

Universidades de Burgos, León y
Valladolid

Máster universitario

Inteligencia de Negocio y Big Data en Entornos Seguros



TFM del Máster Inteligencia de Negocio
y Big Data en Entornos Seguros

Uso de técnicas de aprendizaje no
supervisado para la ayuda en
videojuegos tipo MOBA

Presentado por Iván Iglesias Cuesta
en Universidad de Burgos — 4 de agosto
de 2021
Tutores: Jose Francisco Díez Pastor y
~~César-Ignacio García Osorio~~

Universidades de Burgos, León y Valladolid



Máster universitario en Inteligencia de Negocio y Big Data en Entornos Seguros

D. Jose Francisco Díez Pastor, profesor del departamento de Ingeniería Informática, área de Lenguajes y Sistemas Informáticos.

D. César-Ignacio García Osorio, profesor del departamento de Ingeniería Informática, área de Lenguajes y Sistemas Informáticos.

Exponen:

Que el alumno D. Iván Iglesias Cuesta, con DNI 45573756S, ha realizado el Trabajo final de Máster en Inteligencia de Negocio y Big Data en Entornos Seguros titulado Uso de técnicas de aprendizaje no supervisado para la ayuda en videojuegos tipo MOBA.

Y que dicho trabajo ha sido realizado por el alumno bajo la dirección del que suscribe, en virtud de lo cual se autoriza su presentación y defensa.

En Burgos, 4 de agosto de 2021

Vº. Bº. del Tutor:

Vº. Bº. del tutor:

D. Jose Francisco Díez Pastor

D. César-Ignacio García Osorio

Resumen

En este primer apartado se hace una **breve** presentación del tema que se aborda en el proyecto.

Descriptores

MOBA, League of Legends, Riot Games, aprendizaje no supervisado, conjuntos frecuentes, ETL, Django, MongoDB.

Abstract

A **brief** presentation of the topic addressed in the project.

Keywords

MOBA, League of Legends, Riot Games, unsupervised items, frequent itemsets, ETL, Django, MongoDB.

Índice general

Índice general	iii
Índice de figuras	iv
Índice de tablas	v
 Memoria	 1
1. Introducción	3
2. Objetivos del proyecto	5
2.1. Objetivos principales	5
2.2. Objetivos personales	5
3. Conceptos teóricos	7
3.1. ETL	7
3.2. Aprendizaje no supervisado	7
3.3. Conceptos sobre <i>League of Legends</i>	7
4. Técnicas y herramientas	13
4.1. Técnicas	13
4.2. Herramientas	13
5. Aspectos relevantes del desarrollo del proyecto	15
5.1. Control de velocidad de peticiones por ratios de la API . . .	16
5.2. Tolerancia a fallos	16
5.3. Elección del algoritmo	16

5.4. Uso de MongoDB	16
6. Trabajos relacionados	17
7. Conclusiones y Líneas de trabajo futuras	19
 Apéndices	 20
Apéndice A Plan de Proyecto Software	23
A.1. Introducción	23
A.2. Planificación temporal	23
A.3. Estudio de viabilidad	27
 Apéndice B Especificación de Requisitos	 29
B.1. Introducción	29
B.2. Objetivos generales	29
B.3. Catalogo de requisitos	30
B.4. Especificación de requisitos	31
 Apéndice C Especificación de diseño	 33
C.1. Introducción	33
C.2. Diseño de datos	33
C.3. Diseño procedimental	33
C.4. Diseño arquitectónico	33
 Apéndice D Documentación técnica de programación	 35
D.1. Introducción	35
D.2. Estructura de directorios	35
D.3. Manual del programador	37
D.4. Compilación, instalación y ejecución del proyecto	37
 Apéndice E Documentación de usuario	 39
E.1. Introducción	39
E.2. Requisitos de usuarios	39
E.3. Instalación	39
E.4. Manual del usuario	39

Índice de figuras

3.1. Mapa de <i>League of Legends</i>	8
---	---

Índice de tablas

Memoria

Introducción

Las competiciones de deportes electrónicos, o *esports*, al igual que los deportes tradicionales, mueven grandes cantidades de dinero a la vez que atraen a un número muy elevado de espectadores a sus retransmisiones.

