Evaluation einer Statistiklehrveranstaltung mit dem JACK R-Modul

Benjamin Otto[1], Till Massing[2], Nils Schwinning[1], Natalie Reckmann[2], Alexander Blasberg[2], Sandy Schumann[2], Christoph Hanck[2] und Michael Goedicke[1]

Abstract: Neben klassischen Multiple-Choice oder Lückentextaufgaben bieten moderne elektronische Lernsysteme die Möglichkeit, Programmieraufgaben automatisiert zu bewerten. Das E-Assessment System JACK wurde um einen Aufgabentyp für Programmieraufgaben mit der statistischen Programmiersprache R ergänzt, da diese innerhalb der statistischen Gemeinschaft weit verbreitet ist. Sie wird daher auch Studierenden der Wirtschaftswissenschaften an der Universität Duisburg-Essen bereits in der Studieneingangsphase vermittelt. Dieser Artikel schildert die Nutzung des neuen Aufgabentyps in einem Erstsemester-Statistikkurs und zeigt, dass ein positiver Effekt von Programmieraufgaben auf klassische Klausuraufgaben festzustellen ist.

Keywords: E-Assessment, R, Statistik, Lehrveranstaltung, Evaluation, Autograder

1 Einleitung

Der Einsatz von elektronischen Übungssystemen in der Studieneingangsphase ist ein weit verbreitetes Konzept, um wachsenden Studierendenzahlen angemessen begegnen zu können. Der in vielen Studienfächern übliche, klassische Übungsbetrieb wird dabei durch Online-Assessments ersetzt, um Studienanfängern die Möglichkeit zu geben Lerninhalte zu üben. Neben der schnellen Rückmeldung in Form von automatischen Bewertungen spielt auch die hohe Flexibilität bei dem Einsatz von E-Assessment Systemen eine Rolle, denn Studierende sind in der Lage beliebig oft und zu jeder Tageszeit mit ihnen zu arbeiten. Konzepte reichen von klassischen Aufgabentypen, wie Multiple-Choice-Aufgaben[3] oder Lückentextaufgaben[4] bis hin zu Programmieraufgaben, z.B. mit Java [St16].

In den Vorlesungen der Statistik an der Universität Duisburg-Essen wird das selbst entwickelte E-Assessment-System JACK zur Unterstützung des Vorlesungsbetriebs eingesetzt. Das Konzept beinhaltete bisher wöchentliche formative und summative Assessments in Form von parametrisierten Lückentextaufgaben [Sc17]. Analysen haben gezeigt, dass die Note der Abschlussklausur mit dem Lernerfolg bei diesen Assessments

[1] Universität Duisburg-Essen, Paluno - The Ruhr Institute for Software Technology, Gerlingstraße 16, 45127 Essen, vorname.nachname@paluno.uni-due.de
[2] Universität Duisburg-Essen, Wirtschaftswissenschaften, Universitätsstr. 12, 45117 Essen, vorname.nachname@vwl.uni-due.de
[3] Z.B. Moodle: https://moodle.de/
[4] Z.B. Moodle, Stack: http://stack.bham.ac.uk/, ActiveMath: http://www.activemath.org/

positiv korreliert [Ma17]. Jedoch stoßen die Lückentextaufgaben bei komplexeren Problemstellungen an ihre Grenzen. So spielt die Programmiersprache R [RC17] in der Statistikausbildung eine wichtige Rolle und dient als Hilfswerkzeug für viele Berechnungen. Studierende lernen daher bereits in Einführungsveranstaltungen den Umgang mit der Sprache, da ihnen dadurch die Anwendung der erlernten Methoden erleichtert wird. Es wäre somit wünschenswert, auch die R-Programmierung durch E-Assessments unterstützen zu können.

Dieser Artikel beschreibt einen neuen Aufgabentyp, welcher es ermöglicht, Aufgaben zu stellen, die mit R zu lösen sind. Der Aufgabentyp wurde dem E-Assessment-System JACK hinzugefügt und im Wintersemester 2016/17 in das bestehende Konzept von semesterbegleitenden formativen und summativen E-Assessments in einer Vorlesung zur Deskriptiven Statistik integriert. Lernende müssen dabei zu einer vorgegebenen Aufgabenstellung R-Code schreiben und können diesen anschließend hochladen. Das System prüft diese Lösung mittels dynamischer und statischer Tests und gibt den Lernenden ein detailliertes Feedback. Es wurden insgesamt 72 Aufgaben für die Vorlesung erstellt und eingesetzt. Wir evaluieren die Nutzung dieser Aufgaben durch die Studierenden und beschreiben ihren Lernerfolg hinsichtlich der Abschlussklausur.

