Módulo de fidelización de clientes en base a las compras realizadas

Ramon Zalabardo Bosch

Junio 2022

Trabajo final de grado

Director: Joan Subirats Salvans (CTO & Co-Founder de Stockagile)

Ponente: Javier Bejar Alonso (Departamento de Ciencias de la Computación)

Tutor de GEP: Ferran Marfil Sanchez (Departamento de Organización de Empresas)

Grado en Ingeniería Informática

Mención en Computación

Facultad de Informática de Barcelona

Universidad Politècnica de Catalunya · BarcelonaTech

Índice

1 Contexto

1.1 Introducción

Este proyecto consistirá en la elaboración de un módulo de fidelización de clientes que deberá cumplir las siguientes funciones:

- El almacenamiento y la gestión de un valor que representará el saldo acumulado de un cliente de tienda física.
- La aplicación del descuento de este valor en compras realizadas por el cliente.
- La aplicación del descuento de este valor en todo el módulo de facturación.
- La capacidad de identificar los clientes más propensos a comprar un producto teniendo en cuenta sus compras realizadas.
- La capacidad de visualizar y exportar la información obtenida mediante los puntos anteriores.

El módulo se implementará como parte del sistema de planificación de recursos empresariales (ERP) de Stockagile.

Stockagile es un Saas de gestión rápida y eficiente de inventario, productos, finanzas y varios factores esenciales para el correcto funcionamiento de una empresa. Una vez los usuarios importan la información necesaria en la aplicación pueden gestionar todas las etapas desde la fabricación del producto hasta su venta a un tercero.

El objetivo del software es centralizar todas las gestiones de la empresa en un único sistema usable, eficiente y funcional.

El software consta de 9 módulos que cubren diferentes funcionalidades:

- Analítica
- Contactos
- Inventario
- Fabricación
- Mayorista
- Tiendas
- Ecommerce
- Facturación

Es accesible a través de una aplicación web que consta de frontend y backend conectados a través de una API REST.

Las tecnologías con las que están implementados son:

- El frontend está desarrollado con Angular, JavaScript, HTML, y CSS. [1] [2]
- El backend utiliza Django y otras librerías de Python para gestionar una base de datos PostgreSQL. [3] [4] [5]

Algunas de las librerías de Python que se han utilizado para desarrollar ciertas funcionalidades de los distintos módulos de Stockagile son easy_pdf, logging, shopify y rest_framework.

Aparte de estas tecnologías, a nivel interno se usan otros frameworks y aplicaciones para manejar correctamente la gestión del proyecto:

- Se utiliza el servidor de Jenkins para automatizar todo el proceso de compilar, testear y desplegar el proyecto. [6]
- El motor de analítica de Elasticsearch se utiliza para la indexación de varios de los datos del proyecto.
- La aplicación de frontend Kibana se utiliza sobre los logs almacenados por Elasticsearch para mostrar y representar los errores, avisos e información que resultan del software. [7]
- El sistema de Docker permite almacenar las diferentes partes del proyecto en contenedores para poder ejecutarlo desde cualquier ordenador. [8]
- Se utiliza el cloud de Amazon Web Services (AWS) para almacenar las bases de datos utilizados en la aplicación. [9]
- Git es el sistema de control de versiones que se utiliza para gestionar las funcionalidades que se desarrollan en paralelo por los miembros del equipo técnico.
- Se utiliza Bitbucket para gestionar el código Git del proyecto y realizar los repasos del código entre los miembros del equipo antes de ser publicado.
- El almacén de datos Redis funciona como la caché de la aplicación para poder acceder a los datos de forma más rápida.

1.2 Términos y conceptos

A lo largo de este proyecto se mencionan ciertos términos y conceptos que pueden no ser familiares para el lector. Para facilitar la lectura de este documento se definirán a continuación:

- *Módulo*: En el contexto de stockagile el software está distribuido en distintos módulos que se ocupan de gestionar las distintas necesidades de una empresa. Cada uno de estos módulos tiene sus propias características y es normalmente independiente de los demás.
- *Sistema de recomendación*: Un sistema de recomendación es una herramienta que establece un conjunto de criterios y valoraciones sobre los datos de los clientes de cada usuario para realizar predicciones sobre recomendaciones de productos que puedan ser de utilidad o valor para el cliente.
- *Usuario*: A lo largo de este proyecto, el término usuario se referirá al principal consumidor del software, es decir, los clientes directos de Stockagile.
- *Cliente*: Por lo general el uso del término cliente se utilizará para referirse a los clientes de los usuarios, también se mencionarán como clientes finales.
- *ERP*: El término ERP se refiere a Enterprise Resource Planning, que significa “sistema de planificación de recursos empresariales”. Estos programas se hacen cargo de distintas operaciones internas de una empresa, desde producción a distribución o incluso recursos humanos. [10]
- Front-end: El frontend de un software o una página web es todo con lo que interactúa el usuario. Desde el punto de vista del usuario el frontend es un sinónimo de interfaz de usuario.
- Back-end: En el mundo de la informática, el backend se refiere a cualquier parte de la web o el software que el usuario no puede ver. Contrasta con el frontend. En términos de programación el backend es la capa de acceso a los datos mientras que el frontend es la capa de presentación de los datos.

1.3 Problema

La métrica más importante para la empresa en estos momentos, por la que se guían la mayoría de las decisiones es la cantidad de ventas que realizan nuestros usuarios. Es por eso que cuando planeamos un nuevo desarrollo la pregunta principal que nos hacemos es: "¿Esto hará que nuestros usuarios obtengan clientes nuevos o los existentes realicen más compras?".

Reformulando esta misma pregunta con el objetivo de saber cuál podría ser el próximo desarrollo, nos podríamos preguntar qué es lo más importante para que nuestros usuarios llamen la atención de sus clientes actuales y de posibles nuevos clientes.

Está claro que esta pregunta podría tener muchas respuestas distintas e igual de válidas, sin embargo la que nos ha parecido más adecuada a nosotros es "el marketing".

El marketing es esencial para toda empresa, es una forma de dar a conocer tu producto a nuevos clientes e incentivar a tus clientes existentes a seguir interesados en él. En un momento de grandes cambios y novedades tecnológicas se hace vital que las empresas vayan de la mano con las nuevas tecnologías y definitivamente esto ha marcado una evolución a pasos agigantados en el mundo del Marketing, ya que se ha convertido en parte necesaria del universo on line.

A partir de aquí nos realizamos una nueva pregunta, ¿Cómo podemos ayudar a nuestros usuarios en sus estrategias de marketing de forma eficiente y automatizada?.

Cada estrategia de marketing es única para cada empresa, producto y cliente, por lo que desarrollar manualmente una estrategia para cada uno de nuestros usuarios y a su vez para cada uno de sus productos y clientes, resultaría en un gran coste de los recursos de Stockagile.

Es debido a esta complicación por lo que se nos ocurrió elaborar un módulo especializado en la fidelización y clasificación de clientes, que permitiera desarrollar diferentes estrategias de marketing personalizadas para estos de forma automatizada.

1.4 Actores implicados

En este proyecto hay distintos actores implicados:

- El *personal del proyecto*: Yo mismo y el CTO de Stockagile nos hemos comprometido a desarrollar y finalizar este proyecto que aportará un gran valor a la aplicación.
- Los *fundadores de Stockagile*: Esperan hacer crecer tanto su empresa como su producto, con el objetivo de obtener beneficios y de ofrecer una herramienta de gran utilidad a otras empresas.
- Los *miembros del equipo de Stockagile*: Les interesa que el proyecto avance, crezca y obtenga más beneficios, lo cual hará que ellos mismos también crezcan en el mundo laboral y obtengan más recompensas.
- Los *inversores de Stockagile*: Requieren que la empresa crezca y obtenga los beneficios esperados para que puedan no solo recuperar su inversión inicial sino también para seguir obteniendo ingresos de esta en un futuro.
- Los *usuarios de Stockagile*: Quieren hacer crecer su negocio y obtener mayores beneficios como resultado de las compras realizadas por sus clientes.
- Los *clientes de los usuarios de Stockagile*: Quieren descuentos y ofertas en los distintos productos que les interesan además de conocer nuevos productos que les puedan interesar.

2 Justificación

2.1 Situación actual

Como se ha comentado en el apartado de introducción, el software de Stockagile está compuesto por 8 módulos distintos que cubren prácticamente todas las necesidades que pueda tener una empresa de un software de gestión. Pero como es de esperar el objetivo ideal sería llegar a cubrirlas absolutamente todas.

Cada desarrollo que se realiza en Stockagile está orientado a cumplir este objetivo o al menos a acercarnos un poco más a él. En cuanto a nuestro proyecto, la intención de este es acercar a Stockagile a poder proporcionar una ayuda para las necesidades de marketing que pueda tener una empresa.

Es necesario para nuestros usuarios que puedan ofrecer unas ofertas específicas y personalizadas a sus propios clientes, y Stockagile es quien tiene la información y los recursos para hacer esto posible.

2.2 Opciones de marketing digital

Como se ha mencionado anteriormente el marketing digital es una herramienta prácticamente esencial para cualquier empresa que quiera triunfar en la actualidad, sin embargo no es demasiado común disponer de las tecnologías, recursos o habilidades para poder diseñar y ejecutar las estrategias de marketing digital dentro de la misma empresa, por lo que normalmente se suele contratar a un tercero que se encargue de ese trabajo. [11]

Muchas empresas, grandes y pequeñas, han visto el potencial de este mercado y han desarrollado aplicaciones para realizar un marketing digital personalizado para los usuarios.

Algunas de estas empresas son:

- Google (Google Ads): Con este servicio se puede crear publicidad específica para una empresa y mostrarlo a los usuarios que buscan cosas similares. Se utiliza principalmente para llamar la atención de nuevos clientes potenciales, pero puede llegar a ser molesto para clientes existentes de la empresa. Dado que Google es hoy en día una de las empresas más importantes a nivel mundial, una de las grandes ventajas que ofrece este servicio es la capacidad de acceder a una gran cantidad de usuarios. [12]

- Meta (Meta For Business): La empresa Meta, propietaria de algunas de las redes sociales más utilizadas en todo el mundo (WhatsApp, Instagram, Facebook…), dispone de un servicio llamado Meta For Business que hace uso de estas mismas redes sociales para publicitar todo tipo de empresas y productos. [13] Igual que Google, la gran ventaja de Meta es su gran rango de accesibilidad ya que hoy en día unos 3,600 millones de personas utilizan aplicaciones pertenecientes a la empresa Meta. [14]

- Amazon (Amazon Ads): La gran ventaja que tiene Amazon, es que la misma compañía de Amazon puede servirte para vender tus productos a través de ella. A pesar de que su algoritmo de publicidad no sea tan eficiente en el sentido de a quien y cuando mostrarla, es un servicio a tener en cuenta a la hora de comparar opciones. [15]

A pesar de estas opciones, muchas de las empresas digitalizadas tienen sus propias técnicas de marketing digital que suelen consistir en enviar mails de forma repetida u ofrecer descuentos mediante un sistema de fidelización. Sin embargo el servicio que proporcionará este módulo a los clientes de

Stockagile es prácticamente único, ya que ofrece la posibilidad de desarrollar estas técnicas de marketing y targeting de clientes de forma automatizada y sin la necesidad de adquirir los servicios de un tercero.

3 Alcance y obstáculos

3.1 Objetivo

El objetivo final de este proyecto es diseñar e implementar un módulo de fidelización de clientes automatizado, partiendo de los datos relativos a estos que se tienen de ventas anteriores. Este módulo debe ser eficiente, funcional y seguro. La intención es que permita realizar el análisis de todos los productos pertenecientes a una compañía y los clasifique mediante un modelo de aprendizaje automático entrenado con los datos existentes de los productos y las ventas de los usuarios de Stockagile. También mostrará un esquema de cómo se han clasificado y utilizará esta información para alimentar un sistema de recomendación que encontrará los clientes de la compañía que puedan estar más interesados en cada producto.

