UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

título del TFG
Documentación Técnica

Presentado por Mario Sanz Pérez
en Universidad de Burgos — 14 de mayo de 2024
Tutor: Álvar Arnaiz González

Índice general

Índice de figuras

Índice de tablas

Apéndice A

Plan de Proyecto Software

A.1. Introducción

Este apéndice tiene como propósito proporcionar una visión detallada del proceso de planificación y estudio de viabilidad que ha sido fundamental en el desarrollo del proyecto de software presentado. A través de una metodología estructurada, se han abordado las diferentes tareas a lo largo del tiempo, asegurando un seguimiento exhaustivo y adaptativo de cada fase del proyecto. Además, este documento explora en profundidad el estudio de viabilidad realizado, abarcando aspectos tanto legales como económicos. El análisis legal es crucial para asegurar que el proyecto cumpla con todas las normativas y legislaciones aplicables, minimizando así posibles riesgos legales. Por otro lado, el estudio económico proporciona una evaluación detallada de la viabilidad financiera del proyecto, incluyendo estimaciones de costos, análisis de retorno de inversión y proyecciones de rentabilidad a largo plazo.

A.2. Planificación temporal

Como se explica en la sección 4 de la memoria, la metodología a seguir es Scrum, salvando las distancias con el número de personas que suele haber en un contexto habitual, ya que únicamente habrá un desarrollador. El proyecto se elige antes de empezar el curso, únicamente para poder informarse y leer articulos relacionados con el tema del aprendizaje semi supervisado. Al inicio con sprint más largos y a medida que se avanza en el desarrollo los sprints se reducen a una o dos semanas de duración, realizando una reunión al inicio de cada uno.

Sprint 1

Este sprint corresponde a las fechas entre el 7 de noviembre y el 18 de diciembre. Se comienza con una reunión en la que se establecen las siguientes tareas:

- Crear un repositorio en GitHub [4] donde poder subir los cambios del proyecto, más concretamente, la plantilla de documentación inicial para ir familiarizándose con LaTeX.
- También se manda terminar de leer el artículo [5] y se asigna una nueva lectura acerca de ensembles [3]. Esto es necesario ya que de entre los algoritmos a implementar, alguno de ellos será un ensemble. Concepto que se desconocía antes de iniciar la lectura.
- Encontrar un programa adecuado para el seguimiento del proyecto que soporte Scrum.

El repositorio se crea siguiendo la plantilla de documentación ya creada en 2016 y publicada en Github. Para poder empezar a familiarizarse con LaTeX es necesario instalar los programas necesarios en el equipo local. Se prueban TeXstudio y TeXworks como editores, y por gusto y comodidad se elige TeXstudio como editor de archivos. Tambien se instala MiKTex, que es una distribución de TeX/LaTeX para sistemas operativos Windows. Se continua con la lectura establecida, adquiriendo conceptos de ensembles, estos son, ¿Qué es el boosting?, ¿Qué es el bagging?, ¿Qué posibilidades hay de combinar varios modelos? entre otros. En cuanto al programa utilizado para el seguimiento de las tareas y sprints, se prueban varios como Zenhub (descartado por ser de pago), Jira y Taiga. El primero era la mejor opción dado su relacion con Github, pero al ser de pago se descarta. Entre Jira y Taiga se elige la segunda por su accesibilidad, los numerosos problemas de Jira hacen que la plataforma de Taiga sea la elegida. Esta herramienta se explica en el apartado cuatro de la memoria, pero su sencillez y el hecho de poder comunicarse con GitHub hacen facil su uso. Aun asi, queda abierto a cambiar debido a que no es la herramienta que ofrece mas posibilidades.

Sprint 2

Sprint correspondiente a las fechas entre el 18 de noviembre y el 15 de enero. Se inicia con una reunión previa para establecer las tareas:

- Finalizar la lectura de [3].
- Comenzar la documentación con conceptos teóricos acerca del aprendizaje automático vistos hasta el momento.
- Aprender a usar el entorno de flask en python.

La lectura y documentacion se realiza durante el periodo de vacaciones, mientras que el aprendizaje de flask, se lleva a cabo con la ayuda de la asignatura cursada de Diseño y Mantenimiento del Software, en la que se realiza una web que se implementa con este *framework*.

