Implementación de una arquitectura online para el aprendizaje de la programación en Python

Memoria del Proyecto de Innovación Docente ID2017/006

Alejandro Benito Santos – abenito@usal.es
Rodrigo Santamaría Vicente – rodri@usal.es
Roberto Therón Sánchez – theron@usal.es

10 Julio 2018

Índice

1 Introducción

Cada vez más, el manejo de *herramientas informáticas* supone un factor clave en el desempeño de cualquier tarea profesional cualificada. A su vez, el concepto de *herramienta informática* evoluciona a la vertiginosa velocidad que dicta la ley de Moore. Mientras que en el pasado esta concepción se limitaba principalmente, salvo en algunos pocos casos, al denominado campo de la ofimática y al manejo de alguna herramienta software generalista o específica de la disciplina (ej: AutoCAD, SPSS), en el presente la tendencia está cambiando: El aumento en volumen y en complejidad de los conjuntos de datos requiere de profesionales excelentes en el manejo de computadoras a niveles de maestría impensables hasta hace sólo unos pocos años. Además, el trabajo con ordenadores ha estado tradicionalmente más ligado a las disciplinas STEM (Science, Technology, Engineering & Mathematics) lo cual sigue siendo un hecho patente hoy en día.

Las nuevas generaciones de egresados pertenecen al grupo de los denominados "Nativos Digitales"[1], que se diferencian de las anteriores en que adquirieron familiaridad con los sistemas digitales desde su infancia. Es por esto que los viejos métodos de aprendizaje basados en el papel están quedando obsoletos a marchas forzadas no sólo porque son en su mayor parte ya inútiles para el cometido que se espera que desempeñen los nuevos titulados, sino porque además resultan inaccesibles y lejanos a ojos de aquellos. Es por tanto que como docentes tenemos la responsabilidad de proveer un contexto adecuado de aprendizaje para la Ciencia empleando técnicas y métodos didácticos que, sin comprometer el rigor que se espera de la enseñanza universitaria, resulten comprensibles para la totalidad del alumnado y la comunidad científica del presente y del futuro (Holdgraf et al., 2017).

En este dilema que se plantea se enmarca este Proyecto de Innovación Docente, que se llevó a cabo entre los meses de Septiembre de 2017 y Enero de 2018 con 43 alumnas y alumnos del curso "Informática I" del Grado en Estadística de la Universidad de Salamanca. Mediante la adaptación de una metodología novel y un ecosistema de herramientas interactivas para el aprendizaje creados recientemente en la Universidad de Berkeley en California se impartieron los contenidos prácticos de la citada asignatura, dedicando para ello un tiempo de clase de 2 horas

[1] https://es.wikipedia.org/wiki/Nativo_digital

semanales durante el primer cuatrimestre del curso. El proyecto fue financiado con la cantidad de 210€ por el Vicerrectorado de Docencia.
En este proyecto se pone en práctica una arquitectura de software basada en notebooks interactivos de Python junto a una metodología de aprendizaje adaptada de experiencias previas. De manera general, un notebook es un documento electrónico que puede ser leído como si se tratase de una revista, pero también ejecutado como un programa de ordenador. A pesar de que esta idea no es nueva, el enfoque de notebook computacional ha cobrado fuerza recientemente en el mundo de la enseñanza de las ciencias de la computación (Hamrick, 2016; O'Hara et al., 2015). Existe por tanto un interés dentro de la comunidad docente en alejarse de la configuración tradicional que promociona la comunicación unidireccional profesor-alumno, que en muchas ocasiones dificulta o imposibilita completamente el aprendizaje por parte del estudiante. Recientes avances en la computación en la nube (o cloud) y en la virtualización facilitan enormemente las tareas de configuración necesarias para poner en marcha los llamados MOOC (Massive Open Online Courses), los cuales acarrean importantes ventajas a la hora de facilitar la tarea del aprendizaje al alumno (Cruz-Benito et al., 2015). Nuestro enfoque incorpora algunas de estas tecnologías y las conjuga con las ya existentes dentro del entorno de la Universidad de Salamanca para tratar de obtener una experiencia óptima de aprendizaje en un perfil de alumno que acaba de iniciar su formación universitaria.

El contenido de esta memoria se organiza como sigue: En la sección 2 Equipo de Trabajo se presenta a los responsables de este proyecto. En la sección 3 se citan otros proyectos del grupo de investigación al que pertenecen los autores que han servido como referentes para la realización del presente estudio. En la sección 4 se enumeran los objetivos del proyecto, mientras que en la sección 5 se comentan los métodos y materiales empleados, así como la arquitectura de aprendizaje propuesta. Las secciones 6 y 7 comentan en la evaluación y dispersión de resultados para cerrar este texto con unas conclusiones en la sección 8.

2 Equipo de Trabajo

- **Coordinador**: Alejandro Benito Santos – Prof. Asociado del Dpto. de Informática y Automática. Área de las Ciencias de la Computación e Inteligencia Artificial. abenito@usal.es
- **Responsable de la asignatura**: Rodrigo Santamaría Vicente – Prof. Titular de Universidad del Dpto. de Informática y Automática. Área de las Ciencias de la Computación e Inteligencia Artificial. rodri@usal.es
- **Soporte y evaluación externa**: Roberto Therón Sánchez – Prof. Titular de Universidad del Dpto. de Informática y Automática. Área de las Ciencias de la Computación e Inteligencia Artificial. theron@usal.es

3 Proyectos relacionados

Este Proyecto de Innovación guarda estrecha relación con otros trabajos de investigación (García-Holgado et al., 2017; Santamaría Vicente, n.d.; Therón et al., 2017) de la Universidad de Salamanca y en concreto del Grupo de Investigación en Interacción y e-Learning GRIAL[2].

4 Objetivos del proyecto

4.1 Objetivos generales

Los objetivos generales que se marcaban al comienzo del proyecto eran los siguientes:

4.1.1 Acelerar el proceso de aprendizaje de la programación por medio del empleo de *Jupyter Notebooks* durante las sesiones prácticas de la asignatura

Como se ha mencionado anteriormente, dadas las características del curso se quería conseguir eliminar una serie de tareas repetitivas, tanto para el profesorado como para los participantes en el curso, que pueden suponer un escollo al aprendizaje, a veces tristemente insalvable para cierto perfil de alumno. Estas tareas se identificaron inicialmente como:

- Mantenimiento y configuración de equipos.
- Instalación de la pila de software (entorno de programación, editor de texto, control de versiones…).
- Confección, distribución, recogida y evaluación de ejercicios periódicos, semanales y trimestrales.
- Comunicación de calificaciones y envío de *feedback* al alumnado.

[2] http://grial.eu/

- Otras tareas relacionadas con la toma rápida de decisiones durante el curso (preparación de ejercicios adecuados, revisión de conceptos que se detecten problemáticos, etc…)

4.1.2 Seguimiento completo del progreso y satisfacción del alumnado.

Gracias a la automatización del proceso de revisión de prácticas, es posible proveer al alumno con realimentación rápida y certera sobre su progreso en el curso. Esto, junto a las sesiones de tutorías permite un seguimiento adecuado de los alumnos y una toma rápida de decisiones durante el curso que redundan en la calidad general del curso.

4.1.3 Evaluación y diseminación satisfactorias de los resultados obtenidos a lo largo del curso

Se ha hecho hincapié en generar contenidos públicos y abiertos que puedan ser consultados en Internet. Para ello, todo el código fuente relacionado con el curso está disponible en un repositorio de código con fin de ser consultado y/o reutilizado en el futuro.

4.2 Objetivos específicos de la metodología

La metodología propuesta se centra en la colaboración organizada entre los alumnos para resolver los problemas típicos de un curso introductorio a la programación. Entre los principios clave que se tuvieron en cuenta a la hora de diseñarla se encuentran:

- Promoción del trabajo en equipo y la colaboración entre los alumnos.
- Participación proactiva de los alumnos en las horas de clase
- Trabajo fuera de las horas de clases a través de los recursos online expuestos en la arquitectura que se plantea.
- Incentivación del auto-aprendizaje y estimulación de la curiosidad por medio de notebooks interactivos.

5 Materiales y métodos

5.1 Encuesta inicial

Con el objetivo de conocer las motivaciones del alumnado y la experiencia previa en el uso de ordenadores y programación, al comienzo del curso se distribuyó una encuesta a los alumnos con preguntas generales cuyos resultados se comentan a continuación. La encuesta fue distribuida a través de la plataforma Studium y fue completada por 32 de 43 participantes en el curso (74.4%).

