Facultade de Informática
UNIVERSIDADE DA CORUÑA

TRABALLO FIN DE GRAO
GRAO EN CIENCIA E ENXEÑARÍA DE DATOS

Eliminación de batch-effects en cohortes de microbioma 16S-rRNA mediante variational autoencoders para estudios de cáncer colorectal

Estudante: Carla Rodríguez Rodríguez

Dirección: Carlos Fernández Lozano

Diego Fernández Edreira

José Liñares Blanco

Vigo, junio de 2024.

Dedicatoria

A mis padres, por su incansable labor de enseñanza y educación, siempre acompañada de todo el cariño y la paciencia que un ser humano puede llegar a dar.

Agradecimientos

Expresar mi profundo agradecimiento a mis tutores, Carlos, Jose y Diego, por el apoyo y orientación brindados en cada etapa de este proceso; haciendo de un simple trabajo académico obligatorio un proyecto estimulante que ha despertado en mí el interés por el ámbito y que, sin duda, supondrá un significativo punto de partida de cara a mi futuro laboral y profesional.

Además, agradecer también este trabajo a todos y cada uno de los compañeros que me han acompañado a lo largo de este grado. Cada gesto de ayuda brindado, ya fuera menor o mayor, directo o indirecto, ha contribuido a que hoy esté entregando este trabajo.

Por último, quiero expresar mi agradecimiento a la cátedra CICAS, por promover la investigación científica y brindarme la oportunidad de recibir apoyo económico para llevar a cabo este proyecto. Saber que este trabajo puede ser útil para otras personas ha sido una motivación adicional durante todo el proceso.

Resumen

El término microbioma se refiere al conjunto de comunidades de microorganismos (como hongos, bacterias y virus) que existen en un entorno en particular. Estas comunidades son cada vez más estudiadas con el objetivo de investigar su papel en numerosos hábitats. Sin embargo, estos estudios por lo general son difíciles de reproducir debido a la existencia de fuentes de variación no deseada y no relacionadas con ningún factor de interés, que pueden conducir a conclusiones inexactas y/o erróneas. A estas fuentes de variación se les conoce como efectos de lote (batch effects).

Entre los numerosos estudios existentes acerca del microbioma y su funcionamiento, el microbioma humano ha cobrado en los últimos años un gran interés, debido a los resultados de varios estudios que demuestran la existencia de una estrecha relación entre el equilibrio del mismo y el riesgo del desarrollo de diferentes tipos de cáncer. Entre ellos, el cáncer colorectal.

El objetivo de este trabajo está centrado en la búsqueda de la forma más adecuada de eliminación de los ya nombrados efectos de lote en cohortes de microbioma para estudios de cáncer colorectal.

Abstract

The term microbiome refers to the set of microbial communities (such as fungi, bacteria, and viruses) existing in a particular environment. These communities are increasingly being studied to investigate their role in numerous habitats. However, these studies are generally difficult to reproduce due to the existence of sources of unwanted variation unrelated to any factor of interest, which can lead to inaccurate and/or erroneous conclusions. These sources of variation are known as batch effects.

Among the numerous existing studies on the microbiome and its functioning, the human microbiome has garnered significant interest in recent years, due to the results of several studies demonstrating the existence of a close relationship between its balance and the risk of developing different types of cancer, including colorectal cancer.

The objective of this work is focused on finding the most appropriate way to eliminate the aforementioned batch effects in microbiome cohorts for colorectal cancer studies.

Palabras clave:

- microbioma
- cohorte
- MOBER
- efectos de lote
- señal biológica
- normalización
- proyección

Keywords:

- microbiome
- cohort
- MOBER
- batch effects
- biological signal
- normalization
- projection

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introdución

La incidencia del cáncer colorrectal (CCR) en España es de 37.172 casos al año, siendo la principal causa de cáncer en el país para ambos géneros. La tasa de supervivencia a 5 años es del 63%, aunque si es diagnosticado en un estadio localizado, la tasa asciende al 90%. Si el cáncer alcanza partes distantes del cuerpo, el ratio de supervivencia a 5 años es del 15%. A pesar de los recientes avances en el campo de la oncología, estas estadísticas revelan la necesidad de seguir investigando para la mejora en el diagnóstico y tratamiento de la enfermedad.

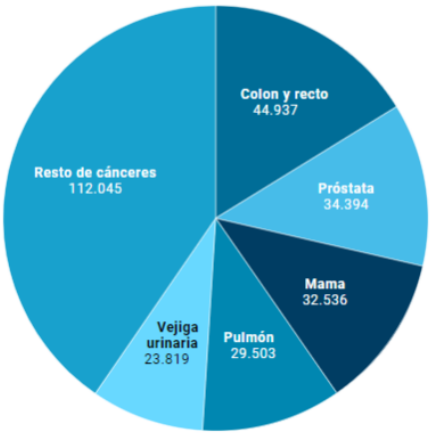


Figura 1.1: Estimación del número de nuevos casos de cáncer en España (año 2019) [1]

Como observamos en la Figura 1.1 el cáncer colorrectal ha sido y sigue siendo de manera diferenciada el más diagnosticado en relación con cualquier otro tipo de reacción tumoral [1].

Un aspecto singular del CCR es que tiene una relación muy íntima con el equilibrio del microbioma intestinal, que forma una parte esencial del microentorno tumoral. Las investigaciones de la última década establecieron que la disbiosis de bacterias, hongos, virus y arqueas intes-

tinales acompaña a la tumorigénesis colorrectal, y que estos cambios podrían ser causales.

Para identificar los cambios, diferencias y alteraciones del microbioma que podrían estar causando el CCR es necesario utilizar datos de diferentes cohortes de pacientes, pero eso supone un reto. Primero, existen diferentes tecnologías de secuenciación de microbioma, principalmente secuencia del gen 16S-rRNA y de los metagenomas completos de la muestra (shotgun). Segundo, existen diferentes protocolos, plataformas de secuencia, y técnicas para secuenciar las muestras de los pacientes. Esto es, además de poder obtener diferentes artefactos técnicos, puede haber diferencias en la representatividad de las señalizaciones biológicas obtenidas. La propuesta de Trabajo Fin de Grado (TFG) se centra en el segundo punto, estudiando la forma más adecuada de eliminar estos efectos no deseados, también llamados batch-effects, de varias cohortes de CCR 16S-rRNA mediante la aplicación de una red condicional variational autoencoder (VAE) optimizada para generar embeddings que puedan reconstruir la entrada original.

Esto se hará principalmente mediante la implementación de un sistema de eliminación de efectos de lote para cohortes de CCR 16S-rRNA a partir de una aproximación propuesta ya existente y en uso llamada método MOBER (**M**ulti-**O**rigin **B**atch **E**ffect **R**emover) [4], que se explicará de forma más detallada en otros capítulos.

1.1 Motivación

La motivación de este proyecto está basada en la preocupación por la gravedad y mortalidad asociada a una enfermedad cómo es el cáncer. Con este trabajo se propone continuar una línea de investigación que a día de hoy ya está siendo abordada por numerosos investigadores debido a su alta influencia e importancia, con el objetivo tanto de reducir la mortalidad como de mejorar el diagnóstico asociado a esta enfermedad.

Además, otra fuente de motivación se sitúa en el hecho de poder demostrar a la sociedad la utilidad y relevancia de un grado como es el Grado en Ciencia e Ingeniería de Datos (GCED), proporcionando este los conocimientos necesarios para permitir a los estudiantes abordar problemas de gran relevancia a nivel mundial, marco en el que se sitúa el cáncer.

1.2 Objetivos

El objetivo principal de este trabajo, como ya se ha introducido en el punto anterior, es tanto aportar valor a una línea de investigación científica ya en curso, como servir como punto

de partida o puente para otras investigaciones relativas en el ámbito. En cuanto a los objetivos concretos:

- Implementar un sistema de eliminación de batch-effects para cohortes de CCR 16S-rRNA mediante la aproximación propuesta.
- Mantener actualizada la base de datos de cohortes 16S-rRNA disponibles en CCR de los directores del proyecto para localizar cohortes de muy recente publicación.
- Selección de las cohortes a incluír en el estudio.
- Comparación de resultados.
- Elaboración de una memoria descriptiva del trabajo y exposición de los resultados.

1.3 Estructura de la memoria

- **Introducción**: Se presenta una visión general de lo que se ha realizado en el trabajo, incluyendo la motivación del mismo, los objetivos y el plan de trabajo seguido.
- **Estado del arte**: Se estudian diversas muestras de técnicas y proyectos que han abordado el mismo problema que el presente TFG, pero con enfoques distintos. Se proporciona una breve explicación de estos y se comentan sus resultados.
- **Tecnologías y herramientas**: Descripción del software, tecnologías y herramientas utilizadas para la realización del proyecto.
- **Metodología**: Descripción de la metodología seguida para la realización del proyecto con una base de organización e interacción con los directores del mismo.
- **Diseño y desarrollo de la solución**: Se detalla la solución desarrollada, incluyendo tanto representaciones (visuales o de cualquier otro tipo) como explicaciones de las técnicas utilizadas, que ayuden a comprender la implementación de la herramienta.
- **Pruebas**: Se describen las pruebas realizadas para verificar el correcto funcionamiento de lo implementado.
- **Conclusiones**: Se resumen los principales descubrimientos a raíz de lo implementado, así cómo los puntos de mejora y las limitaciones del estudio. Además, se hace una breve discusión de los resultados y su posible implicación en investigaciones futuras.

1.4 Plan de trabajo

El plan de trabajo seguido para la realización de este proyecto ha sido la metodología Ágil Scrum [2, 5], caracterizada por el desarrollo interactivo e incremental, dividiendo el trabajo en 12 sprints y organizando reuniones periódicas con los directores del proyecto para revisión, seguimiento y planificación. Se explica de forma más detallada tanto la metodología en la sección 4.1, como la planificación en la sección 4.4.

Capítulo 2

Estado del arte

En este apartado se hace una revisión de diferentes estudios del microbioma en los que se han obervado efectos de lote.

2.1 Microbioma humano

Estrechamente relacionado con el objetivo de este proyecto y tratando de forma amplia la casuística del microbioma y los efectos de lote, está el *"Human Microbiome Project (HMP)"* [6] desarrollado por el Instituto Nacional de Salud (NIH) [7] que tuvo como objetivo la investigación de la diversidad de los microorganismos que se encuentran asociados a humanos, tanto en la salud como en la enfermedad, es decir, del microbioma humano.

A lo largo del proyecto, se identificaron efectos de lote debidos a diferencias en los protocolos de muestreo, secuenciación y análisis entre los diversos centros de investigación participantes.

Para lidiar con ellos, se emplearon principalmente métodos estadísticos y técnicas de normalización de datos, destacando la secuenciación, la corrección de los efectos de lote basada en los lotes conocidos, el análisis de covarianza ANOVA y la validación cruzada.

2.2 Esponja A. aerophoba

La tesis *"Ecología química y microbiana de la demosponja Aplysina Aerophoba"* [8] desarrollada por Oriol Sacristán Soriano por la Universidad de Barcelona, fue concebida para estudiar la ecología química y microbiana de la esponja *Aplysina aerophoba* (Nardo, 1833). En ella se investigó el posible papel de esta especie de esponja en la biosíntesis de alcaloides bromados (ABs). Se comparó la composición microbiana y la concentración de ABs en dos tejidos diferentes (ectosoma y coanoma) para investigar la relación entre la composición bacteriana y la

concentración de ABs.

Además, se evaluó el efecto de múltiples factores ambientales, centrándose en la luz, sobre los perfiles químicos y microbianos. En cuanto a los ABs, se concluyó que las diferencias en los perfiles bacterianos no solo se debieron a la variación en los tejidos (el principal efecto de interés), sino también a que las muestras se procesaron en dos geles de gradiente desnaturalizante separados. Por lo tanto, el gel actuó como un efecto técnico de lote. Asimismo, se concluyó que distintas condiciones lumínicas podían causar variaciones en los perfiles bacterianos y en los productos naturales de la esponja, considerándose esto también un efecto de lote.

En este caso, se abordó la existencia de los efectos de lote y no su eliminación. En lugar de erradicarlos, se estudiaron correlacionando los resultados del estudio con los patrones de variación, con el objetivo de inferir posibles asociaciones entre productos naturales y simbiontes microbianos.

2.3 Digestión anaeróbica

En el artículo *"Las concentraciones crecientes de fenol afectan progresivamente la digestión anaeróbica de la celulosa y las comunidades microbianas asociadas"* [9] publicado en el NIH se expone una exploración sobre indicadores microbianos que podrían mejorar la eficacia del bioproceso de AD y prevenir su falla. La AD es un proceso microbiológico de degradación de materia orgánica que produce biogás que se utiliza en la producción de energía eléctrica y térmica pero que, sin embargo, experimenta inhibición durante su etapa de desarrollo, que no está bien caracterizada.

Perfilizaron la microbiota (231 OTUs) de 75 muestras de AD en diversas condiciones. Aquí consideraron dos rangos diferentes de concentración de fenol como tratamientos. El experimento se llevó a cabo en diferentes fechas, lo que constituye una fuente técnica de variación no deseada (efecto de lote). Además de este, existieron otras variaciones no deseadas, como la temperatura. Para mitigar estos factores de variación, primero se determinaron las condiciones más adecuadas para el estudio y, posteriormente, se estudiaron los cambios poblacionales ante las diferentes perturbaciones, con el objetivo de tener en cuenta estas diferencias y lograr con ello dar una mayor estabilidad al estudio.

2.4 Modelos de ratones con la enfermedad de Huntington

El estudio *"El perfil del microbioma revela disbiosis intestinal en un modelo de ratón transgénico de la enfermedad de Huntington"* [10] publicado también en el NIH tuvo como objetivo el estudio de diferencias en la composición microbiana entre ratones con enfermedad de Huntington (HD) y ratones de tipo salvaje (WT).

De nuevo el establecimiento de comunidades microbianas también estuvo influenciado por efectos biológicos de lote: el ambiente de la jaula y el sexo.

En cuanto al sexo, se observó que, al examinar toda la población en conjunto en un gráfico de PCoA, las muestras tendieron a agruparse según el sexo. Para mitigar este efecto, la mayoría de las métricas y estudios se realizaron por separado para machos y hembras, asegurando que el sexo no interfiriera en los resultados.

Para evaluar los efectos de la jaula dentro de cada grupo de sexo, se utilizó un modelo lineal sobre los recuentos proporcionales transformados con CLR. En este modelo, la jaula se consideró un efecto aleatorio, lo que significa que las diferencias atribuibles a la jaula se trataron como una fuente de variabilidad no principal, aunque potencialmente influyente en los resultados.

2.5 Conclusiones

Como conclusión general, se destaca la importancia de tener en cuenta los efectos de lote en la investigación del microbioma, ya que estos pueden influir significativamente en los resultados. Es crucial abordar y controlar estos efectos para garantizar la validez y fiabilidad de los estudios realizados.

Además, se concluye también en la existencia de una alta variabilidad de efectos de lote presentes en estudios de microbioma, pudiendo ser estos causados por factores diversos como: las condiciones experimentales (como se observa en el estudio de la AD), los procesos de muestreo (proyecto Microbioma Humano y estudio de la esponja Aplysina Aerophoba), los factores biológicos (observados en el estudio sobre la enfermedad de Huntington en ratones) u otros muchos que no se ven reflejados en estos ejemplos como podrían ser las diferencias geográficas o en los métodos de almacenamiento. Por lo tanto, es de gran importancia en este tipo de estudios estar siempre atentos a cualquier fuente de variación externa.

Capítulo 3

Tecnologías y herramientas

En este capítulo se hablará de las diferentes tecnologías y herramientas que se han utilizado a lo largo del desarrollo del proyecto y han permitido la realización del mismo.

3.1 Lenguajes de programación

Dentro del marco tecnológico para la realización del TFG, se ha trabajado con diferentes lenguajes de programación (definidos como un conjunto de instrucciones y reglas que se utilizan para escribir programas informáticos) como base para el desarrollo de la solución. En este caso los utilizados han sido R y Python.

3.1.1 R

R [11] es un entorno y lenguaje de programación abierto, libre y gratis con un enfoque al análisis estadístico. Es un proyecto GNU (sistema operativo de tipo Unix), lo que significa que cualquier persona tiene derecho a estudiar, usar, modificar y compartir el software sin que pertenezca a nadie. Además es un lenguaje accesible ya que funciona con paquetes fáciles de descargar que nos permiten procesar grandes conjuntos de datos, excelente para el análisis y el cálculo estadístico y con una potente oferta de producción de gráficos.

3.1.2 Python

Python [12] es un lenguaje de alto nivel de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código. Tiene estructuras de datos de alto nivel eficientes y un simple pero efectivo sistema de programación orientado a objetos. La elegante sintaxis de Python y su tipado dinámico, junto a su naturaleza interpretada, lo convierten en un lenguaje ideal para scripting y desarrollo rápido de aplicaciones en muchas áreas. Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional.

3.2 Programas: Entornos de desarrollo

Para el trabajo con los lenguajes mencionados y la gestión del desarrollo de la solución y los resultados obtenidos se ha hecho uso de diferentes aplicaciones y/o entornos de desarrollo.

3.2.1 Rstudio

Rstudio [13] es una herramienta informática sumamente potente para realizar distintos cálculos científicos, numéricos y estadísticos, así como para crear gráficas y figuras de gran calidad. Incluye una consola y un editor de sintaxis que apoyan la ejecución de código, así como herramientas para el trazado, la depuración y la gestión del espacio de trabajo.

3.2.2 Spyder

Spyder [14] es un entorno de desarrollo gratuito y de código abierto para programación en lenguaje Python y diseñado por y para científicos, ingenieros y analistas de datos. Presenta una combinación de funciones avanzadas de edición, análisis, depuración y creación de perfiles de una herramienta de desarrollo integral con la exploración de datos, ejecución interactiva, inspección profunda y potentes capacidades de visualización de un paquete científico.

3.2.3 Visual Studio Code

Visual Studio Code [15] es un editor de código fuente desarrollado por Microsoft para Windows, macOS y Linux. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código y soporta multitud de lenguajes de programación (como C++, C#, Java, Python, PHP, Go, .NET).

3.3 Software: Control de versiones

3.3.1 Git

Git [16] es un software de control de versiones distribuido gratuito y de código abierto pensando en la eficiencia, la confiabilidad y compatibilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Su propósito es llevar registro de los cambios en archivos de computadora incluyendo coordinar el trabajo que varias personas realizan sobre archivos compartidos en un repositorio de código.

3.3.2 GitHub

GitHub [17] es un servicio basado en la nube que aloja el sistema de control de versiones ya mencionado Git. Es un plataforma para el desarrollo colaborativo que permite a los desarrolladores colaborar y realizar cambios en proyectos compartidos, a la vez que mantienen un seguimiento detallado de su progreso.

3.3.3 Singularity

Singularity [18] es una plataforma de virtualización a nivel del sistema operativo, también conocida como containers. Uno de los principales usos de Singularity es llevar contenedores y reproducibilidad a la informática científica y al mundo de la informática de alto rendimiento (HPC).

Los contenedores de Singularity se pueden usar para empaquetar flujos de trabajo científicos completos, software y bibliotecas, e incluso datos.

Capítulo 4
Metodología

En esta sección se detallará la metodología seguida para el desarrollo del TFG. Se especificarán los puntos clave y los roles involucrados en la metodología, así como la forma en que el proyecto se ha llevado a cabo siguiéndola. A mayores, se presentarán los costes asociados a la implementación de esta metodología en el presente proyecto.

4.1 SCRUM

El término ”Scrum” [2, 5, 19, 20, 21] tal como lo conocemos hoy, se introdujo por primera vez en un artículo de la Harvard Business Review en 1986 *El nuevo juego de desarrollo de nuevos productos (The New New Product Development Game)*, escrito por Hirotaka Takeuchi e Ikujiro Nonaka. Takeuchi y Nonaka tomaron el término “Scrum” del rugby, explicando que *“como en el rugby, los miembros del equipo se pasan la pelota entre sí, a medida que avanzan como una unidad por el campo de juego”*.

De ahí, y contando con ciertos pasos intermedios, nace la metodología Scrum como creencia de que los desarrolladores podrían beneficiarse de un enfoque más flexible e interactivo, dejando de lado el modelo más utilizado en aquel momento, conocido como modelo de cascada de desarrollo de software, dónde los trabajos se dividían en fases y cada fase realizada desbloqueaba a la siguiente.