3.2 Sub-objetivos e indicadores

Los sub-objetivos e indicadores son los siguientes:

1. **Desarrollo de la funcionalidad de fidelización de clientes**
    (a) Cada cliente tiene una tasa de acumulación de saldo
    (b) La tasa de acumulación de saldo incrementa cuando el cliente realiza compras
    (c) La tasa de acumulación de saldo varía con el tiempo
    (d) El usuario de Stockagile puede modificar la tasa de acumulación de saldo
2. **Desarrollo de la aplicación del descuento en las compras según su acumulación de saldo**
    (a) La acumulación de saldo puede suponer un descuento en las compras del cliente
    (b) El punto de venta muestra la acumulación de saldo del cliente
    (c) Si la compra sigue abierta el usuario puede modificar el descuento por acumulación de saldo
3. **Adaptación de la facturación para aplicar el descuento por acumulación de saldo**
    (a) Se pueden crear facturas a partir de una compra con descuento por acumulación
    (b) Las facturas permiten modificar el descuento como porcentaje y como absoluto
    (c) El cálculo del total de la factura se realiza correctamente independientemente de si es de mayorista o minorista
4. **Entrenamiento del modelo de clasificación**
    (a) Existe un dataset lo bastante grande para entrenar el modelo
    (b) La información del dataset proviene de la base de datos de Stockagile
    (c) La información es suficientemente general como para no causar sobreajuste
    (d) Los datos y atributos del dataset son relevantes y adecuados para el modelo
    (e) Se ha realizado un preprocesamiento adecuado en el dataset

(f) Se obtienen buenos resultados al analizar el modelo entrenado

5. **Modelo de clasificación de productos según su información**
   (a) El modelo clasifica de forma correcta los productos
   (b) Las decisiones que toma el modelo al clasificar tienen sentido
   (c) El modelo obtiene resultados rápidos y concretos
6. **Sistema de recomendación de productos a clientes**
   (a) El sistema de recomendación obtiene resultados con sentido
   (b) Utiliza las técnicas más eficientes
   (c) El sistema de recomendación actúa sobre una sola compañía
7. **Desarrollo de ofertas de productos según la información obtenida**
   (a) A partir de los resultados obtenidos se crean ofertas que encajan con los clientes
8. **Visualización de las ofertas**
   (a) La visualización de las ofertas es fácilmente entendible
   (b) Las ofertas muestran posibles cambios o fallos
   (c) El usuario puede modificar las ofertas antes de enviarlas
9. **Optimización de eficiencia y seguridad del módulo**
   (a) El modelo de clasificación tiene un tiempo de ejecución optimizado
   (b) El sistema de recomendación tiene un tiempo de ejecución optimizado
   (c) El módulo contempla casos extremos y trata los errores como es debido
   (d) El módulo no tiene fallos internos

3.3 Requisitos

Los distintos requisitos que contemplamos para que el módulo funcione correctamente son:

1. Es necesario que los usuarios puedan ejecutar el modelo en cualquier momento para analizar y clasificar nuevos productos o volver a clasificar productos ya existentes.
2. Cuando el usuario cree las ofertas deberá rehacerse la clasificación y recomendación para todos los productos, debido a que podría haber nueva información en la base de datos, y podría alterar los resultados.
3. La interfaz tiene que ser simple y fácil de usar y entender para los usuarios de Stockagile, para evitar posibles errores humanos en los datos.
4. Es necesario que el tiempo de ejecución de cualquier funcionalidad sea óptimo para conseguir una mayor satisfacción de los clientes.
5. El sistema tiene que informar a los clientes de todos los pasos que da para justificar las decisiones que toma y que los usuarios se sientan más seguros a la hora de enviar las ofertas.

6. Es necesario que el módulo complemente al resto de módulos del software, pero no interfiera en estos causando errores o fallos.

3.4 Obstáculos y riesgos

Al ser un proyecto considerablemente grande que forma parte de un software con una estructura muy compleja, hay varios obstáculos que podrían perjudicar el proyecto:

- *Acoplamiento*. El software de Stockagile por lo general está bastante acoplado, lo que significa que realizar un pequeño cambio en una parte del código puede derivar en realizar una refactorización del código a gran escala. Esto más que un proceso difícil es un proceso largo que requiere muchas validaciones por lo que podría resultar en una gran pérdida de tiempo.
- *Falta de datos*. Como se ha mencionado anteriormente en los sub-objetivos del proyecto, para que el modelo sea eficiente, funcional y no se sobreajuste, es necesario un dataset muy grande y bien definido. Debido a que el dataset debería provenir de la base de datos de Stockagile, este puede no ser lo suficientemente grande y por lo tanto generar un modelo sobre ajustado.
- *Factores externos*. Stockagile es una start-up que ha empezado a crecer de forma exponencial durante el último año por lo que cada vez tiene más usuarios, componentes y por lo tanto cambios. Estos cambios pueden afectar de forma directa a este proyecto ya que lo que es necesario y urgente para los clientes ahora mismo, puede que no sea lo mismo en un futuro. Esto puede significar que a lo largo del desarrollo lleguen a modificarse varias partes del módulo.
- *Tiempo*. Dado el tamaño y la complejidad del proyecto y teniendo en cuenta la gran cantidad de obstáculos y cambios que pueden surgir el tiempo necesario para acabarlo puede ser demasiado. Esto podría resultar en la necesidad de hacer menos complejas algunas partes del proyecto, o de sustituirlas por desarrollos más sencillos y rápidos.

4 Metodología y seguimiento

4.1 Metodología de trabajo

Este proyecto está dividido en varias partes, cada una con diferentes niveles de complejidad. Es un proyecto vinculado a la mención de Computación, pero con mucha implicación por parte de la Ingeniería de Software

En cuanto a la mención de Computación se refiere, se estudiará cómo programar, entrenar y optimizar el modelo de aprendizaje automático que se usará para la clasificación de productos así como el análisis de datos previos y el preprocesamiento de estos. Además se diseñará e implementará un sistema de recomendación complejo basado en los datos obtenidos de los clientes y productos. Por último se estudiarán los algoritmos utilizados para intentar optimizarlos en la medida de lo posible.

Por otro lado, como ya se ha mencionado anteriormente, el proyecto se desarrollará como parte de la aplicación web de Stockagile. Esto significa que para adaptar el proyecto al software de Stockagile será necesario diseñar el UML del módulo, explorar los casos de uso y las funcionalidades. También habrá que investigar sobre el cloud para ver cómo se le puede sacar partido para optimizar la experiencia del usuario en lo que respecta a la transferencia de datos entre front-end y back-end.

En el desarrollo de Stockagile se utiliza la metodología Scrum, con la intención de facilitar el progreso a los nuevos integrantes de la empresa y coordinar el trabajo realizado en paralelo por distintos desarrolladores sobre el mismo código. Dada la complejidad del proyecto y la necesidad de sincronización con el resto del equipo, creo que también es la metodología más adecuada para el proyecto.

4.2 Seguimiento

El seguimiento del proyecto se llevará a cabo a través de reuniones bimensuales. Ya que al tratarse de una start-up el contacto y la asistencia diaria es muy común, se ha decidido que no es necesario realizar reuniones semanales.

Durante el día a día se irán desarrollando las diferentes tareas propuestas en el apartado 7 de este documento y durante el contacto diario con el director del proyecto se hará un seguimiento poco escrupuloso, mediante el método SCRUM, de en qué estado se encuentra cada una de las tareas.

En las reuniones bimensuales se revisarán los avances realizados en el último periodo de quince días y se establecerán los objetivos y las tareas a realizar preferiblemente antes de la siguiente reunión. Se comprobará constantemente los objetivos y los sub-objetivos conseguidos para contabilizar el avance del proyecto, para lo que se tendrán en cuenta los indicadores.

5 Extensión temporal

Hemos estimado que este proyecto nos tomará unas 811 horas de dedicación a lo largo de dieciséis semanas. Dado que este proyecto formaba parte del roadmap de Stockagile, los conceptos principales ya estaban definidos. Sin embargo, hasta después de matricular el proyecto, no establecimos sus detalles.

El proyecto se inició el lunes 14 de febrero de 2022, el día que se realizó la primera reunión para especificar los objetivos y las tareas del proyecto y la intención es terminar el proyecto el jueves 30 de junio de 2022.

Como se ha mencionado anteriormente este proyecto es bastante complejo, lo que complica la estimación exacta de una fecha de finalización. Es por esto que se ha decidido estimar la fecha de finalización tan pronto.

6 Personal y material

El personal necesario para este proyecto se reduce a tres perfiles distintos, el director del proyecto, un programador y un grupo de testers.

- El *director del proyecto* se encargará de dirigir, supervisar y facilitar el desarrollo de las diferentes tareas del proyecto. El es quien se encargará de describir cada una de estas tareas y de establecer cuales son las funcionalidades que se deberían cumplir. Durante el desarrollo de estas tareas ofrecerá la ayuda necesaria para que se desarrollen de forma rápida y adecuada.

- El *programador* desarrollará las tareas ideadas por el director del proyecto. Aparte de programar el código que dará forma a todas las tareas, se encargará de investigar todo lo necesario para que las funcionalidades más complejas, el modelo de clasificación y el sistema de recomendación, funcionen de forma eficiente y sin problemas.

- El *grupo de testers* serán los encargados de validar que las funcionalidades cumplen los requisitos necesarios y se adaptan bien con el resto de módulos de la aplicación.

El material necesario para este proyecto es:

- Al menos tres ordenadores personales (uno para cada persona implicada en el proyecto)

- Un espació donde trabajar con acceso a internet

7 Tareas del proyecto

En esta sección se detallan las tareas a realizar de forma individual, pero se agrupan por bloques para distinguir con mayor facilidad las distintas fases del proyecto.

GP - Gestión del proyecto

La gestión del proyecto es esencial para planificar, definir y documentar el trabajo a realizar, además, engloba las reuniones para la validación y propuesta de objetivos semanales. Se estima que en global el grupo de gestión tendrá una duración de 224 horas.

GP.1 - Alcance

Antes de empezar el proyecto es necesario definir la extensión que tendrá este, debido a esto, se ha dedicado un tiempo inicial para definir qué se quiere conseguir con este proyecto, qué se va a desarrollar y qué medios serán necesarios. La duración ha sido de 40 horas. Antes de definir el alcance del proyecto ha sido necesario investigar el estado del arte de los sistemas de recomendación y fidelización para poder definir cuales se desarrollarán.

GP.2 - Planificación

Para una mayor certeza en el alcance del proyecto definido, se realiza una planificación temporal, de recursos y de requerimientos para cada una de las distintas tareas de las que consta el proyecto. También se determinarán los posibles riesgos y obstáculos que podrían interferir con la planificación y se proponen distintas soluciones para estos. La duración de esta tarea ha sido de 32 horas.

GP.3 - Presupuesto

Se determinará el coste total del proyecto teniendo en cuenta tanto los recursos materiales o tangibles (ordenadores, pantallas, router…) como los intangibles (trabajadores, conexión a internet, electricidad…). Para prevenir posibles contratiempos será necesario ser preciso al definir el presupuesto del proyecto. Se estima que la tarea requerirá una dedicación de 24 horas.

GP.4 - Informe de sostenibilidad

Se analizará a partir de un informe el impacto medioambiental, económico y social del proyecto, en concreto, de las fases de planificación y desarrollo. El tiempo estimado para realizar el informe es de 4 horas.

GP.5 - Reuniones

Dado que es necesario comprobar que el avance del proyecto va según lo esperado sin ningún inconveniente será necesario realizar reuniones bimensuales para poder establecer los objetivos cumplidos y si hay necesidad de cambiar aspectos de la planificación para adaptarse mejor a ellos. La duración total de estas reuniones tendrá un tiempo estimado de 40 horas.

GP.6 - Documentación

Para realizar un proyecto en condiciones lo más importante es crear una memoria final que ayude a entender todo el desarrollo del proyecto, por eso, se va a ir documentando las distintas fases y tareas de este a medida que se desarrollen. La documentación se realizará de forma paralela al resto de las tareas de forma que se pueda redactar con máximo detalle. La duración estimada de la documentación será de 60 horas.

GP.7 - Presentación

Una vez finalizado el proyecto y la documentación será necesario preparar la presentación para el tribunal que evaluará el TFG. Se preparará el material de soporte visual y el guión, así como también se realizarán ensayos para practicar y se repasará el proyecto completamente. La duración estimada es de 24 horas.