5.1.1 Edad y Género

La clase estaba compuesta por 19 mujeres (44.1%) y 24 hombres (55.9%). En los resultados de la encuesta se observó la distribución de edades que se adjunta en la Figura 4:

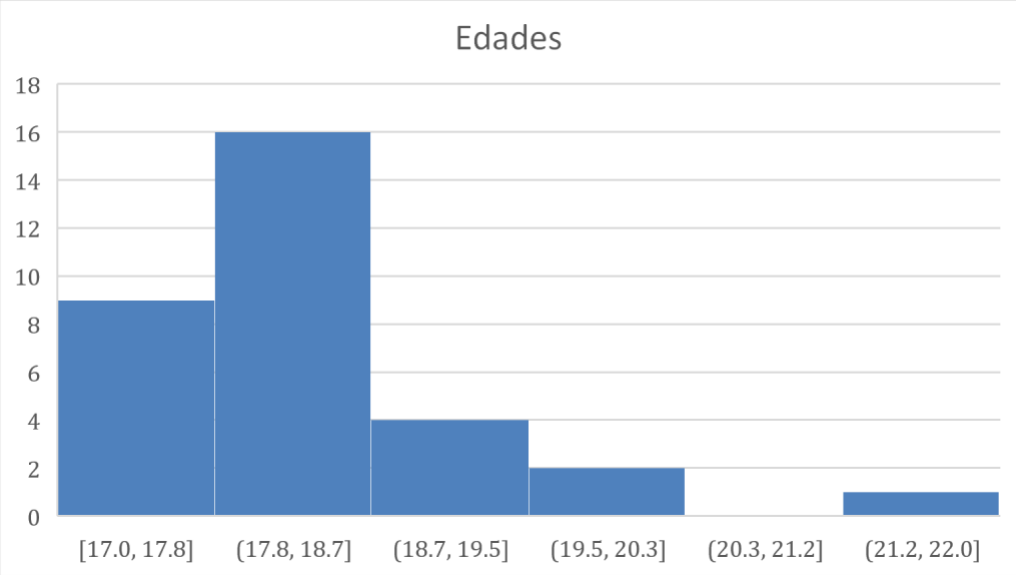

Figura 1: Distribución por edades de los participantes en la encuesta inicial, típica de un primer curso universitario

5.1.2 Conocimientos previos

26 de los alumnos que respondieron a la encuesta (81%) había cursado el Bachillerato en su modalidad de Ciencias y Tecnología, mientras que sólo 6 (19%) provenían de la modalidad de Humanidades y Ciencias Sociales.

En cuanto a conocimientos de programación 6 alumnos afirmaron contar con algún tipo de experiencia previa, de los cuales 3 eran repetidores de la asignatura. Los 3 restantes habían adquirido dicha experiencia en el bachillerato, en los lenguajes C (1), C++ (1) y Python (1). Con estos datos pudimos corroborar que el alumno medio era completamente inexperto en tareas de programación e incluso en tareas de informática básica.

5.1.3 Maestría en STEM e inglés

En el siguiente bloque de preguntas se trató de evaluar el nivel de maestría en disciplinas STEM ya que esta es importante para comprender conceptos de programación. Para dilucidar esta cuestión se les preguntó por las asignaturas en las que obtuvieron la nota más alta y más baja en la EBAU. Además, y debido a que la programación está íntimamente ligada al idioma inglés, se les pidió también que aportaran su nota en dicha asignatura. Los resultados fueron los siguientes:

- La nota media del grupo en la EBAU fue de 8.21 sobre 14, equivalente a un 5.8 si se pondera sobre 10 (Aprobado).

- 12 alumnos (37.5%) obtuvieron su nota más alta en la asignatura de Matemáticas. La nota media de estos alumnos en dicha prueba fue
- 5 alumnos obtuvieron la nota más alta en inglés. La nota media en inglés fue de 6.77 y la mediana, 7.

5.1.4 Predisposición del alumnado hacia los contenidos

Se trató de evaluar la predisposición a la materia que tenían los alumnos mediante recogida de su preferencia a la hora de elegir titulación universitaria. Los datos recogidos se muestran en la Figura 2 y muestran una clara inclinación hacia la Estadística y las Matemáticas, materias clave de la asignatura, de lo cual podemos deducir que el interés en la materia era, a priori, alto.

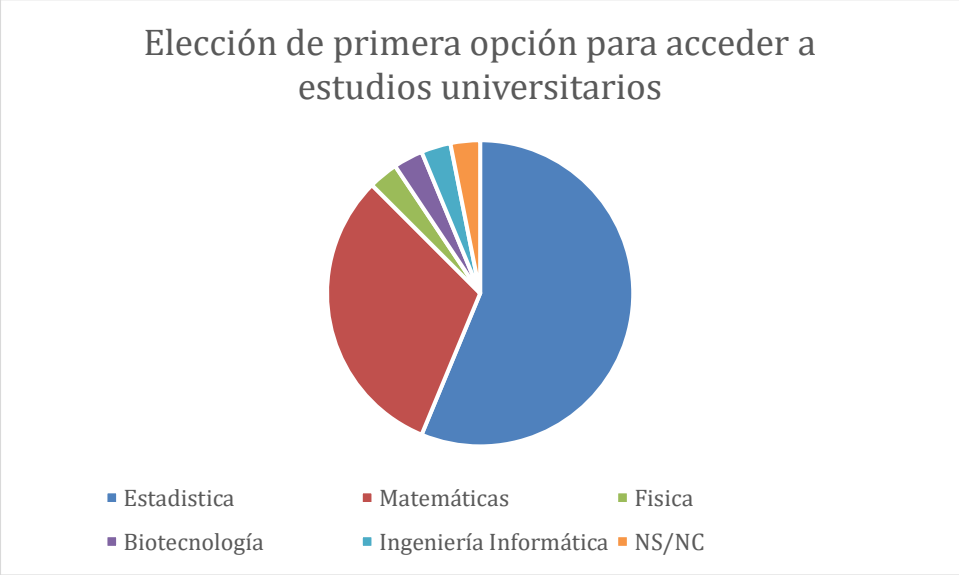

Figura 2: Distribución de resultados de la elección de primera opción para acceder a estudios universitarios. La mayoría de estudiantes encuestados escogieron Estadística o Matemáticas.

5.1.5 Acceso a recursos informáticos

La totalidad de los alumnos encuestados afirmó contar con acceso a un ordenador personal en propiedad. 31 de 32 alumnos contaban con el sistema operativo Windows (en sus versiones 8 y 10) instalado en sus equipos. El alumno restante disponía de un equipo Apple.

5.1.6 Expectativas sobre el curso

Para finalizar, se les pidió a los alumnos que indicasen cuáles eran sus expectativas sobre el curso que comenzaba. En la nube de palabras de la Figura 3 se cuantifica la frecuencia de términos de las respuestas recogidas, indicando una peso destacado de los términos “aprender” y “aprobar”, que se colocan a la par.

Figura 3: Nube de palabras que recoge las respuestas a la pregunta "Resume brevemente tus expectativas para este curso". Los términos más repetidos fueron aprender y, por supuesto, aprobar.

En general, las conclusiones en líneas generales que el equipo de trabajo extrajo de esta encuesta fueron las siguientes:

- Un gran número de alumnos tiene interés por aprender los contenidos del curso.
- Una parte significativa de los alumnos domina con solvencia las matemáticas, de acuerdo con los estándares educativos marcados en el Bachillerato.
- La gran mayoría de los alumnos, a pesar de pertenecer a la llamada generación de "Nativos Digitales", tiene un nivel bajo de alfabetismo digital. Dado que la mayoría de ellos provenía de la rama de Bachillerato de Ciencia y Tecnología, esto podría indicar carencias en el diseño e implantación de los planes de estudio a este nivel.
- El nivel de inglés de los participantes es alto de acuerdo con los estándares marcados en el Bachillerato.

5.2 Metodología

La metodología didáctica propuesta se organizó en torno a tres pilares de evaluación: **Evaluación continua, prácticas obligatorias y examen práctico** de programación en ordenador. Se describen en el resto de esta sección las particularidades de cada una de ellas:

Las sesiones prácticas semanales se dividen en dos partes y se llevan a cabo por parejas (siguiendo técnicas de colaboración por pares y pair programming) en un equipo compartido del aula de informática. En un estudio de referencia del año 2002, los autores McDowell et al. demuestran las virtudes de este enfoque en un curso de nivel introductorio a la programación como era nuestro caso (McDowell et al., 2002) y por este motivo se eligió esta forma de trabajo.