Der Rest dieses Artikels gliedert sich wie folgt: Zunächst werden in Kapitel 2 verwandte Arbeiten beschrieben. Kapitel 3 erläutert den neuen Aufgabentyp und Kapitel 4 das Konzept der Vorlesung sowie den Einsatz der verschiedenen Assessmentformen. In Kapitel 5 wird das Lernverhalten der Studierenden während des Semesters analysiert und mit den Ergebnissen der Abschlussklausur verglichen. Zum Schluss wird in Kapitel 6 ein Fazit gezogen und ein Ausblick auf weitere Forschung gegeben.

2 Verwandte Arbeiten

Der Einsatz von elektronischen Übungssystemen in universitären Lehrveranstaltungen kann sowohl das Themenverständnis als auch die Leistung in Prüfungen der Studierenden erheblich verbessern [CM02], [BB14], [Me13]. Für den Bereich der Statistik hat [So11] dies in einer Meta-Analyse ausführlich untersucht und kommt zu dem Ergebnis, dass die simultane Verwendung von Präsenzveranstaltung und Online-Assessments einen positiven Effekt auf den Lernerfolg von Studierenden hat. Insbesondere bei datenintensiven Disziplinen zeigt sich jedoch, dass eine alleinige Vermittlung von traditionellen Lehrbuchinhalten heute nicht mehr ausreicht und die Datenanalyse sowie -aufbereitung mit entsprechenden Software-Programmen eine wichtige zu beherrschende Fähigkeit darstellt [TL12], [Ch07].

In ersten Studien konnte gezeigt werden, dass die Verwendung von statistischen Softwareprogrammen zu Analyse- und Visualisierungszwecken das Verständnis für komplexe Zusammenhänge noch weiter verbessert und bessere Prüfungsergebnisse erzielt werden. So erforschte [Ba05] in einer einführenden Statistiklehrveranstaltung, ob Studierende mit Hilfe des Software Programms SPSS ein besseres statistisches

Verständnis erhalten. In der abschließenden Prüfung erreichten Studierende, die während des Semesters wöchentlich in einem Computerlabor Zugriff auf SPSS hatten, eine höhere Punktzahl als jene, die nur die klassische Vorlesung besuchten. In diesem Kontext erscheint es sinnvoll, das Lernen der theoretischen Konzepte auf der einen Seite mit der praktischen Anwendung dieser auf der anderen Seite zu verknüpfen. Entsprechende Möglichkeiten, wie diese Verknüpfung in einer E-Assessment-Umgebung realisiert werden kann, werden von [Mc15] ausführlich vorgestellt. Neben der konkreten Ausgestaltung des Übungssystems stellt sich hierbei die Frage nach der Wahl des zu verwendenden Programms. Aufgrund seiner Popularität, Flexibilität sowie Erweiterbarkeit bietet sich hierfür R an [Mu15]. Eine entsprechende Lehrplattform existiert in der Form von datacamp [Da17] bereits, allerdings liegt der Hauptfokus dort eher auf dem Aspekt des Selbstlernens und weniger auf dem summativen Assessment. Eine entsprechende, auf Hochschullehre zugeschnittene, E-Assessment-Plattform mit integrierter R-Anbindung existiert nach bestem Wissen der Autoren noch nicht.

3 Der JACK-R-Aufgabentyp

Im Rahmen des ProViel-Projekts zur Verbesserung der Lehramtsausbildung wurde ein neuer Aufgabentyp entwickelt, der Aufgaben für die statistische Programmiersprache R im E-Assessment-System JACK ermöglicht. In diesem Aufgabentyp sehen die Studierenden zunächst eine Aufgabenstellung, welche in R ausgerechnete (z.B. randomisierte) Werte enthalten kann. Hierbei steht den Studierenden ein Texteingabefeld zur Verfügung, welches sowohl der Eingabe und dem Einreichen der Lösung dient, als auch als Texteingabefeld für die Live-R-Konsole. Diese ermöglicht es (ohne Installation zusätzlicher Software) den Studierenden R-Programme im Browser zu schreiben und auch live auszuwerten. Dazu steht ein Editor mit Syntax-Highlighting zur Verfügung. Durch Klicken des Buttons „Auswerten“ wird der gesamte angezeigte Code an einen R-Server gesendet, der die Programmausführung übernimmt und die Rückgaben in das Ausgabefeld einträgt. So sind Studierende in der Lage, ihre Lösungen direkt auf dem Server zu testen und sich die Rückgaben anzusehen. Die Studierenden bekommen dadurch sofort einen Eindruck davon, ob die Syntax ihres Programms korrekt ist, da alle in R auftretenden Fehler an die Benutzeroberfläche durchgereicht werden. Aber es kann auch unverzüglich gesehen werden, ob die Programmausgabe einerseits das erwartete Format zurückgibt oder andererseits im erwarteten Zahlenbereich liegt. Durch die schnelle und einfache Berechnung von R-Befehlen im Browser sollen die Studierenden zum Ausprobieren angeregt werden.