Por lo general los juegos de los que se realizan competiciones son gratuitos, por lo que cualquier persona puede adentrarse en ese mundillo para pasar un rato entretenido, o ponerse la meta de llegar a ser profesional.

Sin embargo, esto es un objetivo complicado por la gran cantidad de horas necesarias para conseguir las capacidades necesarias para ser profesional. Además, el rango de edad en el que más necesario dedicar más horas de *práctica* coincide con etapas de escolarización todavía obligatorias, pudiendo crear conflicto de intereses.

En ambos ámbitos, profesional y casual, se genera constantemente una gran cantidad de datos, tomando la forma de un registro de partidas jugadas. Para este trabajo me he propuesto analizar ese histórico de partidas en uno de los *esports* más predominantes del momento, *League of Legends*, un videojuego dentro del tipo MOBA (*Multiplayer Online Battle Arena*).

Mediante el uso de aprendizaje no supervisado, se puede extraer conocimiento del juego y ponerlo a disposición de las personas que empiezan a jugar y hacer más fácil esta entrada. Además de servir de ayuda en el ámbito profesional, para facilitar la preparación de un equipo ante una partida de competición.

Ideas

- Importancia del mercado de los deportes electrónicos.

- Cifras de beneficios
- Cifras de espectadores
- <https://www.esportmaniacos.com/business/excel-esports-inversion-20-mill>
- <https://www.esportmaniacos.com/comunidad/jugadores-futbol-esports-clube>
- <https://newzoo.com/insights/trend-reports/newzoos-global-esports-live-s>
- <https://dotesports.com/league-of-legends/news/league-of-legends-reporter>
- <https://techacake.com/league-of-legends-player-count/>
- <https://www.bdsesport.com/en/actualites/hot-news/team-bds-acquires-fc-s>
- https://www.openbank.es/superliga-esportsg_80TUK1qc3YybD6qYP-bygTCX2aS19UoGb5t0SSf-Lf1C8

Objetivos del proyecto



Este apartado explica de forma precisa y concisa cuales son los objetivos que se persiguen con la realización del proyecto. Se puede distinguir entre los objetivos marcados por los requisitos del software a construir y los objetivos de carácter técnico que plantea a la hora de llevar a la práctica el proyecto. Adicionalmente también se incluyen objetivos de carácter personal para el alumno.

2.1. Objetivos principales

- Desarrollar un proceso ETL que sea capaz de recopilar los datos necesarios usando la API oficial de Riot Games¹.
- Aplicar técnicas de aprendizaje no supervisado sobre los datos recopilados, en este caso algoritmos para la obtención de conjuntos frecuentes de objetos.
- Ser capaz de obtener conocimiento útil a partir de los datos obtenidos.
- Desarrollar una aplicación en la que se pueda consultar el conocimiento extraído.
- Que el producto final sea capaz de ayudar a los nuevos jugadores.

2.2. Objetivos personales

- Aplicar lo aprendido durante el máster en un campo novedoso.

¹Riot Games es la desarrolladora de League of Legends

- Dar a conocer el mundo de los deportes electrónicos en un ambiente donde sean menos conocidos.
- Desarrollar un proyecto de ideación propia.

Conceptos teóricos

3.1. ETL

El término ETL (~~Extract, Transform, Load~~) se refiere al proceso de obtención de datos desde una o varias fuentes, su transformación y carga final en un lugar centralizado para su posterior uso.

3.2. Aprendizaje no supervisado

Dentro de la disciplina de aprendizaje automático, nos referimos al aprendizaje no supervisado como el conjunto de algoritmos que son capaces de generar conocimiento a partir de datos de los que no se conocen relaciones entre ellos. En el contrario se encuentra el aprendizaje supervisado, donde tenemos un conocimiento previo por la relación entre los datos de entrada y su resultado, y el objetivo es la obtención de un modelo que generalice ese conocimiento para realizar predicciones.