TP - Trabajo previo

Esta fase contiene las tareas a realizar antes de empezar con el desarrollo del proyecto, incluye todo lo relacionado con la preparación y el estudio previo necesario para poder desarrollar el proyecto en condiciones óptimas. Dado que se trata de una fase de preparación se realizará paralelamente a la mayoría de las tareas de gestión de proyectos. Se estima una duración de 30 horas.

TP.1 - Preparación del entorno de trabajo

Dado que el proyecto va a requerir varias interacciones entre el programador y el director del proyecto, se necesita una preparación del entorno de trabajo para permitir una mayor flexibilidad a la hora de comunicarse sin molestar o interrumpir al resto del equipo. La estimación temporal de esta tarea es de 5 horas.

TP.2 - Estudio del estado del arte

Dado que el proyecto va a ser una parte importante del software de Stockagile, va a ser necesario que este sea eficiente y funcional en todos los sentidos, por lo que se estudiarán las posibles técnicas y sistemas de recomendación a utilizar para el proyecto. Principalmente se analizarán las herramientas

utilizadas por las grandes empresas de software (Spotify, Google, Amazon…). La estimación de esta tarea es de 24 horas.

DASC - Diseño y desarrollo del valor de acumulación de saldo de los clientes

DASC.1 - Diseño

Esta tarea consistirá en diseñar el UML de la arquitectura necesaria para implementar el valor de la acumulación de saldo a nivel de cliente y global de la empresa de forma que se pueda editar de forma masiva o específicamente para cada cliente. Es importante asegurarse de que el UML no va a causar errores ni interferir con otras partes de la arquitectura independientes de ésta, así como también, tener en cuenta las posibles modificaciones en un futuro. El tiempo estimado de dedicación a esta tarea será de 24 horas.

DASC.2 - Desarrollo

Esta tarea consistirá en desarrollar y programar el código necesario para implementar el UML mencionado en el apartado anterior. Esta tarea tendrá un tiempo de dedicación estimado de 44 horas.

DDBC - Diseño y desarrollo de descuento por baremos en compras

DDBC.1 - Diseño

Esta tarea consistirá en diseñar el UML de la arquitectura necesaria para implementar un descuento por baremos en las compras de tienda física. Es importante asegurarse de que el UML no va a causar errores ni interferir con otras partes de la arquitectura independientes de ésta, así como también, tener en cuenta las posibles modificaciones en un futuro. El tiempo estimado de dedicación a esta tarea será de 24 horas.

DDBC.2 - Desarrollo

Esta tarea consistirá en desarrollar y programar el código necesario para implementar el UML mencionado en el apartado anterior. Esta tarea tendrá un tiempo de dedicación estimado de 44 horas.

SCC - Sistema de clasificación de productos

SCC.1 - Investigación

A partir de la información obtenida durante la preparación del proyecto, será necesario estudiar cuál es el tipo de modelo que mejor se adaptará a la información de la que se dispone. También será necesario pensar en la mejor forma de estructurar el dataset con la información de los productos de forma que al entrenar al modelo con este, obtenga los mejores resultados. La duración estimada es de 24 horas.

SCC.2 - Preparación del dataset

Una vez pensado el tipo de modelo que se utilizará y cómo se estructurará el dataset para alimentarlo, va a ser necesario obtener el dataset adecuado, esto se conseguirá a partir de la información que tiene Stockagile sobre los productos de una empresa o a partir de datasets de dominio público. Esta tarea tiene una duración estimada de 40 horas.

SCC.3 - Entrenamiento del modelo

Esta tarea consistirá en utilizar el dataset para entrenar al modelo de la forma más adecuada posible. Cabe la posibilidad de que tras entrenar al modelo no obtengamos las métricas esperadas por lo que quizás será necesario modificar el dataset del modelo para que se adapte mejor. La duración de esta tarea se estima en 52 horas.

SCC.4 - Adaptación del modelo en el software

Tras tener un modelo entrenado con buenas métricas y buenos resultados, será necesario adaptarlo dentro del software de Stockagile de forma que se adapte a la interfaz gráfica y al resto de la arquitectura. El tiempo estimado de esta tarea será de 60 horas.

SRC - Sistema de recomendación de clientes

SRC.1 - Investigación

A pesar de haber realizado una parte de la investigación durante la preparación del proyecto, será necesario comprobar cuál es el sistema de recomendación más adecuado para la información que podemos obtener de los clientes y cómo se podría adaptar a nuestra plataforma. La estimación del tiempo de esta tarea es de 32 horas.

SRC.2 - Diseño

Tras tener claras las técnicas y el sistema de recomendación que se adapta mejor al software de Stockagile, será necesario diseñar el UML de la arquitectura que implementará el sistema de recomendación. Es importante tener en cuenta que es una herramienta que tocará varias partes del Software y se tendrá que diseñar la arquitectura de forma que no interfiera con el resto del funcionamiento de esta. La duración de esta tarea será de 32 horas.

SRC.3 - Implementación

Una vez realizado el diseño de la arquitectura, se procederá a implementar el sistema de recomendación y adaptarlo en el software de Stockagile. Como se ha mencionado anteriormente, para desarrollar esta herramienta se tendrán que tener en cuenta muchas de las partes que componen el software por lo que podríamos encontrarnos con la necesidad de modificar el diseño inicial. Esto es algo que nos podría tomar mucho tiempo a la larga. Esta tarea tiene un tiempo de duración estimado de 60 horas.

SVE - Sistema de visualización y exportación de datos

SVE.1 - Diseño

Se diseñará un sistema de visualización y exportación de datos para que el usuario pueda comprobar y analizar los resultados obtenidos por el sistema de clasificación y el de recomendación de clientes. Está tarea puede variar bastante en cuanto a la extensión temporal, ya que el formato de los datos obtenidos no estará claro hasta haber terminado las tareas anteriores. La duración de esta tarea será de 8 horas.

SVE.2 - Desarrollo

Se desarrollará el sistema de visualización y exportación de datos adaptándolo a la arquitectura del proyecto y a la de Stockagile. Esta tarea tendrá una estimación temporal de 24 horas.

P - Pruebas

Tras haber finalizado el desarrollo del módulo, es importante realizar varias pruebas exhaustivas en todas las funcionalidades y detalles del módulo. Ésta tarea se realizará entre el grupo de testers y el programador del proyecto. Dado que la herramienta que se desarrollará con este proyecto se utilizará en un software accesible por todos los usuarios de Stockagile y que este es esencial para el correcto funcionamiento de sus negocios, es muy importante que no pueda producir ningún error en el software o en la base de datos, por lo que se deberá testear entero a conciencia. Esta tarea tendrá una duración de 60 horas.

8 Estimaciones y gantt

El tiempo que se le va a dedicar a este proyecto es de 811 horas. Dado que el proyecto se realizará en horas de jornada laboral, hemos distribuido el tiempo a dedicar a las distintas tareas en días, teniendo en cuenta que cada día se dedicarán unas 8 horas al proyecto, sin embargo para poder mostrar correctamente el diagrama lo hemos separado por semanas.

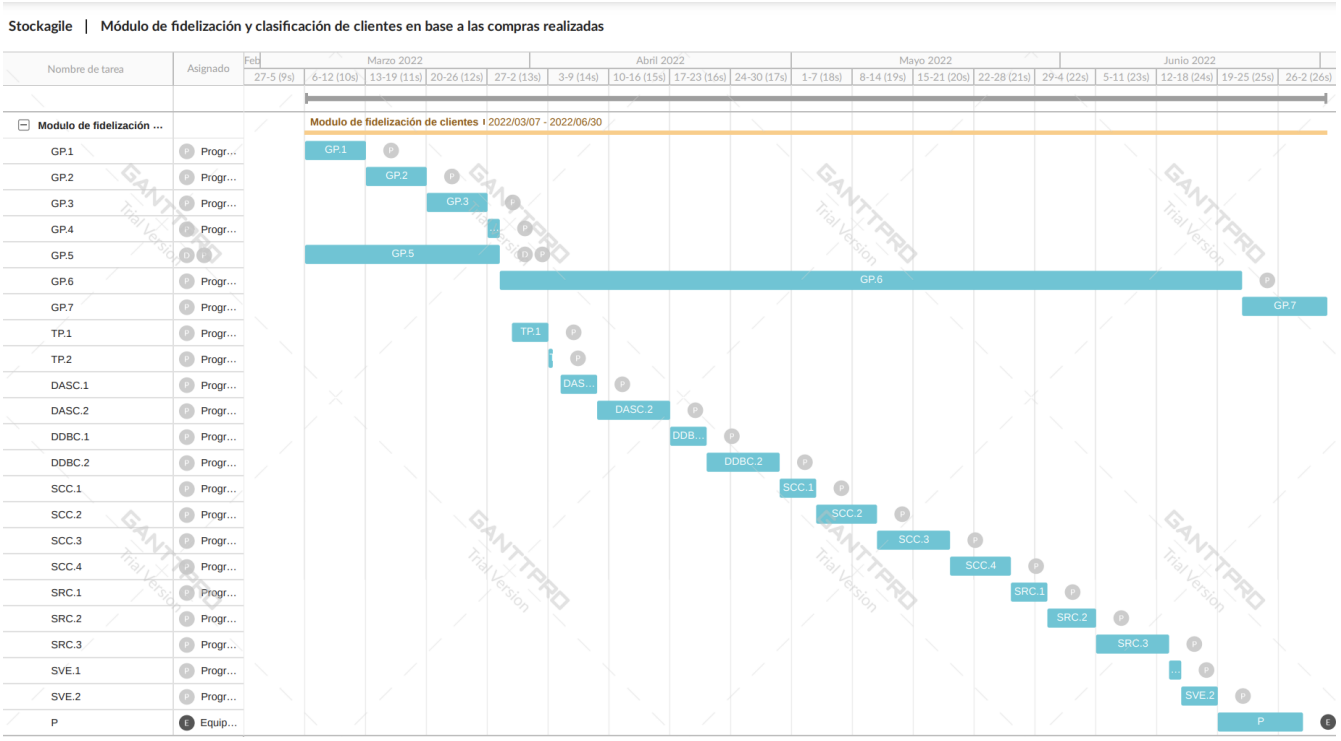


Figura 1: Diagrama de Gantt del proyecto.

9 Tabla de resumen de tareas

Se muestra la tabla resumen de las tareas a realizar en la Tabla 1. La documentación se realizará en paralelo a las tareas, ocupando una hora de la jornada laboral diaria.

Puede parecer que hay un exceso de tiempo dedicado a cada tarea, sin embargo es algo previsto que se comentará en el apartado 11.

Id.	Tarea	Horas	Perfil
GP	**Gestión del proyecto**	**224**	**Programador**
GP.1	Alcance	40	Programador
GP.2	Planificación	32	Programador
GP.3	Presupuesto	24	Programador
GP.4	Informe de sostenibilidad	4	Programador
GP.5	Reuniones	40	Director de proyecto y programador
GP.6	Documentación	60	Programador
GP.7	Presentación	24	Programador
TP	**Trabajo previo**	**29**	**Programador**
TP.1	Estudio del estado del arte	24	Programador
TP.2	Preparación del espacio de trabajo	5	Programador
DASC	**Diseño y desarrollo valor acumulación de saldo**	**68**	**Programador**
DASC.1	Diseño	24	Programador
DASC.2	Desarrollo	44	Programador
DDBC	**Diseño y desarrollo descuento por baremos en compras**	**68**	**Programador**
DDBC.1	Diseño	24	Programador
DDBC.2	Desarrollo	44	Programador
SCC	**Sistema de clasificación de productos**	**176**	**Programador**
SCC.1	Investigación	24	Programador
SCC.2	Preparación del dataset	40	Programador
SCC.3	Entrenamiento del modelo	52	Programador
SCC.4	Adaptación del modelo en el software	60	Programador
SRC	**Sistema de recomendación de clientes**	**124**	**Programador**
SRC.1	Investigación	32	Programador
SRC.2	Diseño	32	Programador

SRC.3	Implementación	60	Programador
SVE	**Sistema de visualización y exportación de datos**	**32**	**Programador**
SVE.1	Diseño	8	Programador
SVE.2	Desarrollo	24	Programador
P	**Pruebas**	**60**	**Grupo de testers y programador**
Duración total del proyecto		**781**	

Tabla 1: Tabla resumen de tareas.