La metodología Scrum es un marco de trabajo ágil para la gestión y desarrollo de proyectos, a través del cual las personas pueden abordar problemas complejos adaptativos a la vez que se entregan productos de forma eficiente y creativa con el máximo valor. Se caracteriza por su enfoque iterativo e incremental, donde el trabajo se divide en ciclos cortos llamados ”sprints”. Durante cada sprint, un equipo multidisciplinar trabaja para entregar un conjunto de funcionalidades completas y potencialmente entregables. Fomenta el trabajo en equipo y

permite aprender a partir de la experiencia y adaptarse al cambio.

4.1.1 Eventos y puntos clave

- **Refinement**: Consiste en la organización del trabajo pendiente. El punto de partida para comenzar a trabajar en un Sprint (posteriorermente se detallará en qué consiste) dentro de esta metodología se sitúa en la identificación de qué trabajo extraer de la lista de tareas pendientes, es decir, el trabajo que debe realizarse. Para un óptimo desarrollo de los sprints, el trabajo pendiente debe estar claramente documentado y organizado en un solo lugar y habitualmente clasificado por prioridad, dificultad, etc. A este punto de almacenamiento de las tareas pendientes (historias de usuario) se le suele conocer como Product Backlog.

- **Sprint**: El Sprint es el punto clave de la metodología Scrum y es el nombre que recibe cada uno de los ciclos o iteraciones que se llevan a cabo dentro de dentro de un proyecto Scrum. Se basa en periodos breves de tiempo fijo en el que un equipo de Scrum trabaja para completar la cantidad de trabajo previamente establecida. Generalmente su duración es de entre 2 y 4 semanas.

- **Sprint planning**: El Sprint Planning es la reunión previa a dar inicio a un Sprint, dónde se establece el trabajo a realizar durante el Sprint. El objetivo de esta reunión se sitúa en detérminar QUÉ se va a hacer y CÓMO se va a hacer. Es por tanto una reunión de planificación que servirá para planificar el trabajo que se desarrollará durante ese sprint, mediante el estudio y discusión del backlog existente en ese momento. En esta reunión participa todo el equipo Scrum.

- **Daily**: Otra reunión importante en la metodología Scrum es la ”daily” o ”daily standup”. Se realiza todos los días y en ella el equipo revisa y discute lo que se hizo el día anterior, lo que se planea hacer ese día y cualquier obstáculo o problema que se esté enfrentando. La reunión generalmente se realiza de pie para asegurar que sea breve y enfocada, promoviendo la comunicación de información relevante. Todos los miembros internos del equipo asisten a esta reunión.

- **Sprint Review**: Sprint Review es el nombre que recibe la reunión que se celebra al final de cada Sprint con el propósito de evaluar los resultados que ha obtenido el equipo en ese Sprint y adaptar el backlog de cara al siguiente. Esta permite, a su vez, analizar el progreso que está teniendo el desarrollo global del proyecto con miras a cumplir con los objetivos establecidos.

  Parte de lo que caracteriza el Sprint Review son los participantes del mismo, ya que

a diferencia del resto de reuniones, en esta también participan los stakeholders (cualquier persona o entidad que tenga un interés o impacto en el proyecto), que son los principales interesados en conocer el avance del proyecto. La importancia de su participación reside, además de en su conocimiento de los avances conseguidos en el Sprint, en un aporte de su opinión a manera de retroalimentación. En esta reunión participan por lo tanto, tanto los miembros del equipo Scrum como generalmente un grupo de stakeholders, entre los que pueden estar patrocinadores, clientes, usuarios finales, propietarios de productos, etc.

- **Sprint Retrospective**: El Sprint Retrospective (retrospectiva) es el último evento en un Sprint de Scrum. Es una oportunidad para el equipo de inspeccionarse, y crear un plan de mejora que se pondrá en marcha inmediatamente, en el siguiente Sprint. El objetivo de un sprint retrospective es, básicamente, mejorar: mejorar la productividad, mejorar las habilidades del equipo, mejorar la calidad del producto. Se realiza al final de cada Sprint y acuden a él los asistentes habituales del resto de reuniones.

4.1.2 Artefactos

- **Product Backlog**: Consiste en un inventario que contiene cualquier tipo de trabajo que haya que hacer en el producto: requerimientos, casos de uso, tareas y dependencias. Está formado en casi su totalidad por las llamadas historias de usuario y es la principal fuente de información sobre el producto en Scrum. Consiste en una lista, en cualquier formato, que contiene todos los requerimientos que se necesitan implementar en el producto. Esta lista es el resultado del trabajo del Product Owner (PO) con el cliente, los distintos stakeholders, patrocinadores, comités, etc, y refleja el estado real del trabajo pendiente de implementar en el producto. Este es gestionado por el PO, quien debe priorizar los elementos con mayor valor en cada etapa y detallarlos para que el equipo de desarrollo sea capaz de valorarlos y ejecutarlos

- **Sprint Backlog**: Se refiere al conjunto de elementos seleccionados del Product Backlog sobre los que se trabajará durante un Sprint específico. Estos elementos suelen consistir en pequeñas tareas técnicas que permitan conseguir un incremento de objetivos cumplidos en el proyecto. El Sprint Backlog se define principalmente durante el Sprint Planning, donde, como ya se ha explicado, el equipo decide qué elementos del Product Backlog se incluirán en el Sprint y cómo se organizarán para lograr los objetivos establecidos.

- **Incremento**: El incremento es básicamente el resultado del Sprint. Es la suma de todos los desarrollos, casos de uso, historias de usuario, nuevas funcionalidades, etc, que será puesto a disposición del usuario final, aportando así un valor de negocio al producto

que se está desarrollando, que es básicamente el objetivo de la metología. Es la suma de todos los ítems del Product Backlog completados durante un Sprint y el valor de los incrementos de todos los Sprints pasados.

4.1.3 Roles

- **Product Owner**: Es la persona encargada de dirigir el equipo hacia el mejor resultado posible. Conoce muy bien el producto final y es el responsable de la toma de decisiones, de la gestión efectiva del Product Backlog, y del equipo.
  Entre sus funciones destacan:
  - Suministrar al equipo de trabajo de orientación clara sobre lo que se debe desarrollar y entregar.
  - Servir de puente entre lo que desea negocio (parte interesada en el resultado) y lo que el equipo de desarrollo debe hacer para conseguirlo.
  - Administrar el Product Backlog y priorizar los objetivos de cada sprint.
- **Scrum Master**: Es la persona que mantiene en movimiento al equipo de trabajo. Gestiona los impedimentos, alerta los riesgos, trata de mitigar los mismos e impulsa la búsqueda de soluciones. Como funciones principales:
  - Entrenar y motivar al equipo.
  - Comunicarse con grupos externos para resolver cualquier reto al que se enfrente el equipo.
  - Garantizar que el equipo de desarrollo tenga las mejores condiciones posibles para cumplir sus objetivos y producir productos entregables.
- **Equipo de desarrollo**: Formado por el grupo de personas responsable de llevar a cabo el trabajo necesario para entregar el producto o incremento del producto al final de cada sprint. Entre sus responsabilidades:
  - Implementar las funcionalidades definidas en el Sprint Backlog.
  - Trabajar de manera colaborativa para resolver problemas y conseguir alcanzar los objetivos del Sprint.
  - Aportar valor, es decir, ser capaces de entregar un producto o incremento de producto funcional al final de cada Sprint.

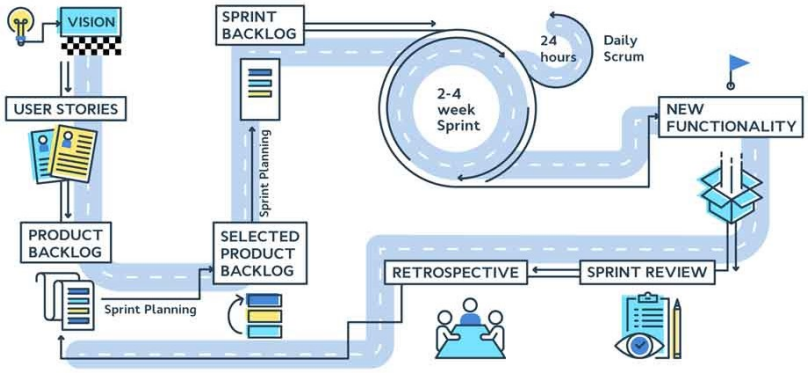


Figura 4.1: Resumen metodología SCRUM [2]

En la Figura 4.1 se muestra un resumen de lo detallado en los anteriores apartados sobre la metodología SCRUM.

4.2 Historias de usuarios y toma de requisitos

Las historias de usuario son una técnica utilizada en la metodología ágil SCRUM para captar los requisitos del cliente en un formato comprensible y centrado en el usuario. Lo que las distingue de una simple petición por parte de un usuario es la redacción; en las historias de usuario se describen las funcionalidades deseadas desde la perspectiva del usuario final, utilizando un lenguaje sencillo y claro, sin tecnicidades. Esto permite ofrecer contexto al equipo de desarrollo a la vez que se pone a los usuarios finales reales en el centro, ya que el marco de las mismas está centrado en ellos, que expresan de manera directa sus necesidades, lo que el equipo de desarrollo debe hacer.

Cada historia de usuario es una explicación general e informal de una función de software escrita desde la perspectiva del usuario final o cliente, siendo estas la unidad de trabajo más pequeña en el marco de la metodología Ágil SCRUM.

Estas historias de usuario serán las componentes mayoritarias del Product Backlog y, en caso de que proceda, del Sprint Backlog. Las historias añadidas a un sprint representan qué debe hacer el equipo en ese sprint y cómo debe hacerlo.

Entre los beneficios de las historias de usuario se sitúan:

- **Comprensión compartida**: Al describir las funcionalidades deseadas en un lenguaje sencillo y claro, las historias de usuario promueven una mejor comprensión y alineación entre el equipo de desarrollo y el cliente o usuario final.

- **Priorización efectiva**: Permiten una priorización clara de las peticiones del cliente. Teniendo el objetivo final del proyecto claro y con la existencia de historias de usuario basadas en lo que aporta valor al cliente, el equipo puede identificar y abordar las funcionalidades más importantes en primer lugar, lo que maximiza el valor entregado en cada iteración del desarrollo.

- **Acuerdo mutuo**: La colaboración en la redacción y refinamiento de las historias de usuario puede ayudar a agilizar los procesos que se llevan a cabo dentro de una organización al promover el consenso y la toma de decisiones compartidas entre todas las partes interesadas.

- **Dinamización del trabajo**: El estar en contacto directo o casi directo con el usuario final, genera una adaptabilidad automática de la conversación, maximizando así la transmisión de información entre peticionarios y receptores. Además, permite al equipo de desarrollo recibir feedback casi inmediato del trabajo que se está realizando, lo que va a generar un trabajo mucho más eficiente.

- **Reducción de errores**: De la mano del anterior punto, el contacto directo o casi directo con el cliente, permite y potencia una validación continua de las funcionalidades que se van desarrollando. Esta interacción constante ayuda en la identificación y corrección eficiente de posibles errores o malentendidos de manera temprana, evitando así que se conviertan en problemas más difíciles de solucionar durante el desarrollo del producto.

En Scrum, la toma de requisitos es una parte fundamental del proceso de desarrollo ágil. Aunque Scrum se caracteriza por su enfoque flexible y adaptable, la identificación y comprensión clara de los requisitos del proyecto siguen siendo esenciales para el éxito del equipo. Este ejercicio consiste en entender lo que se va a hacer antes de llevarlo a cabo. Es una actividad crucial, especialmente en entornos más tradicionales, donde la planificación previa a la ejecución es fundamental.

Dentro de la toma de requisitos, se abordan varios puntos:

- **Identificación de Requisitos**: El Product Owner trabaja estrechamente con el cliente/usuario final para entender y definir las necesidades del producto.

- **Priorización de Objetivos**: Se establece la importancia y el orden de los requisitos en función de su valor para el producto y el cliente.

- **Descomposición de Requisitos**: Los requisitos se desglosan en elementos de trabajo más pequeños y manejables, como historias de usuario o tareas específicas.

- **Refinamiento del Backlog**: Durante las reuniones de refinamiento del backlog, se revisan y detallan los requisitos para asegurarse de que estén claros y listos para ser implementados.

Como herramientas para la toma de requisitos, se utilizan las reuniones periódicas ya comentadas y las historias de usuario. Estas herramientas facilitan la comunicación y la comprensión de los requisitos entre todos los miembros del equipo y los interesados.

4.3 Adaptación de la metodología Scrum al proyecto

Este proyecto se ha ajustado a la metodología Ágil SCRUM de la manera más cercana posible, teniendo siempre en consideración el reducido número de participantes y las funcionalidades a desarrollar.

4.3.1 Eventos

- **Refinement**: Antes de comenzar cada sprint, se llevó a cabo una reunión de refinamiento para revisar el trabajo pendiente. Durante esta práctica, se tomaron decisiones sobre qué tareas o historias de usuario incluir en el próximo sprint, con el objetivo de cumplir tanto con la planificación del proyecto como con sus objetivos. Todo lo decidido quedó reflejado en un tablero de Trello (software de administración de proyectos utilizado para este trabajo).

- **Sprint**: Se llevaron a cabo sprints de dos semanas de duración, siguiendo lo habitual en el uso de esta metodología. Esta decisión se tomó como una adaptación de la misma a la duración del proyecto y al número de participantes involucrados en el mismo.

- **Sprint Planning**: Una vez celebrada la reunión de refinamiento, se procedió a la planificación del Sprint. Frecuentemente estas dos reuniones fueron realizadas en conjunto, para ahorrar tiempo y facilitar la asistencia de los participantes. En este bloque de la reunión, y teniendo en cuenta lo visto en el anterior bloque, se decidió a cuáles de las tareas pendientes vistas y estudiadas en el refinamiento, se dedicaría el Sprint. Con esta información, se construyó una checklist por cada sprint en el tablero de Trello, de la manera que de detallará en la sección 4.2. De nuevo el punto de mira siguió siendo cumplir tanto con la planificación del proyecto como con sus objetivos.

- **Daily**: Dado que el equipo de desarrollo está conformado por una única persona, no se llevó a cabo una reunión diaria de seguimiento.

- **Sprint Review + Sprint Retrospective**: Estos dos eventos se realizaron de manera conjunta, consistiendo en una revisión de los logros alcanzados durante cada sprint y

un análisis sobre el desarrollo del proyecto en general. Se relacionaron los resultados obtenidos con lo discutido durante el refinamiento y la revisión del sprint. La utilización de Trello facilitó el estudio de estos aspectos.

4.3.2 Artefactos

A lo largo de este proyecto se ha utilizado la herramienta Trello para la gestión del proyecto y el seguimiento de los artefactos.

Trello es una herramienta de gestión de proyectos basada en tarjetas que se centra en la colaboración y la visualización del flujo de trabajo. Utiliza un formato de tablero con listas y tarjetas que representan tareas o elementos del proyecto. Es flexible y adaptable, lo que lo hace ideal para equipos pequeños donde la colaboración y la agilidad son prioritarias.

A través de Trello, se implementó una adaptación de los artefactos de la metodología Scrum para este proyecto. Se configuró un tablero en el que la alumna y los tutores colaboraron estrechamente. Los tutores utilizaron este tablero para detallar los pasos a seguir para lograr la realización del proyecto, mientras que la alumna, en colaboración con ellos y mediante lo acordado en las reuniones realizadas, utilizó checklists por sprints, con los que se hacía seguimiento del progreso de cada tarea. De esta manera, se pudo visualizar de forma clara y precisa qué debía completarse en cada sprint y cómo avanzaba el proyecto en general.

- **Product Backlog**: representado por los pasos a seguir detallados por los tutores para la realización del TFG.
- **Sprint Backlog**: se refleja en cada checklist, que contiene las tareas a completar en cada sprint.
- **Incremento**: representado mediante el marcado de elementos de la lista del sprint como checks, indicando así el progreso realizado en cada tarea y cada sprint.

4.3.3 Roles

Como adaptación a los roles tradicionales de la metodología Scrum, en este proyecto se estableció lo siguiente:

- **Product Owner + Scrum Master**: En lugar de tener roles separados, los tutores del proyecto asumieron conjuntamente las funciones del Product Owner y del Scrum Master. Ellos llevaron a cabo las responsabilidades específicas de estos puestos detalladas en secciones anteriores.

- **Equipo de desarrollo**: la alumna a cargo del TFG desempeñó el papel del equipo de desarrollo. Como única integrante de este equipo, fue la responsable de la implementación de las tareas definidas en el Sprint Backlog y de asegurar el progreso del proyecto de acuerdo con las metas establecidas para cada sprint.

4.3.4 Historias de usuario y toma de requisitos

Durante el proyecto, tanto tutores como alumna colaboraron en la creación de las diversas historias de usuario necesarias para la realización del proyecto. Estas historias fueron quedando reflejadas en el tablero de Trello para su seguimiento y gestión.

Breve descripción de las historias de usuario creadas y completadas a lo largo de la realización de este proyecto:

1. **Lectura de documentación relativa al proyecto**: Revisar y estudiar la documentación pertinente relacionada con el proyecto a realizar.
2. **Creación de repositorio en GitHub**: Establecer un repositorio en GitHub para gestionar y hacer seguimiento de los avances del proyecto.
3. **Estudio del funcionamiento del CESGA**: Investigar y comprender el funcionamiento del Centro de Supercomputación de Galicia (CESGA), en el contexto del proyecto.
4. **Creación de contenedor en Singularity**: Desarrollar un contenedor en Singularity que contenga todos los elementos necesarios para la ejecución de la herramienta.
5. **Modificaciones en la implementación de MOBER**: Realizar mejoras en la implementación de la herramienta MOBER.
6. **Puesta en marcha de MOBER**: Ejecutar las diferentes versiones de MOBER utilizando datos aleatorios de prueba.
7. **Selección de cohortes y preprocesado de datos**: Seleccionar las cohortes a incluir en el estudio y realizar el preprocesamiento necesario de los datos.
8. **Extracción y conversión de matrices de datos**: Extraer matrices a partir de los datos preprocesados y convertirlas al formato .h5ad para su utilización en la herramienta.
9. **Ejecución de MOBER con los datos extraídos**: Realizar la ejecución de MOBER utilizando los datos previamente procesados y extraídos.
10. **Realización y estudio de PCAs**: Aplicar Análisis de Componentes Principales (PCAs) a los datos previos y posteriores a la ejecución de MOBER, representar los resultados gráficamente y realizar las comparaciones y los análisis pertinentes.

11. **Normalización de datos**: Aplicar normalización a los datos del estudio.
12. **Ejecución de MOBER con datos normalizados**: Ejecutar MOBER utilizando los datos normalizados.
13. **Repetición de la H10 con los datos normalizados**: Realizar nuevamente la historia de usuario número 10 utilizando los datos normalizados.
14. **Experimento cohortes ML - Estudio de la eliminación del batch effect a través de Machine Learning**: a través de Machine Learning (ML), estudiar y analizar la eliminación de los efectos de lote una vez ejecutado MOBER.
15. **Experimento status ML - Estudio de la señal biológica presente en los datos a través de Machine Learning**: a través de ML, estudiar y analizar la presencia de señal biológica en los datos una vez ejecutado MOBER.
16. **Desarrollo de la memoria**: Elaborar el documento de memoria del proyecto.
17. **Revisión final del proyecto**: Revisión conjunta entre tutores y alumna de lo realizado para la conclusión del proyecto.

Para la toma de requisitos, se llevó a cabo una reunión inicial donde se detallaron los objetivos del proyecto y se discutió cómo se abordarían. Durante esta reunión, se dividió el trabajo en las historias de usuario mencionadas anteriormente y se establecieron prioridades. Además, cada historia de usuario fue descrita en detalle, con el fin de asegurar una comprensión clara de los requisitos y lo que se esperaba implementar. Esto garantizó que tanto los tutores como la alumna estuvieran alineados en cuanto a las expectativas y el alcance del proyecto.

4.4 Planificación y seguimiento de la planificación

En esta sección se detallan los siguientes aspectos:

- Planificación en Sprints y desarrollo de los mismos: Se expone la planificación realizada para llevar a cabo estas historias de usuario, dividiéndolas en sprints, así como el desarrollo de cada sprint en términos de tareas completadas y objetivos alcanzados.
- Desviaciones de la planificación inicial: Se analizan las desviaciones surgidas durante la ejecución del proyecto respecto a la planificación inicialmente establecida.

Además, se incluye una figura con un diagrama de Gantt que detalla visualmente el contenido expuesto.

4.4.1 Planificación y desarrollo

La planificación inicial de este proyecto se basó en los principios de la metodología a utilizar, así como en la equivalencia de horas de trabajo por crédito ECTS y el número de créditos asignados al TFG.