El lenguaje de programación elegido fue Python en su versión 3 en contraposición al lenguaje C que venía siendo la elección por defecto en años anteriores. Se decidió optar por este lenguaje debido a los siguientes motivos que se consideraron de peso:

1. Promueve un estilo de programación **imperativo** y **funcional**. Permite aprender conceptos de alto nivel de la programación de manera sencilla.
2. Es un lenguaje **interpretado** o de *scripting*, de manera que es más susceptible a ser integrado en entornos **dinámicos** e **interactivos** como *Jupyter*[3]. La ejecución del código es por tanto más transparente a ojos de usuarios inexpertos ya que se evita la tarea del compilado.
3. Posee un **modo interactivo** en el que se pueden escribir instrucciones que son evaluadas inmediatamente por el intérprete. Esto, en conjunción con el entorno Jupyter, permite que los alumnos puedan probar pequeños trozos de código rápidamente online.
4. Posee licencia de **código abierto** y su uso es **libre y gratuito**, haciéndolo ideal para su uso en entornos educativos.
5. El tipado es **dinámico** y la **gestión de memoria**, **automática**, lo cual resta complejidad a la consecución de los objetivos de un curso de iniciación a la programación.
6. Posee una **fuerte relación con la estadística** y en concreto con el denominado "**Data Science**". Junto al lenguaje R, Python se ha convertido en **estándar de facto** en esta disciplina y recientemente ha superado en interés y popularidad a otros lenguajes clásicos como C (Ver Figura 4).

[3] Jupyter es el servidor de notebooks empleado para la impartición de los contenidos de la asignatura y es pieza clave en el organigrama de este proyecto. Para más información ver http://jupyter.org/

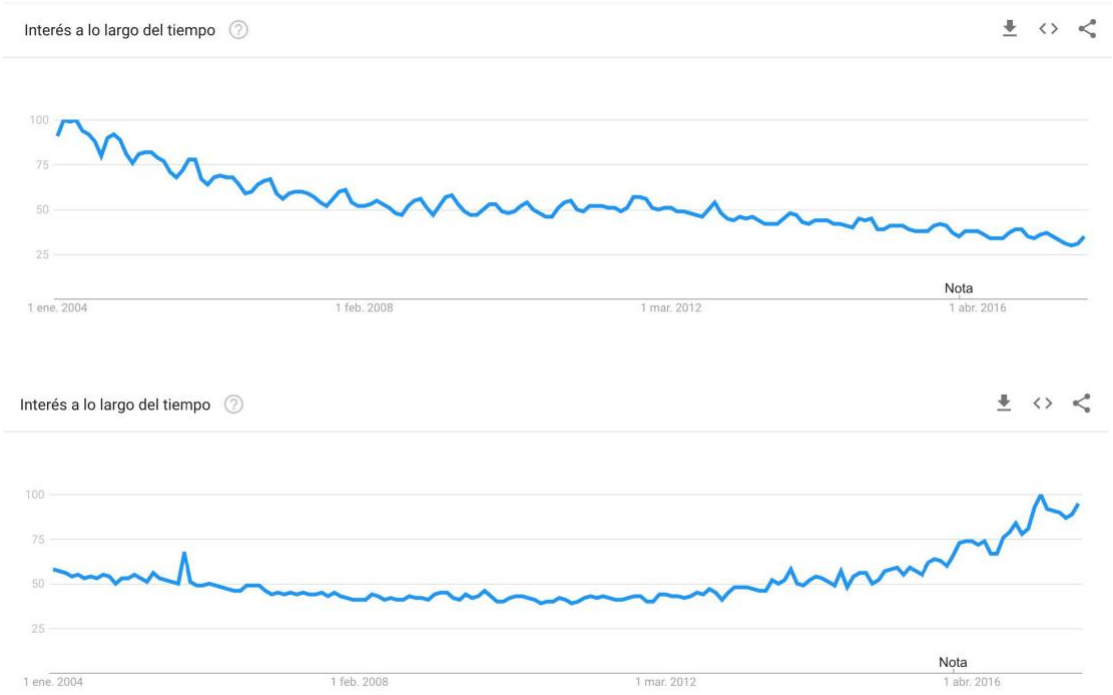

Figura 4: Gráfico que muestra la evolución en Google Trends del interés de búsqueda en los lenguajes de programación C (arriba) y Python (abajo). El segundo es un lenguaje mucho más popular en la actualidad.

5.2.1 Evaluación continua

La evaluación continua se llevó a cabo durante las sesiones de prácticas del curso, con una duración de 2 horas en horario de tarde. Durante la primera parte de la sesión se realizaba una introducción a los contenidos a través de un notebook interactivo creado a tal efecto. A este notebook se le denominó "*lab*", para diferenciarlo de los trabajos prácticos ("tareas") que se realizarían en la segunda parte de la sesión. Al inicio de cada una de las sesiones, se publicaba un enlace en la plataforma Studium para que los alumnos pudieran "clonar" los contenidos del *lab* en su instancia personal de *Jupyter Notebook*. Dicha instancia se encontraba ejecutándose en el servidor de la asignatura (ver sección 5.3) y de manera transparente, los contenidos eran capturados desde un repositorio de código alojado en una URL de la plataforma colaborativa *GitHub* y copiados a la carpeta personal del alumno.

5.2.1.1 Labs

El repositorio de código fue siendo actualizado paulatinamente durante el curso con nuevos *labs*. El estado final del mismo está disponible de forma pública bajo la licencia Creative Commons en la URL: *https://github.com/ale0xb/materiales-informatica-i*. Si se accede a dicho enlace, se puede navegar por los diferentes contenidos que fueron estructurados de la siguiente manera:

- Introducción (Lab 0): Lección introductoria en la que se explican los fundamentos de Python, Jupyter Notebook, y posible resolución de errores.
- Variables (Lab 1): Variables y tipos. E/S estándar.
- Cadenas y flujo (Lab 2): Cadenas. Caracteres especiales. Operadores. Formateado de cadenas. Sentencias *if, for, range, while, break y continue*.
- Funciones y módulos. E/S no estándar (Lab 3): Definición de funciones. Paso por referencia/valor, parámetros obligatorios y por defecto. Funciones anónimas. Importando módulos. Apertura, cerrado, lectura y escritura desde ficheros (*tell*, *seek*).
- Listas y tuplas (Lab 4): Declaración de listas y tuplas. Operaciones básicas con listas y tuplas (*reverse, len, del*…).
- Copias superficiales y profundas. Diccionarios (Lab 5): Concepto de copia en Python (*slice, deepcopy).* Acceso y Operaciones con diccionarios (*pop*, *popitem, clear, update).*
- Conjuntos y Conjuntos Congelados (Lab 6): Introducción al concepto de conjunto. Posibles Aplicaciones. Conjuntos inmutables y congelados. Operaciones con conjuntos.

5.2.1.2 Tareas

Para afianzar la comprensión de los contenidos se propusieron una serie de tareas relacionadas con el contenido explicado en la primera mitad de la sesión, que los alumnos tendrían que trabajar durante el resto de la clase de manera colaborativa (tanto con su pareja como con el resto de alumnos). Las dudas que surgían se resolvían de manera pública en la clase. Las tareas se ponían a disposición de los alumnos a través del módulo de *Jupyter Hub nbgrader* [4] *(Hamrick, 2016).* Con posterioridad a la finalización del curso se añadieron también al repositorio de código, en el que ahora descansan en esta dirección. Los alumnos que no terminaban la tarea en la hora dedicada a tal efecto, debían completarla en horario fuera de clase y entregarla a través de la plataforma con anterioridad al comienzo de la siguiente sesión. Los resultados de la calificación de las tareas eran enviados por correo electrónico en un fichero HTML generado automáticamente por la herramienta a partir del *notebook*, como se muestra en la Figura 5. En la sección "Resultados obtenidos" se comentan aspectos sobre el progreso de los participantes en este sentido.

[4] https://github.com/jupyter/nbgrader

Ejercicio 2:

Escribe una función python que actúe como el método update() explicado en clase, pero que en vez de sobreescribir valores para claves repetidas los combine en una lista.

In [4]: Student's answer (Top)

```
def une(diccionario_1, diccionario_2):
    diccionario_3 = {} #Definimos el diccionario vacío
    for curso in diccionario_1:
        if curso in diccionario_2: #Buscamos las claves que esten en ambos diccionar
ios
            diccionario_3[curso] = [diccionario_1[curso], diccionario_2[curso]] #Aña
dimos al nuevo diccionario la lista con los valores de las claves que estén en ambos
diccionarios
        else: diccionario_3[curso] = diccionario_1[curso] #Añadimos las claves única
s del primer diccionario
    for curso in diccionario_2:
        if curso not in diccionario_1: #Buscamos las claves del segundo diccionario
que no estén en el primero
            diccionario_3[curso] = diccionario_2[curso] #Añadimos las claves únicas
del segundo diccionario
    return diccionario_3 #La función devuelve el nuevo diccionario
```

Comments:
¿Podrías usar un sólo diccionario? Esta sería la opción más correcta.