10 Gestión del riesgo

Como se ha mencionado en el apartado 3.4, se prevén cuatro tipos de riesgos que podrían afectar a la duración y a la realización del proyecto: acoplamiento, falta de datos, factores externos y tiempo.

El riesgo que comporta el acoplamiento actual de la plataforma acaba derivando en un riesgo temporal, ya que a partir de pequeños cambios realizados es posible que se generen errores en otros módulos de la aplicación en los que habrá que invertir tiempo para resolverlos.

Para cubrir este riesgo junto al riesgo del tiempo, se ha previsto una estrategia simple pero eficiente que consiste en organizar las tareas teniendo en cuenta que para completar cada una se tardará más tiempo del previsto. En el caso de que aun así el tiempo disponible para finalizar el proyecto no fuera suficiente se reducirá la complejidad de algunas tareas para poder finalizar las tareas troncales a tiempo.

En cuanto al riesgo de la falta de datos, es el que más puede preocupar ya que parece complicado encontrar un dataset que se adapte completamente al problema de este proyecto. En el caso de que no se encontrará una fuente de datos fiable, o no se pudiera extrapolar esta a partir de la información de la que ya se dispone, se han previsto algunos cambios en la tarea que disminuirían su complejidad. Sin un dataset adecuado, no sería posible entrenar de forma efectiva a un modelo de clasificación, por lo que la clasificación de clientes se realizaría de una forma más trivial, como podría ser un algoritmo básico que tuviera en cuenta ciertas características de cada cliente, o hasta se podría dejar esta tarea al usuario de forma que el decidiera que clientes deberían recibir mejores o peores ofertas.

El riesgo de factores externos es algo ambiguo, ya que teniendo en cuenta la variabilidad de una empresa en crecimiento como lo es una start-up, es difícil predecir la cantidad de ruido externo que estará recibiendo en un futuro. El factor más preocupante es que el módulo que se desarrollará en este proyecto deje de ser prioritario para los clientes y no se necesite incorporar al software. Para evitar que esto pueda causar problemas en un futuro se desarrollará todo el módulo en un entorno local y se irán publicando los avances funcionales y necesarios a medida que estén listos y operativos.

11 Presupuesto

11.1 Identificación de costes

Consideramos que los costes del proyecto se pueden clasificar en las categorías siguientes:

- *Personal*: director del proyecto, grupo de testers y programador
- *Desarrollo*: Espacio de trabajo y ordenadores personales
- *Energéticos*: Consumo eléctrico de los ordenadores

11.2 Costes de personal

Como se ha indicado anteriormente el proyecto requiere un director de proyecto, un grupo de testers y un programador. En la Tabla 2 se puede ver el coste por hora de cada rol, los datos se han obtenido de la empresa de reclutamiento *Hays*. [16]

Rol	**Coste por hora**
Director de proyecto	30€/h
Programador	16€/h
Tester	16€/h

Tabla 2: Tabla de costes de personal a partir de la guía de mercado laboral de Hays.

Id.	**Tarea**	**Horas**	**Coste**	**Perfil**
GP	**Gestión del proyecto**	**224**	**4.784**	**DP,P**
GP.1	Alcance	40	640€	P
GP.2	Planificación	32	512€	P
GP.3	Presupuesto	24	384€	P
GP.4	Informe de sostenibilidad	4	64€	P
GP.5	Reuniones	40	1.840€	DP,P
GP.6	Documentación	60	960€	P
GP.7	Presentación	24	384€	P
TP	**Trabajo previo**	**29**	**464€**	**P**
TP.1	Estudio del estado del arte	24	384€	P
TP.2	Preparación del espacio de trabajo	5	80€	P
DASC	**Diseño y desarrollo valor acumulación de saldo**	**68**	**1.088€**	**P**

DASC.1	Diseño	24	384€	P
DASC.2	Desarrollo	44	704€	P
DDBC	**Diseño y desarrollo descuento por baremos en compras**	**68**	**1.088€**	**P**
DDBC.1	Diseño	24	384€	P
DDBC.2	Desarrollo	44	704€	P
SCC	**Sistema de clasificación de clientes**	**176**	**2.816€**	**P**
SCC.1	Investigación	24	384€	P
SCC.2	Preparación del dataset	40	640€	P
SCC.3	Entrenamiento del modelo	52	832€	P
SCC.4	Adaptación del modelo en el software	60	960€	P
SRC	**Sistema de recomendación de clientes**	**124**	**1.984€**	**P**
SRC.1	Investigación	32	512€	P
SRC.2	Diseño	32	512€	P
SRC.3	Implementación	60	960€	P
SVE	**Sistema de visualización y exportación de datos**	**32**	**512€**	**P**
SVE.1	Diseño	8	128€	P
SVE.2	Desarrollo	24	384€	P
P	**Pruebas**	**60**	**960€**	**GT, P**
Total			**13.696€**	

Tabla 3: Tabla total coste personal.

11.3 Costes de desarrollo

El espacio debe poder contener a las seis personas que participarán en el proyecto, junto con los ordenadores y el resto del equipamiento que se utilizará. El proyecto se llevará a cabo en una de las incubadoras de Barcelona Activa, concretamente en la primera planta del 162 de la calle Llacuna, al lado del centro comercial de Glories [17]. El módulo alquilado es de 70 m^2 y dispone de electricidad y aire acondicionado. El precio de alquiler mensual es de aproximadamente 700€ (m2).

En el módulo se instalará un router wifi de gama alta y los participantes del proyecto dispondrán de un ordenador personal, una pantalla, un teclado y un ratón cada uno.

Material	**Coste**
TP-Link Archer C80	54,39€
Pantalla Lenovo	169€
MacBook Pro reacondicionado	900€
Teclado subblim smart designs	49,99€
logitech pebble M350	25,99€
nonda cable usb C a HDMI	20€
Total	**1.219,37**

Tabla 4: Tabla de costes de material tangible (Costes fijos)

11.4 Costes energéticos

Estos son los costes variables del proyecto, es decir costes que variarán dependiendo del tiempo que se dedique al proyecto. Entre estos costes están el coste del módulo en las oficinas de barcelona activa, el coste de la tarifa de WiFi mensual y el coste de los gastos genéricos como podrían ser la comida, los cafés y el material de limpieza.

El coste total se calculará multiplicando el coste mensual por cinco, ya que el proyecto durará cinco meses.

Dado que tanto la luz como el aire acondicionado vienen incluidos en el precio del módulo estos no se tendrán en cuenta. Sin embargo sí sé tendrá en cuenta el coste de la conexión a internet.

Tipo de coste	**Coste por mes**	**Coste total**
Módulo (70m^2)	700€/mes	3.500€
Tarifa WiFi	24€/mes	120€
Costes genéricos	50€/mes	250€
Total		**3.870€**

Tabla 5: Tabla de costes de material intangible (Costes variables)

11.5 Presupuesto final

El presupuesto final se calcula mediante la suma de los costes personales, de desarrollo y energéticos y multiplicando por 1.15, añadiendo un 15% al coste estimado para prevenir posibles obstáculos e imprevistos.

El coste final estimado del proyecto es de 18.785,37€ por lo que si lo multiplicamos por 1,15 obtenemos un presupuesto final de proyecto de 21.603,175€.

12 Sostenibilidad

En todo proyecto es importante realizar un análisis de sostenibilidad teniendo en cuenta tres dimensiones: económica, ambiental y social.

El coste del proyecto no es demasiado alto teniendo en cuenta lo mucho que puede aportar a la empresa un proyecto de este calibre. La gestión del marketing de la empresa es algo completamente necesario sobretodo para pequeñas empresas, por lo que un proyecto como este el cual permitirá generar de forma automatizada el marketing necesario para los clientes de la empresa, aportará un gran valor al producto.

En cuanto al coste ambiental solo se podría tener en cuenta si habláramos del consumo electrónico que aportará esta nueva funcionalidad al software de Stockagile. Teniendo en cuenta el único proceso que realmente podría tener un gran tiempo de ejecución y por lo tanto consumir más energía, es el entrenamiento del modelo y que este solo se va a realizar una vez, no se considerado que el proyecto tenga ningún tipo de impacto en el factor ambiental.

Por otro lado el factor social es quizás el que más se vea afectado ya que sin ir más lejos se está trabajando con la información de muchas personas, y se está utilizando para conseguir vender un producto. Esto se podría considerar poco ético, sin embargo este proceso también hace un bien hacia la persona ya que el uso de su información se utiliza para presentarle un producto que probablemente le interese y quiera adquirir.

13 Identificación de las leyes y regulaciones

Uno de los puntos más importantes de este proyecto consiste en el tratamiento de los datos obtenidos a través de la aplicación de Stockagile. Como se menciona en el apartado de sostenibilidad, el uso de la información de los usuarios para vender y/o recomendarles productos puede ser éticamente cuestionable. Es por esto que es muy importante tener en cuenta todas las leyes y regulaciones que estén relacionadas con la protección de datos.

La normativa que regula la protección de datos a nivel europeo es el Reglamento Europeo de Protección de Datos (RGPD)[19]. Esta normativa tiene como objetivo proteger los derechos de las personas físicas a preservar su información personal y asegurar los derechos recogidos en la Constitución Española. La normativa fue aprobada el 14 de abril de 2016 pero no se implantó de forma efectiva hasta el 2018.

La adaptación de esta normativa en España se ha hecho mediante la Ley Orgánica de Protección de Datos Personales y garantía de los derechos digitales (LOPDGDD)[20]. Esta ley fue consolidada el 5 de diciembre de 2018.

En Stockagile se siguen todos los pasos a seguir en la adaptación del RGPD, informando a los clientes del uso de sus datos, pidiendo su consentimiento, firmando un contrato de confidencialidad con los empleados…

Al ser una startup en contínuo crecimiento, es necesario contemplar todos los posibles casos legales de protección de datos por lo que actualmente, se está trabajando con un equipo de abogados para repasar, mejorar y consolidar todos los puntos relacionados con este tema.

14 Integración de conocimientos

En este proyecto se aplicarán varios de los conocimientos adquiridos en el grado en ingeniería informática. Algunos de estos conocimientos se listan a continuación:

14.1 Aprendizaje automático

En la asignatura de aprendizaje automático (APA), parte de la especialidad de computación, se estudia y practica con distintos modelos de aprendizaje automático. También se estudia los distintos algoritmos que se pueden utilizar para realizar predicciones a partir de unos datos de entrada, o para entrenar al propio modelo para conseguir que sea lo más preciso posible.

En el proyecto se usarán varios modelos de aprendizaje automático para obtener predicciones sobre ciertas características de los clientes, a partir de los datos almacenados en la información. Se utilizarán los conocimientos aprendidos en APA para analizar y estudiar cual de estos modelos se adapta mejor al problema y permite una predicción de datos más adecuada para los usuarios.

14.2 Sistemas de recomendación

En la asignatura de búsqueda y análisis de información masiva (CAIM), se estudiaron en profundidad los sistemas de recomendación, y cómo estos pueden funcionar con distintos modelos de forma que se adapten correctamente al problema.

En el módulo de fidelización de clientes se programará un sistema de recomendación para obtener los productos que se adapten mejor a las necesidades de un cliente. Para hacerlo se usarán los conocimientos aprendidos en CAIM. La intención es programar un modelo de aprendizaje automático el cual se integrará en el sistema de recomendación para que sea adaptable a todas las situaciones con las que se puedan encontrar los usuarios de la aplicación.

14.3 Ingeniería del software

En la asignatura de introducción a la ingeniería del software (IES) se aprende a diseñar un gráfico UML que sea lo más eficiente posible teniendo en cuenta las características y necesidades del software.

En cada una de las tareas de este proyecto es necesario diseñar los UML correspondientes al desarrollo, aplicando todos los conocimientos adquiridos en IES. Es muy necesario que el diseño del módulo sea lo más correcto posible ya que se quiere que sea completamente independiente del resto de módulos.

