Actas de las Jenui, vol. 6. 2021.
Páginas: 27-34

Aprendizaje basado en proyectos de analítica de software en estudios de ciencia e ingeniería de datos

Silverio Martínez-Fernández, Cristina Gómez, Xavier Franch
Departamento de Ingeniería de Servicios y Sistemas de Información (ESSI)
Universitat Politècnica de Catalunya (UPC) – BarcelonaTech
Barcelona, España
{silverio.martinez, cristina.gomez, xavier.franch}@upc.edu

Resumen

El Grado en Ciencia e Ingeniería de Datos (GCED) de la Universidad Politécnica de Cataluña (UPC) forma a sus estudiantes en análisis e ingeniería de datos con una base matemática y habilidades propias de la ingeniería que les permiten modelizar y resolver problemas complejos. En el último curso, los estudiantes aún no han analizado conjuntos de datos que permitan ayudar a mejorar la calidad del software, ni explotado buenas prácticas de ingeniería del software en proyectos de ciencias de datos. En el curso 2020/2021 se cursa por primera vez la asignatura Temas Avanzados en Ingeniería de Datos 2 (TAED2) en el GCED. Esta experiencia docente relata la introducción del aprendizaje basado en proyectos en el laboratorio de la asignatura, donde los estudiantes aplicaron la ciencia de datos y buenas prácticas de ingeniería del software para analizar y detectar mejoras en la calidad del software. Realizamos un caso de estudio en siete sesiones de laboratorio con 33 estudiantes del GCED. Los estudiantes encontraron la realización del proyecto positiva para su aprendizaje. En el proyecto, destacaron la utilidad de usar un proceso de minería de datos CRISP-DM y una buena práctica de ingeniería del software: una convención de estructura de un proyecto software aplicada a un proyecto de ciencia de datos.

Abstract

The Bachelor's Degree in Data Science and Engineering (GCED) at UPC-BarcelonaTech trains its students in data analysis and engineering with a mathematical basis and engineering skills that allow them to model and solve complex problems. In the last year, students have not yet analyzed data sets that help improving software quality, nor have they exploited good software engineering practices in data science projects. In the 2020/2021 academic year, the subject Advanced Topics in Data Engineering 2 (TAED2) is taken at the GCED for the first time. This teaching experience shows the introduction of project-based learning in the subject laboratory, where students applied data science and good software engineering practices to analyze and detect improvements in software quality. We carried out a case study in seven laboratory sessions with 33 GCED students. The students found the execution of the project positive for their learning. In the project, they highlighted the utility of using a CRISP-DM data mining process and good software engineering practice: a software project structure convention applied to a data science project.

Palabras clave

Ingeniería del software, ciencia de datos, procesos de minería de datos, analítica de software, buenas prácticas de software, docencia.

1. Introducción

Debido a la creciente adopción de las tecnologías Big Data, la ciencia de datos ha ganado atención en múltiples áreas de conocimiento, incluyendo la ingeniería del software. Las empresas de software tienen la necesidad de comprender las habilidades de aprendizaje necesarias para organizar equipos multidisciplinarios con profesionales expertos en software y en datos. Por ejemplo, Microsoft e IBM han informado de la necesidad de organizar equipos multidisciplinarios compuestos por científicos de datos e ingenieros del software para llevar a cabo el desarrollo y mejora de calidad de software basados en datos [12, 15].

Consecuentemente, se están realizado acciones de innovación docente para enseñar ingeniería del software para la creación de sistemas basados en inteligencia artificial [10, 11] (v. Sección 5).

Esta ponencia muestra la experiencia docente de diseñar, ejecutar y evaluar una nueva actividad usando la metodología de *aprendizaje basado en proyectos (ABP)* [6, 19], en el laboratorio de la asignatura TAED2 del GCED en la UPC. El objetivo de dicha actividad es llevar a cabo un proyecto (en adelante, el proyecto) de analítica de software para detectar mejoras en la calidad de software de código abierto. El proyecto tiene dos aspectos de ingeniería del software innovadores para los alumnos: *(i)* la aplica-

ción de ciencia de datos en un nuevo dominio: detección de problemas en la calidad de software; y *(ii)* la generación de una estructura de proyecto de datos basada en una buena práctica de ingeniería del software.

El diseño, ejecución, y evaluación de este estudio tiene tres contribuciones principales:

- Estudiar la viabilidad de aplicar ABP para detectar problemas en la calidad de software por alumnos de último curso del GCED, así como identificar sugerencias de mejora para tratar los problemas encontrados.
- Analizar la utilidad de un proceso de minería de datos (CRISP-DM, v. Sección 2.1) y una convención de estructura de proyecto de ciencia de datos (Cookiecutter Data Science, v. Sección 2.2) por parte de los estudiantes del GCED en el contexto de su proyecto.
- Reportar los objetivos de analítica de software (v. Sección 2.3) escogidos por los estudiantes a partir de un conjunto de datos relativos a proyectos de software de código abierto (TechDebt, v. Sección 2.4).