Se encuentra una aplicación llamada Zappier, la cual permite construir triggers entre aplicaciones. En este caso sirve para que cada vez que se cree una *issue* en github, se cree como historia de usuario en el product backlog de Taiga, donde despues se gestionará independientemente. Esta aplicación está de nuevo explicada en el apartado 4 de la memoria.

Sprint 3

Sprint correspondiente a las fechas entre el 15 de enero y el 1 de febrero. Se tiene una reunión previa para asignar las tareas de:

- Lectura del algoritmo Co-Forest [2] y su pseudocódigo.
- Búsqueda de trabajos relacionados para coger ideas propias para el proyecto.
- Continuar documentando los conceptos teóricos (ensembles, Co-Forest).

En este periodo se completa la lectura del artículo del algoritmo Co-Forest y del apartado del trabajo de Patricia y sus estudios relacionados con este algoritmo. El principal estudio que realiza consiste en resolver un error del pseudocódigo, donde un valor podía coger el valor 0 cuando se utiliza como divisor. Mediante tres propuestas, se inicializa este valor con diferentes operaciones y se muestran varias gráficas para poder evaluar la mejor opción.

Se realiza una búsqueda amplia de aplicaciones web para visualización de algoritmos, apuntando y explicando las más interesantes en el apartado 6 de la memoria. Se realiza la implementación de la técnica de Scrum en el apartado 4 de la memoria. Se deja para sprint posteriores la documentación del CoForest, ya que puede que los conceptos adquiridos no sean los correctos hasta que no se haga su implementación y se vean los resultados.

Sprint 4

Sprint correspondiente a las fechas entre el 1 de febrero y el 14 de febrero. Las tareas asignadas para este sprint son:

- Primera implementacion del algoritmo Co-forest, basado en [2].
- Evaluar esta primera implementación.
- Actualizar documentación, incluyendo este apartado.
- Continuar con la búsqueda de trabajos relacionados en la web.

Para la primera tarea, es importante comentar que se utiliza como referencia el pseudocódigo del articulo mencionado pero también la implementación de Patricia Hernando, [1]. Una vez se tiene la primera implementación del algoritmo, se compara con el algoritmo de Patricia, el cual se considera una solución muy buena como ensemble. El estudio realizado se explica mejor en el apartado de aspectos relevantes de la memoria, pero cabe mencionar que los primeros resultados no son muy fiables, lo que hace pensar que algo está mal implementado. Además de la búsqueda de páginas anteriores, se encuentra una muy buena opción: https://ml-visualizer.herokuapp.com/. Esta página tiene algo diferente, ya que permite configurar y ver el resultado del algoritmo en la misma ventana. Este será uno de los objetivos en la web.

Sprint 5

Sprint que corresponde a las dos semanas siguientes, se asignan las tareas:

- Corregir fallos en la implementación del Co-forest.
- Comenzar a mirar el código correspondiente a la web del trabajo de David, el cual podría resultar interesante para el futuro.
- Evaluar correctamente el Co-forest.
- Continuar con la documentación.

Se muestra al tutor la página encontrada, se acuerda que puede ser una buena idea para la web, pero antes hay que familiarizarse con el entorno de la web y su código. Esto incluye decidir si las llamadas a la web, una vez ejecutas el algoritmo, se realizan de una en una, devolviendo un gráfico en cada iteración, o una sola llamada que calcule todo el algoritmo y que reciba

un diccionario con toda la información necesaria para su representación. Se resuelven dudas del algoritmo Coforest, como la necesidad del uso de *random state* para tener un estudio reproducible. El estudio sigue teniendo bastantes diferencias en comparación con el de Patricia, lo que hace pensar de nuevo que algo en el código no esta bien programado.

Sprint 6

Se inicia con una reunión el 7 de marzo, surge un cambio de planes, dejando las siguientes tareas por hacer:

- Continuar documentación (Co-Forest, trabajos relacionados)
- El trabajo será una versión 2.0 del desarrollo de David.
- Esto implica tener que familiarizarse con el trabajo de David, su estructura, web, etc.
- Corregir estilo de programación. Estandarizar mediante la guía de estilo de python, PEP8 [6].
- Revisar Co-Forest debido a que la comparación con el de Patricia no es del todo buena.