15 Desarrollo de las funcionalidades básicas del modulo

Durante el desarrollo del proyecto nos hemos encontrado con varios de los obstáculos que se preveían en el apartado 3 de este documento. Debido a esto hemos tenido que cambiar ciertos puntos de la planificación para finalizar el proyecto lo más completo posible y dentro de las fechas estimadas.

En este apartado se describen las tareas realizadas hasta el momento y los obstáculos encontrados durante su desarrollo.

15.1 Trabajo Previo

El primer paso que se realizó para llevar a cabo el proyecto fue preparar el espacio de trabajo. Para una mejor comunicación con el director del proyecto se cambió el puesto de trabajo del desarrollador al lado del director del proyecto. De este modo puede haber una comunicación directa y constante en caso de ser necesaria.

Durante la primera semana se realizó una investigación previa sobre el estado del arte de las técnicas y los sistemas de recomendación para averiguar cuál era el que mejor encajaba con las necesidades del proyecto. Tras estudiar los distintos algoritmos que se utilizan en algunas de las compañías más importantes del mundo en la actualidad, se decidió enfocar el estudio en el sistema de recomendación que utiliza Spotify, ya que se considera uno de los mejores sistemas de recomendación por los usuarios. Además Spotify no cuenta con un sistema de calificación de las canciones, por lo que utiliza otros datos para decidir si al usuario le gusta o no la canción. [18]

Dado que Stockagile tampoco cuenta con calificaciones de los productos ofrecidas por los usuarios, se buscó un método para extraer la información necesaria de los usuarios para definir sus gustos, a partir de los datos de los que sí se consta y empleando un modelo de filtrado colaborativo.

15.2 Desarrollo del valor de acumulación de saldo

Esta etapa del proyecto fue tal y como se esperaba, la duración fue de dos semanas.

Durante la primera semana se diseñó el modelo en UML, teniendo en cuenta todos los factores que podían afectar.

La segunda semana se implementó, añadiendo un campo de acumulación de saldo a nivel de la compañía y otro a nivel del cliente. Así mismo también se implementó un campo que definía si la compañía tiene o no activada la acumulación de saldo. De este modo si el cliente realiza una compra y no tiene una acumulación de saldo específica se le aplica el de la compañía general, siempre y cuando esté activada.

En un diseño inicial del UML se pretendía añadir un campo de activación de acumulación de saldo a nivel de usuario, pero durante el desarrollo se decidió que no era necesario y que sólo complicaría el funcionamiento general.

15.3 Aplicación de la acumulación de saldo en las compras

Durante esta tarea hubo ciertas complicaciones, que concluyeron en un aumento del tiempo de dedicación.

Durante la primera semana se diseñó el UML que establecía como se incorporaría la acumulación de saldo en las compras y como se guardaría el valor del descuento obtenido. Se decidió añadir un campo en la compra donde se guardase simplemente el valor absoluto del descuento y se calculará a partir del saldo acumulado del cliente en cuestión.

La segunda semana se dedicó al desarrollo del diseño anterior. Se desarrolló una interfaz para mostrar de forma entendible para los usuarios como y cuándo podían utilizar el saldo acumulado de los clientes en una compra y se implementó toda la lógica para cumplir los casos de uso. Esto debería haber costado una semana de dedicación, sin embargo durante el desarrollo se concluyó que hacía falta recalcular los impuestos una vez aplicado el descuento para mostrarlos de forma correcta.

Como se mencionó en el punto 3.4, uno de los posibles obstáculos de este proyecto es el acoplamiento de este. En este caso afectó bastante, ya que por el simple hecho de cambiar el cálculo de los impuestos se le dedicó una semana más de tiempo.

15.4 Aplicación de la acumulación de saldo en la facturación

Previamente a este proyecto en la facturación solo se tenía en cuenta un posible descuento porcentual a nivel de la factura completa, por lo que al añadir un descuento de saldo en las funcionalidades del programa era necesario modificar las facturas para que permitieran guardar y calcular el precio total teniendo en cuenta un descuento de un valor absoluto.

Esta modificación iba a suponer una semana de dedicación, pero se alargó hasta 3 semanas. Cuando se empezó a diseñar el UML para esta modificación, se vió que era necesario modificar todo el sistema de facturación, ya que estaba tan acoplado que era más complicado añadir un pequeño cambio que refactorizar todo el módulo.

Se diseñó de nuevo el UML del módulo de factorización y se tuvo en cuenta los posibles cambios que pudiera haber en el futuro.

Una vez empezada la refactorización fue necesario consultar con una asistente financiera cuál era la forma correcta de aplicar los descuentos y calcular el precio total de las facturas a partir de los precios excluyendo impuestos, y se confirmó que era necesario cambiar por completo el algoritmo que se utilizaba, ya que estaba pensado tan sólo para facturas a nivel de mayorista y no de minorista.

15.5 Desarrollo de descuento por baremos en compras

Este desarrollo finalmente no va a formar parte del proyecto.

Tras una de las reuniones bimensuales, se decidió que dado el tiempo perdido en las tareas anteriores no iba a ser posible finalizar el proyecto en el tiempo estimado, por lo que era necesario realizar un cambió en la planificación. Ya que este desarrollo es principalmente independiente de los demás y no es una de las funcionalidades principales del proyecto, se realizará posteriormente a la finalización del proyecto.

De este modo, ha sido posible trabajar en las partes del proyecto más relacionadas con la especialidad de computación.

15.6 Reuniones bimensuales

A lo largo del desarrollo del proyecto se han ido realizando prácticamente todas las reuniones bimensuales. Como ya se comentaba en el apartado 3.4 uno de los mayores obstáculos eran los factores externos. Al ser Stockagile una startup en crecimiento, el director del proyecto tiene muchas otras responsabilidades aparte del proyecto en sí. Por lo que no le ha sido posible asistir a todas las reuniones bimensuales. Sin embargo esto ha sido compensado por un contacto menos oficial pero más dinámico y periodico que ha sido suficiente para ir planificando el proyecto.

16 Desarrollo del sistema de recomendación

Los sistemas de recomendación son herramientas cuya función es, como bien dice su nombre, proporcionar recomendaciones a los usuarios basándose en ciertos criterios y valoraciones que obtiene tras analizar los datos correspondientes a estos.

Los datos de los usuarios pueden obtenerse de forma directa o indirecta. La obtención de datos directa se produce cuando el usuario proporciona los datos específicamente. Por otro lado, la obtención de datos de forma indirecta se da cuando se analizan los datos a partir de las acciones que ha tomado el usuario en la aplicación.

Un buen ejemplo sería el sistema de recomendación de Spotify [21], este utiliza varios tipos de sistema de recomendación, y obtiene los datos tanto de forma directa como indirecta.

Al crear la cuenta, Spotify pide al usuario que seleccione los distintos géneros de música que le pueden interesar y los artistas que le gustan entre otras cosas. Este es un buen ejemplo de obtención de datos directa ya que a partir de esta información podrá recomendarle principalmente la música que corresponda a las características descritas por el usuario.

Spotify también obtiene información de forma indirecta, por ejemplo guarda la información de cuánto tiempo ha tardado el usuario en cambiar una canción, las veces que la ha escuchado… de este modo el sistema se puede hacer una idea de cuánto le gusta la canción al usuario.

Entre los tipos de sistemas de recomendación, hay tres que destacan por encima de los demás [22]:

- **Sistemas de popularidad:** Los sistemas basados en la popularidad consisten en recomendar teniendo en cuenta la popularidad del objeto a recomendar. En el ejemplo anterior un sistema de popularidad recomendaría a los usuarios las canciones más escuchadas del momento, del año, de la historia… Una de las grandes desventajas de los sistemas de popularidad es que no permite personalizar la recomendación para el usuario.
- **Sistemas de contenido:** Los sistemas basados en el contenido utilizan la información del historial de los usuarios para realizar las recomendaciones. En el ejemplo anterior un sistema de contenido mostraría al usuario canciones similares a las escuchadas recientemente o canciones relacionadas (del mismo autor, genero, época…). La mayor desventaja de este sistema es que es poco probable obtener recomendaciones innovadoras ya que todo lo recomendado será similar a lo que ya has escuchado. Además existe cierto problema en definir los criterios para definir una similitud, de lo cual se hablá en el punto 16.3 de este documento.
- **Sistemas colaborativos:** Los sistemas colaborativos o de filtrado colaborativo son bastante innovadores en los sistemas de recomendación ya que no solo aprovechan la información y los datos del usuario al cual se quiere recomendar sino que también hacen uso de la información de un colectivo de usuarios. En el caso de Spotify, este guarda una matriz de información de todos los usuarios de la aplicación con la cual utiliza el filtrado colaborativo para agrupar perfiles similares y ofrecer recomendaciones muy personalizadas. Su desventaja principal es que necesita de una gran cantidad de datos para que sea siempre eficiente y funcional.

En este proyecto se intenta conseguir mezclar estos tres tipos de sistemas de recomendación para cubrir con las ventajas de uno, las desventajas del resto.

16.1 Sistema de clasificación de productos

En un principio se planeaba realizar un sistema de clasificación de productos para luego utilizar esta clasificación en pro del sistema de recomendación. La intención era entrenar un modelo de aprendizaje automático para que distinguiera distintas clases entre los productos y pudiera clasificar otros que se fueran añadiendo a la tienda.

Esta idea se ha descartado por varias razones:

- Los datos que dan información sobre los productos, son proporcionados por los usuarios del sistema en el momento de importar los productos al programa. Esto podría influir mucho en el entrenamiento del modelo, ya que varios de los productos podrían tener información corrupta, estar mal categorizados, etiquetados, tener un precio erróneo…
- El modelo requiere de una supervisión humana. Es decir alguna forma de indicarle durante el entrenamiento si está realizando las predicciones correctas o no. Hacerlo de forma manual no era una opción debido a la gran cantidad de datos y dado a la gran cantidad de datos corruptos tampoco era posible automatizarlo.
- La clasificación de los productos tan solo permitía definir una similitud entre estos, lo cual permitiría recomendar a un cliente productos similares a los que ya ha comprado. Sin embargo se decidió que en una situación real, el cliente no siempre busca comprar cosas similares a las que ya ha adquirido. Un ejemplo sería un cliente que se compra un televisor, no tiene mucho sentido recomendarle otro televisor ya que lo más probable es que el primero ya cumpla sus necesidades.

16.2 Red neuronal para estimar la posibilidad de compra de un producto

Tras descartar la idea inicial se realizó una lluvia de ideas para concluir en que debería basarse el sistema de recomendación. Después de hablarlo con varios miembros del equipo, incluyendo al director, se acordó que el sistema debería estimar qué es lo que un cliente querría comprar teniendo en cuenta los productos que ya ha adquirido. Por ejemplo, si un cliente compra un móvil, lo más probable es que necesite una funda de teléfono.

Para conseguir que el sistema de recomendación cumpliera con este objetivo se decidió programar una red neuronal.

16.2.1 Definición de una red neuronal

Las redes neuronales, son un tipo de modelo de aprendizaje automático que intenta imitar el comportamiento de una red neuronal del cerebro humano.

Su componente principal son lo que llamamos neuronas. Estas, al igual que las de una red neuronal humana, reciben ciertos datos de entrada y los transforman de distintas formas enviando estos nuevos datos de salida.

Las neuronas se distribuyen en distintas capas, las cuales tienen diferentes funciones y características.

Siempre disponen de una capa de entrada o "*input layer*" y de una capa de salida o "*output layer*" las cuales se encargaran de recibir los datos brutos y de realizar las transformaciones necesarias para obtener unos datos de salida con las dimensiones especificadas.

Por otro lado, entre estas dos capas se encuentran las llamadas capas ocultas o "*hidden layers*". Estas capas se encargan de transformar los datos recibidos y mandarlos a la siguiente capa siguiendo el criterio de la función de activación que les haya sido especificada.

Las funciones de activación se especifican por capas y son las que definen la salida de una neurona dada una entrada o un conjunto de entradas.