In Abb. 3.1 ist eine typische Aufgabe dargestellt. In dieser Aufgabe sollen die Quintile eines Datensatzes mit R dargestellt werden. Hier hat ein Studierender zwar die richtige R-Funktion (`quantile`) gefunden, aber dieser wurden noch nicht die richtigen Parameter für die Aufgabenstellung (`probs=seq(0,1,0.2), type=1`) übergeben. Da der Studierende den Button „Auswerten“ geklickt hat, ist im unteren Textfeld die Ausgabe von R zu sehen. Eine Besonderheit ist hier, dass dem Studierenden schon

vorausgefüllter Code (der sog. „Initial-Code“ `urliste <- c(11,…,15)`) in dem Texteingabefeld präsentiert wurde. Dieser kann optional vom Lehrenden angegeben werden und auch dynamisch berechnete Werte enthalten.

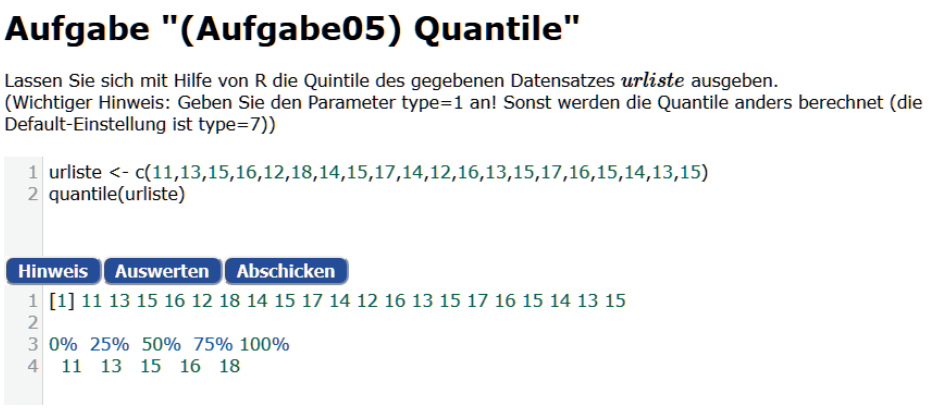

Abb. 3.1: Aufbau einer Jack-R-Aufgabe

Automatische Bewertung der Benutzereinreichungen und Feedback

Um große Anzahl an Benutzereinreichungen sowohl automatisch zu bepunkten, als auch dem Lernenden ein hilfreiches Feedback anbieten zu können, bedienen wir uns zweier Testtechniken aus der Softwaretechnik, dem statischen und dem dynamischen Testen von Programmen [KJH16]. Beim statischen Test (White-Box-Test) können Aussagen über die interne Struktur des Codes gemacht werden. Im dynamischen Test (Black-Box-Test) können im Gegensatz dazu Aussagen über die Funktionalität des Codes gemacht werden.

3.1 Statischer R-Checker

Zunächst wurde für die Entwicklung des statischen R-Checkers eine R-Grammatik entworfen, auf deren Basis R-Programme in eine Graphenstruktur geparst werden können. Auf den so generierten Graphen können nun mit GReQL [Un10] von Lehrenden beliebige Graphabfragen formuliert werden.

Auf diese Art kann vom Lehrenden beispielsweise überprüft werden, ob Studierende die in der Lösung gewünschten Funktionen mit korrekten Argumenten genutzt haben, ob Benamungskonventionen eingehalten wurden oder ob Schleifenkonstrukte zur Lösung genutzt wurden.

3.2 Dynamischer R-Checker

Für den dynamischen Checker kann der Lehrende einen oder mehrere Testfälle anlegen. Jeder Testfall vergleicht die Ausgabe der Studierendeneinreichung mit dem erwarteten Ergebnis. Zum Bestehen oder Fehlschlagen jedes Testfalls kann ein Feedback von Lehrenden angegeben werden.

R wird in der betrachteten Veranstaltung eher wie eine Skriptsprache benutzt, so dass das klassische Konzept des Unittestings auf von Usern geschriebenen Unterfunktionen nur bedingt greifen kann. Diese basieren darauf, dass Userfunktionen mit verschiedenen Argumenten aufgerufen werden und die Rückgabewerte auf vorher bestimmte Werte geprüft werden. Man kann sich so nach einer bestimmten Anzahl von Tests sicher sein, dass die Funktion richtig funktioniert. Deshalb generiert der R-Checker zunächst eine R-Unterfunktion aus der Einreichung. Diese kann dann wie im klassischen Unittesting mit den in den Testfällen definierten Übergabeparametern und den entsprechenden erwarteten Ergebnissen verglichen werden. Zum Vergleich der Userergebnisse mit den erwarteten Ergebnissen verwendet der dynamische Checker das Testthat-R-Modul [Wi16].