Figura 5: Feedback en formato HTML proporcionado a un alumno del curso.

5.2.2 Prácticas obligatorias

Se propusieron dos prácticas obligatorias, de dificultad y duración más elevadas que las tareas descritas previamente. El objetivo de estas prácticas era que los alumnos conectasen los diferentes bloques didácticos en programas más complejos. En concreto se propusieron dos ejercicios, que también se subieron al repositorio de código una vez acabado el curso. A la izquierda en la Figura 6 muestra la interfaz Jupyter ejecutándose en el servidor de la asignatura tal como la veían los alumnos. La manera de resolver las prácticas se basó en técnicas de TDD y en la consecución de subobjetivos con el fin de motivar al alumno para el aprendizaje (Edwards, 2003). En este enfoque los alumnos ejecutan una serie de tests embebidos en el cuaderno que comprueban que la implementación cumple el enunciado de la práctica y, por lo tanto, conocen su nota final antes de recibir realimentación por parte del profesor. En la parte derecha de la Figura 6 pueden consultarse los tests que se diseñaron para la primera práctica.

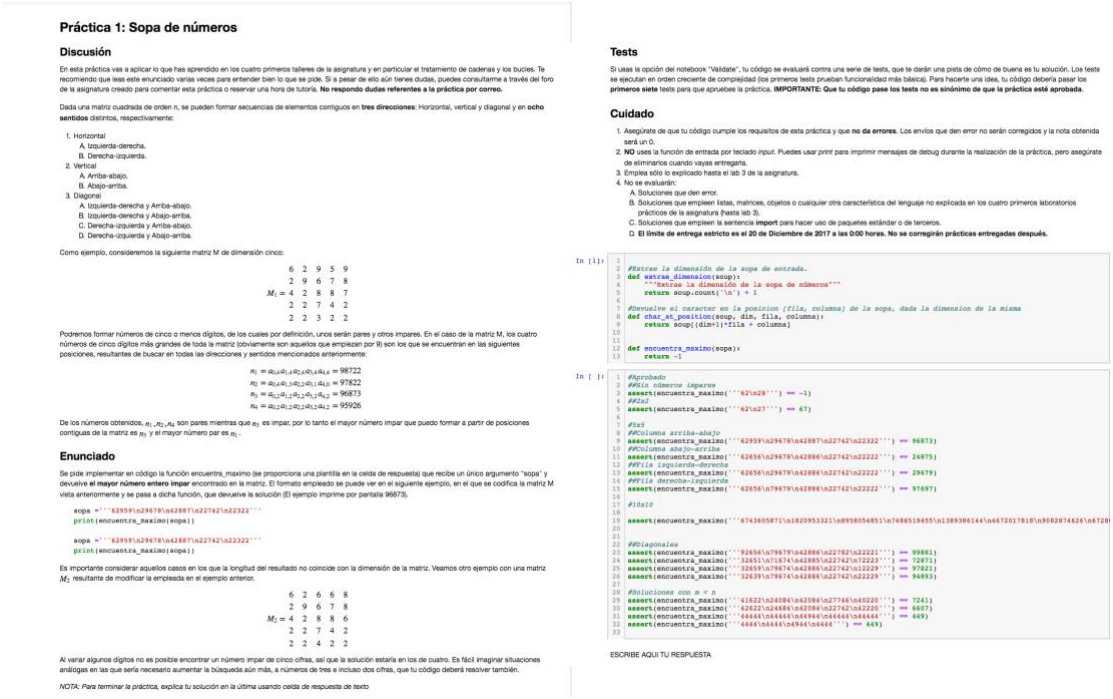

Figura 6: Captura de pantalla del cuaderno de trabajo de la práctica 1 ejecutándose en Jupyter. En la izquierda, en el enunciado de la práctica el lenguaje Markdown permite la inclusión de matrices y otros símbolos matemáticos. En la derecha, los tests proporcionados al alumno por los que progresará durante la realización del ejercicio.

5.2.3 Examen práctico en ordenador

El examen práctico de la asignatura se realizó en los ordenadores de un aula de informática de la Facultad de Ciencias y tuvo una hora de duración utilizando un notebook similar a los otros empleados en las tareas y prácticas obligatorias. Están disponibles de manera pública en el [repositorio en GitHub] del proyecto.

5.3 Arquitectura software

La solución arquitectónica de software que se planteaba contaba con *Jupyter Notebook* como pieza central, ya que era clave para conseguir el objetivo principal de eliminar tareas repetitivas como la instalación de software en los equipos de los alumnos. *Jupyter Notebook* es un entorno computacional interactivo basado en web para crear cuadernos o *notebooks* Jupyter. Como explicación sencilla, un cuaderno Jupyter es un documento JSON con un esquema asociado que contiene una lista de celdas de entrada/salida, las cuales pueden albergar código, texto (en formato *Markdown),* fórmulas matemáticas, gráficos y otro contenido multimedia como vídeo o imágenes. Normalmente adquiere la extensión ".ipynb". El proceso de instalación de *Jupyter Notebook* es relativamente sencillo para alguien con experiencia en la programación pero puede resultar complicado para usuarios inexpertos (Brown, n.d.). Por esta razón se decidió abstraer este proceso empleando *JupyterHub*[5], un servidor multiusuario de *Jupyter Notebook* que permite que cada alumno disponga de su propia instancia individual que es ejecutada desde el navegador de la máquina cliente de forma transparente. La extensión

Los alumnos accedieron a dicho servidor durante el curso, que fue desplegado en una máquina configurada a tal efecto dentro del dominio de la universidad, en concreto en el Instituto de Biología Funcional y Genómica. Se conectó el servicio con el servicio de identidad idUSAL por lo que no se necesitó de ningún proceso de registro por parte de los alumnos para acceder al servicio. El servidor estuvo en funcionamiento hasta Marzo de 2018, cuando tuvo que ser retirado para desempeñar otras tareas dentro del Instituto al que pertenecía.

6 Resultados obtenidos

En esta sección se presentan los principales resultados obtenidos durante la realización del presente proyecto de innovación docente. Se enfocará en base a los siguientes parámetros y marcadores de calidad que se tenían previstos al comienzo del proyecto.

6.1 Seguimiento de la participación

Se realizó un estudio de la participación y asistencia a los talleres, así como la entrega de tareas y prácticas obligatorias, con el objetivo de medir la tasa de abandono por parte del alumnado. El número de asistentes a la primera sesión (número máximo alcanzado) fue de 35 alumnos (81%), número más o menos constante a lo largo del curso, aunque decreció hacia el final del mismo. Hecho

[5] https://github.com/jupyterhub/jupyterhub

llamativo es que existen 8 alumnos que nunca acudieron a ninguna de estas sesiones, ni entregaron ninguna de las tareas propuestas.

En lo referente a las entregas de ejercicios prácticos, la Figura 7 muestra la evolución del número de entregas y aprobados. Como era de esperar, se produce un decremento de la participación en la parte final del curso, cuando la acumulación de tareas es mayor. A pesar de ello se mantuvieron unos porcentajes altos de asistencia a las sesiones presenciales (sobre todo si se pondera con el número de asistentes a las sesiones al comienzo del curso), así como también de aprobado de las prácticas obligatorias.