El resto de este documento se estructura de la siguiente manera. La Sección 2 introduce el contexto y artefactos usados en la asignatura. La Sección 3 resume la guía docente, junto a objetivos de aprendizaje y las actividades del proyecto de laboratorio. La Sección 4 muestra los resultados de la actividad en la primera edición de la asignatura, y reflexiona sobre aspectos fuertes y a considerar en próximas ediciones. La Sección 5 muestra el trabajo relacionado. Finalmente, la Sección 6 concluye este documento.

2. Contexto y artefactos de la asignatura

Esta asignatura es obligatoria y del séptimo cuatrimestre del GCED en la UPC. El punto de partida de esta experiencia docente fueron las habilidades aprendidas por los estudiantes durante los tres primeros años del GCED. A continuación, mostramos los artefactos más importantes utilizados en el laboratorio (CRISP-DM y estructura de proyecto), así como la problemática de analítica de software y el conjunto de datos usado.

2.1. CRISP-DM

CRISP-DM es el acrónimo de "*Cross-industry standard process for data mining*". Según dos encuestas realizadas en 2014 y 2007, es el proceso de minería de datos más usado en la industria[1]. De hecho, en la actualidad, adaptaciones suyas se aplican no sólo en minería de datos, sino en el contexto de aprendizaje automático [18]. Otros procesos de minería de datos son KDD [5] y SEMMA[2] (v. Kurgan et al. para comparativa [13]).

CRISP-DM fue el resultado de un proyecto europeo de investigación. El objetivo era crear un proceso estándar para la comunidad de minería de datos. Está basado en la experiencia práctica de cómo profesionales conducen proyectos de minería de datos. Entre las 300 empresas que colaboraron en una serie de talleres, cabe destacar tres principales del consorcio europeo: Daimler Chrysler (después Daimler-Benz), SPSS (después ISL), y NCR. Destacamos que el estándar no es propietario, está disponible gratuitamente, y es agnóstico respecto a la industria, las herramientas, o los dominios de aplicación.

Una de las contribuciones principales de CRISP-DM es su modelo de referencia. Este modelo de referencia contiene seis fases (v. Figura 1): comprensión del negocio, comprensión de los datos, preparación de los datos, modelado, evaluación, y despliegue. Asimismo, el estándar incluye la metodología, guía de usuario, y plantilla para informes de cada una de las fases del proyecto. Toda la información se puede consultar en la guía de usuario[3].

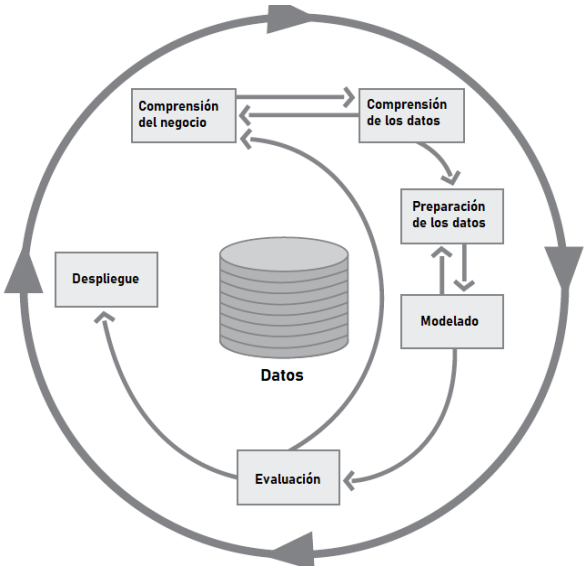

Figura 1: Fases de CRISP-DM.

Recientemente, el uso de este proceso en equipos de desarrollo software ha mostrado ser efectivo para la mejora de calidad de software dirigida por datos en empresas [7]. Por este motivo, estudiamos su viabilidad y utilidad en un contexto académico en esta asignatura y experiencia docente.

[1] https://www.kdnuggets.com/2014/10/crisp-dm-top-methodology-analytics-data-mining-data-science-projects.html

[2] https://documentation.sas.com/?docsetId=emref&docsetTarget=n061bzurmej4j3n1jnj8bbjjm1a2.htm&docsetVersion=15.1

[3] https://spss.ch/upload/1107356429_CrispDM1.0.pdf

2.2. Estructura de proyecto de ciencia de datos

Los proyectos de ciencia de datos contienen scripts, cuadernos interactivos, archivos de configuración, archivos de datos, informes, documentación, pruebas y figuras. Un problema común de estos proyectos de ciencia de datos es su reproducibilidad. Si algún científico de datos o desarrollador necesita ejecutar un script, cuaderno interactivo, o modelo de inteligencia artificial, necesita saber dónde se encuentra la versión correcta de los modelos, los datos de entrenamiento, etc. Este problema es análogo al que se encuentran los equipos de desarrollo de software cuando necesitan encontrar algún artefacto dentro de la estructura de su proyecto software.