Según el sistema universitario español *"Un crédito ECTS equivale a 25 horas de trabajo del estudiante."* [22]. Este TFG tiene una carga de 12 créditos, lo que implica un total de 12 x 25 = 300 horas de trabajo por parte del estudiante. En base a esto, al tiempo disponible, y a las tareas a realizar para el proyecto, se optó por dividir el trabajo en 16 Sprints de 2 semanas de duración cada uno, sumando esto un total de 32 semanas. Cada semana se destinaron 10 horas al proyecto, lo que resultó en una duración total del proyecto de 10 x 32 = 320 horas, una cifra muy próxima a la estimada por el sistema universitario.

Con esta información y la relativa a las historias de usuario la planificación del trabajo en Sprints fue la siguiente:

- **SPRINT 1**: Realización de las historias de usuario 1 y 2.
- **SPRINT 2**: Realización de la historia de usuario 3.
- **SPRINT 3**: Realización de la historia de usuario 4.
- **SPRINT 4**: Realización de las historias de usuario 5 y 6.
- **SPRINT 5**: Realización de la historia de usuario 7.
- **SPRINT 6**: Realización de las historias de usuario 8 y 9.
- **SPRINT 7**: Realización de la historia de usuario 10.
- **SPRINT 8**: Realización de las historias de usuario 11 y 12.
- **SPRINT 9**: Realización de la historia de usuario 13.
- **SPRINT 10**: Realización de la historia de usuario 14.
- **SPRINT 11**: Realización de la historia de usuario 15.
- **SPRINT 12**: Realización de la historia de usuario 15.
- **SPRINT 13**: Realización de la historia de usuario 16.
- **SPRINT 14**: Realización de la historia de usuario 16.

- **SPRINT 15**: Realización de la historia de usuario 16.
- **SPRINT 16**: Realización de la historia de usuario 17.

Sin embargo, a pesar de esta planificación, surgieron pequeños imprevistos durante el desarrollo del proyecto que afectaron e impidieron cumplir al completo con la programación prevista. Estos imprevistos se presentaron durante el trascurso del Sprint 3, mientras se estaba trabajando sobre la creación de un contenedor en Singularity para almacenar todo lo necesario para la ejecución de la herramienta.

Durante el proceso, la alumna se encontró con dificultades a la hora de desarrollar este contenedor, y estas dificultades estaban consumiendo más tiempo del previsto. Además, se llegó a la conclusión de que quizás sería más conveniente realizar este paso una vez que todas las funcionalidades del proyecto estuvieran desarrolladas por completo. De esta manera, se podría crear un contenedor completo en lugar de uno incompleto, evitando así la necesidad de editar el contenedor posteriormente.

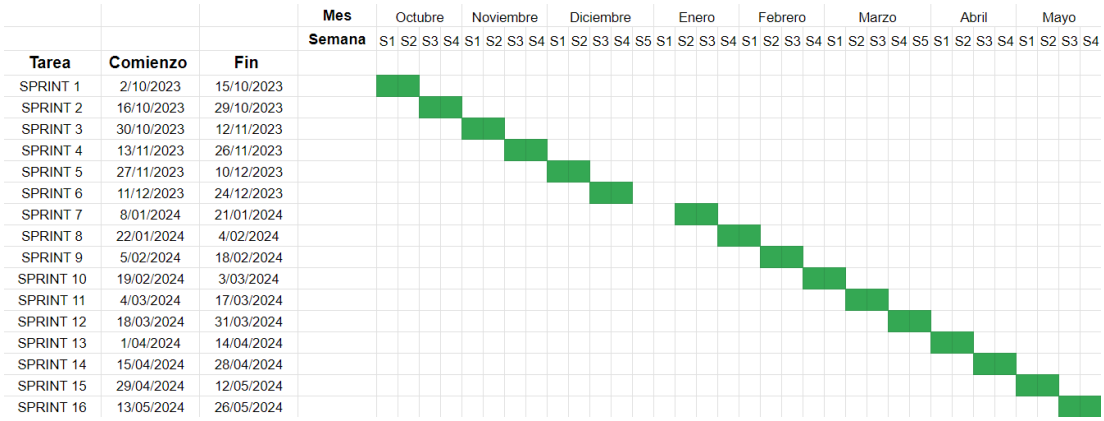
Cómo solución a esto, se realizó un ajuste a la planificación, añadiendo un sprint entre los que en la planificación inicial eran el sprint 12 y el 13, de manera que la planificación constó finalmente de 17 sprints y quedó de la siguiente manera:

- ...
- **SPRINT 12**: Realización de la historia de usuario 15.
- **SPRINT 13**: Realización de la historia de usuario 4.
- **SPRINT 14**: Realización de la historia de usuario 16.
- ...
- **SPRINT 17**: Realización de la historia de usuario 17.

Este ajuste en la planificación también tuvo un ligero impacto en la relación entre las horas trabajadas y los créditos ECTS. La duración total del proyecto fue finalmente de 17 sprints de 2 semanas de duración cada sprint, es decir, 34 semanas de duración total del proyecto. Estas 34 semanas multiplicadas por las 10 horas destinadas al proyecto en cada semana, resulta en una duración total del proyecto de 10 x 34 = 340 horas de trabajo. Aunque esta cifra excede ligeramente la estimación inicial, sigue siendo razonable en comparación con lo establecido por el sistema universitario.

4.4.2 Diagrama de Gantt

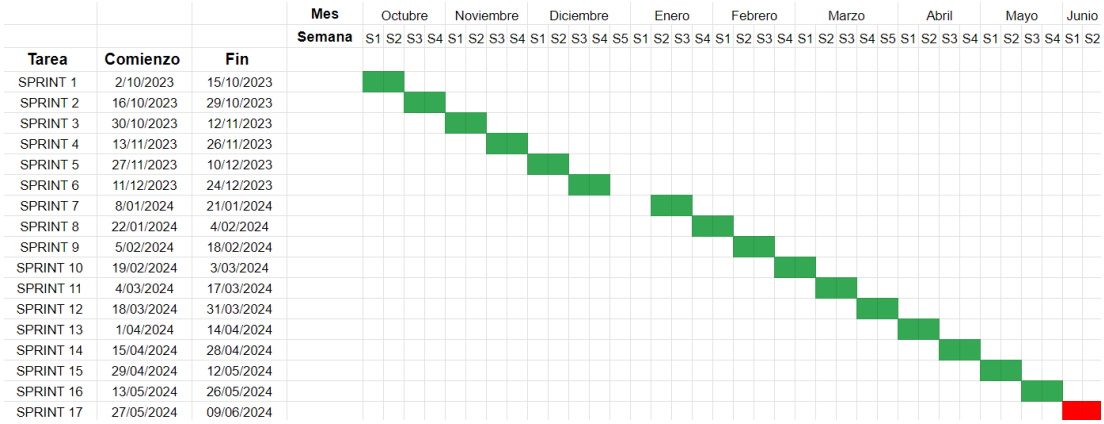
En este apartado se presentan dos diagramas de Gantt: uno correspondiente a la planificación inicial del proyecto y otro reflejando la ejecución real. Estos diagramas ilustran el trabajo realizado a lo largo del proyecto, permitiendo una comparación clara entre lo planificado y lo finalmente ejecutado.



Tarea	Comienzo	Fin
SPRINT 1	2/10/2023	15/10/2023
SPRINT 2	16/10/2023	29/10/2023
SPRINT 3	30/10/2023	12/11/2023
SPRINT 4	13/11/2023	26/11/2023
SPRINT 5	27/11/2023	10/12/2023
SPRINT 6	11/12/2023	24/12/2023
SPRINT 7	8/01/2024	21/01/2024
SPRINT 8	22/01/2024	4/02/2024
SPRINT 9	5/02/2024	18/02/2024
SPRINT 10	19/02/2024	3/03/2024
SPRINT 11	4/03/2024	17/03/2024
SPRINT 12	18/03/2024	31/03/2024
SPRINT 13	1/04/2024	14/04/2024
SPRINT 14	15/04/2024	28/04/2024
SPRINT 15	29/04/2024	12/05/2024
SPRINT 16	13/05/2024	26/05/2024

Figura 4.2: Diagragrama de Gantt planificación inicial

En la Figura 4.2 se presenta la planificación inicial del proyecto, que consta de 16 sprints de dos semanas cada uno. Cabe destacar un leve desplazamiento entre los Sprints 6 y 7 debido a un período de descanso de dos semanas durante las vacaciones de Navidad.



Tarea	Comienzo	Fin
SPRINT 1	2/10/2023	15/10/2023
SPRINT 2	16/10/2023	29/10/2023
SPRINT 3	30/10/2023	12/11/2023
SPRINT 4	13/11/2023	26/11/2023
SPRINT 5	27/11/2023	10/12/2023
SPRINT 6	11/12/2023	24/12/2023
SPRINT 7	8/01/2024	21/01/2024
SPRINT 8	22/01/2024	4/02/2024
SPRINT 9	5/02/2024	18/02/2024
SPRINT 10	19/02/2024	3/03/2024
SPRINT 11	4/03/2024	17/03/2024
SPRINT 12	18/03/2024	31/03/2024
SPRINT 13	1/04/2024	14/04/2024
SPRINT 14	15/04/2024	28/04/2024
SPRINT 15	29/04/2024	12/05/2024
SPRINT 16	13/05/2024	26/05/2024
SPRINT 17	27/05/2024	09/06/2024

Figura 4.3: Diagragrama de Gantt ejecución final

Por su parte, en la Figura 4.3 se muestra la planificación final del proyecto, la cual es muy similar a la inicial. Sin embargo, se añade un Sprint adicional, señalado en color rojo, debido a los problemas encontrados con el manejo de Singularity. Como resultado, el proyecto constó finalmente de 17 Sprints.

4.5 Costes

En esta sección se detalla la estimación de costes del proyecto, considerando el número de sprints y las horas de trabajo necesarias para completar cada uno. Además, se incluirá el coste asociado al hardware necesario, como el ordenador utilizado para el desarrollo del proyecto.

Para la estimación de los costes del trabajo realizado en el proyecto, se han comparado varias fuentes para determinar los salarios medios de un Ingeniero de Datos Junior y un Scrum Master en España [23, 24]. Se encontró que el sueldo medio de un Ingeniero de Datos Junior es de 24.000€ brutos anuales, mientras que el sueldo medio de un Scrum Master es de 42.700€ brutos anuales.

Para calcular los costes del proyecto por horas, se considera que un año tiene aproximadamente 273 días laborables, y cada día laborable se trabajan 8 horas. El coste por hora de trabajo se establece en 20€ para el Scrum Master y 11€ para el Ingeniero de Datos Junior.

Dado que hay tres tutores, se estima que el proyecto contará con tres Scrum Masters y un Ingeniero de Datos Junior. Por lo tanto, los costes se calculan en base a esta distribución de roles y, además, se incluyen los costes de hardware, que será el coste relativo al precio del ordenador utilizado por el estudiante durante la ejecución del proyecto.

Con estos datos, se obtiene lo siguiente:

Función	Coste por hora	Horas	Coste total
Ingeniero de datos	11	340	3740€
Scrum Master	20	30	600€
Scrum Master	20	30	600€
Scrum Master	20	30	600€
Hardware			800€
			6340€

Tabla 4.1: Estimación de costes del proyecto

Como se puede observar en la Tabla 4.1, la estimación de coste total del proyecto es de 6340€.

Capítulo 5

Diseño y desarrollo de la solución

Esta sección constituye el apartado central del proyecto, donde se detalla exhaustivamente su desarrollo. Este núcleo del trabajo se subdivide en los Sprints mencionados anteriormente, y cada sección ofrece una explicación detallada de las actividades realizadas durante el Sprint correspondiente.

Como se mencionó en la introducción de este proyecto, el objetivo principal fue abordar la eliminación de los efectos de lote en cohortes de microbioma para estudios de cáncer colorrectal, con el fin de poder estudiar de forma global varios conjuntos de datos. Normalmente, esto no es posible debido a las diferencias externas entre ellos (efectos de lote). Al eliminar estas diferencias, es posible realizar estudios que proporcionen mayor cantidad de información, y en consecuencia que esta sea más relevante y detallada. Para lograr esto, se ha adaptado y utilizado la herramienta existente MOBER [4].

A pesar de haber sido mencionados en varios puntos previamente, es fundamental comprender con mayor profundidad qué son los efectos de lote. En el contexto de los datos de microbioma, los efectos de lote se refieren a variaciones que surgen en los perfiles microbiómicos. Estas variaciones pueden manifestarse, por ejemplo, en diferencias en la abundancia relativa de distintos microorganismos presentes en las muestras, en la presencia o ausencia de ciertas especies, entre otros aspectos.

Estas variaciones pueden surgir debido tanto a las diferencias biológicas entre las comunidades microbianas en diferentes muestras como a los efectos de lote. Si estas variaciones son el resultado de diferencias biológicas, no interferirán con el estudio, ya que reflejan la verdadera diversidad microbiológica entre las muestras. Sin embargo, si son atribuibles a efectos de lote, pueden complicar el análisis y la interpretación de los datos.

Los efectos de lote surgen como resultado de factores técnicos o experimentales, como la manipulación de las muestras durante el procesamiento, la variabilidad en los protocolos de secuenciación, diferencias en la calidad de los reactivos utilizados, condiciones de laboratorio y almacenamiento de muestras, entre otros.

Identificar cuándo estas variaciones son resultado de diferencias biológicas y cuándo son atribuibles a efectos de lote es crucial para llevar a cabo un estudio preciso de los datos de microbioma. La eliminación de estos efectos facilita enormemente el análisis de los datos, lo cual es, al fin y al cabo, el objetivo principal de su eliminación.

Es por este motivo que el proyecto se centra en esta tarea como objetivo central, ya que aunque pueda parecer superfluo, la correcta identificación y eliminación de los efectos de lote garantizará la fiabilidad y precisión de los resultados obtenidos en el análisis de datos de microbioma.

Una vez se haya estudiado la eliminación de los efectos de lote, se evaluará si esta eliminación añade valor al análisis de los datos, aumentando la cantidad de señal biológica capturada.

Cómo apoyo y complemento a estas tareas principales, se realizan comprobaciones intermedias, análisis de resultados y visualizaciones gráficas para evaluar y comprender los avances en el proceso. Estas actividades complementarias ayudan a alcanzar una conclusión sólida sobre la tarea principal y a obtener una comprensión más profunda acerca los resultados obtenidos.

Teniendo en cuenta esto, cada sección de este capítulo detalla el proceso de trabajo y los avances logrados hasta llegar a una conclusión firme tanto sobre la eliminación de los efectos de lote, como de su posible repercusión en el análisis de los datos.

5.1 SPRINT 1 - Lectura de documentación y creación de repositorio en GitHub

En el primer Sprint, se llevó a cabo una primera toma de contacto con el tema a tratar. Esto implicó la lectura de documentación relacionada con el tema del proyecto y la creación de un repositorio en GitHub, que se utilizó como plataforma centralizada para almacenar, gestionar y colaborar en el desarrollo del proyecto de manera eficiente.

Esta fase inicial sentó las bases para el desarrollo del proyecto, permitiendo a la alumna familiarizarse con el tema y establecer un punto de partida sólido para el trabajo futuro.

5.1.1 Documentación

En cuanto a la lectura de documentación, los tutores proporcionaron a la alumna diversos artículos de investigación sobre MOBER, microbioma, cáncer de colon, entre otros temas relevantes. Además, la alumna realizó una búsqueda independiente para recopilar información sobre las tecnologías que se utilizarían en el proyecto, como los paquetes de RStudio necesarios y el funcionamiento de Singularity.

Entre los artículos proporcionados por los tutores, dos de ellos destacaron por su relevancia para el desarrollo del proyecto. El primero de ellos, titulado *"Biologically relevant integration of transcriptomics profiles from cancer cell lines, patient-derived xenografts and clinical tumors using deep learning"* [25], fue especialmente importante debido a su explicación y discusión sobre MOBER [4].

Este documento es la referencia oficial proporcionada por los creadores de MOBER sobre la herramienta. En él se explican los fundamentos y el funcionamiento de la misma, así como sus diversos casos de uso. Además, se ofrece un breve resumen sobre cómo los creadores aplicaron MOBER en su investigación, junto con los resultados obtenidos. También se incluye un conjunto de gráficas detallando el proceso, lo que brinda una visión muy completa de la aplicación de MOBER y sus resultados.

Esta herramienta, como se presentó en la introducción del trabajo, es un método de eliminación de efectos de lote que tiene como objetivo extraer simultáneamente aspectos biológicamente significativos (señal biológica) y eliminar los efectos de lote de conjuntos de datos transcriptómicos de diferentes orígenes. Esto se logra a través de dos redes neuronales: un autoencoder variacional condicional y una red neuronal discriminadora de origen.

El autoencoder variacional condicional (cVAE) es una arquitectura de red neuronal que, además de reconstruir los datos de entrada, incorpora información condicional para capturar aspectos específicos y significativos de los datos. Mientras tanto, la red neuronal discriminadora de origen es un componente esencial en este proceso, encargado de distinguir entre muestras del dominio de origen y del dominio de destino. Al trabajar conjuntamente, estas dos redes permiten que el método extraiga simultáneamente características biológicamente relevantes y elimine los efectos de lote, lo que resulta en una mejor comprensión de la biología presente en los conjuntos de datos con los que se trabaja.

A pesar de que MOBER se ha desarrollado principalmente para trabajar con datos de transcriptoma, desempeñó un papel crucial en el desarrollo del proyecto, siendo un punto clave

para la alumna en el trabajo de eliminación de efectos de lote en datos de microbioma y el posterior estudio de la señal biológica captada por los mismos.

El segundo artículo significativo fue *"Managing batch effects in microbiome data"* [26], un documento publicado en el NIH que aborda la presencia de efectos de lote en datos de microbioma y sus diversas fuentes de origen. Este estudio proporciona ejemplos concretos de estudios de microbioma en los que se identificaron y cuantificaron los efectos de lote y, además, ofrece métodos y estrategias tanto para el análisis de datos de microbioma como para la gestión y eliminación de estos efectos de lote.

Este artículo proporcionó una comprensión más amplia acerca del contexto de los efectos de lote en datos de microbioma y las herramientas para manejarlos. Aunque su enfoque fue más general, esta información fue fundamental para comprender aspectos específicos asociados con los datos de microbioma y su relación con los efectos de lote, y orientar de esta manera el proyecto.

Además de los dos documentos destacados, como se mencionó anteriormente, la alumna realizó también la lectura y estudio de documentación adicional. Alguna relacionada con los aspectos ya mencionados, como microbioma y efectos de lote, y otra sobre tecnologías y herramientas que se utilizarían a lo largo del proyecto.

Dentro de esta documentación se incluye el portal sobre el *"Human Microbiome Project (HMP)"* [6] también del NIH, recurso que permitió a la alumna ampliar su conocimiento y comprensión acerca del tema de estudio y profundizar en la investigación activa sobre la función de los microbiomas humanos en diversas partes del cuerpo; y algunos fragmentos de la tesis *"Chemical and microbial ecology of the demosponge Aplysina aerophoba"* [8], de la cual se extrajo información relevante sobre un estudio con datos de microbioma en el que se identificaron efectos de lote.

Dentro de la documentación del aspecto tecnológico, se incluyó un tutorial sobre Singularity [27] que detalla el propósito de esta herramienta, sus diversos usos y cómo utilizarla eficazmente. Además, se realizaron diversas lecturas sobre los paquetes de R y Python más relevantes para el proyecto, esto incluyó tutoriales sobre phyloseq [28, 29] y mlr3 [30] en R, así como sobre pandas [31] y AnnData [32] en Python.

La lectura de toda esta documentación resultó sumamente útil tanto en la obtención de contexto como en la optimización del progreso del trabajo. Además, fue especialmente útil du-

rante el desarrollo de esta memoria, destacándose en los apartados 2 de Estado del Arte y 3 de Herramientas.

5.1.2 GitHub

Como se mencionó en la sección 3, GitHub fue una de las herramientas clave utilizadas para respaldar el desarrollo del proyecto.

Se creó un repositorio en GitHub con acceso para la alumna y los tutores. Este repositorio se utilizó como una plataforma centralizada para subir y compartir los progresos del proyecto. Su propósito fue permitir a los tutores verificar los avances de la alumna y validar o corregir lo que la alumna iba desarrollando.

Además, en él se incluyó un breve README, donde se proporciona una explicación sobre los avances realizados en el proyecto, así como una descripción general del contenido y la estructura del repositorio.