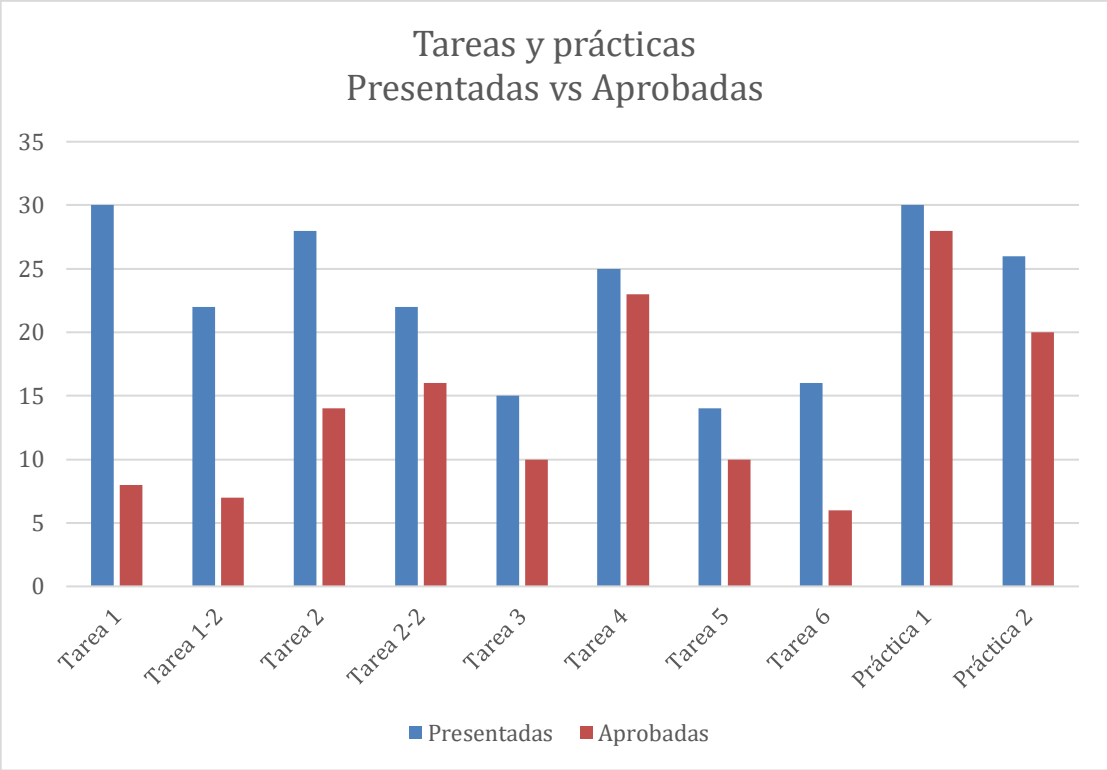

Figura 7: Evolución del número de alumnos que entregó las tareas y prácticas comparado con el número de aprobados. La participación bajó sensiblemente en aquellas últimas, quizás debido al hecho de que se solapaban con la entrega de las prácticas obligatorias

6.2 Encuesta de satisfacción con el Proyecto de Innovación Docente

Como parte final del proyecto, se llevó a cabo una encuesta en una de las sesiones finales a la que contestaron 22 alumnos (51% del total). El formulario se distribuyó en papel y consistía de una serie de preguntas relacionadas con los siguientes marcadores de calidad previstos al inicio del proyecto. A continuación se pasa a comentar los resultados del cuestionario revisando los bloques temáticos de preguntas inspirados por otros estudios (García-Holgado et al., 2017).

6.2.1 Metodología de trabajo personal

En líneas generales en este apartado se trata de comprender la percepción, en relación a su utilidad, que tiene el alumnado sobre el contenido de la asignatura. En la Figura 8 se muestran los resultados de este primer bloque, de los que se extraen las siguientes conclusiones:

- Los alumnos encuestados afirmó no haber comprendido completamente los contenidos de la asignatura.
- En líneas generales confiaban en que el contenido de la asignatura les sería útil como futuros profesionales de la Estadística, pero no así para comprender los contenidos de otras asignaturas del grado.
- Los encuestados coincidieron en haber consultado los recursos disponibles y en que la asistencia a las sesiones prácticas les fue útil para comprender los contenidos.
- Por último, mayoritariamente consideraron la asignatura como bastante difícil.

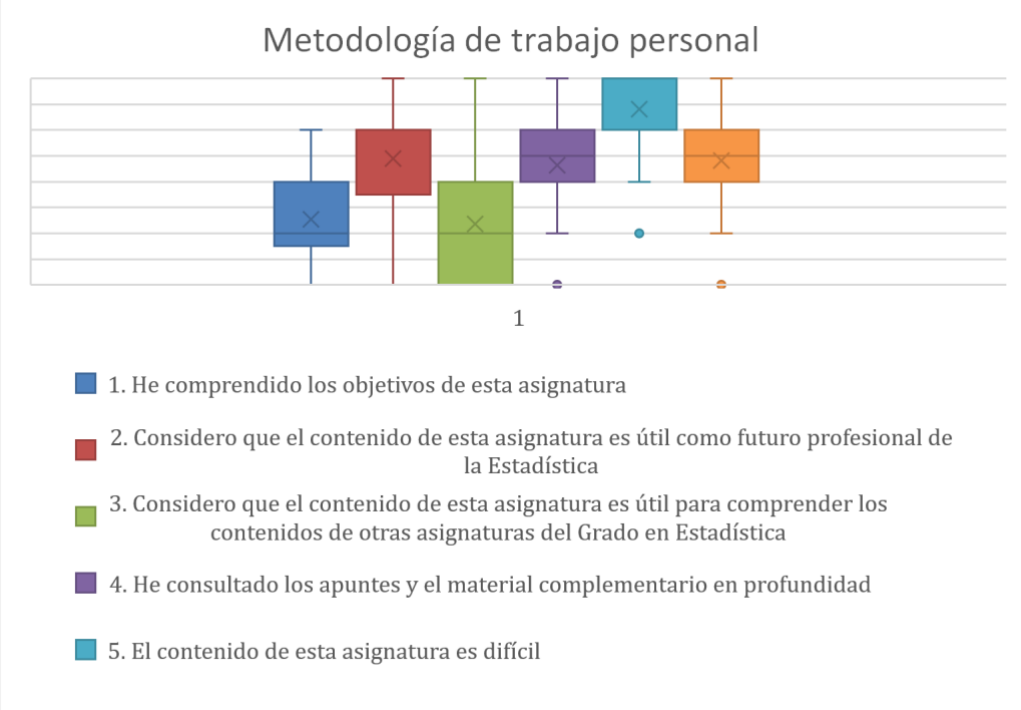

Figura 8: Visualización del primer bloque temático del cuestionario de satisfacción. La mayoría de alumnos percibieron la asignatura como difícil y valoraron la asistencia a clase de utilidad para comprender los contenidos del curso.

6.2.2 Modo de usar los notebooks

El siguiente bloque trataba de conocer cómo los usuarios emplearon los notebooks interactivos durante el curso. Los resultados se muestran en la Figura 9 de la que se extrae que:

- La mayoría de alumnos emplearon los notebooks para estudiar los contenidos del curso.

- Sin embargo, pocos los usaron para tomar apuntes en clase lo que habría supuesto una sustitución completa del formato papel. Especulando podríamos quizás alegar esto a la falta de soltura con el manejo de los notebooks o el lenguaje *Markdown,* aunque sería necesario ahondar de manera más rigurosa en esta cuestión para conocer las razones específicas de este hecho.
- Un alumno de los encuestados afirmó haber usado los notebooks para tareas de programación no relacionadas directamente con la asignatura. Este hecho es interesante ya que da indicios de que los notebooks podrían tener el efecto positivo de fomentar la programación creativa como se extrae de estudios de otros autores (García Monsálvez, 2017; Llorens Largo et al., 2017).

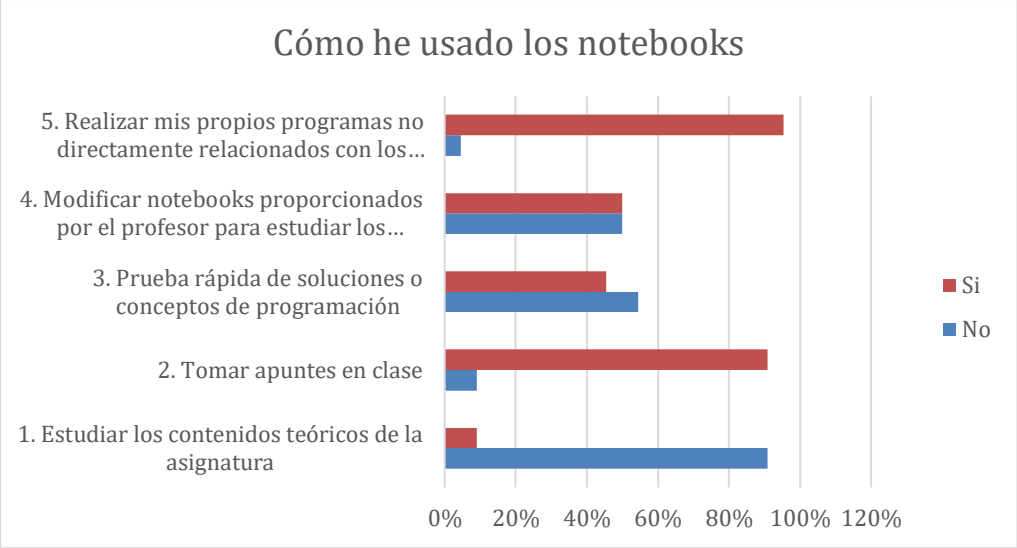

Figura 9: Distribución de resultados referente al uso de los notebooks. La mayoría de usuarios los emplearon para estudiar los contenidos teóricos del curso.