Por este motivo, recientemente se ha impulsado el uso de múltiples convenciones de estructuras de proyecto de ciencia de datos, como la ofrecida para Tensorflow[4] y Azure[5]. En el momento del inicio de la asignatura, la estructura de proyecto de ciencia de datos con mejor valoración en GitHub era Cookiecutter Data Science[6], lo que motivó su elección.

Cookiecutter Data Science: *(i)* proporciona una estructura de repositorio genérica para modelos, datos, configuración, etc.; *(ii)* permite iniciar un proyecto personalizado por el autor, el nombre del proyecto, la licencia, la versión de Python, etc.; *(iii)* especifica una convención de nombres para los cuadernos, proporciona un directorio para los modelos entrenados y una estructura de carpetas para separar los datos en bruto de los datos externos/procesados.

2.3. Analítica de software

La analítica de software consiste en utilizar enfoques basados en los datos para obtener información reveladora y procesable que ayude a los profesionales del software en sus tareas relacionadas con los datos [8]. Tiene como objetivos mejorar la productividad del proceso de desarrollo, la calidad de los sistemas de software, y/o la experiencia de los usuarios finales [19]. En este laboratorio, el proyecto se centra en la detección de mejoras de la calidad de los sistemas software a partir del análisis de sus datos. Para ello, se comienza ejecutando tareas de minería en datos brutos (p.ej., código fuente, repositorios de fallos) para descubrir patrones (p.ej., construir un modelo de predicción de fallos), y finalmente producir información procesable (p.ej., programar tareas específicas de garantía de calidad, asignar recursos).

2.4. El conjunto de datos de deuda técnica

Existen múltiples conjuntos de datos para realizar analítica de software para diversas problemáticas de ingeniería del software[7] (p.ej., estimación de costes, elicitación de requisitos, etc.). Se puede encontrar en repositorios propios como PROMISE[8] o incluso en Kaggle[9].

En esta experiencia docente nos centramos en un problema específico de calidad de software: la deuda técnica. La deuda técnica refleja el costo implícito del trabajo adicional causado por elegir una solución fácil en un punto concreto de desarrollo en lugar de utilizar una alternativa de más calidad, pero más costosa en su implementación. Para estudiar este problema, usamos el conjunto de datos "TechDebt" [14].

En el momento del inicio de la asignatura, el conjunto de datos TechDebt consistía en 33 proyectos Java de código abierto con diferentes datos calculados y/o extraídos por diferentes herramientas. Para cada proyecto podemos encontrar: métricas de análisis estático de código realizado por la herramienta de SonarQube, las tareas de los backlogs de Jira, commits, datos sobre refactorización extraídos por la herramienta RefactorMiner, y fallos identificados por el algoritmo SZZ (para más información, v. [14]).

El conjunto de datos TechDebt, con su documentación y trabajo de limpieza de los datos previo, facilita que realizar el proyecto en siete semanas sea factible.

3. Guía docente de la asignatura

En esta sección explicamos los objetivos de aprendizaje específicos de la asignatura, y nos centramos en el objeto de estudio de esta experiencia docente: el proyecto que realizarán los estudiantes.

3.1. Objetivos de aprendizaje

La asignatura TAED2 tiene cuatro objetivos. En los cuatro objetivos el sujeto es el estudiante, se utiliza un verbo relacionado con un nivel de la taxonomía de Bloom [2, 4], y se especifican el contenido y la circunstancia. Los objetivos son:

- Interpretar los conceptos básicos de la Ingeniería del Software, especialmente en relación al uso y explotación de datos.
- Aplicar y analizar conceptos y métodos referentes al uso de datos provenientes del proceso de desarrollo en la gestión de la calidad del sistema software.

[4] https://github.com/Mrgemy95/Tensorflow-Project-Template
[5] https://github.com/Azure/Azure-TDSP-ProjectTemplate
[6] https://github.com/drivendata/cookiecutter-data-science
[7] https://danrodgar.github.io/DASE/repositories.html
[8] http://promise.site.uottawa.ca/SERepository/
[9] https://www.kaggle.com/

- Aplicar conceptos y métodos referentes al uso de datos obtenidos durante la utilización del sistema por parte de sus usuarios en la planificación de nuevas versiones evolutivas.
- Describir conceptos y métodos referentes al uso de datos de contexto obtenidos durante la utilización del sistema, para la adaptación autónoma de los sistemas en tiempo de ejecución en respuesta a cambios.

El objetivo número dos se trata en profundidad en el proyecto de laboratorio, alcanzando los niveles 3 (aplicación) y 4 (análisis) de la taxonomía de Bloom.

3.2. El proyecto

El proyecto es la actividad principal del laboratorio de TAED2. Se ha escogido la metodología de *aprendizaje basado en proyectos (ABP)* debido a la necesidad de equipos multidisciplinarios en analítica de software, la complejidad del objetivo a alcanzar, y para favorecer la participación activa y crítica de los estudiantes identificando oportunidades de mejora de calidad de software. El ABP es una metodología de aprendizaje activo en la que los alumnos trabajan sobre un problema del mundo real durante un periodo de tiempo.

