet, dass sie für Itemanalyse und Notenberechnung verwendet werden können.

Schritt 1: Daten einlesen Durch klicken auf *Datei auswählen* kann eine beliebige CSV-Datei, welche der spezifizierten Struktur genügt, geladen werden. Die gelesenen Tabelle wird im Hauptfenster angezeigt. Sollte diese nicht korrekt angezeigt werden, müssen die CSV-Einstellungen in der Seitenleiste links oder die Struktur der Datei angepasst werden.

Schritt 2: Daten transponieren Für jede Matrikelnummer wird eine Zeile mit den erreichten Punkten pro Frage erstellt. Es resultiert eine Tabelle mit $m \times n$ Einträgen, wobei m der Anzahl Studenten/Studentinnen und $n - 1$ der Anzahl Items entspricht. Diese beiden Werte werden im Hauptfenster angezeigt.

Schritt 3: Maximal erreichbare Punkte Die erwartete Anzahl maximaler Punkte pro Item wird aus der zuvor generierten Tabelle gelesen. Sie entspricht dem Maximum für jede Spalte. Die erste Spalte wird ignoriert, da sie die Matrikelnummer beinhaltet. Wird angenommen, dass bei einem Item niemand die maximale Punktzahl erzielt hat, so wären diese Angaben fehlerhaft, was Auswirkungen auf die Itemanalyse hätte. Deshalb müssen unbedingt *alle* erwarteten Maximalpunkzahlen kontrolliert und gegebenenfalls angepasst werden.

2.4.3 Itemanalyse

Der zweite Hauptteil des Programms befasst sich mit der Itemanalyse. Bortz und Döring (2006, S. 730) fassen den Begriff Itemanalyse wie folgt zusammen:

> Überprüfung der Gütekriterien einzelner Items, um die Qualität eines Tests oder Fragebogens einschätzen zu können und ggf. durch Austausch oder Veränderung einzelner Items (Testrevision) zu Verbesserungen zu kommen. Insbesondere werden die Trennschärfe, die Itemschwierigkeit, die Homogenität und die Validität der Items geprüft [...].

Im Kontext dieser Arbeit sind vor allem Schwierigkeiten und Trennschärfen der Items von Interesse. Diese sind intuitiv verständlich und somit gut zu interpretieren. Die auswertende Person sieht, ob ein Item zu einfach oder zu schwierig war und/oder kann Ambiguitäten aufdecken.

Schritt 4: Leere Prüfungen Unter der Annahme, dass keine Person, welche versucht hat, die Prüfung zu lösen, 0 Punkte erreicht, sollten Prüfungen mit einer Gesamtpunktzahl von 0 für die Itemanalyse ausgeschlossen werden. Anderenfalls werden die Itemschwierigkeiten in negativer und die Itemtrennschärfen in positiver Richtung verfälscht.

Schritt 5: Rohwertverteilung Nun kann die Verteilung der erreichten Punkte betrachtet werden. Ob leere Prüfungen angezeigt werden oder nicht, hängt von der Entscheidung im vorherigen Schritt ab. Es sind zwei unterschiedliche Visualisierungen verfügbar: ein Histogramm und ein Quantil-Quantil-Plot. Letzterer bildet die tatsächliche Verteilung gegen eine Normalverteilung (rote Gerade) ab. Abweichungen können so leicht festgestellt werden.

Schritt 6: Itemkennwerte Es werden die beiden unter Abschnitt 2.4.3 angekündigten Itemkennwerte berechnet. Wahlweise können mit dem cutoff-Regler in der Seitenleiste links oder manuell durch Klicken auf die Checkboxen Items hinzugefügt oder entfernt werden.