6.2.3 Percepción sobre la metodología experimental

En esta sección se evalúa la percepción del alumno sobre las innovaciones introducidas en la tecnología en referencia a la calidad de acceso al servicio, percepción general de la utilidad de la metodología y otros parámetros (Ver Figura 10). Los resultados a remarcar que se extraen de este bloque son:

- La tendencia vista en otros bloques relacionada con la dificultad de la asignatura se replica aquí también. Los alumnos presentan dudas sobre la idoneidad de las prácticas semanales y el tiempo disponible para realizarlas.
- No está claro si la metodología de aprendizaje les ha servido para comprender mejor el contenido o lograr los objetivos del aprendizaje.
- El nivel de incidencias de acceso al servicio no es elevado.
- Hay una tendencia alcista clara en las últimas cuatro preguntas. En líneas generales se puede decir que los alumnos perciben el sistema de notebooks

y el trabajo en tareas como útil. Una gran mayoría coincide en que los notebooks son una buena manera de aprender a programar, aunque si nos remitimos a lo visto en la encuesta inicial no podrían compararlo con otros métodos debido a su inexperiencia en el tema. Sería necesario investigar las causas por las cuales perciben positivamente el sistema.

- Finalmente, el trabajo por parejas fue satisfactorio para la mayoría.

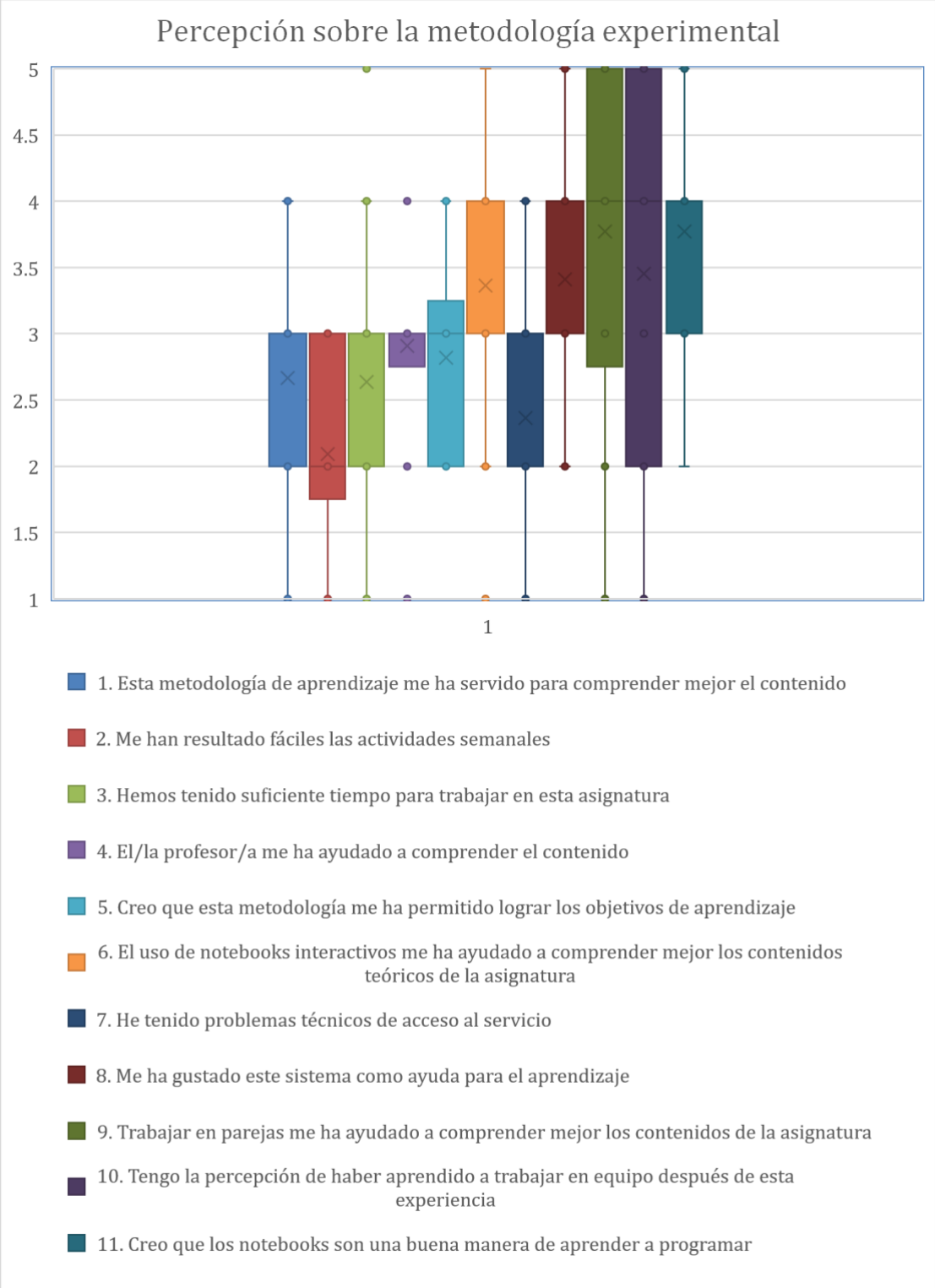

Figura 10: Resultados referentes a cómo los alumnos perciben la metodología experimental. La mayoría coincide en que la manera de trabajar propuesta había sido de su agrado.

6.2.4 Satisfacción general con la asignatura

Los resultados de este bloque ( ) muestran un nivel de satisfacción no muy alto con la asignatura, quizás debido a la alta dificultad percibida. Sin embargo parece haber un consenso en las otras dos preguntas: La mayoría de alumnos opinan que

hubiesen aprendido menos estudiando por su cuenta los contenidos y, en menor medida, creen también que el enfoque empelado podría ser exportado para la docencia de otras asignaturas del grado.

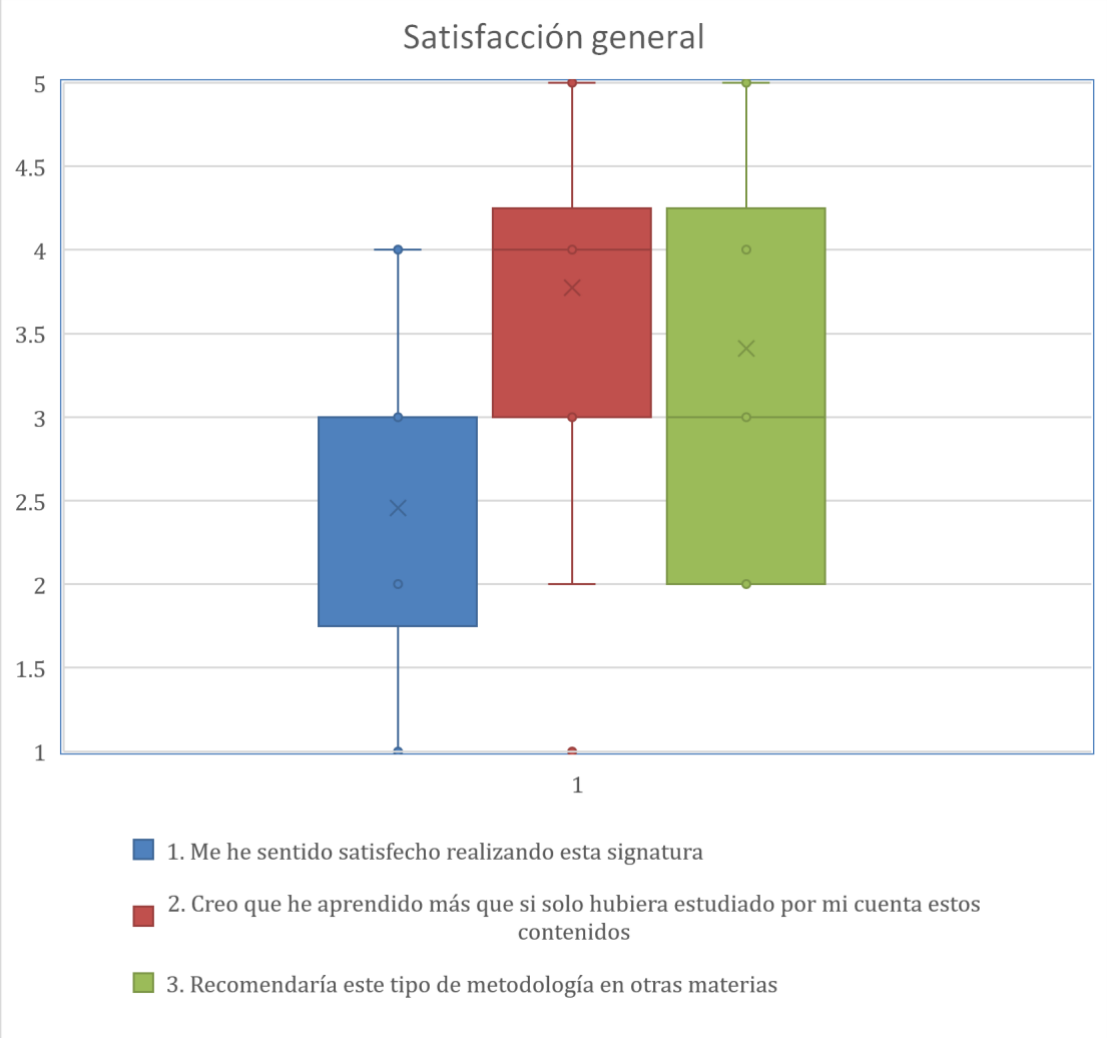

Figura 11: Satisfacción general de los alumnos con la asignatura.