La decisión de continuar el trabajo de David se da ya que la idea original para esta implementación iba a ser prácticamente igual. Por ello, se aprovecha la base de este trabajo, sobretodo el de la web, ya que los algoritmos serán diferentes. La idea es seguir con un estilo propio en la configuración de la web, pero todo sobre la plantilla ya creada por David. En cuanto al código implementado hasta el momento, se considera que está bastante desordenado, sin documentar y sin seguir una guía de estilos. Por ello, se establece el idioma español para nombrar a todas las variables, se documentan todos los métodos de la manera correspondiente en python, y con la auyda de librerias como *pylint* y *mypy* se sigue la guía de estilo PEP8. El error que se estaba cometiendo en cuanto a la implementación del Co-forest estaba en el calculo del error, ya que uno de los parámetros (los índices de los datos de entrenamiento que se usan en cada árbol) se estaba repitiendo en cada ejecución, cuando cada árbol debería tener los suyos. Es decir, el error se estaba calculando de manera incorrecta. Una vez corregido esto, el algoritmo se considera eficiente.

Sprint 7

Este sprint es aún más reducido debido a las condiciones dadas, correspondiente entre los días 14 y 20 de marzo. Se utiliza para avanzar en todas aquellas tareas retrasadas, asignando las tareas:

- Documentación de conceptos teóricos: tanto teoría de ensembles como el propio algoritmo Co-forest.
- Documentación de trabajos relacionados.
- Documentación de aspectos relevantes.
- Documentación de anexos.
- Prototipo de ventana pensada para la configuración del algoritmo.

Sin mucha más explicación, se avanza todo lo que se puede en la documentación y, una vez visto y entendido la mayor parte del proyecto web de David, se empieza el prototipo de una nueva ventana.

Sprint 8

Debido a que el calendario corresponde con la semana santa, este Sprint corresponde entre los días 20 de marzo y 4 de abril. Se asignan las siguientes tareas:

- Continuación de toda la documentación del Sprint anterior.
- Ver tutoriales de JavaScript y de BootStrap
- Introducir correctamente el Co-Forest en la web
- Investigar la manera en la que se pasan los parametros de ejecución del algoritmo (JSON).

En cuanto a los tutoriales, se siguen los de la web https://www.w3schools.com/ y también algún video de YouTube. HTML y css son dos lenguajes que se pueden ir aprendiendo sobre la marcha, pero javascript puede ser complicado de entender sin unos conceptos previos, y más cuando se trata de un proyecto complejo. Por esto se dedica gran parte del tiempo a aprender las bases del lenguaje. En cuanto a la web, se integra gran parte del Co-Forest, permitiendo realizar todos los pasos hasta la visualización, donde surge un

error de servidor. Para conseguir la comunicación entre la ejecución del algoritmo y su visualización se realizará siguiendo la misma técnica que David, para ello primero hay que comprender todo lo que recoge del propio algoritmo. Se decide fijarse en el algoritmo Democratic Co-Learning por su gran parecido con el Co-Forest.

Además de todo esto, aparece un error en la parte de la gestión de usuarios, no permitiendo registrar ni logear usuarios. Esto se debe a algún problema con el método que se utiliza para la encriptación de las contraseñas, el cual se deja a resolver para el siguiente sprint.

Sprint 9

De nuevo se vuelven con las reuniones semanales, estableciendo las siguientes tareas:

- Introducir la parte de visualización del co-forest en la web.
- Modificar el código del Co-Forest para tener el JSON.

Ambas tareas están relacionadas, ya que para poder ver una visualizacion final, es necesario almacenar todos los datos de la ejecución. La tarea de conseguir la visualizacion consiste más bien en corregir el error mencionado anteriormente. Ya que un error 400 no da muchas pistas de donde puede estar el error, se hace un seguimiento de todo el proceso. Esto conlleva ir mostrando por consola los diferentes valores y resultados. Finalmente se localiza el error en la forma en la que se denominan los *div* en los formularios. Para el caso del Co-Forest, se usa lo que ya existía de los árboles de decisión. Esto hace que no haya necesidad de crear un *div* para seleccionador el clasificador. Sin embargo, para aprovechar el código de David, es necesario definir uno y darle el mismo nombre en los diferentes formularios para que funcione. El JSON se consigue basandose de nuevo en el Democratic Co-Learning. Puede que se guarden los mismos datos, pero la manera de recogerlos es distinta en cada algoritmo.