3.3. Conceptos sobre *League of Legends*

3.3.1. El juego

League of Legends es un videojuego de estrategia multijugador del tipo MOBA (*Multiplayer Online Battle Arena*), desarrollado por Riot Games y lanzado en 2011, en el que dos equipos de cinco jugadores se enfrentan para destruir la base del equipo enemigo [?]. Cada jugador controla dentro del juego a un personaje llamado campeón, que pueden seleccionar antes de empezar a jugar, y es único entre los diez jugadores.

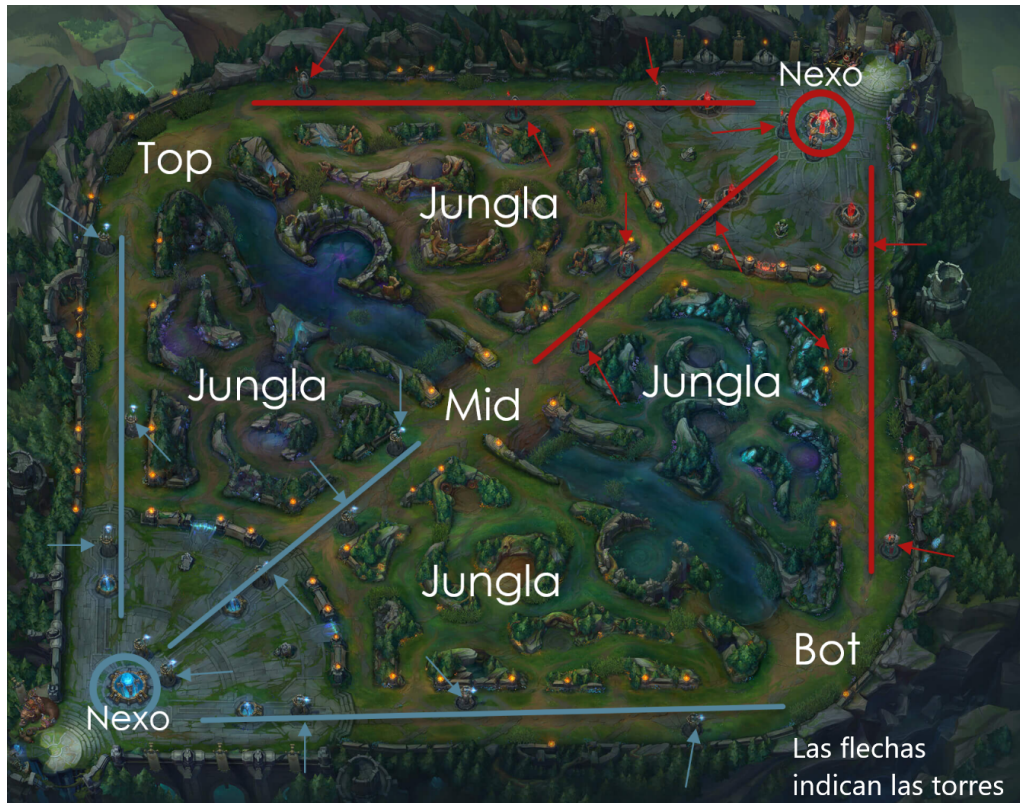


Figura 3.1: Mapa de *League of Legends*.

Los jugadores se enfrentan en un mapa 3.1 en forma de cuadrado, con las bases de cada equipo localizadas en zonas opuestas del mapa, una en la esquina inferior izquierda, y la otra en la superior derecha. Conectando cada base se encuentran tres líneas o calles, superior o *top*, central o *mid* e inferior o *bot*. El espacio entre las calles se denomina *jungla*. Conectando las dos esquinas que no pertenecen a las bases se encuentra el río, que se encarga de separar el terreno del mapa dominado por cada equipo. De forma general los jugadores se reparten de la siguiente manera, uno en *top*, uno en *mid*, dos en *bot* y el restante en la *jungla*.