6.2.5 Otros recursos y su utilidad para el estudio de la asignatura

En el siguiente bloque se preguntaba a los alumnos sobre la utilidad percibida de otros elementos del ecosistema de aprendizaje, como el campus virtual Studium o las tutorías virtuales por correo electrónico y presenciales. A grandes rasgos, todos los recursos resultaron útiles salvo las lecturas recomendadas aunque desconocemos los hechos que motivaron esta circunstancia. En siguientes revisiones de este método será necesario hacer un seguimiento más profuso del acceso a estas fuentes y su comprensión por parte de los alumnos.

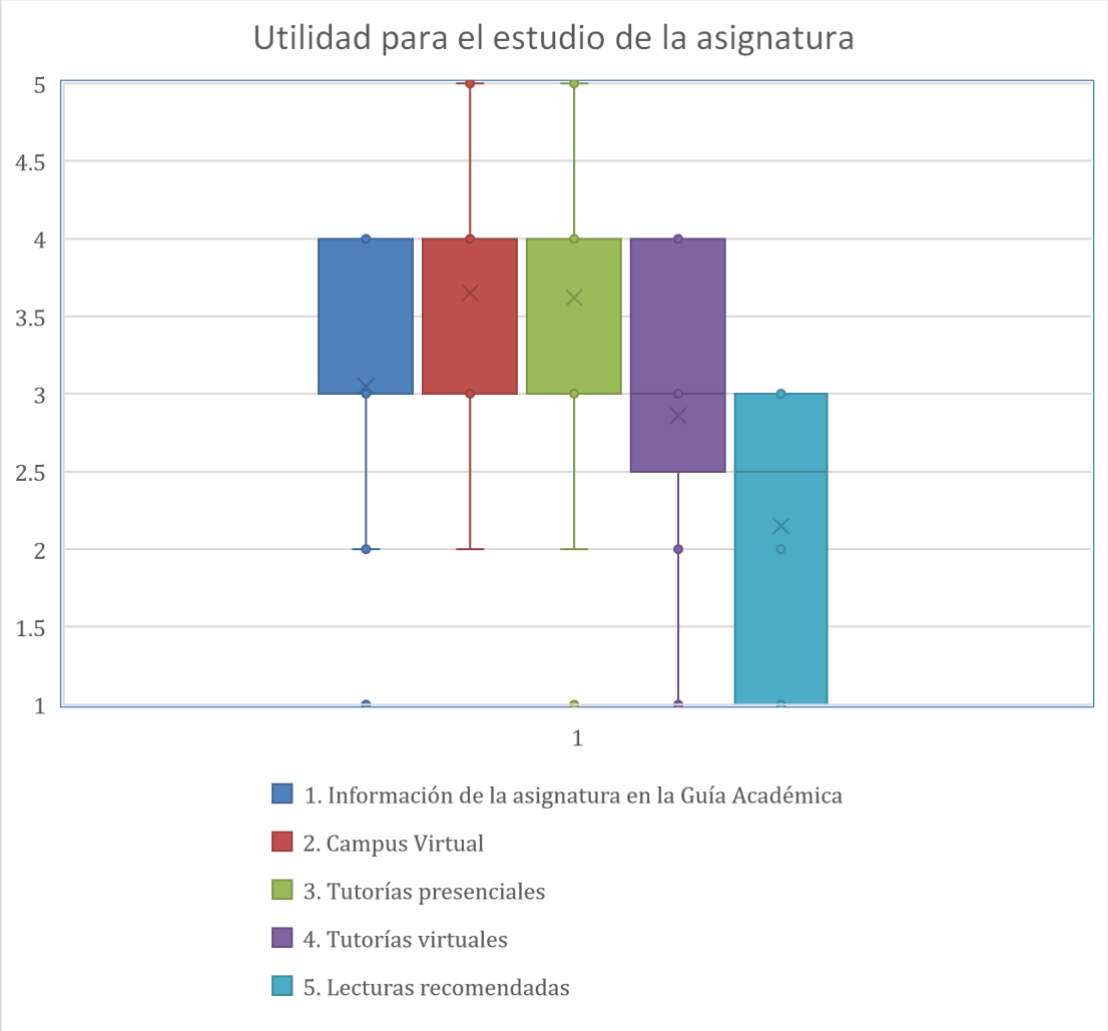

Figura 12: Utilidad de otros elementos, diferentes al notebook, para el estudio de la asignatura.

6.2.6 Valoración de los distintos métodos de aprendizaje

En este bloque temático de preguntas, los alumnos valoraron los distintos métodos de aprendizaje (Figura 13), situando en el lugar más alto de utilidad el aprendizaje mediante el trabajo en grupo, en el que se incluye el trabajo en parejas. Este marcador confirma que la técnica de Programación por Pares escogida fue adecuada.

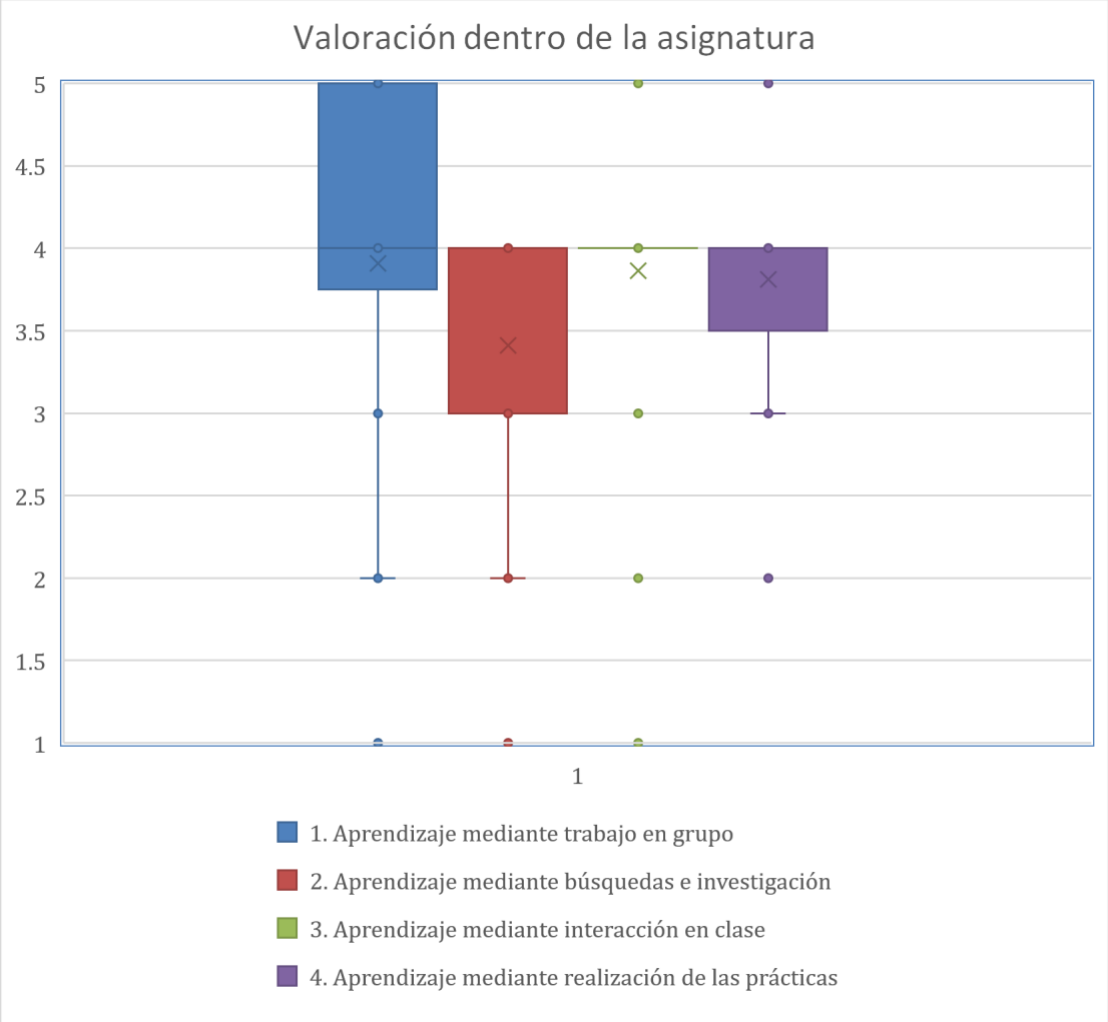

Figura 13: Valoración de los distintos métodos de aprendizaje por el alumnado. Como más valorado se reveló el trabajo en grupo.