5.2 SPRINT 2 - Estudio del funcionamiento del CESGA

La Fundación Pública Galega Centro Tecnolóxico de Supercomputación de Galicia (CESGA) [33] es el centro de cálculo, comunicaciones de altas prestaciones y servicios avanzados de la Comunidad Científica Gallega, Sistema Académico Universitario y del Consejo Superior de Investigaciones Científicas (CSIC).

Para este proyecto, se hizo uso del CESGA a través del entorno de supercomputación que ofrece, interfaz donde se llevó a cabo la ejecución de los grandes experimentos del proyecto. Este entorno proporcionó la infraestructura necesaria para ejecutar las tareas computacionalmente intensivas requeridas por el proyecto.

Además, se utilizó el CESGA para probar el funcionamiento del contenedor creado en Singularity. Este contenedor, como ya se mencionó, fue utilizado para ejecutar las funcionalidades desarrolladas en el proyecto. Destacar que el CESGA facilita este proceso al proporcionar un módulo de Singularity integrado en su infraestructura.

La alumna siguió los siguientes pasos para usar el CESGA:

- Solicitud de acceso al sistema completando el formulario correspondiente.
- Acceso al portal de usuarios mediante las credenciales proporcionadas.

- Apertura de una terminal o un escritorio, según preferencia.
- Importación de los datos necesarios desde el equipo local para poder trabajar con ellos en el CESGA.
- Realización de los desarrollos pertinentes en este entorno de supercomputación.
- Exportación de datos y/o resultados en caso de ser necesario.
- Cierre de sesión.

5.3 SPRINT 3 - Creación de contenedor en Singularity

Como se mencionó anteriormente, Singularity desempeñó una función muy importante en este TFG, siendo otro de los softwares de apoyo para la realización del proyecto. Se utilizó como una herramienta para la portabilidad de las funciones desarrolladas, permitiendo encapsular el entorno y las dependencias del proyecto, a través de un contenedor.

Este contenedor, creado con Singularity, facilitó la ejecución de las funcionalidades del proyecto de manera sencilla y directa para cualquier persona que tenga acceso al mismo. Esto significa que los usuarios pueden ejecutar las funcionalidades del proyecto en sus propios entornos sin tener que preocuparse por configurar o instalar las dependencias necesarias, lo que proporciona una mayor accesibilidad y reproducibilidad de los resultados del proyecto.

Para la creación del contenedor de singularity, a grandes rasgos, los pasos a seguir por la alumna fueron:

1. Creación de un ”recipe” de Singularity, que se trata de un archivo de texto que contiene las instrucciones necesarias para construir un contenedor Singularity, incluyendo la instalación de todos los paquetes y dependencias necesarias para su posterior ejecución.
2. Construcción de una imagen .sif (imagen del contenedor Singularity). Durante este paso, se incluyeron todos los elementos especificados en el ”recipe” para garantizar que la imagen resultante contenga todo lo necesario para su funcionamiento.
3. Exportación de la imagen al equipo local de la alumna para su posterior uso.

Por último, importante señalar que, como se mencionó en la sección 4.4 de planificación del proyecto, la tarea principal prevista para este sprint presentó algunas complicaciones. Después de enfrentar estas dificultades y reflexionar sobre la mejor manera de proceder, se llegó a la conclusión de que sería más beneficioso posponer esta tarea hasta tener desarrolladas todas

las funcionalidades del proyecto. Por lo tanto, se realizó un reajuste en la planificación de los Sprints.

Con esto, gran parte de las actividades descritas en este apartado, a pesar de haber sido abordadas en este sprint, se desarrollaron de manera más efectiva en el sprint 13, sección 5.12.

5.4 SPRINT 4 - Modificaciones en MOBER y puesta en marcha de la herramienta

Una vez completado el estudio del funcionamiento de las diversas infraestructuras y herramientas necesarias para el proyecto, se procedió con el desarrollo.

Como se mencionó en el Sprint 1, sección 5.1, MOBER fue un punto crucial y el punto de partida para la solución de este TFG. Para su utilización, se hizo un clonado de los ficheros de la herramienta desde GitHub y se procedió a la instalación de las dependencias y paquetes necesarios para su posterior uso.

Además de poner en marcha la herramienta, se llevó a cabo un análisis exhaustivo de su funcionamiento, examinando detalladamente tanto sus archivos como su documentación. Esto proporcionó una comprensión profunda de su estructura y capacidades.

Como se comentó, el funcionamiento de MOBER se basa en el uso de dos redes neuronales: un codificador automático variacional condicional y una red neuronal discriminadora de fuente entrenada adversariamente. Estas redes operan conjuntamente para preservar las señales biológicas presentes en los conjuntos de datos originales, al mismo tiempo que clasifican los datos. El proceso genera embeddings o representaciones numéricas de los datos en un espacio dimensional reducido. Estas representaciones mantienen las propiedades y características esenciales de los datos originales, a la vez que eliminan la información confusa o redundante.

Cuando se menciona que las embeddings generadas no codifican información de confusión, significa que estas representaciones tienen la capacidad de capturar las características biológicas esenciales de los datos, separándolas de cualquier variabilidad no deseada o ruido presente en los conjuntos de datos originales donde, en este caso concreto, se sitúan los efectos de lote, que provienen de juntar varias cohortes diferentes para hacer un estudio global.

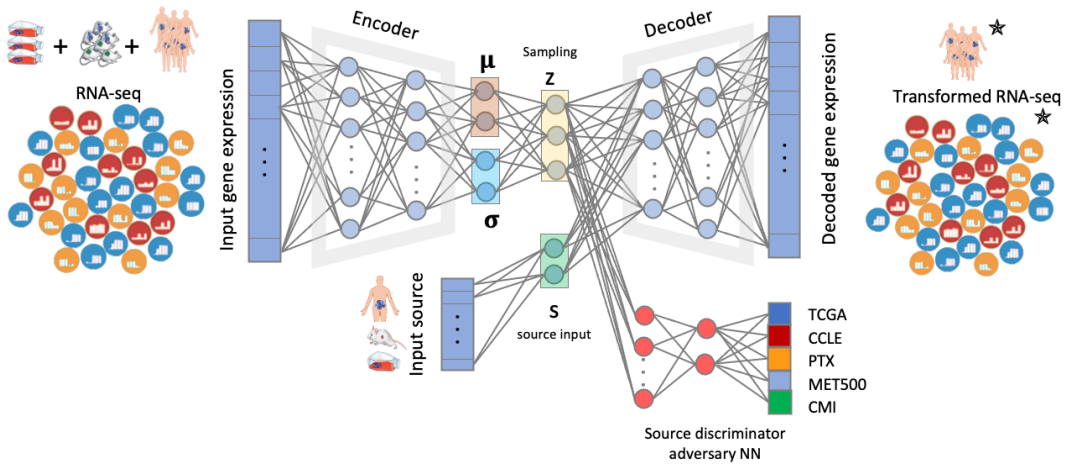


Figura 5.1: Funcionamiento de MOBER

En la Figura 5.1 se muestra una representación más detallada del funcionamiento interno de MOBER.

Para la utilización de MOBER, la documentación de la herramienta proporciona comandos de instalación que permiten configurarla correctamente en el entorno de trabajo. Una vez instalada, se requiere un objeto en formato .h5ad para aplicar los comandos también proporcionados y hacer así uso de la herramienta. Estos comandos incluyen funciones para entrenar y proyectar sobre el conjunto de entrenamiento, los cuales cuentan con diversos parámetros de entrada, modificables según el conjunto de datos específico y el estudio a realizar.

Además, el código fuente de la herramienta, que se ejecuta para entrenar el modelo, incluye una serie de hiperparámetros, los cuales son adaptables a las necesidades de cada situación específica. La adaptación de estos hiperparámetros al caso de estudio de este proyecto fue uno de los retos de este Sprint.

Los hiperparámetros presentes en MOBER son los siguientes:

1. **use_sparse_mat**: Booleano que indica si se debe utilizar un dataloader disperso o no.
2. **src_adv_weight**: Peso de la pérdida adversarial de origen.
3. **src_adv_lr**: Tasa de aprendizaje para la pérdida adversarial de origen.
4. **batch_ae_lr**: Tasa de aprendizaje para el lote de autoencoders.
5. **val_set_size**: Fracción de muestras que constituyen el conjunto de validación.

6. **encoding_dim**: Tamaño de los embeddings.
7. **balanced_sources_ae**: Indicador booleano que habilita pesos de muestra para equilibrar según la fuente en la pérdida de autoencoder.
8. **balanced_sources_src_adv**: Indicador booleano que habilita pesos de muestra para equilibrar según la fuente en la pérdida adversarial de origen.
9. **batch_size**: Tamaño del lote durante el entrenamiento.
10. **epochs**: Número máximo de ciclos completos de entrenamiento.
11. **random_seed**: Semilla aleatoria para reproducibilidad.
12. **kl_weight**: Peso para la pérdida de divergencia de Kullback-Leibler (KL).
13. **patience**:Número de épocas durante las cuales el entrenamiento continuará si no se observa una mejora en el rendimiento en el conjunto de validación.
14. **output_dir**: Ruta de salida.

Se estudiaron con detenimiento estos puntos y sus correspondientes valores en el código MOBER original para determinar la mejor manera de adaptarlos a los datos con los que se iba a trabajar. A pesar de no contar en este punto con el conjunto de datos a entrenar, se pudo hacer una estimación de los ajustes más convenientes. Como resultado de este análisis, se optó por realizar modificaciones en dos parámetros específicos: *encoding_dim* y *batch_size*.

La variable *encoding_dim* se ajustó para controlar la dimensionalidad en la que se representa la información, mientras que la variable batch_size se modificó con el objetivo de mejorar la eficiencia del proceso de entrenamiento del modelo. Se realizó una reducción significativa en ambos valores teniendo en cuenta el tamaño y la dimensión de los datos con los que se iba a trabajar. El objetivo de estos cambios fue optimizar tanto el rendimiento del modelo como su precisión. Sin embargo, es importante destacar que, en ese momento, estos cambios estaban todavía sujetos a modificaciones, una vez se observara cómo funcionan con el conjunto de datos específico con el que se trabajó.

Una vez estudiada toda esta información, la alumna se dedicó al otro reto de este Sprint, la construcción de una matriz de datos aleatorios de prueba, y el uso de MOBER con estos datos, tanto con los valores de los parámetros por defecto como con los valores después de las modificaciones realizadas, queriendo observar con esto si el modelo funcionaba de manera similar con las modificaciones, sin esperar en ningún caso resultados significativos debido a

la naturaleza aleatoria de los datos.

Por lo tanto, el objetivo principal de esta parte del proceso fue probar el funcionamiento básico de la herramienta, evaluar sus parámetros e hiperparámetros, así como las modificaciones realizadas sobre estos, y adquirir un mayor contexto sobre su uso.

Para la construcción de la matriz, se empleó Python y, una vez creada, se ejecutó MOBER con esa matriz como entrada. Este proceso sirvió para familiarizarse con el funcionamiento de la herramienta, así como con sus parámetros de entrada y sus outputs. Además, se observó que los resultados obtenidos no variaban significativamente entre las diferentes configuraciones de hiperparámetros, lo que dio pie a pensar que no se estaban realizando modificaciones sin sentido.

5.5 SPRINT 5 - Selección de cohortes y preprocesado de los datos

Durante el Sprint número 5, la alumna se enfocó en una tarea fundamental para el desarrollo del proyecto: la selección de cohortes a incluir en el estudio y el preprocesado de las mismas. Además, y una vez hecho esto, las cohortes de microbioma seleccionadas se volvieron a procesar con el objetivo de aplanarlas y quedarse sólamente con los datos necesarios para el posterior estudio.

Las llamadas cohortes de microbioma representan grupos de individuos que comparten características similares y que son estudiados en relación con la composición y función de su microbioma. En este contexto, el término ”cohorte” se refiere a grupos de personas que comparten ciertas características o criterios de inclusión en el estudio.

Para la búsqueda de las cohortes a incluir en el estudio se utilizó el European Nucleotide Archive (ENA) [34]. El ENA es una base de datos que proporciona acceso gratuito y sin restricciones a secuencias anotadas de ADN y ARN. Además, almacena información complementaria como procedimientos experimentales, detalles del ensamblaje de secuencias y otros metadatos relacionados con proyectos de secuenciación. Estos datos son utilizados por investigadores de todo el mundo en una variedad de campos, incluyendo la genómica, la bioinformática, la evolución molecular y la biología molecular. Los datos almacenados en el ENA están disponibles públicamente para su consulta y descarga, lo que facilitó en gran medida la recopilación de los datos a incluir en el estudio que se desarrolla en este proyecto.

Para obtener las cohortes, se siguieron varios pasos. Inicialmente, se accedió a las secuencias de ARNr 16S en el ENA utilizando herramientas específicas para la búsqueda y descarga de datos. En este caso, se utilizó enaBrowserTools, un conjunto de herramientas desarrolladas por el ENA para facilitar la visualización y el análisis de los datos almacenados en su plataforma. Luego, se aplicaron filtros y criterios específicos para seleccionar las secuencias adecuadas para el estudio en desarrollo.

Una vez obtenidas las secuencias, se procedió a su procesamiento utilizando software bioinformático especializado, en este caso, DADA2. El uso de esta herramienta fue fundamental, ya que ayudó a eliminar errores y a agrupar secuencias similares en *amplicon sequence variants* (ASVs). El objetivo fue obtener un conjunto de datos homogéneo, comprensible y en un formato fácil de manejar.

Una vez procesadas las ASVs, estas fueron convertidas a unidades taxonómicas operativas (OTUs), que se asignaron taxonómicamente utilizando la base de datos de referencia Silva (v138.1) [35]. Al finalizar este proceso se tenían las cohortes que se utilizaron en el estudio, listas para poder trabajar con ellas.

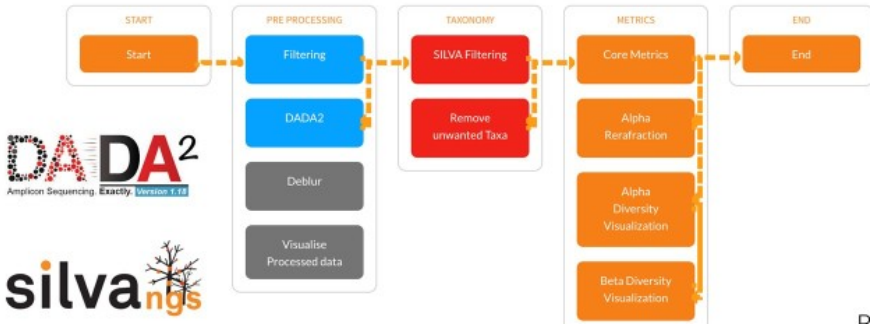


Figura 5.2: Flujo de preprocesado de datos [3]

En la Figura 5.2 se muestra un resumen global del flujo seguido para el preprocesado de los datos.

A través de los llamados ENA IDs, que son identificadores únicos asignados a proyectos, secuencias, muestras u otros datos alojados en la plataforma, se llevó a cabo un estudio de extracción de información acerca de las cohortes con las que se iba a trabajar en el proyecto. Estos identificadores se utilizaron por lo tanto para recopilar información detallada sobre las muestras, proyectos y metadatos asociados. Este proceso proporcionó un marco estructurado para comprender la procedencia y la composición de los datos que se utilizarían en el proyec-

to, facilitando su análisis y la interpretación de los resultados obtenidos.

Estas cohortes fueron:

- **PRJEB33634_B1**: Primer lote de muestras de microbioma asociadas al proyecto PRJEB33634, desarrollado por el centro CIC BIOGUNE y dividido en 3 lotes, que se centra en secuenciación de ADNr 16S para control, adenoma avanzado y cáncer colorrectal para biomarcadores tempranos de patogénesis y desarrollo de enfermedades.
- **PRJEB33634_B2**: Seguno lote de muestras de microbioma también asociadas al proyecto PRJEB33634.
- **PRJEB33634_B3**: Tercer lote de muestras de microbioma también asociadas al proyecto PRJEB33634.
- **PRJEB6070_F**: Conjunto de muestras fecales utilizadas en estudios de secuenciación metagenómica asociadas al proyecto PRJEB6070. En este proyecto se realizó un estudio de secuenciación metagenómica de muestras fecales. El objetivo del estudio fue identificar marcadores taxonómicos que pudieran distinguir entre pacientes con cáncer colorrectal (CCR) y controles sin tumores en una población de estudio de 156 participantes. En este caso, el sufijo "_F" indica que las muestras utilizadas en el estudio fueron obtenidas en Francia.
- **PRJEB6070_G**: Conjunto de muestras de tejido asociadas, de la misma manra que las anteriores, al proyecto PRJEB6070. En este caso, el sufijo "_G" indica que las muestras utilizadas en el estudio fueron obtenidas en Alemania.
- **PRJNA290926**: Conjunto de muestras de metagenoma del intestino humano utilizado el proyecto PRJNA290926 desarrollado por la universidad de Michigan, basado en la utilización de modelos basados en microbiomas para diferenciar pacientes sanos de aquellos con lesiones colónicas.
- **PRJNA763023_H**: Conjunto de muestras de metagenoma del intestino humano, uilizadas en el desarrollo del proyecto PRJNA763023 desarrollado en Huadong (de ahí el sufijo H), basado en la caracterización de manera integral de la asociación entre la disbiosis del microbioma intestinal y los riesgos de CCR en grupos de jóvenes o ancianos. En él se destaca el gran potencial de los biomarcadores de la microbiota intestinal como una herramienta no invasiva prometedora para la detección y distinción precisas de la aparición del CCR según la edad.

- **PRJNA67873**: Muestras pareadas de tumores de colon y ADN de colon normal mediante secuenciación microbiana de ADNr 16S utilizadas en el proyecto PRJNA67873 para determinar la composición microbiana del cáncer colorrectal humano.
- **PRJNA318004**: Conjunto de muestras de metagenoma del intestino humano, uilizadas en el desarrollo del proyecto PRJNA318004, basado en la utilización de ADN extraído de cartuchos de pruebas inmunoquímicas fecales, en lugar de directamente de las heces, para detectar el cáncer colorrectal basándose en el microbioma intestinal.
- **Zackular**: Esta cohorte específica no está identificada por un ENA ID ya que se trata de una cohorte individual, no vinculada a ningún proyecto documentado en el ENA. Se utilizó en estudios publicados en el NIH sobre el potencial de la microbiota fecal para la detección temprana del cáncer colorrectal.

Una vez seleccionadas, estudiadas y preprocesadas las cohortes, se pasaron a formato .rds para poder seguir trabajando con ellas en R.

Se dispone por lo tanto en este momento de 10 cohortes en formato .rds. Partiendo de esto el siguiente paso consistió en continuar el procesamiento de los datos utilizando R. El objetivo de esto fue el obtener un conjunto de datos plano y sencillo que facilitara el trabajo con la herramienta y permitiera trabajar de manera eficiente.

Para iniciar este procesamiento de las muestras, se comenzó accediendo a RStudio y cargando las 10 muestras desde sus respectivos archivos. Una vez cargadas, se procedió a crear un objeto phyloseq (estructura de datos especializada en el análisis de datos de microbioma en R) por cada muestra, con ayuda del paquete ”phyloseq” de R. El término ”objeto phyloseq” se repetirá de forma recurrente durante este trabajo, ya que es el tipo de objeto principal con el que se trabajó en R durante todo el proyecto.

Estos objetos phyloseq almacenan los datos de las muestras una vez procesados (generalmente en forma de ASVs u OTUs) y otros importantes datos asociados. En el caso de este proyecto, los objetos phyloseq con los que se trabaja contienen:

- **otu_table**: Representa la abundancia relativa de cada OTU en cada muestra. Consiste en una tabla de datos con muestras en filas y OTUs en columnas. Una OTU (Operational Taxonomic Unit) es una unidad utilizada para representar agrupaciones de secuencias de ADN o ARN que comparten un nivel específico de similitud, generadas normalmente mediante métodos de clustering. Los valores en esta tabla representan la abundancia relativa de cada OTU en cada muestra.

- **sample_data**: Incluye información sobre las muestras, como condiciones experimentales, variables ambientales, información de tiempo, etc.
- **tax_table**: Proporciona información sobre la taxonomía de las OTUs, es decir, a qué categoría taxonómica pertenece cada OTU (filo, familia, género, especie, etc).

```
phyloseq-class experiment-level object
otu_table()   OTU Table:         [ 7526 taxa and 129 samples ]
sample_data() Sample Data:       [ 129 samples by 21 sample variables ]
tax_table()   Taxonomy Table:    [ 7526 taxa by 7 taxonomic ranks ]
```

Figura 5.3: Objeto phyloseq generado para una de las cohortes

En la Figura 5.3 se observa la forma y el tamaño de uno de los objetos phyloseq creado para una de las cohortes.