Para ganar la partida hay que destruir el *nexo* del equipo enemigo, es la estructura que está más alejada de la base propia. Las *torres* son otro tipo de estructuras situadas por el mapa, que disparan a los campeones del equipo contrario. Existen tres torres por cada línea y equipo, además de dos adicionales que protegen cada *nexo*. Las torres se tienen que derribar en el orden que se van encontrando en cada línea, y para destruir el *nexo*, como mínimo, hay que haber derribado todas las torres de una calle.

La forma en la que un equipo consigue ventaja sobre el rival es mediante el oro, este se consigue de varias formas. La principal forma es asesinando a los campeones enemigos. Otra forma de conseguir oro es destruyendo las estructuras enemigas. La última forma de conseguir oro es matando monstruos y súbditos, los primeros se encuentran en la jungla, los segundos recorren las calles en oleadas. Tanto los campeones como los monstruos y súbditos vuelven a aparecer pasado un tiempo concreto, las estructuras una vez destruidas se mantienen así.

El oro permite comprar objetos, que mejoran las habilidades del campeón, haciendo que sea más sencillo derrotar a los campeones enemigos, lo que proporciona más oro para más objetos, causando un efecto bola de nieve y la eventual victoria del equipo.

Para ver estos conceptos de una forma más visual, Riot Games preparó un vídeo informativo de cara a los mundiales de 2020 para que la gente que no tuviera mucho conocimiento del juego y su funcionamiento pudiera ver y disfrutar la competición. El vídeo se titula *¿Así que queréis ver el Mundial? | Mundial 2020 - League of Legends*² y se encuentra disponible en YouTube.

3.3.2. Campeones

Los campeones son los diferentes personajes disponibles que un jugador puede seleccionar antes de la partida. Actualmente hay 156 disponibles, añadiéndose a la lista uno nuevo en franjas de tiempo que van desde uno a seis meses. Cada uno tiene un rol asignado que determina su forma de jugar, su función e incluso posición dentro del mapa.

- Tirador
- Apoyo
- Asesino
- Luchador
- Tanque
- Mago

Para que un campeón acabe en una categoría u otra hay que prestar atención a varios factores, entre los que se encuentran su tipo de ataque básico, el efecto de sus habilidades y la forma en la que sus estadísticas modifican las habilidades. En las secciones 3.3.3 y 3.3.4 se explican en detalles estos conceptos.

²https://youtu.be/ERkt_1TY1kU

3.3.3. Habilidades

Cada campeón tiene un conjunto único de habilidades, una **pasiva** y cuatro **activas**. Se pueden definir como las operaciones que un jugador puede realizar para interactuar con el mapa, con otros jugadores, monstruos y súbditos.

Cada habilidad puede tener un efecto o combinar varios, los cuales se aplican sobre uno mismo, un **aliado** o **enemigo**. Los efectos más comunes son las **curaciones**, realizar daños o control de adversario, donde se engloban **ralentizaciones** o **inmovilizaciones**. Estos efectos tienen unos valores numéricos que constan de dos partes, un valor base y uno variable que depende de las estadísticas del campeón.

Explicado con un ejemplo, una **habilidad** de un campeón hace daño a un enemigo con un valor base de 150 puntos de vida y un valor variable que corresponde al 30 % **daño** de ataque del campeón. Si en un momento determinado el campeón tiene 300 de **daño** de ataque, el **daño** total de la habilidad se calcula como $150 + (0,3 * 300) = 240$.


En el caso de realizar **daño** a un rival, existen tres formas en las que puede realizar: **daño físico**, **mágico** y **verdadero**. Esto está determinado por la habilidad y rol del campeón.

3.3.4. Estadísticas


Las estadísticas son diferentes valores numéricos que determinan las capacidades de cada campeón en un área en concreto del juego. Estos valores se ven modificados por la compra de objetos (3.3.5). A continuación se describen las estadísticas y que representan.