Itemschwierigkeit Die Itemschwierigkeit ist definiert als „Index [...], der dem Anteil derjenigen Personen entspricht, die das Item richtig lösen oder bejahen“ (Bortz & Döring, 2006, S. 218-219). An gleicher Stelle findet sich auch die Formel für mehrstufige Items gemäss Dahl (1971) und weitere Ausführungen:

$$p_i = \frac{\sum_{m=1}^{n} x_{im}}{k_i \cdot n}$$

n = Anzahl der Personen, welche die Prüfung gelöst haben
x_{im} = Anzahl Punkte, welche Person m bei Item i erreicht hat
k_i = maximal mögliche Anzahl Punkte bei Item i

Es ergeben sich Werte zwischen 0 und 1. Schwierige Items nehmen tiefe und einfache Items hohe Werte an. Werte im mittleren Bereich sind zu bevorzugen.

Itemtrennschärfe Auch die Trennschärfe eines Items wird bei Bortz und Döring (2006, S. 219) definiert: „Der Trennschärfe eines Items ist zu entnehmen,

wie gut das gesamte Testergebnis aufgrund der Beantwortung eines einzelnen Items vorhersagbar ist.“

Berechnet wird sie als Korrelation zwischen den erreichten Punkten für jedes Item i und dem entsprechenden korrigierten Gesamttestwert t. Für intervallskalierte Daten ist die Berechnung der Produkt-Moment-Korrelation angezeigt (vgl. Bortz, 2005, Tab. 6.11). Die Formel für Item i lautet:

$$r_{it} = \frac{cov(i,t)}{s_i \cdot s_t}$$

Korrelationen können Werte zwischen -1 und 1 annehmen. Es gilt: je höher die Trennschärfe, desto besser. Weise (1975) gibt an, dass mittelmässige Werte zwischen 0.3 und 0.5 liegen, hohe Werte hingegen über 0.5.

2.4.4 Notengebung und Bericht

Der Datensatz enthält an dieser Stelle nur noch jene Items, welche ausgewählt wurden. Nun gilt es, eine zufriedenstellende Verteilung der Noten festzulegen und bei Bedarf die Notentabelle als CSV-Datei und/oder den Auswertungsbericht im PDF-Format herunterzuladen.

Schritt 7: Notengebung Die benötigten Mindestpunktzahlen für die Noten 4 und 6 können mittels zweier Schieberegler festgelegt werden. Die Auswirkungen auf die Anzahl bestandener und nicht bestandener Prüfungen sowie auf den Notendurchschnitt können laufend überprüft werden. ACHTUNG: alle Angaben beziehen sich auf *nicht leere* Prüfungen. Per default wird die Note 6 auf das Punktemaximum gesetzt und die Anzahl Punkte für die Note 4 so, dass 80% der Studierenden die Prüfung bestehen.

Im Hauptfenster werden zusätzlich die lineare Funktion der Notengebung und die Verteilung der Noten visualisiert.

Schritt 8: Bericht Der letzte Schritt der Auswertung beinhaltet die Möglichkeit, die Notentablle, welche im Hauptfenster als Vorschau angezeigt wird, und/oder den Auswertungsbericht herunterzuladen.

2.5 Tests

inf4oec wurde auf zwei Arten getestet. Funktionen, welche nicht einer *R*-Library entstammen, durchliefen einen Komponententest. Die Applikation als Ganzes wurde in der Folge geprüft, indem die Notengebung aus dem Vorjahr zu reproduzieren versucht wurde.

2.5.1 Komponententest

Im Laufe der Implementierung wurden sechs Funktionen definiert. Diese sind am Anfang des Quelltextes in der Datei *server.r* zu finden. Die Funktionen wurden jeweils mit zwei Fällen getestet, von welchen der erste den erwarteten Wert zurückgeben und der zweite eine Fehlermeldung produzieren sollte. Alle Tests verliefen positiv.

Der Ordner *unit_tests* befindet sich im unter Abschnitt 2.4.1 beschriebenen repository auf *GitHub* und ist wie folgt aufgebaut:

- Der Ordner *functions* enthält für jede der sechs Funktionen eine *.R*-Datei mit der dazugehörigen Definition.
- Der Ordner *tests* enthält für jede der sechs Funktionen eine *.R*-Datei mit den Definitionen der dazugehörigen Testfälle.
- Die Datei *run_tests.r* ist für die Ausführung aller Tests zuständig.