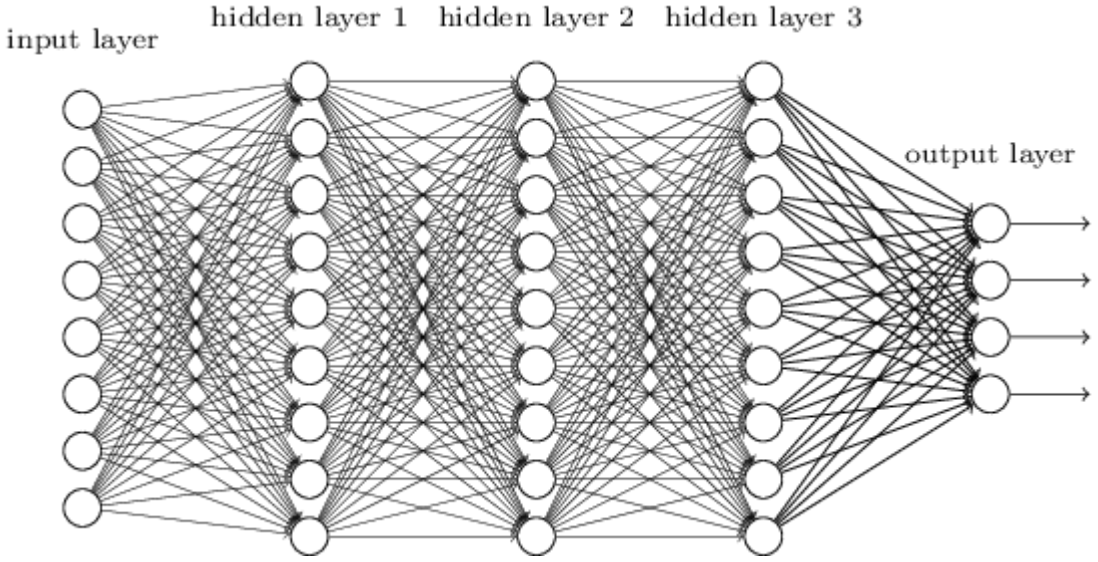


Figura 1: Esquema de una red neuronal

En la Figura 1, se muestra el esquema de una red neuronal compuesta por cinco capas: una capa de entrada, tres capas ocultas y una capa de salida. Los nodos representan las neuronas de cada capa. Se puede ver que la capa de entrada está formada por ocho neuronas, las capas ocultas por nueve y la de salida por cuatro. Las aristas representan las conexiones entre las neuronas de distintas capas. En este caso cada neurona está conectada con todas las de la siguiente capa y las de la anterior.

16.2.2 Funcionamiento de una red neuronal

La red neuronal se entrena a partir de un dataset compuesto por miles de datos de entrada y de salida. El objetivo de la red neuronal es que a partir de una entrada específica obtenga la salida que le corresponde a esta, también busca que esta salida sea lo más cercana a la real. Un ejemplo sería querer obtener el peso de una persona a partir de varias de sus características, como por ejemplo la edad, altura y anchura… En este caso los datos de entrada serían las características de la persona y el único dato de salida sería el peso de esta.

Para que el modelo pueda entrenarse y acabar prediciendo de la forma más certera posible, es necesario entrenarlo con mucha información real que conste tanto de los datos de salida como de los de entrada. En el ejemplo dado antes, se entrenaría a la red neuronal con un dataset compuesto de miles de instancias de personas donde cada instancia tendría la información tanto de las características de entrada, como de su peso. El conjunto de instancias que contienen tanto las variables de entrada como las de salida, con el que se entrenará y se validará el modelo, es llamado conjunto de datos o *dataset*.

El proceso de entrenamiento de la red neuronal, empieza por dividir el conjunto de datos en un conjunto para el entrenamiento, y otro para la validación del modelo.

Seguidamente se hace que el modelo recorra las instancias del conjunto de datos de entrenamiento una a una e intente predecir el peso que le corresponde a cada instancia. El peso que obtenga la red, será comparado después con el peso real de esta instancia, y a partir de estos dos, el modelo calculará el error cometido. Tras obtener el error, el modelo utiliza un proceso llamado *backtrack propagation* que consiste en ir ajustando los parámetros de activación de las capas, empezando desde la más cercana a la capa de salida y terminando en la primera capa, para que el error obtenido se minimice.

La ejecución de este proceso para todas las instancias de entrenamiento es llamada *epoch*. Por lo tanto si el modelo repite el proceso tres veces para todas las instancias se dice que se ha entrenado con tres *epoch*.

El mismo modelo puede ser entrenado con distintos *epoch* y esto produciría normalmente resultados distintos. Por ejemplo, en el caso de utilizar demasiados *epoch* para entrenar el modelo, este podría adaptarse tan bien a los datos de entrenamiento, que funcione perfectamente para estos, y que para cualquier entrada del conjunto obtenga el valor esperado para la salida. Peró lo más probable es que si se probara el modelo con una instancia completamente nueva distinta a todas con las que se ha entrenado, no obtenga el resultado esperado. En este caso se dice que el modelo está sobre ajustado a los datos de entrenamiento. Por otro lado, si solo se entrenara el modelo con unos pocos *epoch*, es probable que este no haya sido capaz de aprender lo suficiente del conjunto de datos, por lo que no obtendría buenos resultados para el conjunto de entrenamiento.

Por esto mismo es muy importante el conjunto de datos de validación.

Una vez el modelo ha sido entrenado con los datos de entrenamiento, se utilizan los datos de validación, los cuales son completamente nuevos para el modelo, para comprobar cómo funciona este con datos desconocidos.

16.2.3 Aplicación de la red neuronal (Clientes/Productos)

Para implementar una red neuronal a un problema real, lo más importante es definir los objetivos. En este caso el objetivo es conseguir una red neuronal a la cual tras entrenarla con la información de las compras realizadas por los clientes, pueda predecir la posibilidad que tiene un cliente de comprar un producto en concreto.

Para hacerlo es necesario pensar cómo estructurar el conjunto de datos que se utilizará para entrenar el modelo, es decir, establecer cuáles serán los parámetros de salida y los de entrada.

En un primer intento de entrenar el modelo, se decidió que la mejor forma de alimentarlo con la información de todas las compras realizadas por los clientes era generar un archivo CSV (Comma Separated Values) en el cual las filas representaran los clientes y las columnas representaran los productos. De este modo la celda posicionada en la fila "*x"* y en la columna "*y"* contendrá el número de veces que el cliente al que pertenece la fila *"x"* ha comprado el producto al que pertenece la columna *"y"*.

En la *Tabla 1* se muestra un ejemplo de cómo se estructuraría el archivo CSV. En este caso se puede ver que el cliente con identificador 1 ha comprado zero veces el producto con identificador 1, una vez el producto con identificador 2 y dos veces el producto con identificador 3. El cliente con

identificador 2 sin embargo, ha comprado una vez el producto con identificador 1, una vez el producto con identificador 2 y zero veces el producto con identificador 3.

		Id Productos		
		1	**2**	**3**
Id Cliente	**1**	0	1	2
	2	1	1	0

Tabla 1: Ejemplo de un conjunto de datos generado como tabla de Clientes/Productos

Al estructurar los datos de esta forma, se puede utilizar cualquiera de las columnas como datos de salida y el resto de ellas como datos de entrada, para poder predecir así que posibilidad tiene un cliente de comprar un producto en base a los productos que ya ha comprado.

Utilizando el ejemplo de la *Tabla 1*, si se quisiera predecir la posibilidad que tienen los clientes de comprar el producto con identificador 3, se usarían las columnas con identificadores 1 y 2 como datos de entrada y la columna con identificador 3 como datos de salida.

El problema de esta estructura de datos es que si el modelo fuera perfecto, y predijese con una precisión del 100% todas las veces, no serviría para nuestro caso real, ya que, lo que estaría diciendo es si el cliente ya ha comprado ese producto o no.

Sin embargo si el modelo no fuera perfecto, entonces en los casos en los que se le pasara información de un cliente que no ha comprado el producto, podría predecir que ese cliente tiene una posibilidad alta de haber comprado el producto, y en este caso significa que el cliente ha comprado objetos relacionados con el producto y que por lo tanto lo más normal sería que también hubiese comprado el producto. Por lo que en este caso el modelo estaría prediciendo que el cliente sería propenso a comprar el producto teniendo en cuenta las compras que ya ha realizado, lo cual es el objetivo que se quería cumplir.

Es por eso que con este modelo realmente no se está buscando que sea una red neuronal precisa y perfecta que tenga una precisión del 100% en todos los casos, lo que se busca es que prediga correctamente todos los casos en los que el cliente ha comprado el producto peró también se equivoque en algunos de los casos en los que el cliente no ha comprado el producto.

16.2.4 Aplicación de la red neuronal (Ventas/Productos)

Estructurar el conjunto de datos de la forma Cliente/Producto, a pesar de ser un formato funcional con el que se cubrían los objetivos, tenía algunas desventajas, y una de estas era que se perdía mucha información relevante.

El software permite que un cliente que ha hecho una compra, no quede registrado en el sistema, de modo que hay es muy probable que para cada empresa existan una gran cantidad de compras sin ningún cliente vinculado, ya que por lo general, uno no se registra en el software de una tienda en la que ha comprado si no és o va a ser un cliente habitual.

Al caer en la cuenta de esta gran desventaja se vió que una gran cantidad de ventas, las cuales relacionaban productos entre ellos, no estaban siendo utilizadas para entrenar ni validar el modelo.

Después de darle varias vueltas a las posibles soluciones, se decidió que la mejor manera de tener toda esta información en cuenta sin perder la funcionalidad anterior del modelo era estructurar el conjunto de datos de forma distinta.

En vez de estructurar el conjunto de datos de forma que cada fila correspondiese a un cliente, se estructuró de forma que cada fila correspondía a una venta y las columnas seguían correspondiendo a los productos. De este modo la celda posicionada en la fila *"x"* y en la columna *"y"* contendrá el número de unidades del producto al que pertenece la columna *"y"* han sido vendidas en la venta a la que pertenece la fila *"x"*.

En la *Tabla 2* se muestra un ejemplo de cómo se estructuraría el archivo CSV. En este caso se puede ver que en la venta con identificador 1 se ha vendido una unidad del producto con identificador 1, zero unidades del producto con identificador 2 y una unidad del producto con identificador 3. En la venta con identificador 2 sin embargo, se ha vendido una unidad del producto con identificador 1, zero unidades del producto con identificador 2 y zero unidades del producto con identificador 3.

		Id Productos		
		1	**2**	**3**
Id Venta	**1**	1	0	1
	2	1	0	0

Tabla 2: Ejemplo de un conjunto de datos generado como tabla de Ventas/Productos

Al estructurar el conjunto de datos de este modo, se conserva la información de todas las ventas realizadas por el usuario y no solo de las ventas asignadas a un cliente registrado.

Sin embargo, esta estructura sólo se usará para entrenar el modelo, ya que lo único que interesa es que la red neuronal aprenda de algún modo a relacionar los productos entre ellos en base a si se han vendido de forma conjunta o no. La validación del modelo se realizará con datos estructurados de la forma Cliente/Producto, para que de este modo podamos obtener el resultado esperado.

Las instancias de los datos de validación serán los productos comprados por cada cliente, ya que, como en las dos estructuras las columnas representan todas las instancias de productos de la empresa, habrá el mismo número de columnas.

16.2.5 Integración de la red neuronal en el sistema de recomendación

Una vez obtenida una red neuronal eficiente con los resultados esperados se pudo usar para obtener la información necesaria para el sistema de recomendación.

Para que el sistema de recomendación sea eficiente fue necesario obtener un valor que muestre la relación entre cada uno de los clientes y cada uno de los productos.

Para ello se realizó un entrenamiento del modelo, usando como valores de salida cada uno de los productos y se guardaron los resultados obtenidos formando una matriz, la cual se ha llamado matriz de afinidad, entre clientes y productos donde cada casilla *(x, y)* tendrá un valor entre 0 y 1 que corresponderá a la posibilidad de que el cliente al que corresponde la fila *x* quiera comprar el producto que corresponde a la columna *y*.

Una vez obtenida esta matriz, el sistema de recomendación puede encontrar N clientes a los que recomendar un producto en 3 pasos:

- Obtener la columna del producto
- Ordenar todos los valores de mayor a menor
- Seleccionar los N primeros.