Daño de ataque  Daño realizado con ataques básicos.

Probabilidad de crítico  Probabilidad de que un ataque básico haga el doble de **daño**.

Velocidad de ataque  Cantidad de ataques básicos que se pueden realizar por segundo.

Poder de habilidad  Modifica el **daño** que realizan las habilidades.

Velocidad de movimiento  Velocidad a la que un campeón se desplaza por el mapa.

Armadura  Cantidad en la que se ve reducida el **daño físico** que se recibe.

Resistencia mágica Cantidad en la que se ve reducida el daño mágico que se recibe.

Penetración de armadura Cantidad de la armadura del rival ignorada a la hora de realizar daño físico.

Penetración mágica Cantidad de la resistencia mágica del rival ignorada a la hora de realizar daño mágico.

Vida Daño que tiene que recibir un personaje para morir.

Maná Coste de usar habilidades.

Robo de vida Porcentaje de vida recuperado al dañar a un rival.

3.3.5. Objetos

Dentro de la base de cada equipo en el mapa está localizada la tienda. Aquí los jugadores pueden comprar objetos con el oro que han ido ganando con el progreso de la partida. Su función es modificar las estadísticas del campeón, para que las habilidades del mismo sean más eficaces contra los rivales. Los objetos que se compran se quedan guardados en el inventario del campeón, el cual está limitado a seis objetos.

En el juego actual existen 222 objetos disponibles clasificados en cinco categorías.

Iniciales Objetos más relevantes al inicio de la partida.

Básicos Objetos que mejoran una estadística.

Épicos Objetos formados por la combinación de varios objetos básicos que mejoran varias estadísticas.

Legendarios Objetos formados por la combinación de objetos épicos y/o básicos que mejoran varias estadísticas y proporcionan algún efecto adicional.

Míticos Igual que los anteriores, pero limitado a uno en el inventario.

3.3.6. Ligas

Al igual que otros deportes, *League of Legends* posee un sistema de ligas que clasifica a los jugadores que lo deseen dentro de su sistema de ligas, en base a la habilidad que demuestren en sus partidas. Ordenadas de menor a mayor habilidad, y con el porcentaje de pertenencia sobre el total [?], las ligas del juego son las siguientes:

- Hierro - 2 %
- Bronce - 20 %
- Plata - 37 %
- Oro - 28 %
- Platino - 10 %
- Diamante - 1,5 %
- Maestro - 0,12 %
- Gran Maestro - 0,029 %
- Aspirante - 0,013 %

Desde Hierro a Diamante, cada una cuenta con cuatro divisiones (subcategorías dentro de cada liga), que van desde IV a I. Las tres restantes tienen una única división. Además, tanto Gran Maestro como Aspirante tienen plazas limitadas, 700 y 300 respectivamente.

La localización de cada jugador se basa en un sistema de puntos, ganándolos al ganar partidas y perdiéndolos al perder. En las ligas con divisiones, cuando el jugador alcanza 100 puntos pasa a la siguiente división, o el caso de estar en la división superior, subiría de liga. Por encima de Diamante no hay límite de puntos, y estando los jugadores ordenados por estos, la clasificación se realiza según el límite de plazas mencionado anteriormente.

Técnicas y herramientas



Esta parte de la memoria tiene como objetivo presentar las técnicas metodológicas y las herramientas de desarrollo que se han utilizado para llevar a cabo el proyecto. Si se han estudiado diferentes alternativas de metodologías, herramientas, bibliotecas se puede hacer un resumen de los aspectos más destacados de cada alternativa, incluyendo comparativas entre las distintas opciones y una justificación de las elecciones realizadas. No se pretende que este apartado se convierta en un capítulo de un libro dedicado a cada una de las alternativas, sino comentar los aspectos más destacados de cada opción, con un repaso somero a los fundamentos esenciales y referencias bibliográficas para que el lector pueda ampliar su conocimiento sobre el tema.