Es wird der komplette Ordner *unit_tests* benötigt, um die Tests nachzuvollziehen. Zur Ausführung reicht es aber, die Datei *run_tests.r* in *RStudio* zu öffnen und das Skript auszuführen bzw. den Instruktionen zu folgen.

2.5.2 Systemtest

Der Systemtest sollte sicherstellen, dass die Applikation den Anforderungen im eigentlichen Anwendungsfeld einwandfrei entgegentreten kann. Es wurde daher versucht, die Noten der letztjährigen Prüfung zu errechnen. Der Test verlief bis auf zwei Studierende, für welche eine abweichende Bewertung resultierte, erfolgreich.

Die Ursachen für die beiden Abweichungen sind zum Zeitpunkt der Abgabe dieser Arbeit noch unklar. Es wird diesem Sachverhalt jedoch nachgegangen.

2.6 Probleme

Insgesamt sind zwei problematischen Episoden zu berichten. Die erste betrifft die Suche nach einer geeigneten Möglichkeit zur Umsetzung einer grafischen Benutzeroberfläche und die damit verbundenen Unklarheiten. Die Kombination einer nicht zu Beginn der Arbeit gefällten Entscheidung und einer Unterschätzung des Aufwands führte zu einer Schlussphase, in welcher die Arbeitsbelastung deutlich höher war, als in der Anfangsphase.

Die zweite problematische Episode geht auf den zwar grösstenteils, aber nicht restlos positiv verlaufenen Systemtest zurück, welcher bereits im vorherigen Abschnitt geschildert wurde.

3 Zusammenfassung und eigene Einschätzung

Zusammenfassend kann ich festhalten, dass die Arbeit für mich in jeder Hinsicht eine Bereicherung darstellte. Hierbei gilt es, die Zusammenarbeit mit dem Betreuer dieser Arbeit, welchem ich an dieser Stelle meinen herzlichen Dank aussprechen möchte, und natürlich die neu erlangten Programmierkenntnisse besonders herauszustreichen.

Den Nutzen für den Lehrstuhl *Dynamic and Distributed Information Systems Group* kann ich zum aktuellen Zeitpunkt nicht abschätzen; ebenfalls nicht, ob diese Arbeit den Erwartungen gerecht wurde. Natürlich hoffe ich sehr, dass die statistische Auswertung und die anschliessende Notenberechnung eine Vereinfachung erfahren haben und damit ein Mehrwert für den Lehrstuhl und die beteiligten Personen aus dieser Arbeit hervorgeht.

Die Kernanforderungen, welche zu Beginn des Semesters formuliert wurden, können - so glaube ich - generell als erfüllt betrachtet werden. In diesem Sinne: enjoy *inf4oec*.

4 Literaturverzeichnis

Bortz, J. (2005). *Statistik für Sozial-und Humanwissenschaftler*. Heidelberg: Springer Medizin Verlag.

Bortz, J. & Döring, N. (2006). *Forschungsmethoden und Evaluation: für Human- und Sozialwissenschaftler*. Heidelberg: Springer Medizin Verlag.

Dahl, G. (1971). Zur Berechnung des Schwierigkeitsindex bei quantitativ abgestufter Aufgabenbewertung. *Diagnostica*, *17*, 139–142.

Weise, G. (1975). *Psychologische Leistungstests: ein Handbuch für Studium und Praxis. 1. Intelligenz, Konzentration, spezielle Fähigkeiten*. Göttingen: Hogrefe.

Anmerkung: Aufgrund der an Gültigkeit nicht verlierenden Grundlagen zu Itemanalyse und verwandten Themen wurden für diese Arbeit nicht die jüngsten Auflagen der Bücher von Bortz (2005) und Bortz und Döring (2006) verwendet.