Del mismo modo también es posible encontrar los N productos que más le pueden interesar a un cliente cambiando tan solo el primer paso por "obtener la fila del cliente".

Esta matriz, debería ser generada para cada uno de los usuarios de Stockagile y debería ser independiente entre compañías, ya que por ahora, los productos y los clientes de diferentes compañías siempre son distintos a nivel de datos.

16.3 Algoritmo de similitud entre productos

Después de obtener una red neuronal funcional y eficiente, se notó en los experimentos que varios productos obtenían un valor de 0 en todas las celdas de su columna en la matriz de afinidad.

Al estudiar el motivo de este suceso se descubrió que tales productos no tenían ninguna venta por el momento, por lo que la red neuronal no tenía información a partir de la que relacionarlos con otros productos. Para poder solucionar este problema se decidió establecer un algoritmo de similitud entre productos para poder usar el sistema de filtrado colaborativo en los productos que no obtuvieran buenos resultados con la red neuronal.

Sin embargo existe un problema al definir la similitud entre dos productos de forma automatizada, y es que la similitud entre objetos es un valor subjetivo. Este problema es mayor en el caso de Stockagile ya que los usuarios tienen todo tipo de rangos de productos, desde muebles hasta cepillos de dientes y la mayoría de estos datos son introducidos por ellos mismos en el programa. Es decir que para solucionar este problema lo ideal sería que los usuarios también influyeran de cierta forma en como obtener el valor de similitud.

Para ello se decidió desarrollar un algoritmo que dados dos productos calculase la similitud entre ellos a partir de un vector de ponderaciones elegidas por el usuario para tener en cuenta si es necesario dar más importancia a las categorías de los productos que a las marcas por ejemplo.

	Mocasines	Sandalias
Marca A		
Marca B		

Tabla 3: Productos clasificados por marca y categoría

En la *Tabla 3* se muestran cuatro cuatro productos los cuales obtendrían similitudes completamente distintas dependiendo de si la ponderación fuera mayor en la marca o en la categoría. En este caso, se puede ver claramente que el usuario debería dar mayor ponderación a la categoría, ya que dentro de una misma marca hay productos muy variados.

Permitiendo que el usuario decida a qué características y atributos dar mayor ponderación, el mismo puede definir si la similaridad entre productos radica en su marca, su etiqueta, el número de variantes que tienen… De este modo el sistema de recomendación puede ser mucho más personalizable.

17 Experimentación

En este proyecto se ha realizado una experimentación e investigación para conseguir la eficiencia y funcionalidad del sistema de recomendación.

17.1 Entrenamiento y validación de la red neuronal

Para el correcto entrenamiento y análisis del modelo se han usado varias librerías de python. El uso de las cuales se describe a continuación:

- sklearn [23]: Esta librería se ha usado para obtener y mostrar las métricas y los resultados del modelo de forma clara y entendible.
- keras [24]: Consiste en una API que permite acceder tanto a distintos modelos de aprendizaje automático, como a sus parámetros y funciones para que puedan ser correctamente compilados y usados.
- matplotlib [25]: Matplotlib es una librería que permite mostrar datos en gráficas de forma ordenada y entendible para el usuario
- numpy [26]: Esta librería permite hacer cálculos y operaciones para distintas estructuras de datos y es especialmente útil para grandes cantidades de datos.
- pandas [27]: Pandas al igual que numpy es una librería especializada en el manejo y análisis de datos.

La red neuronal consta de cinco capas de neuronas:

- La capa de entrada que cuenta con el mismo número de neuronas que el número de productos que tenga el conjunto de datos.
- Tres capas ocultas con la mitad de neuronas que el número total de productos del conjunto de datos cada una y que usan la función de activación relu [28].
- Una capa de salida que cuenta con una neurona y usa una función de activación sigmoide [29].

Tras experimentar varias veces se concluyó que los mejores resultados se obtenían al poner N/2 neuronas en las capas ocultas del modelo, donde N es el número de productos que contiene el conjunto de datos.

Para compilar el modelo se utilizó el algoritmo de *adam* [30] tras ver que era el que obtenía los tiempos de ejecución más pequeños. También se decidió utilizar la función de pérdida [31] llamada *binary_crossentropy* [32].

```
input = keras.Input(shape=(len(x_train[0]),1))
flatten_layer = keras.layers.Flatten()(input)
dense1_layer = keras.layers.Dense(units = int(n_products/2), activation='relu')(flatten_layer)
dense2_layer = keras.layers.Dense(units = int(n_products/2), activation='relu')(dense1_layer)
dense3_layer = keras.layers.Dense(units = int(n_products/2), activation='relu')(dense2_layer)
output_layer = keras.layers.Dense(1, activation='sigmoid')(dense3_layer)
model_nn = keras.Model(input, output_layer)

model_nn.compile(optimizer='adam',
          loss='binary_crossentropy',
          metrics=['accuracy', 'Recall'])
model_nn.summary()
```

Figura 2: Código de la compilación del modelo

Una vez compilado el modelo era necesario entrenarlo y evaluar los resultados obtenidos para poder establecer si era funcional.

Como se ha mencionado en el apartado 16.2, el modelo se ha entrenado con dos estructuras de conjuntos de datos distintos. Por ello el entrenamiento y la evaluación del modelo también ha sido un poco distinta para cada uno de estos.

17.1.1 Estructura Cliente/Productos

Para esta estructura se han analizado los conjuntos de datos de 4 empresas distintas y para cada una de estas se ha entrenado el modelo con el objetivo de predecir la afinidad de los clientes para dos productos distintos.

Al utilizar esta estructura para el conjunto de datos, era necesario dividirlo en un conjunto de entrenamiento y un conjunto de validación, así que se utilizó un 70% de los datos para entrenar al modelo y el 30% restante para validar el modelo entrenado.

El modelo se entrenó con el número de epochs que mejor se adaptase a la empresa y al producto que estuviera analizando por lo que en algunos de los casos se entrenó con 10 epochs y en otros con 3.

Los resultados obtenidos en general fueron muy satisfactorios para todas las empresas.

En las *Figuras 3* y *4* se muestran las matrices de confusión obtenidas de las predicciones del modelo junto a los datos de validación. La matriz de confusión muestra de forma gráfica cuáles de las instancias que el modelo ha predecido como clase 1 eran realmente clase 1 y cuáles no, y también cuáles de las instancias que el modelo ha predecido como clase 0 lo eran y cuáles no. En este caso la clase 1 representa que el cliente compraría el producto y la clase 0 representa lo contrario.

Realmente el resultado que obtiene el modelo es un valor entre 0 y 1 que representa la probabilidad de que el cliente compre el producto según el modelo, así que para poder representar estas gráficas, se ha establecido un baremo con el cuál se establece que todo cliente que haya obtenido una probabilidad mayor que 0.8 se clasifica como clase 1 y de lo contrario se clasifica como clase 0.

Dicho de otro modo la celda superior izquierda de la matriz nos muestra el número de clientes que no tienen el producto y que el modelo ha considerado que no comprarían el producto. La casilla superior derecha representa los clientes que no tienen el producto pero que el modelo considera que si lo comprarían. La casilla inferior izquierda representa los clientes que sí han adquirido el producto y que el modelo considera que no lo comprarían. Finalmente la casilla inferior derecha representa los clientes que sí tienen el producto y que el modelo considera que sí lo comprarían.

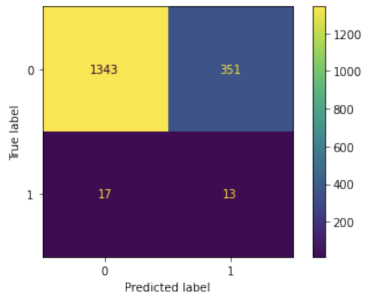
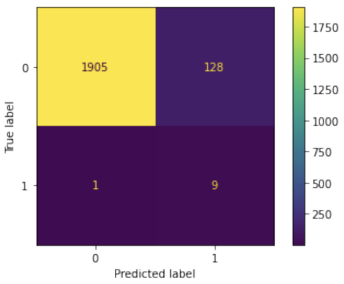
Empresa A	Empresa B
	

Figura 3: Matriz de confusión de las empresas A y B

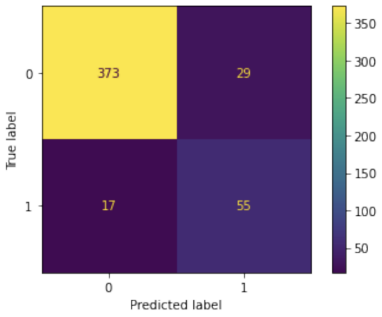
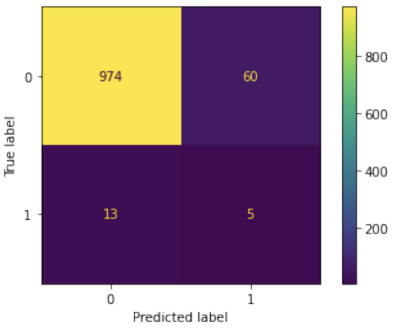
Empresa C	Empresa D
	

Figura 4: Matriz de confusión de las empresas C y D

En estas matrices se puede ver de forma muy clara que el modelo está funcionando de la forma esperada. Esto se debe a que no solo está obteniendo una gran cantidad de aciertos en lo que respecta a los clientes que ya han adquirido el producto y una cantidad muy notable de aciertos en los clientes que no tienen el producto, sino que también está prediciendo una cantidad considerable de clientes que no tienen el producto pero que él considera que sí deberían adquirirlo.

A parte de estas matrices también se han obtenido otras métricas con las que se puede estimar una buena funcionalidad del modelo (el resto de métricas así como los códigos fuente se pueden encontrar en los documentos adjuntos).

Los clientes representados por la casilla superior derecha son los clientes a los que el sistema de recomendación debería ofrecer el producto.

17.1.2 Estructura Ventas/Productos

Para esta estructura solo se ha podido analizar una empresa debido a la falta de tiempo.

En este caso no era necesario dividir el conjunto de datos en datos de entrenamiento y datos de validación, ya que para los datos de entrenamiento se iba a usar el conjunto de datos con la estructura Ventas/Productos y para los datos de validación, el conjunto con la estructura Clientes/Productos.

A pesar de esto se repitió el mismo procedimiento que con la estructura anterior para poder comparar los resultados. Primero se utilizó un 70% del conjunto de datos para el entrenamiento del modelo y un 30% para su validación y se obtuvieron los siguientes resultados.

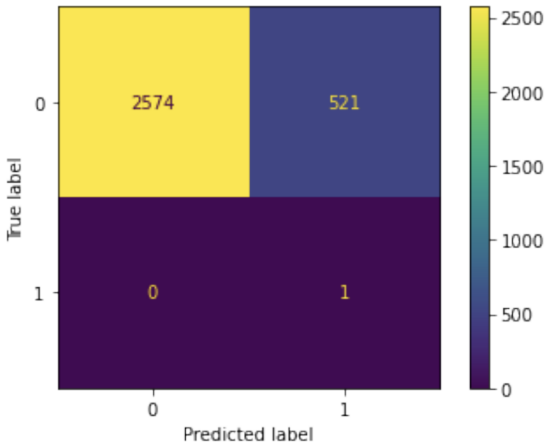
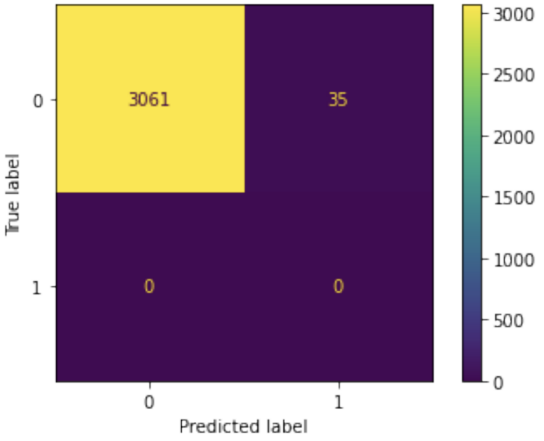
Producto 1	**Producto 2**
	

Figura 5:

Después se volvió a entrenar al modelo, pero esta vez utilizando el conjunto de datos con estructura Ventas/Productos entero y el conjunto de datos con estructura Clientes/Productos para validarlo obteniendo así los siguientes resultados.