4.1. Técnicas

4.2. Herramientas

Aspectos relevantes del desarrollo del proyecto



Este apartado pretende recoger los aspectos más interesantes del desarrollo del proyecto, comentados por los autores del mismo. Debe incluir desde la exposición del ciclo de vida utilizado, hasta los detalles de mayor relevancia de las fases de análisis, diseño e implementación. Se busca que no sea una mera operación de copiar y pegar diagramas y extractos del código fuente, sino que realmente se justifiquen los caminos de solución que se han tomado, especialmente aquellos que no sean triviales. Puede ser el lugar más adecuado para documentar los aspectos más interesantes del diseño y de la implementación, con un mayor hincapié en aspectos tales como el tipo de arquitectura elegido, los índices de las tablas de la base de datos, normalización y desnormalización, distribución en ficheros, reglas de negocio dentro de las bases de datos (EDVHV GH GDWRV DFWLYDV), aspectos de desarrollo relacionados con el WWW... Este apartado, debe convertirse en el resumen de la experiencia práctica del proyecto, y por sí mismo justifica que la memoria se convierta en un documento útil, fuente de referencia para los autores, los tutores y futuros alumnos.

- 5.1. Control de velocidad de peticiones por ratios de la API**
- 5.2. Tolerancia a fallos**
- 5.3. Elección del algoritmo**
- 5.4. Uso de MongoDB**

Trabajos relacionados



Este apartado sería parecido a un estado del arte de una tesis o tesina. En un trabajo final de máster no parece tan obligada su presencia, aunque se puede dejar a juicio del tutor el incluir un pequeño resumen comentado de los trabajos y proyectos ya realizados en el campo del proyecto en curso.

Conclusiones y Líneas de trabajo futuras



Todo proyecto debe incluir las conclusiones que se derivan de su desarrollo. Éstas pueden ser de diferente índole, dependiendo de la tipología del proyecto, pero normalmente van a estar presentes un conjunto de conclusiones relacionadas con los resultados del proyecto y un conjunto de conclusiones técnicas. Además, resulta muy útil realizar un informe crítico indicando cómo se puede mejorar el proyecto, o cómo se puede continuar trabajando en la línea del proyecto realizado.

Apéndice

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En este apartado se va a exponer como se ha llevado a cabo la planificación del proyecto, además de los avances logrados entre cada iteración y reunión.

A.2. Planificación temporal

Durante el desarrollo del proyecto se ha seguido una versión simplificada de *Scrum*. Se han ~~usando~~ *sprints* para revisar los avances, con duración de una semana. Al final de cada *sprint* se realiza una reunión por videoconferencia para ver los avances logrados durante el mismo y planificar las tareas del siguiente.

El inicio oficial del proyecto fue el día 19 de mayo, con la reunión del Sprint 0 A.2.1, habiendo realizado las tareas de ese sprint previamente.

A.2.1. Sprint 0

Sprint dedicado a tareas logísticas y preparación del inicio del proyecto.

- Creación del repositorio de código.
- Solicitud de una clave permanente para la API a la desarrolladora del juego.
- Pruebas con la API para comprobar la existencia de los datos necesarios.

- Reunión para formalizar el inicio del desarrollo del proyecto.

A.2.2. Sprint 1

Primer sprint de desarrollo. Dedicado a la extracción de jugadores. Desde 20/05/2021 hasta el 26/05/2021.

- Documentación Sprint 0.
- Echar un ojo a *wrappers* existentes de la API para comprobar su viabilidad de uso.
- Desarrollo de un *notebook* para la extracción de jugadores, teniendo en cuenta límite de peticiones y tolerancia a fallos.

A.2.3. Sprint 2

Desde 27/05/2021 hasta el 02/06/2021.