6.2.7 Distribución de horas de trabajo

Finalmente se pidió a los alumnos cómo habían distribuido el tiempo total invertido en la asignatura en distintas tareas. Los resultados fueron que los alumnos invirtieron una media de:

- 16.18 horas en asistir a las sesiones presenciales.
- 0.86 horas en acudir a las tutorías presenciales.
- 0.21 horas en tutorías virtuales.
- 15.86 horas en realizar las tareas semanales.
- 12.77 horas en completar las prácticas obligatorias.

Como resultado de estos números, se deduce que el alumno medio de la asignatura invirtió un tiempo de 45.89 horas en las prácticas de la asignatura que, recordemos, constaba de 6 créditos ECTS repartidos entre teoría y práctica.

6.2.8 Comentarios de los alumnos

Como punto final al cuestionario, se dejó que los alumnos incorporaran respuestas de texto libre relacionadas con aspectos positivos, negativos y posibles mejores de la asignatura. Recogemos aquí algunos de ellos:

6.2.8.1 Aspectos positivos

- "Permite probar código para comprobar qué pasaría y cambiarlo de forma dinámica."
- "Me parecen útiles los notebooks ya que se puede mezclar las celdas de texto con las de progamas, de este modo se pueden tomar apuntes, modificar cosas, etc."
- "Lo positivo que veo en esta asignatura son los labs de Jupyter porque son de gran ayuda para las prácticas semanales."
- "Realizar las prácticas de lo explicado semanalmente. Tener los labs siempre a mano (bastante útiles)"
- "Al usar los notebooks las clases son mucho más amenos y se hacen menos aburridas. Me ha gustado, pero me ha parecido bastante difícil la asignatura."

6.2.8.2 Aspectos negativos

- "Hubiese sido más educativo que hubiésemos contado con más ejemplos."
- "Lo peor de la asignatura es la gran dificultad de las tareas semanales y de las prácticas obligatorias."
- "Creo que deberías ser más claro a la hora de ayudarnos con las prácticas porque lo que más nos cuesta es la lógica y aprender a enlazar todos los conceptos que das en los notebooks."
- "En mi opnión son pocas horas a la semana, una sola sesión para una asignatura tan difícil de entender si le dedicas poco tiempo."
- "Hubiera sido mucho más fácil ver la teoría en una presentación que siguiendo el notebook. Creo que hubiera sido más orientativo el tener ,muchos más ejemplos para realizar las prácticas. El horario es muy malo."

6.2.8.3 Posibles mejoras

- "Sacar a los alumnos a la pizarra para resolver las tareas semanales en todas las clases, para que no se vayan acumulando dudas y así todos comprendan mejor la asignatura."
- "Mejorar los labs o hacer las tareas más asequibles. Creoq ue el fallo ha sido nuestra falta de base en programación y las altas expectativas de Alex. Las clases poco a poco han ido siendo más útiles, pero han sido las últimas y no nos ha dado tiempo a recuperarlo. Hacer el contenido más claro ya que es mejor saber poco pero muy bien que mucho y mal ya que esta asignatura es muy importante."
- "Solucionar más ejercicios en clase, clases más interactivas. Que la progresión de dificultad sea menor y no tan brusca de un lab a otro."

- "Creo que con explicar la teoría y mandar los ejercicios no es suficiente. En mi opinión, después de explicar la teoría se debería hacer un ejemplo de ejercicio y posteriormente mandar el ejercicio para trabajar en casa."

7 Difusión de los resultados

La difusión de los resultados se condesna en el repositorio de código del proyecto ubicado en GitHub.

Se está preparando además un artículo científico que se enviará a la International Conference on Technological Ecosystems (TEEM'18) que se celebrará en Salamanca, del 24 al 26 de octubre de 2018.

8 Conclusiones

En este Proyecto de Innovación hemos demostrado cómo implementar una arquitectura software en un curso de iniciación a la programación en Python. Mediante el uso del software adecuado, hemos sido capaces de eliminar tareas redundantes que suponen un obstáculo al aprendizaje en un curso de este tipo. Se demostró como el uso de cuadernos interactivos y la elección del lenguaje de programación fueron las adecuadas para que los alumnos asimilasen un contenido que es percibido como de gran dificultad. Esperamos en el futuro poder ahondar en algunas de las cuestiones planteadas a lo largo de esta memoria, como por ejemplo la implementación de ejercicios personalizados que se adapten de manera más inteligente al progreso del alumno. En esta ocasión se contaba con un repositorio de ejercicios relativamente pequeño por haber sido esta la primera experiencia dentro del grupo empleando estas tecnologías. Sin embargo, y vistos los resultados, la inclusión de más ejemplos (quizás provenientes de otras plataformas) redundaría en una disminución de la percepción de dificultad.

9 Agradecimientos

Los autores quieren agradecer el apoyo recibido por parte del Vicerrectorado de Docencia y en especial del Instituto de Biología Funcional y Genómica de la Universidad de Salamanca para la realización de este proyecto.

10 Referencias

Brown, J., n.d. USING JUPYTERHUB IN THE CLASSROOM: SETUP AND LESSONS LEARNED.

Cruz-Benito, J., Borrás-Gené, O., García-Peñalvo, F.J., Blanco, Á.F., Therón, R., 2015. Extending MOOC ecosystems using web services and software architectures, in: Proceedings of the XVI International Conference on Human Computer Interaction. ACM, p. 52.

Edwards, S.H., 2003. Using test-driven development in the classroom: Providing students with automatic, concrete feedback on performance, in: Proceedings of the International Conference on Education and Information Systems: Technologies and Applications EISTA. Citeseer.

García Monsálvez, C., 2017. Python como primer lenguaje de programación textual en la Enseñanza Secundaria.

García-Holgado, A., García-Peñalvo, F.J., Mena Marcos, J.J., González González, C.S., 2017. Inclusión de la perspectiva de género en la asignatura de Ingeniería de Software I.

Hamrick, J.B., 2016. Creating and Grading IPython/Jupyter Notebook Assignments with NbGrader, in: Proceedings of the 47th ACM Technical Symposium on Computing Science Education, SIGCSE '16. ACM, New York, NY, USA, pp. 242–242. https://doi.org/10.1145/2839509.2850507

Holdgraf, C., Culich, A., Rokem, A., Deniz, F., Alegro, M., Ushizima, D., 2017. Portable Learning Environments for Hands-On Computational Instruction: Using Container- and Cloud-Based Technology to Teach Data Science, in: Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact, PEARC17. ACM, New York, NY, USA, pp. 32:1–32:9. https://doi.org/10.1145/3093338.3093370

Llorens Largo, F., García-Peñalvo, F.J., Molero Prieto, X., Vendrell Vidal, E., 2017. La enseñanza de la informática, la programación y el pensamiento computacional en los estudios preuniversitarios, in: Education in the Knowledge Society (EKS). Ediciones Universidad de Salamanca, pp. 7–17.

McDowell, C., Werner, L., Bullock, H., Fernald, J., 2002. The Effects of Pair-programming on Performance in an Introductory Programming Course, in: Proceedings of the 33rd SIGCSE Technical Symposium on Computer Science Education, SIGCSE '02. ACM, New York, NY, USA, pp. 38–42. https://doi.org/10.1145/563340.563353

O'Hara, K.J., Blank, D., Marshall, J., 2015. Computational notebooks for AI education.

Santamaría Vicente, R., n.d. Wikipedia como herramienta de aprendizaje en Sistemas Distribuidos (Ingenieria en Informática).

Therón, R., Santamaría, R., García Sánchez, F., 2017. Interfaces de usuario imaginadas: el camino de ida y vuelta entre la ciencia y la ficción.