Turnitin

turnitin.com/de/blog/was-ist-ein-code-plagiat-warum-kommt-es-haeufiger-vor

July 26, 2021

Was ist ein Code Plagiat? Einfach ausgedrückt kommt es zu einem Code Plagiat, wenn der Quellcode einer anderen Person als der Eigene ausgegeben wird. Code Plagiate gibt es seit mindestens den 1990ern, als MOSS (Measure of Software Similarity) entwickelt wurde, um Programmieraufgaben auf Plagiate zu prüfen. In diesem Beitrag beschäftigen wir uns nicht nur mit der Zunahme von Plagiaten bei akademischen Programmierarbeiten, sondern auch mit den Gründen dahinter und zeigen auf, was Pädagogen dagegen tun können.

1998 war im Wired-Artikel „Catching Computer Science Cheaters“ zu lesen: „Dozenten von Computer Science schätzen, dass bei jeder Aufgabe zwischen 5 und 20 Prozent der Studierenden über das akzeptable Maß hinaus zusammengearbeitet haben“.

Ein New York Times Artikel von 2017 mit dem Titel „As Computer Coding Classes Swell, So Does Cheating“ bestätigt, dass Code Plagiate auf dem Campus weiterhin ein Problem darstellen. Bidgood und Merill erklären: „An der Brown University war fast die Hälfte der 49 mutmaßlichen Verstöße gegen den akademischen Verhaltenskodex auf Plagiate in Computer Science zurückzuführen. In Stanford, der Alma Mater der Gründer von Google, Snapchat und zahllosen anderen Internet-Wundern, wurden bei einem Computer-Science-Studiengang 2015 bis zu 20 Prozent der Studierenden wegen möglichem Fehlverhalten auffällig.“

Die Doppeldeutigkeit der Zusammenarbeit

Jede Software, die gekauft oder heruntergeladen werden kann, wurde wahrscheinlich von mehr als einer Person entwickelt. Außerdem kann der Code sogenannte „Open-Source“-Elemente enthalten, also kostenlosen Quellcode, der je nach Lizenz mit oder ohne Quellenangabe von jedem benutzt werden kann. In der Welt der Software-Entwicklung ist die Zusammenarbeit auf eine Art und Weise geläufig und erwünscht, die bei einer typischen akademischen Aufgabe inakzeptabel wäre.

Hier müssen Programmierprojekte oft von einer einzelnen Person ausgeführt werden, um diese beurteilen zu können. Dem Code muss eine korrekte Quelle zugewiesen oder sogar voll und ganz die eigene Arbeit des Studierenden sein. Die Gründe für diesen Unterschied liegen auf der Hand: Um die Studierenden bewerten zu können, müssen sie ihr Verständnis der Konzepte und eine originelle Denkweise unter Beweis stellen, und nicht etwas bauen, was verkauft werden soll.

Lösungsverfügbarkeit

In der Welt von Open-Source-Software kann der Code unter unterschiedlichen Lizenzbedingungen weiterverwendet werden. Eventuell tun sich Studierende mit der Hausarbeit schwer und/oder geraten zu sehr in Versuchung, zur Open-Source-Variante

zu greifen. Open-Source-Software ist in Online-Code-Repositorys wie Github verfügbar, wo es eventuell auch Antworten zu Konzepten und Aufgabenstellungen gibt, die von ehemaligen Studierenden desselben Studienganges gepostet wurden, so die New York Times. Eine weitere Ressource, die manchmal von Studierenden zweckentfremdet wird, ist Stack Overflow, eine Frage-/-Antwortseite, die von sowohl neuen als auch erfahrenen Programmierern genutzt wird.

Aus unter anderem den oben genannten Gründen trifft der kollaborative Ansatz, den Programmierer in der Praxis verfolgen, nicht auf Programmieraufgaben in einer Bildungsumgebung zu. Die Grenze bei Absprachen unter Studierenden und Code Plagiaten wirkt daher oft schwammig, nicht zuletzt, weil die Studierenden Zugang zu einer großen Anzahl an Ressourcen haben, die für die Entwicklung von Software in der Praxis gedacht ist. Laut des Forschungsartikels „Eliminating Plagiarism in Programming Courses through Assessment Design“ „neigen Studierende zu Plagiaten, wenn Lösungen zur Aufgabenstellung [sic] leicht im Internet oder anderen Quellen gefunden werden kann“ (Ngo, 2016, p. 873).

Was können Pädagogen tun, um zu verdeutlichen, was ein Plagiat beim Programmieren darstellt und wie sich das verhindern lässt?