Equipos. Los equipos se componen de 3-4 estudiantes. Además, el profesor semanalmente da retroalimentación sobre el proyecto y resuelve dudas.

Plan de trabajo. El proyecto se organiza de forma iterativa e incremental, siguiendo las fases de CRISP-DM, con flexibilidad para saltar de manera ágil a fases anteriores si es necesario, tal y como recomienda el propio proceso. El proyecto se desarrolla en siete semanas, como muestra el Cuadro 1.

Sem.	Tareas de cada sesión de laboratorio
1	Inicio del proyecto: Equipos creados y carpetas de Google drive configuradas. Introducción al conjunto de datos. Los equipos comienzan a definir el objetivo del proyecto.
2	Discusiones a partir de las lecturas asignadas. Presentación de la selección inicial del objetivo del proyecto.
3	Presentación de la comprensión del negocio y los pasos iniciales de la comprensión de los datos.
4	Presentación de la comprensión de los datos (y plazo final para refinar el objetivo si es necesario). Inicio de la preparación de los datos. Sesión “pregúntame lo que quieras” con expertos.
5	Presentación de pasos de preparación de datos y modelado.
6	Presentación de pasos de modelado (si hay alguna actualización) y evaluación
7	Presentación final, incluyendo reflexión del paso de despliegue. Entrega de: (a) informe final (siguiendo CRISP-DM) y (b) software (siguiendo la plantilla del proyecto). Cierre de proyecto.

Cuadro 1: Calendario del proyecto.

En la primera sesión se forman los equipos de trabajo. También se introduce el conjunto de datos y se explican sus características.

A partir de la segunda semana, los equipos presentan sus progresos en el proyecto y reciben retroalimentación semanalmente. Comienzan desde la fase 1 de CRISP-DM, especificando un objetivo de negocio como parte de la comprensión del negocio, hasta el despliegue de la presentación final.

Además del ABP, cabe destacar la integración de otras metodologías. Un ejemplo es la *clase invertida* [3]. En la primera semana, se asigna a cada estudiante de un equipo la lectura de un artículo entre los siguientes [1, 12, 14, 17]. Estos artículos ayudan a escoger un objetivo de analítica de software y a comprender el conjunto de datos. Entre las dos primeras sesiones, los alumnos individualmente leen el artículo asignado. En la segunda semana, deben primero presentar y consensuar un resumen con componentes de otros equipos que tienen el mismo artículo asignado. Posteriormente, presentan el resumen consensuado de cada artículo a los miembros de su equipo. Por otra parte, aunque se presentan las fases de CRISP-DM en clase, cada semana los alumnos tienen que leer detalladamente las actividades a realizar de una fase de CRISP-DM en la guía de uso fuera del aula (presentado en la Sección 2.1). Esto es importante de cara a la presentación sobre los progresos del proyecto en la siguiente semana, pues como muestra el Cuadro 1 y hemos indicado anteriormente, cada semana los equipos van progresando en las fases de CRISP-DM en el proyecto (ver Figura 1).

En el ecuador del laboratorio, se realiza una sesión “*pregúntame lo que quieras*” por videoconferencia. En dicha sesión, en el curso 2020/2021, participaron tres panelistas de ciencia de datos que forman parte de equipos de desarrollo de software. Los panelistas eran de diferente sexo, lugar de trabajo (empresa, instituto de investigación, y universidad), y país. Después de una breve introducción, durante una hora los alumnos podían realizar cualquier pregunta, como por ejemplo consejos sobre el proyecto o sobre su futura carrera laboral.

Hacia el final del proyecto, en la semana 6 se crea un *libro digital* con los informes de todos los equipos al que todos los estudiantes pueden acceder. Este libro está inspirado en otras asignaturas (p.ej., de arquitectura de software) que fomentan el *aprendizaje colaborativo* [21]. El libro, que es un documento en Google Docs, permite que cada equipo pueda ver el resumen de los proyectos de otros equipos en la última semana. Así, cada equipo puede ver qué hace el resto de equipos antes de la presentación final.

Evaluación. La evaluación del laboratorio consistió en un 70% del informe final entregado y un 30% del software generado. El informe debe seguir las guías de CRISP-DM, que también define cómo

realizar un informe. Desde la semana 1, los alumnos van documentando el proyecto con la ayuda de CRISP-DM. En el software se evalúa el código y su adherencia a la convención de estructura de proyecto de ciencia de datos. Los equipos disponen de una *rúbrica* [9] para conocer cómo se evalúan tanto el informe final como el software.