- Documentación Sprint 1.
- Modificar extracción de jugadores para añadir identificador de cuenta, necesario para el siguiente paso de recuperación de partidas.
- Desarrollo de un *notebook* para extraer las partidas de los jugadores recuperador previamente.
- Crear biblioteca con funciones comunes para realizar peticiones y guardar estados de ejecución.

A.2.4. Sprint 3

Desde 03/06/2021 hasta el 09/06/2021.

- Documentación Sprint 2.
- Partiendo de los datos de partidas del sprint anterior, localizar y extraer los objetos que se han comprado para cada campeón dentro de la partida.
- Crear el formato final con las entradas para el algoritmo apriori.

A.2.5. Sprint 4

Desde 10/06/2021 hasta el 16/06/2021. Por conflicto con otras asignaturas durante esta semana se realizaron menos tareas.

- Documentación Sprint 3.
- Escribir la introducción de la memoria.
- Presentación sobre funcionamiento y conceptos del juego a los tutores.

A.2.6. Sprint 5

Desde 17/06/2021 hasta el 30/06/2021. La duración de este sprint se alargó a dos semanas por un problema de salud que impidió realizar avances significativos durante la primera.

- Correcciones en la memoria.
- Escribir objetivos del proyecto.
- Investigar alternativas del algoritmo apriori.
- Buscar y escoger una biblioteca con las implementaciones del algoritmo a utilizar y sus alternativas.
- Aprendizaje de Django.
- Inicio del desarrollo de la aplicación web.

A.2.7. Sprint 6

Desde 01/06/2021 hasta el 07/07/2021. Por conflicto con otras asignaturas durante esta semana se realizaron menos tareas.

- Desarrollo de un notebook en el que probar los algoritmos de conjuntos frecuentes.
- Selección del algoritmo a usar para obtener los resultados que mostrar en la aplicación.

A.2.8. Sprint 7

Desde 08/07/2021 hasta 14/07/2021.

- Instalación de base de datos MongoDB.
- Guardar ejecución del algoritmo en MongoDB.
- Listado de campeones en la aplicación web.
- Al seleccionar un campeón mostrar sus conjuntos de objetos frecuentes.

A.2.9. Sprint 8

- Documentación sprints 4, 5, 6 y 7.
- Incorporación del *notebook* de extracción de jugadores a la aplicación web.
- Incorporación del *notebook* de extracción de partidas a la aplicación web.
- Añadir buscador de campeones.
- Ficha del campeón junto a sus objetos frecuentes.
- Mensaje cuando no existen datos para un campeón.

A.2.10. Sprint 9

- Documentación sprints 8 y 9.
- Extracción de gran cantidad de partidas.
- Incorporación del *notebook* de extracción de detalles de partidas a la aplicación web.
- Incorporación del *notebook* de extracción de ejecución de algoritmos a la aplicación web.
- Escritura de conceptos teóricos sobre el juego.

A.3. Estudio de viabilidad

A.3.1. Viabilidad económica

A.3.2. Viabilidad legal

Apéndice B

Especificación de Requisitos

B.1. Introducción

En este apéndice se describen los objetivos generales de la aplicación y se detallan sus requisitos, tanto funcionales como no funcionales.

B.2. Objetivos generales

- Desarrollar un proceso ETL que sea capaz de recopilar los datos necesarios usando la API oficial de Riot Games¹.
- Aplicar técnicas de aprendizaje no supervisado sobre los datos recopilados, en este caso algoritmos para la obtención de conjuntos frecuentes de objetos.
- Ser capaz de obtener conocimiento útil a partir de los datos obtenidos.
- Desarrollar una aplicación en la que se pueda consultar el conocimiento extraído.
- Que el producto final sea capaz de ayudar a los nuevos jugadores.