Se resuelve el error del sprint anterior de la parte de gestión de usuarios. Una vez aislado el error se ve que lo que falla es un método de una librería utilizada para encriptar contraseñas, lo que da a pensar que alguno de los parámetros pasados deja de ser compatible con la versión de la biblioteca actual. Efectivamente la versión de este proyecto de la biblioteca *werkzeug* no coincidía con la del trabajo base. En concreto el parámetro incompatible que se le estaba pasando a este método es la propia contraseña encriptada,

la cual empezaba por *sha256*. La nueva versión del método establece que para usar *sha256* el texto debe empezar por *pbkdf2:sha256*, por lo tanto se cambia esta opción y también el *hash* definido para el administrador al inciar la aplicación.

Sprint 10

Del 11 de abril al 18 de abril, se establecen las siguientes tareas de cara al siguiente sprint:

- Actualizar documentación.
- Visualizar resultados del Co-Forest en la web
- Crear un gráfico de tarta como tooltip en los datos.

Se consigue la visualización del Co-Forest completa reutilizando gran parte del código de los otros algoritmos, dando pie también a posibles cambios futuros en la visualización. Finalmente se determina que el gráfico de tarta no es la mejor manera de representar este tipo de soluciones y se mantiene el estilo original. Esto tiene la desventaja de que en el Co-Forest existen muchos más clasificadores y en ocasiones provoca un *tooltip* mucho más grande y que se sale de la pantalla.

Además se hacen pequeños cambios como dejar la opción de la etiqueta o clase por defecto al configurar el algoritmo y permitir desmarcar y marcar todas las opciones (clasificadores) en el gráfico de estadísticas específicas.

Sprint 11

Sprint correspondiente a los días entre el 18 y el 25 de abril, en la reunión se establecen las siguientes tareas:

- Finalizar tareas pendiente del Co-Forest. Estas son:
  - Arreglar última iteración ya que no aporta información
  - Quitar parámetro de inicialización de pesos ya que realmente no afecta demasiado.
  - Cambiar la visualización del *tooltip* para que se pueda ver toda la información.
- Leer documentación de construcción de grafos.

- Elegir biblioteca de *python* para la construcción de grafos.

Este Sprint se utiliza sobretodo para leer e investigar acerca de los métodos transductivos o basados en grafos. Más concretamente se manda leer el artículo del algoritmo *GBILI*, que será uno de ellos a implementar en cuánto a la construcción del grafo y el artículo de *Local and Global Consistency (LGC)* como algoritmo de inferencia de etiquetas. En la memoria se especifican las características de ambos algoritmos.

Se realiza un mini estudio para la elección de una biblioteca en el uso de grafos, que se puede encontrar en los aspectos relevantes. Finalmente se decide utilizar *NetworkX* por su facilidad de uso y aprendizaje.

En cuánto a los arreglos del Co-Forest, se realizan las dos primeras tareas marcadas y se deja para más adelante cambiar la visualización del *tooltip*.

Sprint 12

Sprint correspondiente a la siguiente semana entre el 25 de abril y el 2 de mayo. Se mandan las siguientes tareas:

- Implementar algoritmos: tanto el *GBILI* como el *LGC*.
- Documentar los conceptos de cada algoritmo y los aspectos relevantes en su implementación.

En este sprint no hay mucho detalle que comentar ya que se trata de implementar el algoritmo. Finalmente no se utiliza la librería *NetworkX* para almacenar los datos del grafo, si no que se utiliza para su visualización. La visualización en este caso ayuda a ver si realmente el grafo se está construyendo siguiendo los pasos establecidos y si tienen sentido o no. Por ejemplo, si se colorean todos los nodos que son etiquetados, se podra saber si la conexión basada en estos nodos es correcta o no.

Finalmente no se consigue implementar ambos algoritmos de forma definitiva pero si se llega a una buena aproximación. El aspecto a destacar es que de la forma en que se implementa el algoritmo LGC realmente no aprovecha la disposición física final del grafo, sino su matriz de distancias original. Se va con esta premisa a la reunión para tener en cuenta un posible cambio.

Sprint 13

Corresponde a los días entre el 2 y el 9 de mayo. Se comentan las siguientes tareas a realizar:

- Depurar el código del algoritmo *GBILI*.
- Depurar el código del algoritmo *LGC*.

Se trata de finalizar ambos algoritmos para poder empezar su integración con la web.