Una vez construidos estos objetos, y de cara a garantizar la consistencia y la calidad de los datos antes de proceder con su análisis, se procedió a realizar el procesamiento y limpieza de los datos. Los pasos seguidos fueron los siguientes:

1. Filtrar y almacenar únicamente las cohortes con muestras de tipo ”feces”.
2. Aglomerar los datos por el campo ”Genus” (agrupación o sumarización de datos a nivel de género microbiológico).
3. Eliminar géneros sin identificar (NA).
4. Identificar y filtrar por géneros comunes entre todos las cohortes.
5. Unir los objetos obtenidos en los pasos anteriores en uno solo.
6. Filtrar los datos con status = adenoma, ya que no son de interés para el estudio y solo añadirían ruido.

```
phyloseq-class experiment-level object
otu_table()   OTU Table:         [ 175 taxa and 934 samples ]
sample_data() Sample Data:       [ 934 samples by 57 sample variables ]
tax_table()   Taxonomy Table:    [ 175 taxa by 8 taxonomic ranks ]
```

Figura 5.4: Objeto phyloseq generado para una de las cohortes una vez procesadas

Una vez completado el procesamiento de los datos, las 10 cohortes iniciales se redujeron a 7, a partir de las cuales se obtuvo el objeto phyloseq con el que se trabajó en el resto del estudio, mostrado en la Figura 5.4, con un conjunto de 175 géneros, divididos en 8 rangos taxononómicos, con 934 muestras de 57 variables distintas.

Estas siete cohortes obtenidas corresponden a las siguientes: PRJEB33634_B1, PRJEB33634_B2, PRJEB33634_B3, PRJEB6070_F, PRJNA290926, PRJNA763023_H y Zackular; que fueron, por lo tanto, las utilizadas en el resto del estudio.

Entrando en más detalle sobre el contenido de estos objetos phyloseq finales, se destacan los siguientes elementos:

- **otu_table**: Esta tabla contiene las abundancias de cada OTU para cada muestra del estudio. El formato consiste en una matriz numérica que, a simple vista, no proporciona mucha información interpretativa.
- **sample_data**: Este componente contiene los metadatos asociados con cada muestra. Entre estos metadatos, los más relevantes son: el tipo de muestra (*sample_type*), que después del procesamiento debe ser consistente (en este caso, ”feces” para todas las muestras), el estado del paciente (*status*), con categorías ”cáncer” y ”normal” (ya que la categoría ”adenoma” fue eliminada durante el procesamiento de los datos), y que es una de las variables utilizadas en los análisis del proyecto; y el identificador de la cohort (*project*), que es otra variable clave en los análisis y esencial para determinar si los efectos de lote se eliminaron correctamente.
- **tax_table**: Contiene información sobre los niveles taxonómicos de las OTUs, que son 8. Estos niveles incluyen reino, filo, clase, orden, familia, género y especie. Además, se ha añadido una columna (fam_gen) que concatena los valores de familia y género, y que sirve como método de identificación de los microorganismos.

Posteriormente, la alumna generó archivos CSV para las tabla de OTUs y los datos de muestra del objeto phyloseq. Estos archivos fueron exportados a su equipo para poder trabajar en el siguiente punto del proyecto, que consistió en la generación del archivo .h5ad en Python, necesario como entrada para MOBER.

5.6 SPRINT 6 - Extracción y conversión de matrices de datos y ejecución de MOBER con estos datos

Partiendo de los archivos CSV obtenidos en el anterior sprint, el objetivo del Sprint 6 fue la creación del objeto .h5ad de entrada a MOBER, la ejecución de MOBER con ese input y la observación y estudio de los resultados obtenidos.

Para la creación de ese objeto de entrada, se empleó Python, una de las herramientas comentadas en la sección 3 y fundamental para este proyecto. Se emplearon las bibliotecas *pandas* [31]

y *anndata* [32] de Python. Pandas es una biblioteca especializada en el manejo y análisis de estructuras de datos, mientras que anndata está diseñada para el análisis de datos de células individuales (scRNA-seq).

Con la ayuda de las biliotecas mencionadas, se implementó un bucle para iterar sobre los archivos CSV de las tabla de OTU y los datos de muestra exportados. Estos datos fueron cargados en un objeto AnnData en Python. Una vez completado este proceso, a partir de ese objeto, se creó el archivo final .h5ad. El formato .h5ad es ampliamente utilizado en el análisis de datos de expresión génica y otros tipos de datos biológicos en Python; este formato de archivo permite almacenar datos de expresión génica junto con metadatos adicionales, como información sobre las muestras y los genes.

Una vez construido el objeto de entrada y con MOBER correctamente instalado, el objetivo a conseguir fue el entrenamiento y la proyección de los datos utilizando la herramienta; observando y estudiando los resultados obtenidos.

Para llevar esto a cabo, la alumna utilizó los comandos proporcionados por los creadores de MOBER en el repositorio de la herramienta. Para el entrenamiento del modelo, el comando a utilizar requiere dos parámetros de entrada, el conjunto de datos a entrenar y la dirección de salida del modelo entrenado (ya proporcionada por defecto).

Este proceso se realiza utilizando un conjunto de datos de entrenamiento que contiene ejemplos previamente etiquetados o con respuestas conocidas, en este caso, el objeto .h5ad creado con los datos y metadatos sobre las muestras de microbioma ya estudiadas en anteriores Sprints. Teniendo esto, se realizó el entrenamiento del modelo, cuyo objetivo en este contexto es ajustar los parámetros internos del algoritmo para que pueda realizar su tarea específica de manera óptima, que en este caso es la eliminación del batch effect de los datos.

Durante el entrenamiento, el modelo ajusta sus parámetros internos para minimizar la llamada función de pérdida (loss), que evalúa la discrepancia entre las predicciones del modelo y las respuestas reales en el conjunto de datos de entrenamiento.

Se hace un estudio también del output devuelto por MOBER tras entrenar un modelo. Se obtiene como resultado, por una parte, tres ficheros (train_loss_adv, train_loss_ae y train_loss_tot) con información sobre la pérdida comentada obtenida durante el entrenamiento (adversarial, de reconstrucción y total); y por otra parte, varios ficheros CSV con características y parámetros sobre los datos de entrada y el entrenamiento, sumados a los modelos finales

entrenados para las dos componentes claves de MOBER: el codificador automático variacional condicional (batch_ae_final.model) y la red neuronal discriminadora de fuente adversaria (src_adv_final.model).

El funcionamiento de MOBER, como se comentó en el Sprint 4, sección 5.4, consiste en un codificador automático que consta de dos redes neuronales: un codificador automático variacional condicional (VAE) que está optimizado para generar incrustaciones que pueden reconstruir la entrada original y una red neuronal adversaria (aNN) que toma la incrustación generada por el VAE como entrada e intenta predecir el origen de los datos de entrada.

En este estudio, el objetivo del uso de estas redes es lograr una integración de las cohortes eliminando los efectos de lote que las diferencias entre ellas puedan causar. Esto permitirá un análisis global de los datos, resultando en un estudio más robusto y completo gracias a la mayor cantidad de datos disponibles. La eliminación de los efectos de lote busca asegurar que las variaciones debidas a la procedencia de las cohortes no afecten los resultados, permitiendo un enfoque más unificado y preciso en el análisis biológico.

En este contexto, el valor del loss VAE indica la discrepancia entre la entrada original y la reconstrucción producida por el VAE. Un bajo valor de loss VAE sugiere que el VAE es capaz de generar incrustaciones que puedan reconstruir con precisión la entrada original, lo que indica un buen rendimiento en la codificación de datos. Las incrustaciones se tratan de representaciones de datos en un espacio de dimensionalidad reducida, que buscan capturar las características más importantes de los datos.

Por otro lado, el loss ANN mide la habilidad de la red neuronal adversaria para discriminar entre las incrustaciones (embeddings) generadas por el VAE y las incrustaciones reales. Un bajo valor de loss ANN indica que la ANN tiene dificultades para distinguir entre las incrustaciones generadas por el VAE y las reales, lo que sugiere que el VAE ha aprendido a generar incrustaciones que son difíciles de distinguir de las reales, capturando así la información relevante mientras elimina los efectos de lote.

El objetivo de este esquema combinado es utilizar el VAE para crear representaciones (incrustaciones) de los datos que mantengan la señal biológica relevante y, al mismo tiempo, utilizar la ANN para garantizar que estas representaciones sean libres de efectos de lote. La interacción adversarial entre el VAE y la ANN permite ajustar el proceso de codificación para cumplir ambos objetivos simultáneamente.

El loss total de MOBER se calcula como una combinación ponderada de los losses VAE y ANN, donde el coeficiente λ determina el peso que se le asigna a la pérdida adversaria en la optimización del modelo. Esta ponderación permite al modelo encontrar un equilibrio entre la generación de incrustaciones precisas y la capacidad de eliminar los efectos de lote. Esto es:

$$Loss_{MOBER} = Loss_{VAE} - \lambda * Loss_{aNN} \tag{5.1}$$

donde λ es el coeficiente que determina el peso que el modelo le da a la pérdida adversaria.

Una vez estudiado el funcionamiento de MOBER y entrenado el modelo con el obejto de datos construido, el siguiente paso fue estudiar los resultados obtenidos. Se obtuvo lo siguiente:

Datos de entrada	Loss VAE	Loss aNN	Loss total
Sin normalización	3314652.9165	0,4267	3314652.9165

Tabla 5.1: Resultados función de pérdida tras aplicación de MOBER a los datos

En la Tabla 5.1 se presentan los diferentes valores obtenidos para la función de pérdida. Destaca especialmente el valor elevado de la pérdida total, principalmente atribuido al alto valor de la pérdida VAE. Esta discrepancia sugiere que MOBER ha tenido dificultades para preservar la señal biológica mientras reconstruye con precisión el objeto de entrada. Además, aunque la pérdida ANN es considerablemente menor que la pérdida VAE, su valor también indica que MOBER no ha logrado generar incrustaciones difíciles de distinguir de las reales, lo que evidencia una falta de eliminación efectiva de los efectos de lote.

Sin embargo, es importante tener en cuenta que este valor es solo un indicador inicial. La verdadera evaluación de la efectividad del modelo en la eliminación de los efectos de lote se realiza mediante la proyección de los datos y un estudio comparativo de los resultados obtenidos por MOBER con los datos originales previos al entrenamiento.

Considerando la suposición previa de que el modelo no funcionaría bien con datos no normalizados y la inconsistencia de los resultados obtenidos (al ser estos demasiado altos para un proceso de entrenamiento), no sería sorprendente que estos resultados revelen la necesidad de normalizar los datos para obtener resultados coherentes en la aplicación de MOBER. Este aspecto se abordó en Sprints posteriores y se detalla más ampliamente en 5.8.

Una vez completado el entrenamiento del modelo y examinados los resultados obtenidos, se procedió a realizar la proyección. En el análisis de datos, realizar una proyección se refiere a

aplicar un modelo previamente entrenado a un conjunto de datos existente para generar predicciones sobre nuevos datos. En el contexto de este proyecto, la proyección implica aplicar el modelo entrenado de MOBER al conjunto de datos de entrada con el fin de intentar eliminar los efectos de lote presentes en los datos.

Este proceso de proyección genera un nuevo archivo de datos como resultado de las predicciones realizadas. Este archivo es el utilizado en el posterior estudio y análisis conclusivo acerca de la efectividad de la eliminación de los efectos de lote, así como en la comparación con los efectos de lote presentes en los datos originales antes de entrenar MOBER. Este análisis en profundidad será el foco del próximo Sprint.

Para la realización de esta proyección, el comando proporcionado cuenta también con diferentes parámetros de entrada a ajustar por el usuario. Eston son:

- **model_dir**: ruta definida por defecto que indica donde se encuentran los modelos comentados en el anterior apartado.
- **onto**: una de las cohortes del estudio sobre la que los valores son proyectados.
- **projection_file**: fichero sobre el que se hará la proyección. Este fichero es el mismo que el utilizado como entrada para el entrenamiento.
- **output_file**: nuevo fichero .h5ad, que contendrá los valores proyectados.
- **decimals**: número de decimales a utilizar en la generación de los datos para el archivo de salida. Valor 4 por defecto.

Estudiada esta estructura, se realiza la proyección con los valores correspondientes, y se obtiene un fichero de nuevo en formato .h5ad como salida que, como se comentó, representa el fichero a estudiar para concluir si se han eliminado o no los efectos de lote.

5.7 SPRINT 7 - Realización y estudio de PCAs

Como se comentó en la sección anterior, el objetivo principal del Sprint número 7 fue el análisis de la efectividad de MOBER en la eliminación de los efectos de lote en su ejecución con el conjunto de datos generado. Esto se hizo mediante comparación. Para ello, se recurrió al Análisis de Componentes Principales (PCAs) tanto de los datos antes de aplicar MOBER como de los obtenidos tras la ejecución de la herramienta, y se realizaron comparaciones entre ambas gráficas.

El Análisis de Componentes Principales (PCAs) es una técnica estadística de síntesis de la información, o reducción de la dimensión (número de variables). Es decir, ante un conjunto de datos con muchas variables, el objetivo será reducirlas a un menor número perdiendo la menor cantidad de información posible. Es común una vez reducida la dimensionalidad, llevar a cabo una representación de los datos mediante un gráfico de dispersión para visualizar la estructura interna de los datos y explorar posibles patrones o agrupaciones.

En el contexto de este proyecto, la realización y graficado de PCAs nos ayuda a determinar la presencia o ausencia de efectos de lote de los datos estudiados. Generalmente, lo que se busca al realizar un PCA, es que un número reducido de componentes principales (PCs) explique una alta variabilidad de los datos. Esto indicaría que los datos se agrupan de manera coherente, lo cual es, generalmente, el objetivo perseguido.

Sin embargo, en este caso concreto, observar una alta variabilidad de los datos explicada por un bajo número de PCs, significa que los efectos de lote están presentes. Esto se debe a que la existencia de efectos de lote tenderá a agrupar los datos por cohortes, ya que cada cohorte representa a un grupo de muestras que fueron procesadas o secuenciadas juntas, estando expuestas así a los mismos efectos de lote. En resumen, en este contexto una alta variabilidad explicada por una baja cantidad de componentes principales en un PCA indica la presencia de efectos de lote.

Por el contrario, cuando la variabilidad de los datos explicada por un mismo número de PCs es baja, podría estar indicando que hay pocos efectos de lote presentes en los datos. Esto se debe a que, al eliminarse los efectos de lote, el análisis no agrupa correctamente las cohortes al no ser capaz de diferenciarlas y, por lo tanto, el PCA explica menos variabilidad. Sin embargo, la variabilidad explicada no dependerá únicamente de los efectos de lote, por lo que es importante hacer un análisis conjunto entre el resultado numérico y la representación gráfica.

Como conclusión, una baja variabilidad explicada por un mismo número de PCs podría indicar una reducción en los efectos de lote, pero es necesario un análisis más detallado para confirmar esta afirmación. Este fue por lo tanto el enfoque utilizado para el estudio y comparación de los PCAs.

En cuanto a la representación gráfica, y siguiendo la misma línea que para los resultados numéricos, una clara agrupación de los datos por cohortes indicará la presencia de efectos de lote. Sin embargo, una dispersión de los datos en el gráfico sugerirá una menor presencia de efectos de lote.

Estudiada esta información, se procedió a la realización de estos PCAs. Esto se llevo a cabo con R, haciendo uso de la biblioteca *PLSDAbatch* [36]. Esta biblioteca se utiliza para realizar análisis de datos de espectroscopía de masas basados en el método de regresión parcial de mínimos cuadrados (PLS-DA) y es un método comúnmente utilizado en estudios ómicos, como es el caso de este estudio, para analizar datos multivariados y encontrar patrones o correlaciones entre variables.

Para la realización de esta tarea, la alumna utilizó dos conjuntos de datos como base: el objeto phyloseq generado a partir de las cohortes de microbioma antes de aplicar MOBER y el objeto .h5ad obtenido como salida de MOBER tras entrenar el modelo y realizar la proyección. Con el segundo conjunto de datos, se llevó a cabo el proceso inverso al realizado con el objeto phyloseq inicial. En lugar de pasar de phyloseq a .h5ad, se realizó el paso de .h5ad a phyloseq. Esto se hizo utilizando tanto Python como R: Python para extraer la matriz de datos (*otu_table*) y las anotaciones de observaciones (*sample_data*) y generar con ellos los archivos CSV necesarios, y R para construir, a partir de estos archivos CSV, el objeto phyloseq a utilizar.

Una vez completado este paso, llegó el momento de aplicar PCA a ambos conjuntos de datos y comparar los resultados.

Como se mencionó anteriormente, se utilizó para esta tarea la biblioteca *PLSDAbatch* de R. Con su ayuda, se llevó a cabo la PCA de las tablas otu de cada uno de los conjuntos de datos, utilizando un valor de 2 para el número de componentes principales a calcular.

Una vez calculadas las PCAs se observaron y estudiaron los resultados. Se observó que para el objeto inicial (1), que no ha sido procesado por MOBER, el PCA consigue explicar un 45,70% de la variabilidad total de los datos utilizando dos componentes principales; 32,38% explicada por la primera variable y 13,32% por la segunda variable. Por otro lado, para el conjunto de datos procesado por MOBER (2), se obtuvo como resultado una variabilidad explicada por dos componentes principales del 97,53%; 88,80% explicada por la primera variable y 8,73% por la segunda.

Como ya se explicó, una mayor variabilidad de los datos explicada por un mismo número de componentes principales indica la presencia de una mayor cantidad de efectos de lote en ese conjunto de datos con respecto al otro.

Teniendo esto claro, se procedió a estudiar los resultados obtenidos. Se observó una gran

diferencia en la variabilidad explicada por cada conjunto de datos, siendo una casi el doble de la otra. Específicamente el PCA realizado sobre (1) mostró un valor mucho menor en comparación al PCA realizado sobre (2). Estos PCAs sugieren una presencia mucho mayor de efectos de lote en los datos que fueron procesados por MOBER.

A diferencia de lo idealmente esperado, los resultados numéricos revelan que MOBER no logró eliminar los efectos de lote; por el contrario, su presencia en los datos aumentó en lugar de disminuir.

Sin embargo, la presencia de resultados tan extremos, como una sola variable que explica casi la totalidad de los datos o una diferencia tan significativa en la variabilidad explicada entre los dos casos, sugiere que estos resultados pueden no ser muy confiables. De hecho, estos resultados no son sorprendentes si consideramos lo observado en el Sprint anterior: un valor de pérdida muy alto durante el entrenamiento del modelo, lo que ya anticipaba resultados deficientes por la falta de normalización del conjunto de datos inicial.

Cómo se comentó, el valor de la variabilidad explicada no depende únicamente de la presencia o ausencia de efectos de lote. Por el contrario, es lógico esperar que el cálculo de un PCA con datos que no han sido previamente normalizados y, con ello, no estén en una misma escala, devuelva resultados anómalos o sin sentido. El PCA se basa en la covarianza entre las variables, y cuando los datos no están normalizados, estas variables pueden tener escalas muy diferentes. Como resultado, algunas variables pueden tener una influencia desproporcionada en la PCA, lo que conduce a resultados sesgados o poco fiables.

A través del estudio de resultados realizado, se sospecha que esto está ocurriendo y que los resultados obtenidos carecen de sentido. Sin embargo, se realizó una representación gráfica de estos PCAs, lo que proporcionó mayor claridad y ayudó en la decisión sobre la fiabilidad de este caso de estudio.

Una vez estudiados estos resultados, se procedió a representarlos gráficamente para un análisis más completo. Para esto, se utilizó nuevamente el paquete *PLSDAbatch* y se empleó una función específica a la que se le pasaron varios parámetros. Entre ellos, el objeto PCA a representar y las variables a diferenciar, en este caso *"Status"* (estado del paciente) y *"Project"* (cohorte a la que pertenece la muestra). En estas gráficas cada cohorte se representa con un color y cada estado con una forma diferente y además en ellas se muestra la cantidad de variabilidad explicada por cada componente principal en forma de porcentaje, valores discutidos en párrafos anteriores.