Ferner kann der Lehrende nach der generierten Testfunktion eigenen Code einfügen (postcode), so dass es möglich ist, dort noch vom Lehrenden benötigte Nachverarbeitungen durchzuführen. Dies ist hilfreich, wenn mehrere mögliche Antworten als richtig erkannt werden sollen oder die Antworten verschiedene Formate haben dürfen (z.B. eine Matrix ohne und eine mit Spaltenbezeichungen).

In Abb. 3.2 ist Feedback zu einer Einreichung eines Studierenden zu sehen, bei dem sowohl Checker-Regeln des statischen, als auch die des dynamischen Checkers zu Fehlern geführt haben.

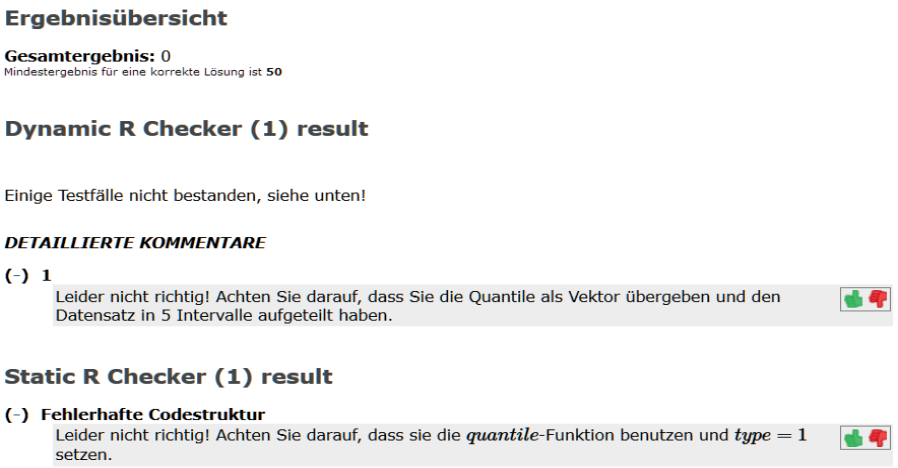

Abb. 3.2: Automatisch generiertes Feedback nach der Einreichung aus Abb. 3.1

4 Didaktisches Konzept

Im Wintersemester 2016/17 wurde JACK im Rahmen der Lehrveranstaltung „Deskriptive Statistik“ an der Universität Duisburg-Essen am Lehrstuhl Ökonometrie eingesetzt. Die Hörerschaft der Veranstaltung setzte sich zusammen aus zu Beginn insgesamt etwa 1000 Studierenden im 1. bis 3. Semester der Bachelorstudiengänge VWL, BWL, Wirtschaftsinformatik und Lehramt Wirtschaftswissenschaften. Von diesen 1000 Studierenden schrieben jedoch nur 457 eine der Klausuren. Lag der Fokus in früheren Semestern alleine auf der Vermittlung und der nicht-computerbasierten Anwendung von Methoden zur statistischen Datenanalyse, so hatte die Veranstaltung im Wintersemester 2016/2017 zusätzlich zum Ziel, die Anwendung der in Vorlesung und Übung erlernten Methoden auch computerbasiert, explizit mit der Statistiksoftware R, zu vermitteln. Zur Umsetzung dieses Ziels wurde das bestehende didaktische Konzept durch die Bereitstellung von statistischen Programmieraufgaben auf der Lernplattform JACK erweitert.

Konzeptionell setzte sich die Veranstaltung „Deskriptive Statistik“ zusammen aus einer wöchentlich stattfindenden Vorlesung, in der die statistischen Verfahren gelehrt werden, sowie einer ebenfalls wöchentlich stattfindenden Übung, in der die erlernten Methoden in praktischen Anwendungen vertieft werden. Beide Veranstaltungen fanden in Form von Frontalunterricht global für die gesamte Hörerschaft statt. Aufgrund der Masse an Studierenden konnte hier nicht, oder nur sehr schwer, auf individuelle Fragen und ein unterschiedliches Lerntempo eingegangen werden. Um dieses Problem abzumildern und das aktive, selbständige Lernen der Studierenden zu fördern, wurden, wie bereits in vergangenen Semestern, Aufgaben zum selbständigen Bearbeiten auf der Lernplattform JACK bereitgestellt. Dieses formative Assessment wurde im aktuellen Semester durch Programmieraufgaben in der Programmiersprache R erweitert. Insgesamt wurden 172 Übungsaufgaben freigeschaltet, davon 100 Übungsaufgaben zu klassischem Vorlesungsstoff ohne Programmierinhalte. Die restlichen 72 Aufgaben wurden als R-Aufgaben konzipiert. Durch die Bereitstellung von Feedback zu häufig begangenen Fehlern beim Lösen einer Aufgabe sowie optional wählbarer Hinweise wird der individuelle Lernerfolg der Studierenden unterstützt. Bei eventuell auftretenden Fragen und Problemen bei der Bearbeitung der JACK-Aufgaben stand ein vom Lehrstuhl betreutes Onlinetutorium zur Verfügung. Es konnte zudem ein wöchentlich stattfindendes freiwilliges Tutorium besucht werden.