4. Evaluación del curso piloto 2020/2021

La primera edición de esta asignatura ha constado de 33 estudiantes divididos en dos grupos. Entre los dos grupos, se formaron nueve equipos de 3-4 estudiantes. Esta sección muestra respectivamente los datos coleccionados antes de comenzar el proyecto, los objetivos escogidos por los estudiantes para el proyecto, y la retroalimentación y opinión del alumnado y profesorado tras realizar el proyecto.

4.1. Cuestionario inicial

Para preparar las sesiones de laboratorio, se realizó un cuestionario inicial dos semanas antes de su inicio. Se preguntó sobre experiencia en lenguajes de programación y entornos de desarrollo, experiencia en procesos de minería de datos, proyectos de ciencia de datos, y conocimientos en ingeniería del software.

La participación en el cuestionario, voluntario, fue del 90% (30 alumnos sobre 33). Para medir la experiencia, usamos una escala ordinal con los siguientes valores sobre experiencia: ninguna (nunca usado), básica (usado en un proyecto o asignatura), intermedio (usado de 2 a 5 asignaturas o proyectos), avanzado (usado en más de 5 asignaturas o proyectos), y n/a (no aplica). Los datos se muestran en la Figura 2.

Respecto a lenguajes de programación, 28 alumnos (93%) tenían un conocimiento avanzado de Python. C++ y R también eran conocidos, con 28 alumnos con un conocimiento entre intermedio y avanzado. Respecto a Java, sólo 8 alumnos tenían algún conocimiento. Aunque no se preguntó explícitamente por ellos, en la sección abierta al menos 5 alumnos mencionaron conocimientos en Matlab y SQL.

El entorno de desarrollo más usado por los estudiantes es Jupyter Notebook, con conocimiento avanzado por 27 de los alumnos, seguido de R Studio, con 13 alumnos con conocimiento avanzado.

Puesto que el objetivo del laboratorio no es la enseñanza de nuevos lenguajes o entornos de programación sino la aplicación de los aprendidos en los tres primeros años del GCED, el laboratorio se diseñó para usar Python y Jupyter Notebook.

A su vez, los equipos tenían libertad para elegir otros lenguajes y herramientas si estaba justificado.

Antes del séptimo cuatrimestre, un 94% de los alumnos no tenían conocimientos de procesos de minería de datos, como CRISP-DM, KDD, o SEMMA. Igualmente, el 83% de los alumnos no conocía qué era la Ingeniería del Software. Esto se alinea con los objetivos de aprendizaje 1 y 2 de la asignatura.

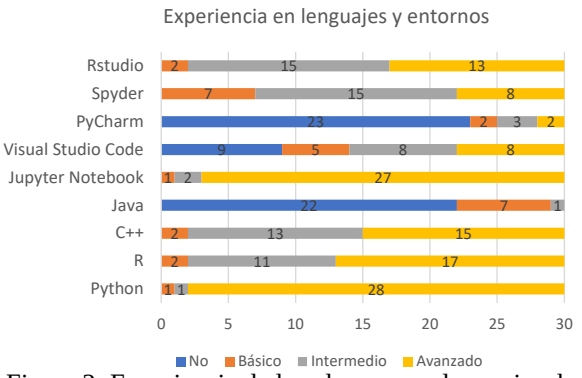

Figura 2: Experiencia de los alumnos en lenguajes de programación y entornos de desarrollo.

Finalmente, destacar que 22 alumnos tenían experiencia previa en conducir proyectos de ciencia de datos (73%), aunque ninguno en el contexto de datos relacionados con el software. Ejemplos de proyectos de ciencia de datos realizados anteriormente a TAED2 por los alumnos son: predecir si un usuario clicará en publicidad web, crear subtítulos de películas, desarrollar algoritmos de reconocimiento de imágenes y de clasificación de textos.

4.2. Objetivos resueltos en los proyectos

Cada equipo tenía que definir un objetivo aplicando conceptos y métodos sobre el uso de datos en el proceso de desarrollo para la gestión de calidad de sistemas software. Por lo tanto, dentro del objetivo genérico de mejorar la calidad de software, cada equipo podía centrarse en un aspecto a su elección. Para ello, disponían de un listado de ideas, y era posible trabajar sobre una nueva idea propuesta. Cinco equipos se centraron en la predicción y clasificación de commits respecto a su introducción de fallos o hediondez del código (en inglés “code smell”). Por ejemplo, un equipo integró su modelo de predicción en un sistema en el cual a partir de característica de un commit (como descripción, número de líneas añadidas/modificadas/eliminados, porcentaje de comentarios, complejidad de código, etc.) predecía la probabilidad de que contuviera fallos. Un equipo clasificó tipos de commits, sin elaborar algoritmos de predicción. Dos equipos se centraron en la clasificación de los desarrolladores creando perfiles en base a las características de su código (p.ej., uso de comentarios, complejidad de las funciones, realización de tareas de refactorización, trabajo en resolver fallos, etc.). Este tipo de perfil de desarrolladores son usados en empresas software para la contratación de personal. Otro equipo clasificó proyectos por su capacidad de mantenimiento y sostenibilidad en el tiempo.