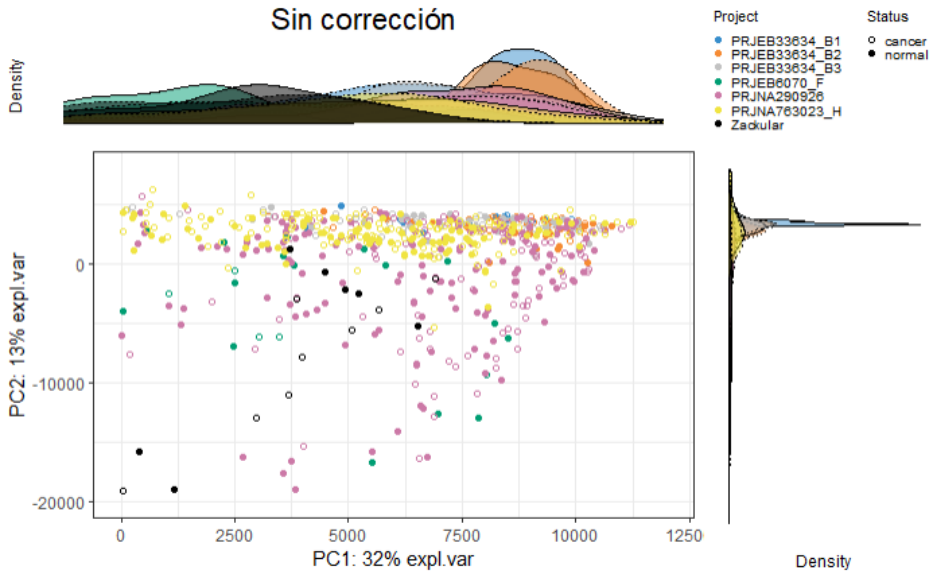


Figura 5.5: Resultado gráfico del PCA previo a MOBER

En la Figura 5.5, representación gráfica del PCA calculado para los datos previos a MOBER, se puede observar una ligera agrupación de los datos por cohortes, lo que sugiere la presencia de efectos de lote. Sin embargo, la realidad es que, confirmando lo que se comenta más arriba, no se pueden extraer conclusiones fiables observando esta gráfica. Esto resalta de nuevo la necesidad de normalizar los datos para obtener conclusiones más precisas y confiables.

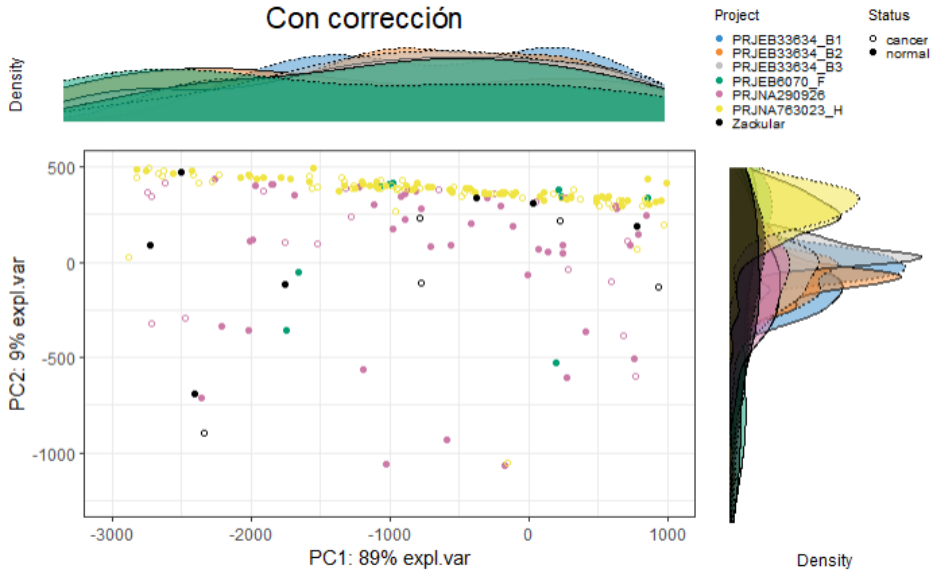


Figura 5.6: Resultado gráfico del PCA tras la ejecución de MOBER

En la Figura 5.6 se observa, de manera similar a la anterior, una ligera agrupación de los da-

tos por cohortes, pero nuevamente no se pueden extraer conclusiones definitivas de la gráfica. Como se comentó, también se muestra el porcentaje de variabilidad explicado por cada componente principal, el cual parece poco realista en comparación con lo que se observa visualmente en la gráfica. Una variabilidad explicada de casi la totalidad de los datos por dos componentes principales debería resultar en una gráfica con grupos perfectamente diferenciados, uno por cada cohorte, lo cual no sucede. La dispersión de los datos en esta gráfica resulta similar a la que se observa en la gráfica anterior, con un resultado de más de el doble de variabilidad explicada en este caso, de nuevo una observación sin sentido. Esta discrepancia puede atribuirse de nuevo al hecho de haber procesado con MOBER un conjunto de datos sin normalizar.

A través de los estudios de resultados realizados, se puede afirmar con casi totalidad que estas PCAs no son válidas. Lo representado en ellas carece de sentido y no está relacionado con los efectos de lote u otros aspectos a estudiar. Por lo tanto, estas PCAs quedaron descartadas del estudio y no se volvieron a tener en cuenta. Esto refuerza la importancia de la normalización de los datos antes de aplicar técnicas como la PCA, ya que garantiza resultados precisos y significativos. Esta conclusión refleja lo observado tanto en este Sprint como en el anterior y, por tanto, la normalización se convierte en el foco principal del Sprint 8.

5.8 SPRINT 8 - Normalización de los datos y ejecución de MOBER con los datos normalizados

Basándonos en lo estudiado y verificado en los Sprints previos, se confirmó la importancia de normalizar los datos del estudio para obtener conclusiones significativas y garantizar que MOBER funcione eficazmente en su tarea de eliminar los efectos de lote. Este proceso de normalización de los datos fue el objetivo de este Sprint.

Para ello, se optó por aplicar diferentes métodos de normalización a los datos con el fin de examinar las diferencias en los resultados obtenidos. Los métodos de normalización escogidos fueron Compositional Log-Ratio (CLR), Additive Log-Ratio (ALR) y normalización por abundancias relativas.

Como contexto sobre los métodos a utilizar:

- **Compositional Log-Ratio (CLR)**: Es una técnica de normalización que transforma datos en una nueva representación logarítmica. Se calcula el logaritmo de las razones entre las proporciones de las diferentes especies en cada muestra, lo que permite comparar las proporciones relativas de las especies dentro de cada muestra.

- **Additive Log-Ratio (ALR)**: Es una técnica similar a CLR, en la que también se transforman lo datos en una nueva representación logarítmica. En este caso, en lugar de calcular el logaritmo de las razones, ALR calcula el logaritmo de las diferencias entre las proporciones de las diferentes especies en cada muestra. Esto puede ser útil para visualizar la estructura de los datos de composición de manera diferente y explorar las relaciones entre las diferentes especies.
- **Abundancias relativas**: Lo que se hace con esta técnica es dividir las abundancias de cada especie en una muestra por la suma total de abundancias en esa misma muestra. Este enfoque muestra la proporción de cada especie en relación con el total de la comunidad microbiana en esa muestra específica, lo que permite tener en cuenta el volumen real de cada especie en la comunidad.

Para llevar a cabo esta tarea, se hizo de nuevo uso de R, esta vez mediante las los paquetes *mixOmics* [37] para la normalización CLR, *compositions* [38] para la normalización ALR y *ampvis2* [39] para la normalización por abundancias relativas. El paquete *mixOmics* se empleó para la normalización CLR mediante la función "logratio.transfer", que forma parte de sus herramientas de análisis multivariado. La normalización ALR se llevó a cabo utilizando el paquete *compositions*, que se especializa en el análisis y modelado de datos composicionales, mediante la función "alr". Por otro lado, se utilizó el paquete *ampvis2*, diseñado específicamente para el análisis de datos de microbioma, para realizar la normalización por abundancias relativas con su función "transform".

Se aplicaron las funciones arriba mencionadas a los datos contenidos en la tabla de OTUs del objeto phyloseq inicial, y como resultados se obtuvieron nuevas tablas OTU para cada uno de los casos.

Una vez completado este proceso, se contaba con tres tablas OTU distintas (CLR, ALR y abundancias relativas), las cuales se exportaron al equipo en formato CSV junto con sus correspondientes datos de muestra (sample_data), coincidentes para las tres.

A partir de estos archivos CSV, se siguió en Python el mismo proceso que se realizó en el Sprint 6, sección 5.6. Utilizando la biblioteca AnnData, se crearon tres archivos .h5ad que se utilizarían como entrada en MOBER, uno para cada tipo de normalización.

Teniendo esto, se aplicó de nuevo MOBER para cada uno de los conjuntos de datos normalizados; realizando el entrenamiento y la proyección siguiendo de nuevo el mismo procedimiento que en el Sprint 6, y se realizó de nuevo un estudio del valor de la función de pérdida.

Como se mencionó previamente, un valor bajo de la función de pérdida podría indicar una reducción efectiva de los efectos de lote y la preservación de la señal biológica al proyectar sobre este modelo entrenado.

Se obtuvieron los siguientes resultados:

Datos de entrada MOBER	Loss VAE	Loss aNN	Loss total
Sin normalizar	3314652,9165	0,4267	3314652,9165
CLR	1,9824	0,0097	1,9824
ALR	0,5432	0,0057	0,5431
Abundancias relativas	3314652,9165	0,4267	3314652,9165

Tabla 5.2: Estudio resultados función de pérdida

En la Tabla 5.2 se puede observar una reducción muy significativa entre el valor de la función de pérdida para los datos sin normalizar y los datos normalizados con CLR y ALR. Esto, como se comentó previamente, podría indicar que MOBER funcionó eficazmente en su tarea de reducción de efectos de lote en estos caso. Sin embargo, como se discutió en el Sprint 6, sección 5.6, esta afirmación debe ser confirmada mediante el análisis de los resultados, en este caso, mediante la aplicación de PCA a los ficheros de salida del proceso de proyección de MOBER.

Los valores obtenidos para los diferentes tipos de normalización muestran que el rendimiento de MOBER varía según el método de normalización utilizado:

- **CLR**: El valor del loss VAE es alto (1,9824), lo que indica que el modelo no ha sido capaz de generar una reconstrucción adecuada de la entrada (incrustaciones), preservando las características más importantes de los datos y, con ello, la señal biológica. Sin embargo, el valor de loss ANN es bajo y muy aceptable (0,0097), lo que sugiere que la ANN enfrenta dificultades para distinguir entre las incrustaciones generadas por el VAE y las reales, indicando esto una posible reducción de los efectos de lote. El loss total (1,9824) podría indicar una reducción de los efectos de lote, pero sin una correcta preservación de las características importantes de los datos.
- **ALR**: En este caso, el valor del loss VAE disminuye con respecto al caso anterior (0,5432), lo que indica que el modelo ha logrado una reconstrucción más precisa de los datos a

través de las incrustaciones, aunque sigue sin ser un valor ideal. El valor de loss ANN es bastante cercano al obtenido en el caso anterior (0,0057), lo que sugiere nuevamente que la ANN enfrenta dificultades para discriminar entre las incrustaciones generadas por el VAE y las reales. En esta ocasión, el loss total (0,5431) indica que el modelo ha sido más eficiente con este tipo de normalización, probablemente logrando una significativa reducción de los efectos de lote y preservando en mayor medida la señal biológica.

- **Abundancias relativas**: Para el método de abundancias relativas, el valor del loss VAE es extremadamente alto (3314652,9165), lo que sugiere que el VAE es incapaz de generar incrustaciones que reconstruyan con precisión la entrada original. El valor de loss ANN también es alto, aunque algo más razonable (0,4267), lo que indica que la ANN tiene ciertas dificultades para distinguir entre las incrustaciones generadas por el VAE y las reales. Con estos valores, el loss total también resulta extremadamente alto (3314652,9165). La diferencia significativa entre los valores y el valor tan extremo del loss VAE sugiere que la normalización por abundancias relativas no ha cumplido su función en este caso, obteniéndose resultados similares a los obtenidos para los datos sin normalizar. Esto se debe a que la normalización por abundancias relativas ajusta los conteos entre 0 y 1, normalizando los datos para que reflejen proporciones relativas en lugar de valores absolutos. Esto puede llevar a un escalado inapropiado de los datos, resultando en valores extremos, ya que entre 0 y 1 hay un amplio rango de variabilidad. Esto sugiere la ineficacia de este método y constituye un primer indicio para descartarlo, pendiente de confirmarse mediante el análisis que se realiza en el Sprint 9, sección 5.9.

Después de examinar los resultados proporcionados por MOBER tras su ejecución con datos normalizados, es esencial validarlos mediante el análisis y la comparación con los datos anteriores a MOBER.

5.9 SPRINT 9 - Realización y estudio de PCAs

Tras lo visto y estudiado en el anterior Sprint, el siguiente paso fue el análisis exhaustivo de los resultados obtenidos. Esto, al igual que en el Sprint 7, sección 5.7, se llevó a cabo mediante la realización, estudio y comparación de PCAs. Se realizó, a partir del archivo de salida de MOBER, un PCA por cada tipo de normalización, y se estudiaron los resultados tanto individualmente, como realizando comparaciones.

Para la realización de los PCAs, se siguió el mismo proceso que en el Sprint 7. En este caso se ejecutaron un total de 6 PCAs: 3 para los datos antes de la ejecución de MOBER (una por cada tipo de normalización) y 3 para los datos obtenidos después de aplicar MOBER.

A continuación, se muestran en dos tablas los resultados de los PCAs. En la primera tabla se muestran los resultados de los PCAs realizados sobre los datos antes de la ejecución de MOBER y en la segunda los resultados obtenidos para los datos de salida de MOBER.

Datos de entrada MOBER	PCA1	PCA2	Total
CLR	9,95%	8,98%	18,93%
ALR	25,39%	7,26%	32,65%
Abundancias relativas	32,38%	13,32%	45,70%

Tabla 5.3: Resultados PCAs sobre los datos previos a la ejecución de MOBER

En la Tabla 5.3 se observan resultados diversos entre las distintas PCAs, siendo el más bajo para CLR y el más alto para abundancias relativas.

Los resultados obtenidos sugieren que el PCA produce resultados diversos dependiendo del método de normalización aplicado, lo que destaca la importancia de evaluar la fiabilidad del método de normalización utilizado. Además, la variabilidad explicada es considerablemente alta en todos los casos, considerando que solo se han utilizado 2 PCs. Esta alta variabilidad podría indicar la presencia de efectos de lote en los datos de estudio, aunque este no sea el único factor que influye en este valor. Otros factores relevantes podrían incluir el tamaño del dataset, la presencia de ruido en los datos o la correlación entre variables. Por lo tanto, es crucial realizar un estudio comparativo entre los resultados numéricos y los gráficos para una comprensión más completa y precisa de los resultados.

Datos de entrada MOBER	PCA1	PCA2	Total
CLR	13,54%	8,37%	21,91%
ALR	20,29%	7,11%	27,40%
Abundancias relativas	88,80%	8,73%	97,53%

Tabla 5.4: Resultados PCAs sobre los datos procesados por MOBER

Por otro lado, en la Tabla 5.4 se observan resultados bastante similares para las PCAs realizadas tras la ejecución de MOBER con los datos normalizados con CLR y ALR; pero estos resultados difieren en gran medida al valor obtenido para abundancias relativas, que es notablemente más alto.

Además de las comparaciones entre los distintos tipos de normalización, lo más importante radica en la comparación entre el resultado del PCA obtenido para los datos antes y después de la ejecución de MOBER (para cada tipo de normalización), con el fin de evaluar si se han eliminado los efectos de lote.

Al estudiar esto, se observa que, a priori, la normalización con la que mejor se han eliminado los efectos de lote ha sido ALR, reduciendo la variabilidad explicada del 32,65% al 27,40% (una reducción del 5,25%). Con CLR, se obtienen resultados muy similares entre ambas PCAs, pasando de un 18,93% a un 21,91% (un leve aumento del 2,98%). Por su parte, en el caso de las abundancias relativas, la variabilidad explicada aumentó notablemente, pasando del 45,70% al 97,53% (un aumento del 51,83%). Esta diferencia tan significativa y un valor tan alto como resultado del segundo PCA sugieren lo que ya se había anticipado en el anterior Sprint con el estudio del loss: la normalización por abundancias relativas no aporta datos fiables para el estudio y MOBER pudo no haber procesado correctamente los datos normalizados mediante este método.

A pesar de que la variabilidad explicada se utilice en este caso de estudio como medidor de los efectos de lote, como ya se indicó, este valor no se debe únicamente a este factor. Otros factores, como la señal biológica, el método de normalización de los datos, y los puntos destacados anteriormente, también pueden contribuir significativamente a esta variabilidad. Teniendo en cuenta esto, y que la forma de distribución de los datos en la gráfica puede proporcionar mucha información acerca de los efectos de lote, se resalta que los resultados gráficos son más útiles para el estudio de este factor. Sin embargo, también se enfatiza en la importancia de realizar diferentes análisis y trabajar con los resultados en conjunto para tomar decisiones.

De la misma manera que en el Sprint 7, sección 5.7, para complementar al análisis numérico de los resultados, se realiza un análisis gráfico. Para esto se representaron gráficamente las 6 PCAs calculadas. En estas gráficas, igual que para las representadas en el Sprint 7, cada cohorte se representa con un color y cada estado con una forma diferente y, además, se muestran en ellas la cantidad de variabilidad explicada por cada componente principal en forma de porcentaje.

5.9.1 CLR

En este subapartado se presentan las gráficas tanto previas como posteriores a la ejecución de MOBER y las conclusiones a las que lleva su estudio para los datos normalizados mediante CLR.

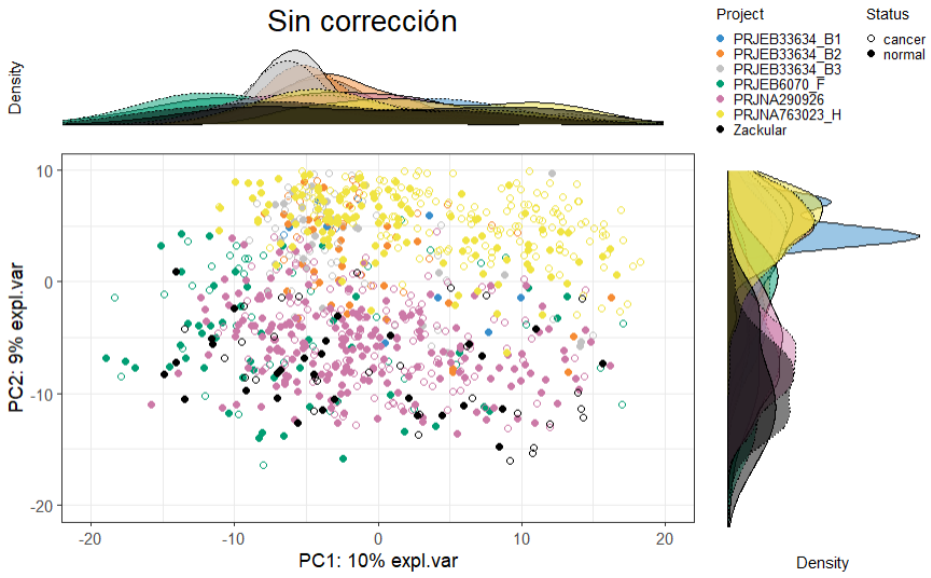


Figura 5.7: Resultado del PCA previo a MOBER con los datos normalizados por CLR

En la Figura 5.7, se aprecia una notable agrupación por cohortes, lo que sugiere la presencia de numerosos efectos de lote. Esta observación se fundamenta en la clasificación definida de los datos en lugar de una distribución dispersa.

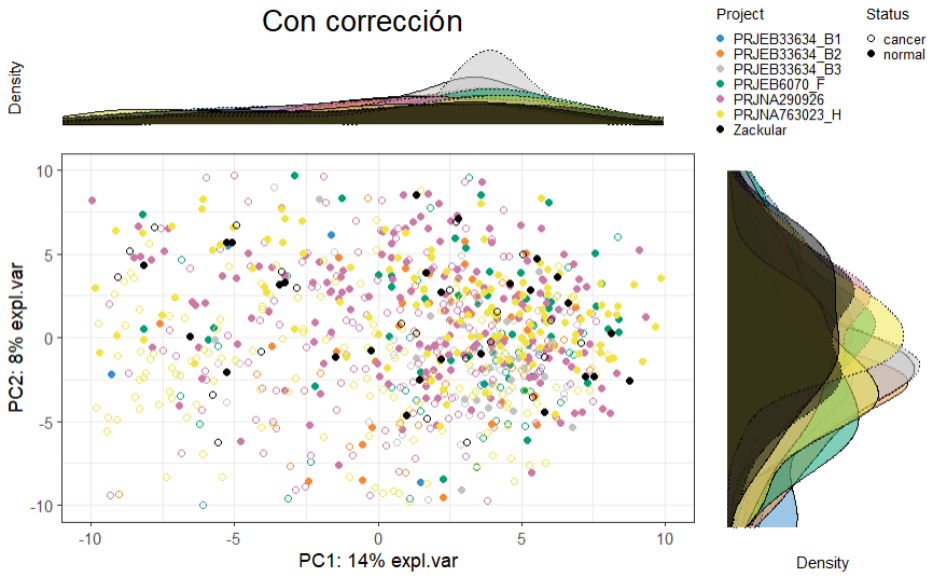


Figura 5.8: Resultado del PCA posterior a MOBER tras normalización CLR

En la Figura 5.8 los datos se muestran con un alto grado de dispersión. Además, en esta ocasión no se pueden observar patrones distintivos o agrupaciones claras de los datos. Esta observa-

ción sugiere una reducción de los efectos de lote y resalta que el resultado numérico no se relaciona únicamente con los efectos de lote, como se explicó anteriormente. Con esto, se destaca de nuevo que el estudio de las gráficas resulta más fiable a la hora de estudiar los efectos de lote.