¹Riot Games es la desarrolladora de League of Legends

B.3. Catalogo de requisitos

B.3.1. Requisitos funcionales

- **RF-1 Proceso ETL:** la aplicación debe ser capaz de recopilar, transformar y almacenar los datos para sus posterior uso.
 - **RF-1.1 Extracción de jugadores:** la aplicación debe ser extraer los jugadores de las ligas más altas.
 - **RF-1.2 Extracción de partidas:** la aplicación debe extraer las partidas de los jugadores obtenidos previamente.
 - **RF-1.3 Generación de transacciones:** la aplicación debe transformar los datos de partidas en listados de objetos agrupados por campeón.
 - **RF-1.4 Generación de conjuntos frecuentes:** la aplicación debe generar conjuntos frecuentes de objetos usando las transacciones obtenidas anteriormente.
- **RF-2 Consulta de información:** la aplicación debe ser capaz de recopilar, transformar y almacenar los datos para sus posterior uso.
 - **RF-2.1 Búsqueda por campeón:** el usuario debe poder buscar conjuntos frecuentes filtrando por un campeón concreto.
 - **RF-2.2 Búsqueda en partida activa:** el usuario debe poder buscar conjuntos de objetos usando su nombre dentro del juego, de tal forma que se muestre la información adecuada para el campeón usado en ese momento.

B.3.2. Requisitos no funcionales

- **RNF-1 Usabilidad:** la aplicación debe ser intuitiva y fácil de usar.
- **RNF-2 Mantenibilidad:** debe ser sencillo añadir funcionalidad nueva a la aplicación.
- **RNF-3 Compatibilidad:** la aplicación debe poder funcionar en los principales navegadores.
- **RNF-4 Responsividad:** la aplicación debe adaptarse al tamaño de la pantalla.

B.4. Especificación de requisitos

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

D.1. Introducción

En este apéndice se presenta todo lo que tiene que conocer un desarrollador para poder continuar con el desarrollo de la aplicación. Se describe la estructura de directorios del proyecto, como instalar la aplicación, etc.

El proyecto tiene dos partes diferenciadas en cuanto a desarrollo. La primera es una colección de *notebooks* en la que se realizar pruebas con la API y algoritmos. La segunda es una aplicación web Django estándar.

D.2. Estructura de directorios

```
/ Directorio raíz
├── memoria/ -Documentación del proyecto
│   ├── img/ -Imágenes de la memoria
│   ├── tex/ -Secciones de la memoria
│   ├── memoria.tex -Código fuente de la memoria
│   ├── memoria.pdf -Memoria del proyecto
│   └── bibliografia.bib -Fuentes bibliográficas
├── scripts and notebooks/ -Colección de notebooks y scripts
│   ├── outputs/ -Directorio donde se almacenan las salidas de los notebooks
│   ├── 0_extract_players.ipynb -Notebook para extraer jugadores
│   └── 1_extract_matches.ipynb -Notebook para extraer partidas
```

- 2_download_matches_details.ipynb -Notebook para obtener detalles de cada partida
- 3_algorithms.ipynb -Notebook para probar algoritmos
- utils.py -Colección de funciones comunes para los notebooks
- api_key.txt -Fichero que contiene la clave de la API
- requirements.txt -Fichero que lista las dependencias
- web/ -Directorio del proyecto en Django
 - betterbuilds/ -App de Django que contiene la web
 - migrations/
 - static/
 - templates/
 - __init.py__
 - admin.py
 - apps.py
 - models.py
 - tests.py
 - urls.py
 - views.py
 - etl/ -App de Django que contiene el proceso ETL
 - management/
 - commands/
 - migrations/
 - __init.py__
 - admin.py
 - apps.py
 - models.py
 - tests.py
 - views.py
 - web/ -Directorio de configuración del proyecto Django
 - __init.py__
 - asgi.py
 - settings.py
 - urls.py
 - utils.py
 - wsgi.py
 - .env -Fichero con [variables](#) de entorno
 - manage.py -Script para interactuar con el proyecto
 - requirements.txt -Fichero que lista las dependencias

D.3. Manual del programador

D.4. Compilación, instalación y ejecución del proyecto

Apéndice E

Documentación de usuario

- E.1. Introducción
- E.2. Requisitos de usuarios
- E.3. Instalación
- E.4. Manual del usuario