- **Geben Sie klare Regeln zur akademischen Integrität vor, zu der auch eine Definition von Zusammenarbeit im Vergleich zu Absprache gehören sollte**. Legen Sie von vornherein fest, welche Grenzen die Studierenden nicht überschreiten dürfen. In „Collaboration Versus Cheating“ veröffentlichte Forschungsergebnisse haben gezeigt, dass „Erklärungen und Betonung der Wichtigkeit von akademischer Ehrlichkeit zu einer statistisch signifikanten ($p < 0,05$) Abnahme von Plagiaten bei allen Programmierarbeiten eines großen Online-Studiengangs für Computer Science führte“ (Mason, Gavrilovska, & Joyner, 2019).
- **Betonen Sie die Richtlinien zur Nutzung von externem Code.** Wenn Sie die Verwendung von externem Code erlauben und/oder eine eingeschränkte Quellenliste führen, sollten Sie die Studierenden darum bitten, ihren Code zu kommentieren und zu verdeutlichen, wann der Code aus externen Quellen entlehnt wurde – das heißt, sie sollen ihre Quellen nennen und ordnungsgemäß zitieren.
- **Denken Sie sich beim Brainstorming neuen und einzigartigen Code aus.** Seien Sie ein Vorbild eigenständiger Denkweisen und erstellen Sie Ihre Aufgaben so, dass sie die Entwicklung von eigenem Code unterstützen.
- Bauen Sie ein Gerüst für originellen Code, **indem Sie Zwischentermine festlegen, an denen es Feedback gibt, um den Lernfortschritt der Studierenden zu fördern und denjenigen zu helfen, die Schwierigkeiten haben**. Manche Pädagogen verwenden Git bzw. ein anderes Tool zur Versionskontrolle, das Studierenden erlaubt, ihre Arbeit in Segmenten über einen längeren Zeitraum hinweg zu speichern.
- **Nutzen Sie Software, die die Arbeit auf Ähnlichkeit prüft**, um Studierende von akademischem Fehlverhalten abzuschrecken. Vor allem in Verbindung mit expliziten Anweisungen und Feedback kann dies die akademische Integrität bei der Programmierung stärken.

- Investieren Sie in die Gestaltung der Aufgabenstellung mit **Hilfe von Item-Analyse und Benotung mit Integrität, damit die Aufgaben zum Lernerlebnis der Studierenden** beitragen und die Studierende den Wert der Benotung und des formativen Feedbacks verstehen. „Eliminating Plagiarism in Programming Courses through Assessment Design“ zufolge „ist im Programmierbereich das Lernen von Beispielen aus der Praxis besonders nützlich, indem Studierende lernen, bestehenden Quellcode zu interpretieren und auf ihre Bedürfnisse auszulegen. Beim Lernen an Beispielen ist es ausschlaggebend, dass die Studierenden die Beispiele verstehen und den Quellcode nicht nur kopieren, um die Aufgaben zu erledigen. Bei unserer Strategie zur Aufgabenstellung werden daher für jede Aufgabe Codeskelette entwickelt, die verhindern, dass die Studierenden Code blind kopieren. Die Präsenzprüfungen werden dann auf der Grundlage des Aufgabeninhalts entworfen, um das Verständnis und die Fähigkeit der Studierenden zu testen, ihren Quellcode an neue Programmanforderungen auszulegen. Erste Ergebnisse und das Feedback der Studierenden zeigen den potenziellen Nutzen unserer Methode zur Verbesserung des Verständnisses und der Leistung der Studierenden. Darüber hinaus verhindert sie, dass Dozenten Arbeit und Zeit in das Aufdecken von Plagiaten stecken müssen“ (Ngo 2016, p. 878).
- **Pflegen und respektieren Sie Feedback-Schleifen zwischen Studierenden und Dozenten**. Die Forschung hat gezeigt, dass starke Beziehungen zwischen Pädagogen und Studierenden Fälle von Plagiaten reduzieren. Computer Science ist da keine Ausnahme. In „Plagiarism and Programming: A Survey of Student Attitudes“ wird die Forschung zur Wahrnehmung von Code Plagiaten durch Studierende sowie deren Prävention zusammengefasst. Die Studie kommt zu dem Schluss: „Diese Studie hat gezeigt, dass die Aufklärung darüber, was akademisches Fehlverhalten bei benoteten Programmieraufgaben darstellt, einen Unterschied in der Wahrnehmung der Studierenden ausmacht. Die Pädagogen müssen deutlich machen, was akzeptabel ist und was nicht, und die Studierenden ständig an die Kursrichtlinien in Bezug auf akademische Unehrlichkeit erinnern.“ Dies deckt sich mit den Ergebnissen von Simkin und McLeod (2010), die das Vorhandensein eines ethischen Fakultätsmitglieds mit Meinungen, die von den Studierenden respektiert wurden, als Grund dafür sehen, dass die Studierenden sich ordnungsgemäß verhalten haben“ (Aashiem, Rutner, Li, & Williams 2012, S. 307). Explizite Anweisungen zur akademischen Integrität in der Programmierung sind ein notwendiger Schritt.

Der ungehinderte digitale Zugang zu Online-Quellcode steigert das Risiko von Code Plagiaten enorm. Quellen sind reichlich vorhanden und für überforderte und gestresste Studierende zugänglich. Und die kollaborative Natur in der Praxis – ideal für die Erstellung von Software – kann für Studierende, die für ihre individuelle Leistung bewertet werden, irreführend sein.

Wir als Pädagogen können den Studierenden helfen, den Zweck des Lernens und die Leitprinzipien der akademischen Integrität zu verstehen, damit sie lernen, originelle Ideen und originellen Code zu erstellen und verstehen, was Zusammenarbeit wirklich bedeutet – auch als Berufsvorbereitung.