Al comparar las gráficas de las Figuras 5.7 y 5.8, se confirma una notable reducción de los efectos de lote presentes en los datos. Esta reducción se evidencia por la ausencia de patrones claros en los datos en el PCA de salida de MOBER. En su lugar, se observa una gráfica con un alto grado de dispersión.

Considerando estas observaciones junto con el análisis previo del valor de pérdida del modelo con este tipo de normalización, se concluye que, a pesar de que el valor de pérdida pueda no parecer excesivamente bueno, el modelo ha logrado cumplir su función al reducir significativamente los efectos de lote. Sin embargo, es posible que no haya logrado preservar adecuadamente la señal biológica de los datos. Este aspecto se estudiará con mayor detalle en apartados posteriores.

5.9.2 ALR

En este subapartado se presentan las gráficas tanto previas como posteriores a la ejecución de MOBER y las conclusiones extraídas para los datos normalizados mediante ALR.

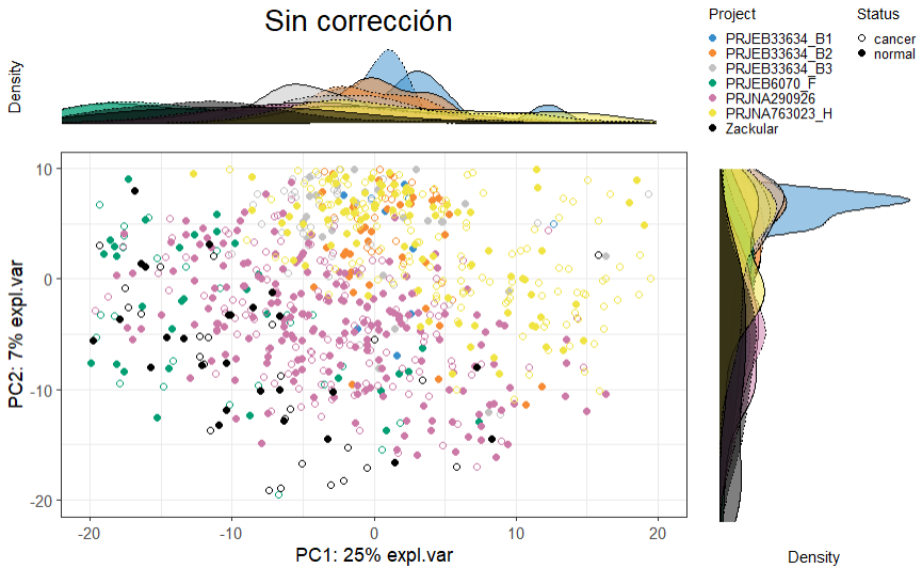


Figura 5.9: Resultado de la PCA previa a MOBER con los datos normalizados por ALR

La Figura 5.9 muestra una agrupación marcada por cohortes, similar a la observada en la Figura 5.7, lo que de nuevo sugiere la presencia de efectos de lote en los datos.

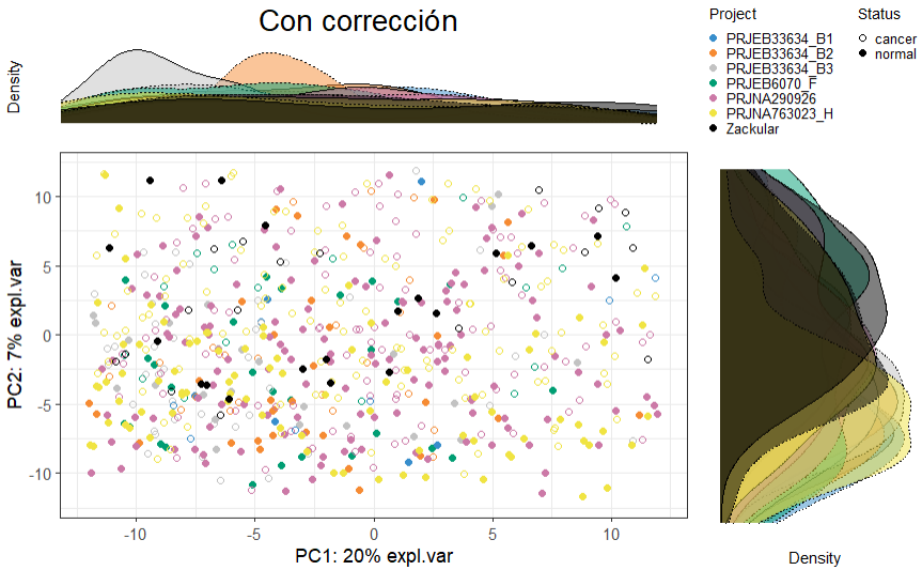


Figura 5.10: Resultado de la PCA salida MOBER tras normalización ALR

También de manera muy similar a su Figura análoga 5.8 pero en este caso para ALR, la Figura 5.10 muestra un alto grado de dispersión de los datos, sin patrones ni agrupaciones.

Analizando y comparando las gráficas de las Figuras 5.9 y 5.10, junto con los resultados numéricos, se puede confirmar una notable reducción de los efectos de lote tras aplicar MOBER a datos normalizados mediante ALR, de forma similar a lo observado para CLR. Esta reducción se evidencia por la dispersión de los datos y la ausencia de patrones claros en la Figura 5.10 en comparación con lo observado en la Figura 5.9.

En resumen, tras el estudio de los PCAs aplicados sobre los datos normalizados mediante ALR, y confirmando esto con el bajo valor obtenido para la función de pérdida, se evidencia que MOBER logró cumplir su función de manera efectiva para este tipo de normalización, logrando una reducción significativa de los efectos de lote y, muy probablemente, preservando gran parte de la señal biológica presente en los datos.

5.9.3 Abundancias relativas

De la misma manera que en los casos de CLR y ALR, se realizó también el estudio gráfico de los PCA de los datos normalizados con abundancias relativas. Como se había anticipado, este

tipo de normalización no era adecuado para este caso de estudio. Al evaluar tanto el resultado del loss del entrenamiento de MOBER con estos datos, como los resultados numéricos de los PCA realizados, se confirmó la falta de validez de este método. A pesar de esto, se llevó a cabo el análisis gráfico para corroborar las evidencias, concluyendo que este método no es válido. El análisis gráfico realizado se puede ver en el Apéndice A de esta memoria.

5.9.4 Conclusiones

Como conclusión final de este capítulo, se puede afirmar que la normalización de los datos ha sido eficaz al utilizar la herramienta, ya que MOBER ha logrado cumplir su función de forma más óptima tras esta normalización. La reducción de los efectos de lote es evidente en los casos de CLR y ALR. Sin embargo, como se ha comentado, para la normalización por abundancias relativas se confirmó su falta de validez. Los resultados obtenidos fueron descartados a la hora de sacar conclusiones, ya que tanto los resultados numéricos como la tendencia de los datos en las representaciones gráficas indican un rendimiento menos óptimo de MOBER y una influencia negativa de este método de normalización en su funcionamiento.

Observando tanto los resultados del entrenamiento con MOBER como los resultados de los PCAs, se concluye que ALR es el método de normalización más óptimo en este caso, dado que obtuvo el menor valor de pérdida (loss) y resultados favorables en el PCA.

En los siguientes Sprints, se llevaron a cabo estudios complementarios utilizando técnicas de aprendizaje automático. En estos estudios, se emplearon los datos normalizados mediante ALR, tanto previos como posteriores a la aplicación de MOBER, ya que este método fue considerado la opción con un funcionamiento más óptimo.

5.10 SPRINT 10 - Experimento cohortes ML

Una vez completado el análisis de la eficacia de MOBER en la eliminación de los efectos de lote, se procedió a continuar investigando su rendimiento mediante el estudio de técnicas de aprendizaje automático desarrolladas por la alumna.

Como se explicó en el Sprint 6, sección 5.6, MOBER consiste en un codificador automático que consta de dos redes neuronales: un codificador automático variacional condicional (VAE) y una red neuronal adversaria (aNN). El objetivo de este esquema combinado es utilizar el VAE para crear representaciones (incrustaciones) de los datos que mantengan la señal biológica relevante y, al mismo tiempo, utilizar la ANN para garantizar que estas representaciones sean libres de efectos de lote.

Estos dos puntos, la eliminación de los efectos de lote causados por la integración de cohortes y la preservación de la señal biológica en los datos tras la eliminación de estos efectos, fueron los estudiados a través de técnicas de aprendizaje automático.

1. **Estudio del rendimiento de MOBER**: En este experimento, se verificó nuevamente si MOBER había conseguido su objetivo de eliminar los efectos de lote de los datos integrados. Se evaluaron métricas de rendimiento para determinar la eficacia de MOBER en la eliminación de los efectos de lote.

2. **Evaluación de la presencia de señal biológica en los datos**: En este segundo experimento se investigó si, al eliminar los efectos de lote, MOBER había logrado preservar la señal biológica presente en los datos. Se examinó si la señal biológica seguía siendo detectable después de la aplicación de MOBER a los datos.

Cómo se comentó, ambos experimentos se llevaron a cabo haciendo uso de los datos normalizados por ALR al ser estos con los que se obtuvieron resultados más óptimos tanto en la ejecución de MOBER como en la realización de los PCAs. Este Sprint se centra en el desarrollo del experimento 1.

Para esto se hizo uso de R, a través del ecosistema *mlr3* [30, 40]. El ecosistema *mlr3* es una suite de paquetes en R diseñada específicamente para el aprendizaje automático. Está construido sobre el marco de trabajo *mlr3*, que proporciona una interfaz unificada y modular para realizar tareas relacionadas con el aprendizaje automático. En este caso se utilizaron los siguientes paquetes de este ecosistema:

- **mlr3learners**: para la definición de modelos.
- **mlr3tuning**: para hacer tuning de los modelos (optimización de hiperparámetros).
- **mlr3pipelines**: para combinar todos los pasos del preprocesado y modelado en un único objeto.

A través de esto, lo que se hizo fue extraer 15 componentes principales de las PCAs de los datos CLR tanto antes como después de aplicar MOBER. Luego, se utilizó esta representación de los datos para entrenar tres modelos de aprendizaje automático: Random Forest (RF), Support Vector Machine (SVM) y Elastic-Net (GLMNET). El objetivo fue comparar el rendimiento de estos modelos antes y después de aplicar MOBER, y evaluar de nuevo si MOBER había realizado correctamente su función.

- **Random Forest (RF)**: Algoritmo de aprendizaje automático que se basa en el concepto de ”ensemble” (utilización de varios modelos y combinación de sus resultados para obtener una predicción final). En este caso se combinan múltiples árboles de decisión para

obtener una predicción más robusta y precisa. Cada árbol se entrena con una muestra aleatoria del conjunto de datos y una selección aleatoria de características, lo que ayuda a reducir el sobreajuste y mejorar la generalización del modelo.

- **Support Vector Machine (SVM)**: Algoritmo de clasificación que busca encontrar el hiperplano óptimo que mejor separa las clases en un espacio multidimensional. Funciona transformando los datos de entrada en un espacio de mayor dimensión donde las clases sean linealmente separables, y luego encuentra el hiperplano que maximiza la separación entre las clases.
- **Elastic-Net (GLMNet)**: Método de aprendizaje automático que se utiliza para ajustar modelos de regresión lineal con regularización elástica, lo que significa que combina la regularización L1 (lasso) y L2 (ridge). La regularización es una técnica que se utiliza para evitar el sobreajuste al penalizar los coeficientes de las variables, lo que ayuda a simplificar el modelo y evitar que sea demasiado complejo.

Para la realización del experimento, se aplicó un enfoque de aprendizaje supervisado, donde se utilizaron como características las 15 PCAs y como variable a predecir la cohorte. Se llevó a cabo una validación cruzada normal con 10 folds (10-fold cross-validation) para evaluar el rendimiento de los modelos de manera robusta. Este proceso permitió comparar cómo los modelos se desempeñaban en la tarea de clasificar las muestras en las cohortes antes y después de la aplicación de MOBER.

Una vez entrenados y evaluados los distintos modelos, se calcularon métricas útiles para estudiar el rendimiento de cada uno. Estas métricas fueron: el error generado por el modelo y su precisión.

Modelo <chr>	Error <dbl>	Precisión <dbl>
SVM_pre	0.1531686	0.6336137
SVM_post	0.4775681	0.2493446
RandomForest_pre	0.1981011	0.5638777
RandomForest_post	0.4914093	0.2555288
GLMNet_pre	0.1638412	0.5898934
GLMNet_post	0.5803249	0.1981365

Tabla 5.5: Resultados experimento ML sobre predicción de cohortes

En la Tabla 5.5 se muestran los resultados de precisión y error obtenidos para los tres modelos de aprendizaje automático, tanto previos a la aplicación de MOBER como posteriores.

El error del modelo es una medida de qué tan lejos están las predicciones del modelo de los valores reales. Se calcula como la diferencia entre el valor predicho por el modelo y el valor real. Por su parte, la precisión del modelo es una medida de qué tan preciso es el modelo en sus predicciones. Se calcula como la proporción de predicciones correctas realizadas por el modelo sobre el total de predicciones realizadas.

En este caso de estudio, a diferencia de lo habitual, se busca un alto error y una baja precisión en los modelos realizados con datos que han sido previamente procesados con MOBER. Esto se debe a que un conjunto de datos sin efectos de lotes lleva a una mala clasificación de los mismos. Por lo tanto, un alto error y una baja precisión indicarían que los efectos de lote han sido eliminados de manera efectiva.

Se realizó un análisis de los resultados obtenidos para cada modelo (medidos en escala 0-1):

- **SVM**: Se observó un notable aumento en el valor del error, pasando de 0,1532 a 0,4776. En cuanto a la precisión, disminuyó desde 0,6336 hasta 0,2493.
- **RF**: En el caso del modelo RF, se observó una diferencia similar. El error aumentó de 0,1981 a 0,4914, mientras que la precisión disminuyó de 0,5639 a 0,2555.
- **GLMNet**: Similarmente, para el modelo GLMNet, el error aumentó de 0,1638 a 0,5803, mientras que la precisión disminuyó de 0,5899 a 0,1981.

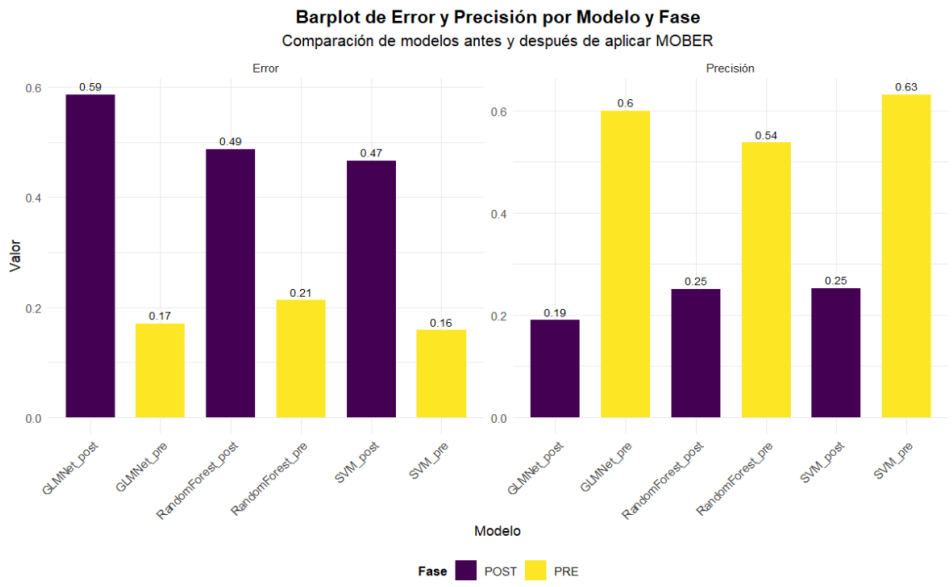


Figura 5.11: Resultados gráficos experimento ML sobre predicción de cohortes

En la Figura 5.11 se observan los resultados comentados de manera gráfica, en forma de barplots de error y precisión por modelo y fase. Observando estos resultados, se reafirma la conclusión previa de que MOBER logró eliminar los efectos de lote. En términos de eficacia de cada modelo, los rendimientos son muy similares, con diferencias entre los valores de error y precisión previos y posteriores a la aplicación de MOBER bastante consistentes entre modelos. Esta uniformidad sugiere que MOBER ha funcionado de manera eficaz con todas las técnicas aplicadas.

Una vez concluido el primer experimento y comprobado que se lograron eliminar los efectos de lote, causados en gran parte por la integración de las diferentes cohortes en un único conjunto de datos, el siguiente paso fue investigar si MOBER había conseguido preservar la cantidad de señal biológica captada por los modelos. Con esto, se evaluó si los datos de las distintas cohortes podían ser estudiados como un conjunto global. Este análisis fue el objetivo principal de los siguientes dos sprints.

5.11 SPRINT 11, 12 - Experimento status ML

Como se acaba de mencionar, una vez estudiada la eliminación de los efectos de lote, el siguiente paso, y el objetivo de este segundo experimento, fue analizar si la señal biológica de los datos se había visto afectada durante este proceso. En otras palabras, se buscó determinar si MOBER logró preservar la información biológica relevante mientras eliminaba los efectos de lote y, con ello, si las cohortes podían ser estudiadas como un conjunto integrado.

Cuando se habla de señal biológica en el contexto de datos de microbioma, se refiere a las características y patrones inherentes que reflejan la composición y función de las comunidades microbianas en diferentes condiciones biológicas. Estos datos incluyen información sobre la presencia, abundancia y diversidad de diferentes especies microbianas en una muestra, y pueden proporcionar insights sobre la salud del huésped, el estado de la enfermedad, el impacto de tratamientos, entre otros factores. Preservar la señal biológica significa que, tras el procesamiento de los datos, se mantienen estos patrones y relaciones importantes, permitiendo una interpretación válida y significativa de los resultados biológicos de las cohortes estudiadas como conjunto integrado.

Sabiendo esto, lo que se propuso hacer para este estudio fue comparar tanto el error como la precisión de los mismos modelos utilizados en el experimento anterior, pero esta vez en la clasificación por la variable *status*. Esta variable consta de dos valores: ”cáncer” o ”normal”. El objetivo fue evaluar qué tan efectivos son los modelos para predecir el valor de esta variable

en los datos previos y posteriores a la aplicación de MOBER.

Una precisión alta en la clasificación de muestras entre las categorías ”cáncer” y ”normal” indicaría que el modelo es capaz de distinguir de manera efectiva las diferencias biológicas entre estas dos condiciones. Esto significa que las características biológicas específicas asociadas con cada condición están siendo capturadas y utilizadas de manera adecuada por el modelo para realizar predicciones precisas. Por lo tanto, una alta precisión sugiere que la señal biológica relevante está siendo correctamente identificada y utilizada en el proceso de clasificación.

Es decir, al eliminar los efectos de lote del conjunto de datos integrado, el modelo debería ser capaz de captar un mayor grado de características biológicas relevantes en comparación con el conjunto de datos integrado sin eliminación de efectos de lote. Por lo tanto, una disminución del error y un aumento de la precisión al comparar los datos previos y posteriores a la aplicación de MOBER indicarían la eficacia de la herramienta.

Para estudiar este factor mediante aprendizaje automático se hizo uso de nuevo del ecosistema de paquetes *mlr3* [30, 40] de R. Se utilizaron los paquetes de este ecosistema *mlr3learners*, *mlr3tuning* y *mlr3pipelines*, y a mayores se utilizó *mlr3filters*, para la utilización de métodos de filtrado en la selección de características.