4.3. Cuestionario final

El objetivo del cuestionario final era conocer el grado de satisfacción del alumnado. La primera pregunta trataba de ver la utilidad de aplicar un proceso de minería de datos. Como muestra el Cuadro 2, la aplicación de CRISP-DM les parece útil para alcanzar los objetivos de la Sección 4.2 (Media = 4.21, sobre 5). Esta pregunta fue seguida de otra pregunta con espacio abierto para dejar comentarios. En esta pregunta abierta, 17 alumnos dijeron explícitamente que volverían a usar CRISP-DM en otros proyectos: "*definitivamente lo tendré en cuenta porque es una manera muy sensata y bien estructurada de trabajar*", "*es bueno para la organización, como una convención y como una buena manera de saltar a los proyectos de ciencia de datos. También creo que es muy eficiente. Por supuesto que sí, lo utilizaré, y también lo he estado utilizando*", "*sí, fue útil para saber lo que había que hacer, no perder de vista el proyecto y lograr nuestros objetivos*", "*creo que es útil ya que nos da la pauta de las partes de un proyecto, así como lo que hay que incluir en cada una de ellas*". Sin embargo, tres alumnos expresaron explícitamente algunas dudas respecto a su uso futuro. Algunos ejemplos: "*no creo que tenga una opinión firme sobre CRISP-DM todavía, ya que no creo que lo haya explorado lo suficiente. Pero creo que la guía era un poco confusa a veces y un poco larga*", "*en mi opinión, lo encuentro un poco rígido y repetitivo, pero supongo que es la mejor manera de diseñar una estructura como esta, ya que hay que otorgarle coherencia*".

La segunda pregunta trataba de ver la utilidad de aplicar una estructura de proyecto. Como muestra el Cuadro 2, los alumnos consideraron útil utilizar la estructura del proyecto "Cookiecutter Data Science" para fomentar la corrección y la reproducibilidad (Media = 4.04). Dicha estructura de proyecto no recibió comentarios negativos. Entre los comentarios positivos podemos encontrar, por ejemplo: "*es bastante interesante ver el equilibrio entre la libertad de organización, la exploración y el trabajo con 'otros ojos', y el orden que se consigue con estructuras de proyectos como 'Cookiecutter Data Science'*", "*encuentro muy útil la estructura del proyecto 'Cookiecutter Data Science'. Me ha ayudado a organizar mejor el software y, sobre todo, a ser limpio. Parece 'profesional' cuando tu software está bien estructurado*". Sin embargo, desde el profesorado, sí tenemos que indicar que durante el proyecto hubo varias dudas respecto a algunos apartados opcionales de la plantilla que fueron planteadas por e-mail.

La sesión "*pregúntame lo que quieras*" fue bien acogida (Media = 4.11). De hecho, los alumnos opinaron que podría haber sido incluso más larga de una hora. Al ser videoconferencia, los panelistas sí mostraron su interés en ver las caras de los alumnos durante la sesión, y no sólo de la alumna que realizaba la pregunta.

La actividad peor acogida fue el libro digital (Media = 3.45). Este libro digital está inspirado en otros cursos para fomentar el *aprendizaje colaborativo* [21]. Los estudiantes no dieron retroalimentación explícita de los problemas. Una hipótesis a investigar en las próximas ediciones es si mejora la acogida si se crea este libro digital desde el comienzo del proyecto.

Las tres últimas preguntas del cuestionario (Cuadro 2) fueron sobre aspectos de ABP. Los alumnos están de acuerdo que les fue útil para aprender (Media = 3.86), con que el proyecto fue factible en un contexto de software (Media = 4.52), y que el trabajo en equipo les ayudó a aprender (Media = 4.31). Estas preguntas fueron seguidas de otra pregunta con espacio abierto para dejar comentarios sobre problemas o retos resueltos durante el proyecto.

Pregunta	**N**	**Mínimo**	**Máximo**	**Media**	**Mediana**	**Moda**	**σ**
La aplicación de CRISP-DM me parece útil para alcanzar los objetivos de nuestro proyecto.	29	2	5	4.21	4.00	4[a]	.819
Considero que utilizar la estructura del proyecto "Cookiecutter Data Science" es útil para fomentar la corrección y la reproducibilidad.	28	2	5	4.04	4.00	5	.999
La sesión "pregúntame lo que quieras" me ayudó a aprender	27	2	5	4.11	4.00	4	.801
El libro digital común en el laboratorio me ayudó a aprender	29	1	5	3.45	3.00	3	.985
El proyecto en el laboratorio me ayudó a aprender	29	1	5	3.86	4.00	4	1.093
El trabajo en equipo en el laboratorio me ayudó a aprender	29	3	5	4.31	4.00	5	.712
Para mi equipo en el laboratorio era factible realizar un proyecto de ciencia de datos en un contexto de software.	29	4	5	4.52	5.00	5	.509

Cuadro 2: Cuestionario final de satisfacción del laboratorio. Las respuestas se calificaron en la escala de 1 a 5 con el siguiente significado: 1 - muy en desacuerdo; 2 - en desacuerdo, 3 - neutral, 4 - de acuerdo, 5 - muy de acuerdo.