Zur Unterstützung einer kontinuierlichen Arbeitsweise der Studierenden wurden verteilt über die gesamte Vorlesungszeit insgesamt 5 Testate abgehalten, für die ebenfalls JACK genutzt wurde. Diese Testate hatten eine Dauer von 40 Minuten und fanden zu vorgegeben Zeiten statt. Zur Teilnahme war lediglich ein internetfähiger PC erforderlich, es bestand aber keine Anwesenheitspflicht in Räumen der Universität. Dieses summative Assessment bot den Studierenden die Möglichkeit, ihren persönlichen Wissensstand kontinuierlich über die Vorlesungszeit hinweg zu überprüfen. Wie auch im Wintersemester 13/14 bildeten Bonuspunkte den Anreiz zur Ablegung dieser Testate, maximal 2,4 pro Testat, die im Falle einer am Semesterende bestandenen Klausur,

angerechnet wurden. Dies bot den Studierenden die Möglichkeit, maximal 12 Bonuspunkte für die Klausur zu sammeln, in der Klausur wurden jedoch maximal 10 Punkte angerechnet. Ein zusätzliches 6. Testat am Ende der Vorlesungszeit beinhaltete, im Gegensatz zu den vorherigen Testaten, R-Aufgaben, die ebenfalls zu einer festgelegten Zeit auf der Lernplattform JACK bereitgestellt wurden. Die Anrechnung von maximal 2 Bonuspunkten auf die am Semesterende abzulegende Klausur auch im Falle des Nichtbestehens bildete einen zusätzlichen Anreiz.

Die abschließende Klausur hatte eine Dauer von 70 Minuten und wurde im ersten Termin als PC-gestützte Klausur auf der Plattform JACK abgelegt. Im zweiten Termin wird eine gewöhnliche handschriftliche Klausur angeboten. In einem zusätzlichen dritten Termin wird den Studierenden die Möglichkeit geboten, erneut an einer PC-gestützten Klausur auf der Lernplattform JACK teilzunehmen. Die Klausuren im zweiten und dritten Termin wurden zum jetzigen Zeitpunkt noch nicht abgelegt. Im Gegensatz zu den abschließenden Klausuren in vorherigen Semestern bieten die Klausuren auf der Lernplattform JACK die Möglichkeit, zusätzlich R-Aufgaben abzufragen. So bestand die erste JACK-gestützte Klausur aus insgesamt 60 erreichbaren Punkten, davon konnten 52 Punkte durch klassische und 8 Punkte durch R-Aufgaben erzielt werden. Aufgrund der Unsicherheit bzgl. des Lernerfolgs der Studierenden bzgl. des in diesem Semester neu eingeführten Konzepts, R in das Lehrangebot aufzunehmen, wurde davon abgesehen, einen größeren Teil der Klausur durch R-Aufgaben abzudecken.

5 Evaluation

In der in Kapitel 4 beschrieben Lehrveranstaltung wurden die Nutzerdaten auf JACK protokolliert und ausgewertet. An der ersten Abschlussklausur nahmen 329 Studierende teil, deren Lernerfolg hier analysiert werden soll. Wie in Kapitel 4 erläutert, bestand die Abschlussklausur zum Vortermin aus insgesamt 60 Punkten, davon konnten 8 Punkte durch eine R-Aufgabe erreicht werden. Wir untersuchen hier die Fragestellung, ob Studierende mit hohem Arbeitseinsatz in R-Übungsaufgaben davon auch in den Standardklausuraufgaben ohne R-Inhalte profitieren. Zur Kontrolle, ob der Effekt alleine aus den R-Aufgaben resultiert, betrachten wir zusätzlich auch die Einreichungen in den klassischen JACK-Aufgaben.

Insgesamt haben 457 Studierende an mindestens einer Klausur teilgenommen. Wir konzentrieren uns in nachfolgender Auswertung auf den ersten Termin, um die unterschiedlichen Modalitäten (elektronisch o. schriftlich, 3 Termine in WS16/17 u. 2 schriftliche Termine in WS14/15) nicht zu stark zu vermischen. Die Ergebnisse sind ohnehin zahlenmäßig sehr ähnlich und sind auf Anfrage verfügbar.