El resumen de las correciones en el *GBILI* (todas ellas comentadas en la sección 5 de la memoria) es: el grafo es no dirigido por lo que se deben almacenar enlances en ambas direcciones, el pseudocódigo se puede interpretar erróneamente y provocar fallos al almacenar las estructuras de datos necesarias y por último, la visualización debe ser con el grafo ordenado para que tenga sentido las nuevas conexiones.

En cuánto a la inferencia se manda tener en cuenta la conexión entre nodos del grafo mediante una matriz de afinidad binaria (1 si hay conexión, 0 en caso contrario).

Se consigue tener el algoritmo *GBILI* bien implementado, pero al inferir etiquetas los resultados no son los esperados con la nueva matriz de afinidad.

Se empieza a integrar en la web una nueva opción para seleccionar la visualización de grafos. La idea es tener una única opción y combinar la construcción del grafo junto con la fase de inferencia de etiquetas. A estas alturas se planea hacer dos algoritmos de construcción de grafos y dos de inferencia (ya teniendo uno de cada).

Sprint 14

Correspondiente a los días entre el 9 y el 16 de mayo. Se comentan las siguientes tareas:

- Corregir definitivamente ambos algoritmos (*LGC* sobretodo).
- Integrar en la web, con la visualización por fases.

Este Sprint se utiliza para corregir el *LGC* para que tome correctamente la matriz de afinidad que debe y también para ponerse al día con la documentación de estos métodos.

A.3. Estudio de viabilidad

Viabilidad económica

Viabilidad legal

Apéndice B

Especificación de Requisitos

B.1. Introducción

Una muestra de cómo podría ser una tabla de casos de uso:

B.2. Objetivos generales

B.3. Catálogo de requisitos

B.4. Especificación de requisitos

CU-1	**Ejemplo de caso de uso**
Versión	1.0
Autor	Alumno
Requisitos asociados	RF-xx, RF-xx
Descripción	La descripción del CU
Precondición	Precondiciones (podría haber más de una)
Acciones	1. Pasos del CU 2. Pasos del CU (añadir tantos como sean necesarios)
Postcondición	Postcondiciones (podría haber más de una)
Excepciones	Excepciones
Importancia	Alta o Media o Baja...

Tabla B.1: CU-1 Nombre del caso de uso.

Apéndice C

Especificación de diseño

C.1. Introducción

C.2. Diseño de datos

C.3. Diseño procedimental

C.4. Diseño arquitectónico

Apéndice D

Documentación técnica de programación

Apéndice E

Documentación de usuario

E.1. Introducción

E.2. Requisitos de usuarios

E.3. Instalación

E.4. Manual del usuario

Apéndice F

Anexo de sostenibilización curricular

F.1. Introducción

Este anexo incluirá una reflexión personal del alumnado sobre los aspectos de la sostenibilidad que se abordan en el trabajo. Se pueden incluir tantas subsecciones como sean necesarias con la intención de explicar las competencias de sostenibilidad adquiridas durante el alumnado y aplicadas al Trabajo de Fin de Grado.

Más información en el documento de la CRUE https://www.crue.org/wp-content/uploads/2020/02/Directrices_Sosteniblidad_Crue2012.pdf.

Este anexo tendrá una extensión comprendida entre 600 y 800 palabras.

Bibliografía

[1] Patricia Hernando Fernández. Aprendizaje semisupervisado y ciberseguridad: detección automática de ataques en sistemas de recomendación y phising. *Universidad de Burgos*, 2023.

[2] Ming Li and Zhi-Hua Zhou. Improve computer-aided diagnosis with machine learning techniques using undiagnosed samples. *IEEE transactions on systems, man, and cybernetics - Part A: Systems and Humans*, 37(6):1088–1098, 2007.

[3] R. Polikar. Ensemble based systems in decision making. *IEEE Circuits and systems magazine*, 6(3):21–45, 2006.

[4] Mario Sanz Pérez. Tfg-semi-supervised-learning. https://github.com/msp1015/TFG-Semi-Supervised-Learning, 2023. Github.

[5] Jesper E. van Engelen and Holger H. Hoos. A survey on semi supervised learning. *Machine Learning*, 109:373–400, 2020.

[6] Guido van Rossum, Barry Warsaw, and Alyssa Coghlan. Pep 8 – style guide for python code. https://peps.python.org/pep-0008/, 2013. [Internet; descargado 10-abril-2024].