[a] Existen múltiples modas. Se muestra el valor más pequeño.

Seis estudiantes mencionaron explícitamente que les costó la parte de comprensión de los datos. Algunos ejemplos: "*los principales problemas fueron la comprensión de los datos y el significado exacto de las variables, a pesar de lo cual fue bastante bien*", "*el principal problema fue lidiar con el nuevo conjunto de datos, es decir, tratar de entender qué significa cada variable, sus valores...*" Otros dos estudiantes encontraron retos al definir un objetivo y criterios de satisfacción como implica la comprensión de negocio en CRISP-DM: p.ej., "*los principales problemas tal vez hayan sido definir el conjunto de datos/objetivos, etc., pero con los comentarios y la comunicación cada semana todo ha funcionado bastante bien*". Finalmente, nueve estudiantes indicaron que el principal reto fue hacer el proyecto en siete semanas: p.ej., "*no fue muy difícil, pero tal vez un poco más de tiempo habría sido apreciado*".

Finalmente, al igual que en el cuestionario inicial, se preguntó sobre conocimientos en ingeniería del software (un 90% afirmó que conocía qué era, respecto al 17% inicial) y CRISP-DM (un 100% afirmó que tenía experiencia, respecto al 6% inicial) en el cuestionario final. Además, en la última pregunta de satisfacción general con el laboratorio, tan sólo un 10% se mostró insatisfecho con los ejercicios de laboratorio.

4.4. Discusiones y posibles mejoras

En general, desde el profesorado hacemos una evaluación positiva y estamos satisfechos con el rendimiento académico (nota media total de TAED2 de 7.5, y nota media de los proyectos 8.9). Con la experiencia del curso piloto, hay datos suficientes para reflexionar sobre la retroalimentación de los estudiantes y realizar mejoras en la próxima edición.

Respecto a la rúbrica de evaluación, pensamos realizar dos mejoras. La primera, simplificar la rúbrica mostrando explícitamente qué partes de la estructura de proyecto son opcionales. Por otra parte, incluir un punto de evaluación en el ecuador del proyecto con la mitad del informe (hasta la comprensión de los datos).

Respecto al contenido, hemos visto que los alumnos han mostrado mucho interés en la convención de estructura de proyecto de ciencia de datos. Con esto, analizaremos reemplazar el último objetivo de aprendizaje de la asignatura y añadir un nuevo objetivo para enseñar otras buenas prácticas de ingeniería del software para este tipo de proyectos [16]. Como un estudiante pidió en los comentarios abiertos: "*más educación sobre buenas prácticas*". Esto puede facilitar el despliegue del modelo realizado en el proyecto en un sistema basado en inteligencia artificial.

Finalmente, para resolver los principales retos mencionados (dificultad de comprensión de los datos y tiempo limitado) la primera semana se introducirán más detalladamente conceptos de las variables del conjunto de datos (en el curso se hizo brevemente con documentación del conjunto de datos), y se pondrá un límite de páginas para el informe realizado para reducir el tiempo empleado en documentación.

5. Trabajo relacionado

En la última década, han proliferado numerosos cursos de analítica de software asociados a minería en repositorios de software y la conferencia MSR (del inglés Mining Software Repositories). Un excelente ejemplo en España es el curso "Data Analysis in Software Engineering using R" de Daniel Rodríguez y Javier Dolado[10].

Con la última ola de inteligencia artificial y aprendizaje profundo, podemos ver también la necesidad de enseñar técnicas de ingeniería del software para crear sistemas basados en inteligencia artificial [16]. Dos experiencias docentes relevantes se han llevado a cabo en el último año en la Universidad Carnegie Mellon [11] y la Universidad Fontys de Eindhoven [10].

La experiencia docente de este documento es innovadora en aplicar analítica de software con estudiantes de ciencias de datos, así como enseñar buenas prácticas de ingeniería del software (p.ej., convención de estructura de proyectos) a este perfil de estudiantes en la UPC. Como trabajo futuro, se estudiará la inclusión de más buenas prácticas de ingeniería del software en TAED2. Además, en la última fase de despliegue, se estudiará evaluar la creación de un pequeño sistema software basado en el modelo creado y evaluado. Un ejemplo creado por un equipo de TAED2 ha sido una aplicación web donde se incluían las características de un commit y predecía la probabilidad de contener fallos.

6. Conclusiones

En esta experiencia docente comenzamos preguntándonos: ¿cómo educar a estudiantes de ciencias de datos para realizar proyectos de analítica de software? Esta ponencia ha mostrado la experiencia docente de diseñar, ejecutar y evaluar una nueva actividad de ABP en el laboratorio de la asignatura TAED2 del GCED en la UPC. El objetivo de dicha actividad de proyecto ha sido detectar oportunidades de mejora de la calidad de software en repositorios de código abierto en base a datos.