Von den Klausurteilnehmern liegen u.a. die folgenden Daten vor: Die Klausurpunkte der klassischen Aufgaben, Klausurpunkte der R-Aufgabe, die Note (setzt sich aus der Summe der Klausurpunkte und der Bonuspunkte zusammen), die Anzahl der Einreichungen zu JACK Aufgaben ohne R und die Anzahl der Einreichungen zu JACK-

R-Aufgaben. Zusätzlich haben wir jeweils die Anzahl an korrekten Einreichungen auf JACK erhoben. Diese korrelieren mit der jeweiligen Anzahl an Einreichungen zu jeweils circa 97%. Wir betrachten im Folgenden die Variablen zu allen Einreichungen, da dies als ein aussagekräftigeres Maß für studentischen Lerneinsatz erscheint als lediglich korrekte Einreichungen. Dieselbe Analyse unter Verwendung der korrekten Einreichungen führt aufgrund der hohen Korrelation zu sehr ähnlichen Ergebnissen.

Tab. 5.1 zeigt den um Zwischennoten bereinigten Notenspiegel des Jahrgangs Wintersemester 2016/17 im Vergleich mit den Noten des Jahrgangs 2013/14, beide jeweils zum Vortermin.

Noten (bereinigt)	Wintersemester 13/14	Wintersemester 16/17
1	22 (5,54%)	39 (11,85%)
2	60 (15,11%)	46 (13,98%)
3	124 (31,23%)	80 (24,32%)
4	27 (6,8%)	15 (4,56%)
5	164 (41,3%)	149 (45,23%)
Summe	397	329
Schnitt (unbereinigt)	3,64	3,59

Tab. 5.1: Notenvergleich für zwei Jahrgänge. Die Noten wurden zur besseren Übersicht um Zwischennoten wie 1,3 bereinigt. Der Notenschnitt bezieht sich auf die unbereinigten Noten.

Im Schnitt haben sich die Noten im Jahrgang mit R-Nutzung leicht verbessert, insbesondere, da sich die Anzahl an sehr guten Studierenden fast verdoppelt hat. Dafür hat sich der Anteil an nicht bestandenen Klausuren von 41,3% auf 45,23% leicht erhöht. Anscheinend wurden durch die Nutzung von R im aktuellen Semester zwar die guten Studierenden gefördert, dafür konnten schwache Studierende eher nicht davon profitieren. Diese Aussagen sind insgesamt jedoch mit Vorsicht zu verstehen, da es sich nur um sehr geringfügige Änderungen handelt.

Die auf JACK bereitgestellten Aufgaben wurden von den Studierenden 97.244 mal bearbeitet. Dies teilt sich auf in 67.015 Einreichungen zu Aufgaben ohne R und 30.229 Einreichungen zu R-Aufgaben. Der Anteil an bearbeiteten R-Aufgaben bzgl. der Anzahl aller Einreichungen betrug also circa 31%.

Abb. 5.1 zeigt Boxplots zur Anzahl an Klausurpunkten ohne die R-Aufgabe, Anzahl an bearbeiteten klassischen Aufgaben auf JACK und der Anzahl an bearbeiteten R-Aufgaben auf JACK je Student. Vergleicht man den mittleren mit dem rechten Boxplot, erkennt man, dass ein größerer Anteil der Studierenden die R-Übungsaufgaben weitestgehend ignoriert hat. Die Ausreißer sind dadurch zu erklären, dass ein Studierender die gleichen Aufgaben sehr oft wiedereingereicht hat.

Wir untersuchen nun den Effekt der R-Aufgaben auf den Lernerfolg und insbesondere auf den Klausurerfolg. Interessant ist hier vor allem, welchen Einfluss die R-Übungsaufgaben auf Klausuraufgaben ohne R-Inhalte haben. Im Falle eines positiven

Effektes würden die R-Aufgaben also auch das allgemeine statistische Verständnis fördern. Hierzu regressieren wir die Variable „Klausurpunkte ohne R“ (minimal 0, maximal 52 Punkte) auf die Variable „Anzahl an Einreichungen in R-Aufgaben“ und kontrollieren für die Variable „Anzahl an Einreichungen in klassischen Aufgaben“ mittels der Standard Kleinste-Quadrate-Methode.

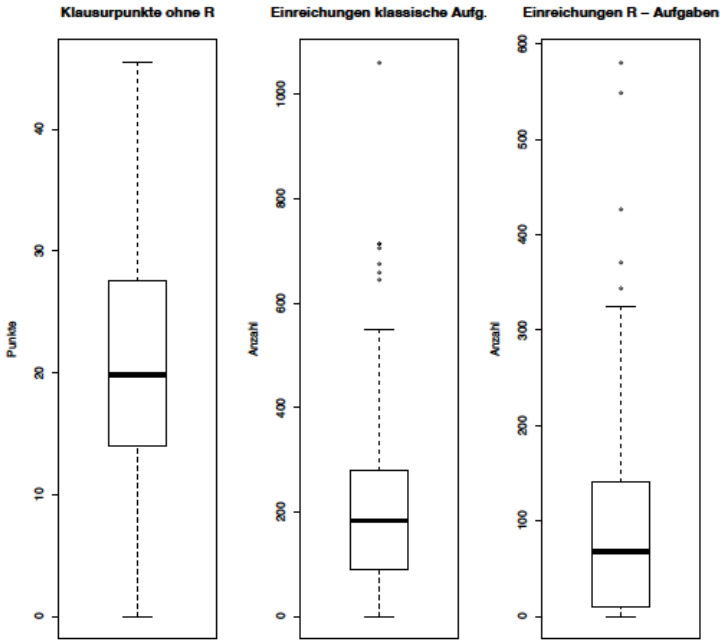

Abb. 5.1: Boxplots zur Veranschaulichung der Verteilung der für die Regression genutzten Variablen.