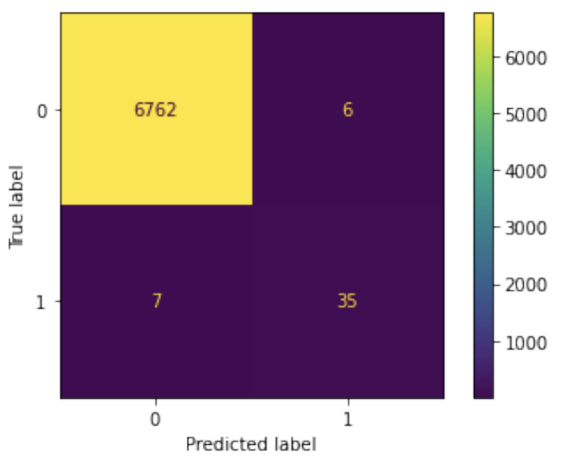
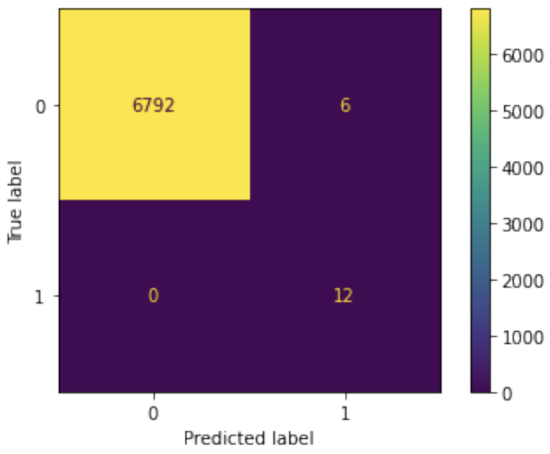
Producto 1	**Producto 2**
	

Figura 6:

Como se puede ver en la *Figura 6* los resultados son hasta mejores que con la estructura Clientes/Productos ya que hay mucha más precisión en las predicciones. A pesar de que en la casilla superior derecha solo se muestren 6 posibles clientes a quien hacer la recomendación, cabe recordar que las clases 1 y 0 están definidas por un baremo de 0.8. Es decir que los 6 clientes que el modelo ha clasificado como posibles compradores, los cuales no tienen aún el producto, considera que tienen más de un 80% de posibilidades de comprarlo.

18 Conclusiones

Para finalizar este proyecto es necesario hacer una autocrítica a mí mismo y al desarrollo de este. Se extraerán también las conclusiones a las que se ha llegado a través de todo el proceso y se comentarán los fallos y obstáculos que han dificultado el avance y la resolución del proyecto. Por último se comentarán los posibles cambios que se podrían haber hecho desde un principio para mejorar el resultado final.

El camino recorrido durante este proyecto ha sido largo y difícil, empecé con muchas ganas de realizar un proyecto vinculado con la inteligencia artificial y el aprendizaje automático ya que es uno de los campos que más me apasionan, sin embargo me he encontrado en más ocasiones peleandome con las funcionalidades del software de la empresa que con el funcionamiento de la red neuronal. Esto ha sido principalmente un error mío, ya que en su momento quise abarcar demasiado para poder ofrecer a la empresa tanta ayuda como fuera posible y esto me llevó a estancarme durante semanas en cosas que eran de mínima importancia para el objetivo principal del proyecto.

Por otro lado, muchos de estos problemas con los que me he ido encontrando, estaban en mi punto de mira desde el principio y ya había planificado como solucionarlos. Y aunque no todo ha salido como esperaba creo que he logrado los objetivos principales del proyecto dejando unas bases preparadas para que el módulo de fidelización de cliente pueda estar al alcance de nuestros clientes para finales de verano.

En el ámbito de la inteligencia artificial y el aprendizaje automático, este primer proyecto dará paso a muchos otros en Stockagile, los cuales no solo partirán de las bases que he ido desarrollando durante el tiempo de I+D dedicado sino que llegarán a ser mucho mejores, por lo que creo que finalmente este proyecto será una gran aportación a la empresa.

Como he mencionado antes este proyecto ha intentado abarcar demasiadas subtareas de las cuales no se han podido llegar a completar todas. El cambio principal que realizaría en el proyecto para obtener un mejor resultado sería abordar tan solo el sistema de recomendación, ya que no solo es la parte del proyecto que más me interesaba, sino que también es la parte que más se adapta a mi especialidad y a pesar de eso ha sido la parte a la que menos tiempo le he podido dedicar.

Teniendo en cuenta lo mencionado anteriormente, no creo que este proyecto haya sido un fracaso ni mucho menos. Se han creado las bases para el módulo de fidelización de clientes del software, se ha adaptado, modificado y arreglado el código relacionado con las facturas a nivel de minorista y mayorista, y se ha desarrollado un sistema de recomendación que hace uso de una red neuronal para predecir el comportamiento de los clientes de distintas empresas y que se adapta a cualquiera de las empresas clientes de Stockagile.

Creo que es un proyecto sin acabar y que con un poco más de mano de obra y tiempo podría llegar a ser mucho mejor, pero creo que ha sido un completo logro.

19 Oportunidades de futuro

Desde un inicio este proyecto no era más que el comienzo de un nuevo campo en la empresa de Stockagile.

No solo se ha empezado a crear un nuevo módulo que puede llegar a incluir miles de nuevas funcionalidades, sinó que también se ha abierto las puertas al uso de la inteligencia artificial para toda clase de funcionalidades del sistema.

El uso del aprendizaje automático en stockagile podría crecer exponencialmente. Se podrían utilizar modelos para predecir qué productos debes comprar en ciertas épocas del año, basándose en las compras realizadas con anterioridad.

También se ha planteado la idea de relacionar de alguna manera la información de todos los usuarios de Stockagile (Clientes, Productos, Marcas…) y utilizarla de forma conjunta. De este modo nuevas empresas que entren con Stockagile y no tengan información en el sistema tendrían mucho más fácil el desarrollo.

Dado a que Stockagile es una startup en crecimiento, permite que las oportunidades relacionadas con este proyecto sean infinitas. Cuanto más crezca la empresa, más información tendrá y más recursos podrá dirigir al campo de la inteligencia artificial.

20 Referencias

[1] (2021).*Wikipedia. Angular (framework)*. Retrieved 25 February 2022, from https://es.wikipedia.org/wiki/Angular_(framework)

[2] (1998-2022). *mdn web docs. ¿Qué es JavaScript?*. Retrieved 25 February 2022, from https://developer.mozilla.org/es/docs/Learn/JavaScript/First_steps/What_is_JavaScript

[3] (2005-2022). *Django*. Retrieved 25 February 2022, from https://www.djangoproject.com/

[4] (2022). *Wikipedia. Python*. Retrieved 25 February 2022, from https://es.wikipedia.org/wiki/Python

[5] (1996-2022). *PostgreSQL*. Retrieved 25 February 2022, from https://www.postgresql.org/

[6] (2022). *Jenkins*. Retrieved 27 February 2022, from https://www.jenkins.io/

[7] (2022). *Elastic. ¿Qué es Kibana?*. Retrieved 27 February 2022, from https://www.elastic.co/es/what-is/kibana

[8] (2022). *¿Qué es Docker? ¿Que son los contenedores? y ¿Por que no usar VMs?*. Retrieved 28 February 2022, from https://platzi.com/contributions/guia-del-curso-de-docker/?utm_source=google&utm_medium=paid&utm_campaign=14603491644&utm_adgroup=&utm_content=&gclid=CjwKCAiApfeQBhAUEiwA7K_UH9H0i5AZTAoaNJ4Xdw842ukXuchaKllbdj8VP0Qcp6pZ8R6C9yLblRoCA-wQAvD_BwE&gclsrc=aw.ds

[9] (2022). *AWS | Cloud Computing - Servicios de informática en la nube.* Retrieved 2 March 2022, from https://aws.amazon.com/es/

[10] (2022). ¿Qué es un sistema ERP y para qué sirve?. Retrieved 2 March 2022, from https://www.ticportal.es/temas/enterprise-resource-planning/que-es-sistema-erp

[11] Eduardo Viteri Luque, F., Alemán Herrera Lozano, L., & Fernando Bazurto Quiroz, A. (2018). *Importancia de las Técnicas del Marketing Digital*. Retrieved February 23, 2022, from https://dialnet.unirioja.es/servlet/articulo?codigo=6732914

[12] (2022). *Google Ads - Crea anuncios online fácilmente y consigue más clientes.* Retrieved 2 March 2022, from https://ads.google.com/intl/es_es/home/

[13] (2022). *Three Insights On the Business Opportunity for the Metaverse.* Retrieved 2 March 2022, from https://www.facebook.com/business/news/metaverse-business-opportunity-insights

[14] (2022). *Infografía: El imperio Meta.* Retrieved 2 March 2022, from https://es.statista.com/grafico/25917/usuarios-activos-mensuales-de-redes-sociales-y-servicios-de-mensajeria-de-facebook/#:~:text=Como%20muestra%20nuestro%20gr%C3%A1fico%20de,una%20de%20estas%20plataformas%20a

[15] (2022). *How Spotify Recommends Your New Favorite Artist.* Retrieved 2 March 2022, from https://towardsdatascience.com/how-spotify-recommends-your-new-favorite-artist-8c1850512af0#:~:text=Well%2C%20Spotify's%20recommender%20system%20provides,it%20deems%20%E2%80%9Csimilar%E2%80%9D%20users

[16] (2022) *Incubadora Glòries*. Retrieved 10 March 2022, from https://emprenedoria.barcelonactiva.cat/web/guest/incubadora-glories

[17] (2022) *Home*. Retrieved 18 March 2022, from https://www.hays.es/

[18] (2021) *Cómo Funcionan los Algoritmos de Recomendación en Spotify*. Retrieved 12 May 2022, from https://promocionmusical.es/como-funcionan-algoritmos-recomendacion-spotify

[19] (2022) *Reglamento General de Protección de Datos y listado de empresas de protección de datos*. Retrieved 26 May 2022, from https://rgpd.es/

[20] (2021) *Ley Orgánica de Protección de Datos Personales y garantía de los derechos digitales.* Retrieved 26 May 2022, from https://www.boe.es/buscar/act.php?id=BOE-A-2018-16673

[21] (2022) *Cómo Funcionan los Algoritmos de Recomendación en Spotify*. Retrieved 21 June 2022, from https://promocionmusical.es/como-funcionan-algoritmos-recomendacion-spotify

[22] (2022) *Sistemas de recomendación | Qué son, tipos y ejemplos*. Retrieved 21 June 2022, from https://www.grapheverywhere.com/sistemas-de-recomendacion-que-son-tipos-y-ejemplos/

[23] (2022) scikit-learn. Machine Learning in Python. Retrieved 31 May 2022 from https://scikit-learn.org/stable/

[24] (2019) *Your First Deep Learning Project in Python with Keras Step-By-Step.* Retrieved 31 May 2022, from https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/

[25] (2022) *Matplotlib: Visualization with Python*. Retrieved 31 May 2022 from https://matplotlib.org/

[26] (2022) *NumPy*. Retrieved 31 May 2022, from https://numpy.org/

[27] (2022) *Pandas documentation.* Retrieved 31 May 2022, from https://pandas.pydata.org/docs/

[28] (2019) *A Gentle Introduction to the Rectified Linear Unit (ReLU).* Retrieved 31 May 2022, from https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

[29] (2022) *Sigmoid function.* Retrieved 31 May 2022 from https://en.wikipedia.org/wiki/Sigmoid_function

[30] (2019) *Gentle Introduction to the Adam Optimization Algorithm for Deep Learning.* Retrieved 31 May 2022, from https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

[31] (2019) *How to Choose Loss Functions When Training Deep Learning Neural Networks.* Retrieved 31 May 2022, from https://machinelearningmastery.com/how-to-choose-loss-functions-when-training-deep-learning-neural-networks/

[32] (2018) *Understanding binary cross-entropy / log loss: a visual explanation.* Retrieved 22 May 2022, from

https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a