[10] https://danrodgar.github.io/DASE/

La experiencia de aplicar ABP en este contexto ha sido positiva. Además, los estudiantes destacaron la utilidad del proceso de minería de datos CRISP-DM y la estructura del proyecto de datos.

Agradecimientos

Nos gustaría agradecer a Davide Taibi y Santiago del Rey, por la ayuda con el conjunto de datos, y también a Adam Trendowicz, Davide Taibi, y Christine Doig-Cardet por su participación como panelistas en la sesión de "pregúntamente lo que quieras". También agradecemos a los profesores del Institut de Ciències de l'Educació (ICE) de la UPC, por las actividades para mejorar la docencia. Finalmente, esta experiencia docente está parcialmente financiada por el programa "Beatriz Galindo".

Referencias

[1] Andrew Begel y Tom Zimmermann. Analyze this! 145 questions for data scientists in software engineering. En *36th International Conference on Software Engineering*, pp. 12-23, 2014.

[2] Mary Besterfield-Sacre, Larry Shuman, Harvey Wolfe, et al. Defining the outcomes: A framework for EC-2000. *IEEE Transactions on Education*, 43(2), 100-110, 2000.

[3] Jacob Bishop y Matthew Verleger. The flipped classroom: A survey of the research. En *ASEE national conference proceedings*, Vol. 30, No. 9, pp. 1-18, 2013.

[4] Benjamin Bloom. Taxonomy of educational objectives. Vol. 1: Cognitive domain. *New York: McKay*, 20, 24, 1956.

[5] Ronald Brachman y Tej Anand. The Process of Knowledge Discovery in Databases: A First Sketch. En *KDD workshop* vol. 3, pp. 1-12, 1994.

[6] Clive Dym et al. Engineering design thinking, teaching, and learning. *Journal of engineering education*, 94(1), 103-120, 2005.

[7] Christof Ebert, Jens Heidrich, Silverio Martínez-Fernández, et al. Data science: technologies for better software. *IEEE Software*, 36(6), 66-72, 2019.

[8] Harald Gall, Tim Menzies, Laurie Williams, et al. Software development analytics (dagstuhl seminar 14261). En *Dagstuhl Reports (Vol. 4, No. 6). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik*, 2014.

[9] Fiorina Gatica-Lara y Teresita del Niño Jesús Uribarren-Berrueta. ¿Cómo elaborar una rúbrica? *Investigación en educación médica*, 2(5), pp. 61-65, 2013.

[10] Petra Heck y Gerald Schouten. Turning Software Engineers into AI Engineers. *arXiv preprint* arXiv:2011.01590, 2020.

[11] Christian Kästner, Eunsuk Kang. Teaching Software Engineering for Al-Enabled Systems. En *IEEE/ACM 42nd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, pp. 45-48, 2020.

[12] Miryung Kim, Thomas Zimmermann, Robert DeLine, et al. Data scientists in software teams: State of the art and challenges. *IEEE Transactions on Software Engineering*, 44(11), 1024-1038, 2017.

[13] Lukasz Kurgan y Petr Musilek. A survey of knowledge discovery and data mining process models. *Knowledge Engineering Review*, 21(1), pp. 1-24, 2006.

[14] Valentina Lenarduzzi, Nyyti Saarimäki, y Davide Taibi. The technical debt dataset. En *15th International Conference on Predictive Models and Data Analytics in Software Engineering*, pp. 2-11, 2019.

[15] P. Santhanam, Eitan Farchi, Victor Pankratius. Engineering reliable deep learning systems. *arXiv preprint* arXiv:1910.12582, 2019.

[16] Silverio Martínez-Fernández, Justus Bogner, Xavier Franch, et al. Software Engineering for AI-Based Systems: A Survey, *arXiv:2105.01984*, 2021

[17] Ezequiel Scott, Fredrik Milani, y Dietmar Pfahl. Data Science and Empirical Software Engineering. En *Contemporary Empirical Methods in Software Engineering*, pp. 217-233, 2020.

[18] Stefan Studer, Thanh Bui, Christian Drescher, et al. Towards CRISP-ML (Q): a machine learning process model with quality assurance methodology. *arXiv preprint* arXiv:2003.05155, 2020.

[19] Dongmei Zhang, Shi Han, Yingnong Dang, et al. Software analytics in practice. *IEEE software*, 30(5), pp. 30-37, 2013.

[20] Miguel Valero, Javier García. Cómo empezar fácil con PBL. *Jornadas de Enseñanza Universitaria de la Informática* (17es: 2011: Sevilla), 2011.

[21] Arie Van Deursen, Maurício Aniche, Joop Aué, et al. A collaborative approach to teaching software architecture. En *ACM SIGCSE Technical Symposium on Computer Science Education*, pp. 591-596, 2017.