Abb. 5.2 zeigt ein Streudiagramm, welches die Klausurpunkte der Studierenden der Anzahl an Einreichungen in R-Übungsaufgaben gegenüberstellt. Eine beträchtliche Anzahl an Studierenden hat die R-Übungsaufgaben nicht oder nur sehr wenig bearbeitet. Diese schneiden zumeist auch in den klassischen Klausuraufgaben unterdurchschnittlich ab. Dafür geht eine gesteigerte Leistungsbereitschaft in R-Übungsaufgaben mit besseren Klausurergebnissen einher. Das Streudiagramm enthält zudem die Regressionsgerade der oben beschriebenen Regression. Dazu wurde in die Variable „Anzahl an Einreichungen in klassischen Aufgaben“ ihr Mittelwert eingesetzt.

Die geschätzten Regressionskoeffizienten sind Tab 5.2 zu entnehmen. Bemerkenswert ist der mehr als 3-mal größere Effekt der R-Übungsaufgaben im Vergleich mit den klassischen Übungsaufgaben auf die Klausuraufgaben. Möglicherweise stärken die Programmieraufgaben im Vorfeld das Verständnis der Materie deutlich stärker als die reproduktiven Rechenaufgaben. Für die Zukunft ist daher ein Ausbau des R-Bereiches

angedacht.

Wenn nun, siehe Tab. 5.3, die Klausurpunkte der R-Aufgabe (max. 8 Punkte) auf die gleichen Regressoren regressiert werden, wird auch hier ein positiver Effekt der R-Übungsaufgaben geschätzt. Der geschätzte Effekt der klassischen Übungsaufgaben ist hingegen nicht signifikant, da diese das Verständnis für Programmieraufgaben wohl nicht schärfen.

Koeffizient	Schätzung	Standardfehler	p-Wert
Konstante	15.53	0.79	~ 0
Anzahl R-Einreichungen	0.037	0.006	8.52*10^(-9)
Anzahl Klass. Einreichungen	0.01	0.004	0.0175

Tab. 5.2: Geschätzte Koeffizienten, Standardfehler und p-Werte der Regression „Klausurpunkte ohne R“ auf „Anzahl Einreichungen in R-Aufgaben“ und „Anzahl Einreichungen in klassischen Aufgaben“.

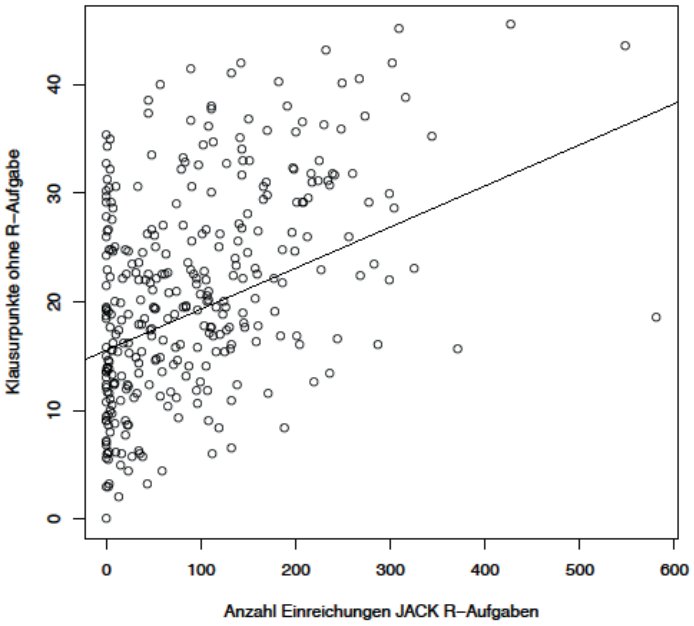

Abb. 5.2: Streudiagramm Klausurpunkte ohne R-Aufgabe gegen Einreichungen von R-Übungsaufgaben. Regressionsgerade der Regression Klausurpunkte ohne R gegen Einreichungen R-Aufgaben und Einreichungen klassische Aufgaben. In der Geraden wurde für die Kontrollvariable ihr Mittelwert eingesetzt.

Koeffizient	Schätzung	Standardfehler	p-Wert
Konstante	1.78	0.209	5.08*10^(-16)
Anzahl R-Einreichungen	0.014	0.002	1.35*10^(-15)
Anzahl Klass. Einreichungen	-0.001	0.001	0.346

Tab. 5.3: Geschätzte Koeffizienten, Standardfehler und p-Werte der Regression „Klausurpunkte der R-Aufgabe“ auf „Anzahl Einreichungen in R-Aufgaben“ und „Anzahl Einreichungen in klassischen Aufgaben“.

6 Fazit und Ausblick

Der eigens für die Statistikausbildung entwickelte JACK-R-Aufgabentyp wurde erstmals im WS 2016/17 genutzt, um den Studierenden der Vorlesung „Deskriptive Statistik“ die Anwendung der Statistiksoftware R zu vermitteln. Bei der Analyse der Nutzung dieses zusätzlichen Lehrangebots zeigte sich ein positiver Effekt von Programmier-Übungsaufgaben auf klassische Klausuraufgaben. Studierende, die das zusätzliche Lehrangebot nutzten, schnitten in der Klausur besser ab, als Studierende, die sich lediglich an den klassischen Aufgaben orientierten. Dies motiviert einen weiteren Ausbau dieses Konzepts auch in weiteren statistischen Veranstaltungen, um Studierende durchgehend besser zu fördern. Darüber hinaus soll die Förderung schwacher Studierender mit einem Ausbau des automatisierten individuellen Feedbacks verbessert werden. Zusätzlich arbeiten wir gerade an einer erweiterten Benutzeroberfläche in JACK, ähnlich dem Paket Rcmdr [FB17]. Damit sollen auch Studierende mit Nebenfach Statistik einen leichteren Zugang zu der Programmiersprache R finden.

Danksagung

Diese Arbeit wurde gefördert durch das Projekt „Professionalisierung für Vielfalt – Qualitätsoffensive Lehrerbildung“ des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 01 JA 1610.

Literaturverzeichnis

[Ba05] Basturk, R.: The Effectiveness of Computer-Assisted Instruction in Teaching Introductory Statistics. Educational Society & Technology 2/05, S. 170-178, 2005.

[BB14] Buttner, E. H.; Black, A. N.: Assessment of the Effectiveness of an Online Learning System in Improving Student Test Performance. Journal of Education for Business 5/14, S. 248-256, 2014.

[Ch07] Chance, B. et al.: The Role of Technology in Improving Learning of Statistics. Technology Innovations in Statistics Education 1/07, S. 1-26, 2007.

[CM02] Chalmers, D.; McAusland, W. D. M.: Computer-assisted Assessment. Technial report, Glasgow Caledonian University, 2002.

[Da17] DataCamp, datacamp.com, Stand: 21.03.2017.

[FB17] Fox, J.; Bouchet-Valat, M.: Rcmdr: R Commander. R package version 2.3-2. 2017.

[KJH16] Keuning, H.; Jeuring, J.; Heeren, B.: Towards a Systematic Review of Automated Feedback Generation for Programming Exercises. In: Proceedings of the 2016 ACM Conference on Innovation and Technology in Computer Science Education, S. 41-46, 2016.

[Ma17] Massing, T. et al.: E-Assessment Using Variable-Content Exercises in Mathematical Statistics. Wiedereingereicht beim Journal of Statistics Education, 2017.

[Mc15] McNamara, A. A.: Bridging the Gap Between Tools for Learning and for Doing Statistics. Dissertation, 2015.

[Me13] Means, B. et al.: The Effectiveness of Online and Blended Learning: A Meta-Analysis of the Empirical Literature. Teachers College Record 3/13, S. 1-47, 2013.

[Mu15] Muenchen, R. A., The Popularity of Data Science Software, http://r4stats.com/articles/popularity/, Stand: 21.03.2017.

[RC17] R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, 2017.

[Sc17] Schwinning, N. et al.: Towards digitalization of summative and formative assessment in academic teaching. Proceedings of LaTiCE 2017. Im Erscheinen, 2017.

[So11] Sosa, G. W. et al.: Effectiveness of Computer-Assisted Instruction in Statistics: A Meta-Analysis. Review of Educational Research 1/11, S. 97-128, 2011.

[St16] Striewe, M.: An architecture for modular grading and feedback generation for complex exercises. Science of Computer Programming 129/16, S. 35-47, 2016.

[TL12] Tishkovskaya, S.; Lancaster, G. A.: Statistical education in the 21st Century: a review of challenges, teaching innovations and strategies for reform. Journal of Statistics Education 2/12, S. 195-272, 2012.

[Un10] Universität Koblenz Landau, AG Ebert, https://www.uni-koblenz-landau.de/de/koblenz/fb4/ist/rgebert/research/graph-technology/GReQL, Stand: 17.03.2017.

[Wi16] Wickham, H.: testthat: Unit Testing for R, https://cran.r-project.org/web/packages/testthat/index.html, Stand 20.03.2017.