A partir de este ecosistema, se realizaron tres estudios diferentes y complementarios:

- **Estudio del funcionamiento de cada cohorte individualmente**: El objetivo de este primer estudio fue evaluar el rendimiento de cada cohorte en la clasificación de pacientes por la variable *status* de forma individual. Para ello, se aplicó validación cruzada o cross validation (CV) estándar a cada cohorte de forma independiente. El propósito de este análisis fue identificar si alguna cohorte obtenía resultados significativamente malos, lo que podría justificar su eliminación antes de proceder con los estudios conjuntos.
- **Estudio del funcionamiento del conjunto de datos mediante LCO-CV**: El segundo estudio se centró en el funcionamiento del conjunto de datos combinados, utilizando validación cruzada dejando uno fuera o Leave-Class-Out Cross-Validation (LCO-CV). Una vez evaluado el rendimiento individual de cada cohorte, se realizaron dos estudios conjuntos. En este caso, con las siete cohortes disponibles, se entrenó el modelo con seis cohortes y se probó con la cohorte restante, repitiendo este proceso para cada una de las cohortes. El objetivo de este estudio fue examinar cómo las cohortes funcionaban en conjunto y determinar la capacidad del modelo para generalizar entre ellas.
- **Estudio del funcionamiento global del conjunto de datos mediante CV**: De ma-

nera similar a la primera prueba, se utilizó CV estándar, pero en este caso, el análisis se enfocó en el rendimiento del conjunto completo de cohortes en lugar de evaluarlas individualmente. Este último estudio permitió analizar el rendimiento global del modelo cuando se consideran todas las cohortes como un solo conjunto, proporcionando una visión integral y optimizando la clasificación de los pacientes por la variable *status*.

Para la evaluación de resultados, en el primer estudio solo se analizaron los datos previos a la aplicación de MOBER. Esto se debe a que, al evaluar cada cohorte de manera individual, el interés radicaba en la precisión de cada cohorte en la tarea de clasificación, sin considerar la integración realizada para el uso de MOBER.

Para los dos estudios restantes, que implican un análisis conjunto de todas las cohortes, se llevó a cabo la evaluación tanto con los datos previos a MOBER como con los datos posteriores a su aplicación. Esto permitió comparar los resultados y determinar el impacto de MOBER en la integración y rendimiento global de las cohortes en esta tarea de clasificación.

En la realización de estos estudios se hizo uso de los mismos modelos que en el experimento tratado en el Sprint 10, sección 5.10. Estos fueron Random Forest (RF), Support Vector Macine (SVM) y Elastic-Net (GLMNet).

5.11.1 Estudio del funcionamiento de cada cohorte individualmente

Como se comentó, el primer estudio se centró en evaluar el rendimiento de cada cohorte en la tarea de clasificación de los datos entre sanos y enfermos. Para ello, se empleó CV estándar por cohorte.

Para analizar mejor las características de las cohortes, se obtuvo el número de muestras de cada una. Esto permitió investigar si el tamaño de la cohorte podría estar influyendo en su rendimiento en la tarea de clasificación.

	Project <chr>	Num_Muestras <int>
1	PRJEB33634_B1	15
2	PRJEB33634_B2	64
3	PRJEB33634_B3	80
4	PRJEB6070_F	91
5	PRJNA290926	314
6	PRJNA763023_H	310
7	Zackular	60

Tabla 5.6: Número de muestras por cohorte

En la Tabla 5.6 se observan diferencias significativas en el número de muestras de cada cohorte, que van desde 15 hasta 314 muestras. Este factor podría influir en el rendimiento de los modelos y, por lo tanto, en los resultados obtenidos durante la evaluación individual de las cohortes.

Una vez analizado este factor, se procedió a la evaluación sobre el rendimiento de los modelos en la tarea de clasificación para estas cohortes.

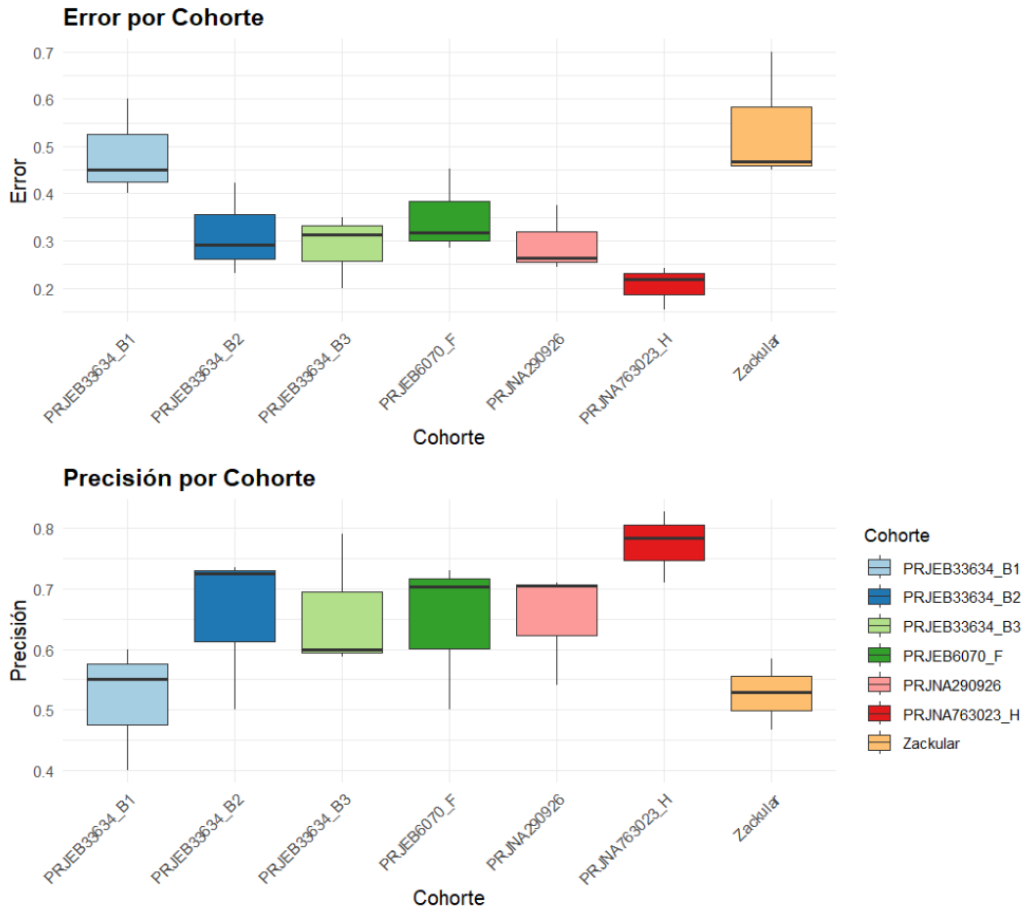


Figura 5.12: Resultados gráficos de Error y Precisión por cohorte

En la Figura 5.12 se muestran dos boxplots que presentan el error y la precisión por cohorte, resultado de la tarea de clasificación comentada, utilizando los tres modelos mencionados. Se puede observar un rendimiento consistente y similar entre las cohortes PRJEB33634_B2, PRJEB33634_B3, PRJEB6070_F y PRJNA290926, caracterizado por un error estable y aceptable (entre 0,3 y 0,4), así como una precisión del modelo relativamente alta (entre 0,6 y 0,75). Por

otro lado, para la cohorte PRJNA763023_H se observa un desempeño sobresaliente en comparación a las demás, con un error muy bajo (alrededor de 0,2) y una precisión excepcionalmente alta en comparación con las demás cohortes (cercana a 0,8).

Sin embargo, se observa una situación contrastante en las cohortes PRJEB33634_B1 y Zackular, donde se registra un error notablemente elevado (superior a 0,5) y una baja precisión en comparación con el resto de las cohortes (sobre 0,5). Estas disparidades resaltan la variabilidad en el rendimiento de los modelos entre las diferentes cohortes.

Relacionando estos resultados con el número de muestras de cada cohorte, se observa que las cohortes mencionadas que obtuvieron peor rendimiento coinciden con las dos cohortes con menor número de muestras (15 y 60 respectivamente). Además, la cohorte que presenta un mejor rendimiento coincide con una cohorte que tiene un alto número de muestras (310). Esto sugiere que el número de muestras de cada cohorte está afectando al rendimiento de los modelos, y no necesariamente indica la influencia de factores específicos de cada cohorte en la capacidad de clasificación de los modelos.

Como se mencionó previamente, el propósito inicial de este estudio fue evaluar el rendimiento de cada cohorte de manera individual en esta tarea de clasificación para determinar si alguna de ellas mostraba resultados significativamente malos, lo que podría justificar su exclusión para evitar introducir ruido en los estudios posteriores. Basándose en este objetivo y en lo observado en la gráfica, a pesar de que parece que el mal rendimiento de las cohortes viene dado únicamente por el número de muestras que tiene cada una, se decidió llevar a cabo los siguientes estudios de dos maneras: uno incluyendo todas las cohortes y otro excluyendo las cohortes PRJEB33634_B1 y Zackular, dado que son aquellas que mostraron el peor rendimiento.

5.11.2 Estudio del funcionamiento del conjunto de datos mediante LCO-CV

En el segundo estudio, y teniendo en cuenta los resultados del anterior, se utilizó la validación cruzada Leave-One-Cohort-Out (LCO-CV) para evaluar el rendimiento de las cohortes en conjunto. Aunque se analizan en conjunto, es importante destacar que cada cohorte individual también podría jugar un papel significativo en este proceso, ya que el método LCO-CV implica usar todos los datos de una única cohorte para el testeo mientras que las demás cohortes se utilizan para el entrenamiento.

Como se menciona anteriormente, tanto este estudio como el siguiente se realizarán de dos maneras: incluyendo todas las cohortes y excluyendo las cohortes PRJEB33634_B1 y Zackular, dado que mostraron el peor rendimiento.

Conjunto de datos completo

Inicialmente se realizó el estudio para el conjunto de datos completo.

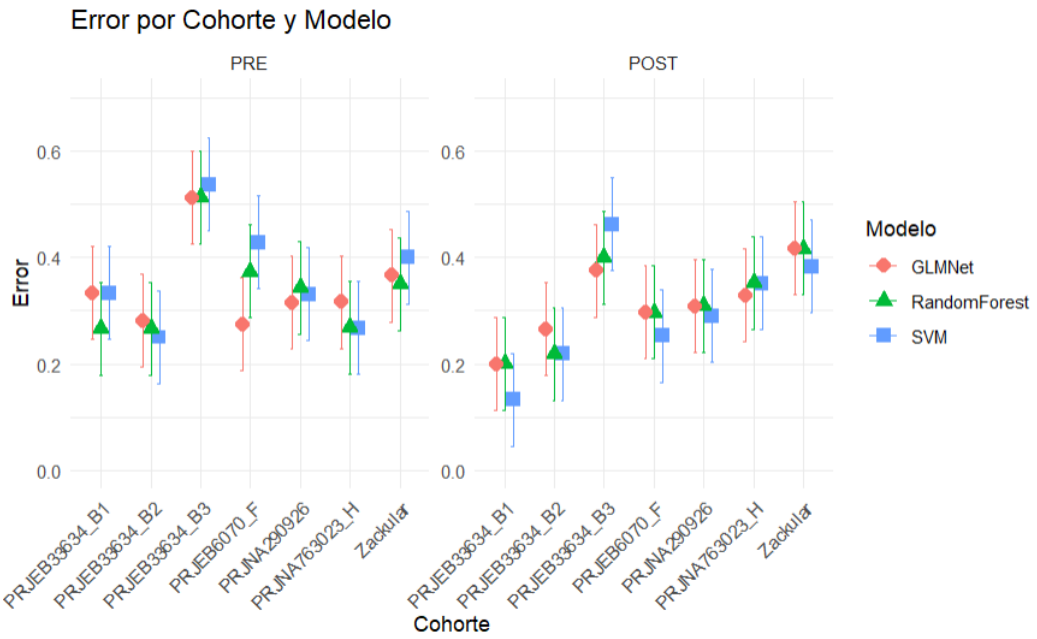


Figura 5.13: Resultados gráficos de Error por cohorte, modelo y fase

En la Figura 5.13 se presenta un gráfico de puntos que muestra los resultados obtenidos para el error en cada fase (con cada cohorte como conjunto de prueba) por cohorte y modelo. La cohorte sobre la que se representan los resultados es la utilizada como conjunto de prueba.

Se observan resultados bastante similares entre cada cohorte y modelo, para cada una de las fases, exceptuando la cohorte PRJEB33634_B3, que presenta un error mayor en los tres modelos. Esto sugiere un rendimiento estable de todas las cohortes y un funcionamiento similar de los modelos empleados, con un pequeño pico de error para la cohorte mencionada.

Comparando los resultados entre fases, que es el objetivo principal de este estudio, se observa una disminución general del error en los datos previamente procesados con MOBER, es decir, en la fase POST MOBER. En los datos previos a MOBER, el error oscila entre 0,2 y 0,4, exceptuando la cohorte PRJEB33634_B3, que asciende hasta alrededor de 0,55. Por su parte, el error para los datos que han sido procesados por MOBER es menor en la mayoría de los casos, rondando entre 0,1 y 0,35, y no superando el 0,45 en ningún caso.

Esta disminución del error en el entrenamiento del modelo indica una mejora del rendimiento

de los modelos para los datos que han sido previamente procesados con MOBER. Esto significa que el modelo comete menos errores en la tarea de clasificación de los datos entre sanos y enfermos, lo que sugiere que está capturando una mayor cantidad de señal biológica y, a través de ella, es capaz de clasificar mejor.

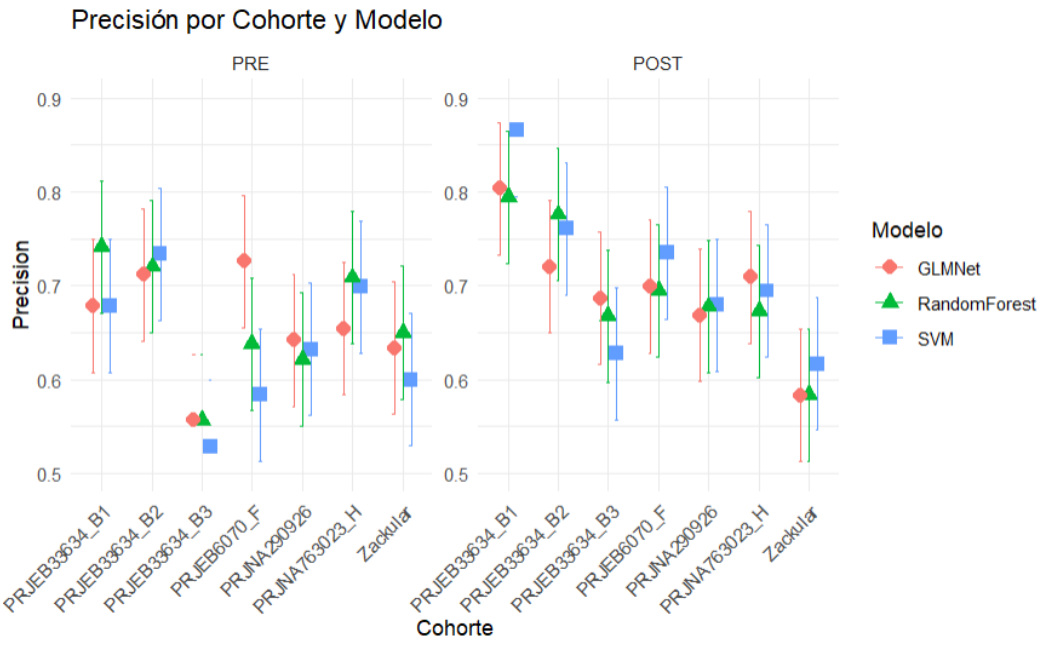


Figura 5.14: Resultados gráficos de Precisión por cohorte, modelo y fase

De manera análoga a lo observado en la Figura 5.13, en la Figura 5.14 se presentan los resultados de la precisión obtenida en el estudio para las distintas cohortes.

En este caso, la precisión global de los modelos es mayor en los datos que han sido procesados por MOBER, oscilando entre 0,55 y 0,85. En los datos previos a MOBER, la precisión se encuentra entre 0,5 y 0,75. Es notable que la cohorte PRJEB33634_B1 muestra la mayor diferencia en precisión y los mejores resultados, a diferencia de lo observado en el primer estudio. Sin embargo, la cohorte Zackular sigue mostrando una precisión baja, de la misma manera que en el estudio individual. Esto resalta la importancia de estudiar tanto los resultados globales como los individuales de cada cohorte.

De manera similar a lo observado con el error, este aumento en la precisión indica una mejora del rendimiento de los modelos para los datos que han sido previamente procesados con MOBER. Es probable que estas observaciones se deba a un aumento en la cantidad de señal biológica capturada en el conjunto de los datos integrados tras su procesamiento con MOBER,

lo que permite una mejor clasificación de los datos entre sanos y enfermos e indicaría que la herramienta ha conseguido realizar su función.

Excluyendo las cohortes PRJEB33634_B1 y Zackular

Una vez realizado el estudio con todas las cohortes, se procedió a excluir las dos cohortes que mostraron el peor rendimiento de forma individual. El objetivo de esta exclusión era determinar si el rendimiento de los modelos se mantenía constante o si, por el contrario, el mal desempeño de estas cohortes de forma individual estaba afectando negativamente el funcionamiento global de MOBER.

Para esto, se eliminaron las cohortes seleccionadas del conjunto de datos inicial, previamente normalizado mediante ALR, y luego se procesó este conjunto con MOBER.

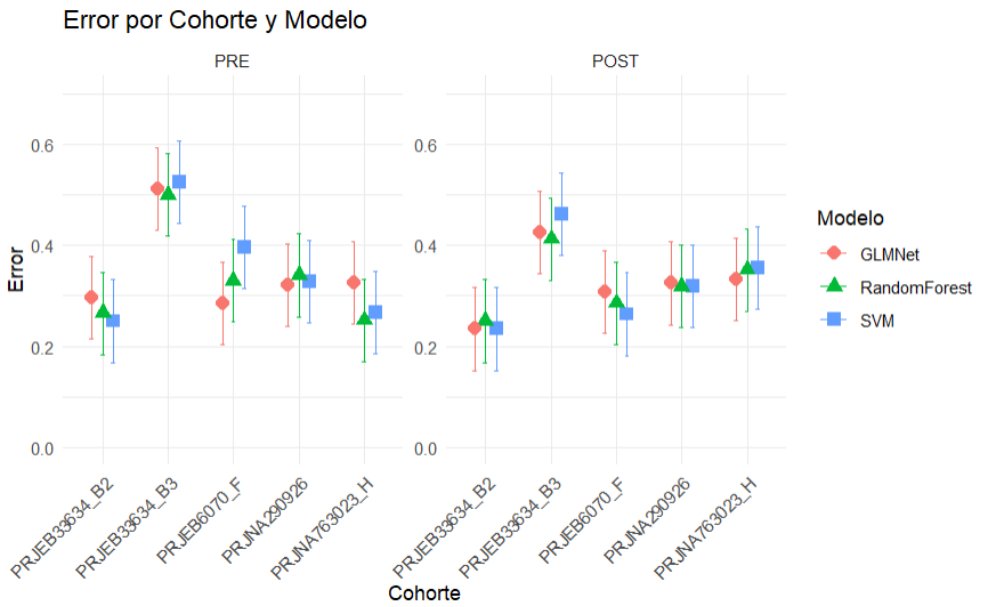


Figura 5.15: Resultados gráficos de Error por cohorte, modelo y fase

En la Figura 5.15 se observan resultados muy similares a los observados en la Figura 5.13, con una ligera disminución del error y un leve aumento de la precisión al comparar los datos previos y posteriores a MOBER. En este análisis, el error con los datos previos oscila entre 0,25 y 0,50, mientras que con los datos posteriores a MOBER varía entre 0,20 y 0,45. Esta tendencia es consistente con lo observado en la Figura 5.13, y nuevamente se identifica un pico en el mal rendimiento cuando se utiliza como prueba la cohorte PRJEB33634_B3.

Estos resultados tan similares indican que el mal rendimiento individual de las cohortes eliminadas no afecta significativamente al rendimiento global de los modelos entrenados con todas las cohortes mediante LCO-CV. Esto confirma lo que ya se suponía: el mal rendimiento individual de esas cohortes solo se debe al pequeño número de muestras con el que contaban y, por lo tanto, resulta lógico que la eliminación de las cohortes de menor desempeño no mejore sustancialmente el rendimiento global.

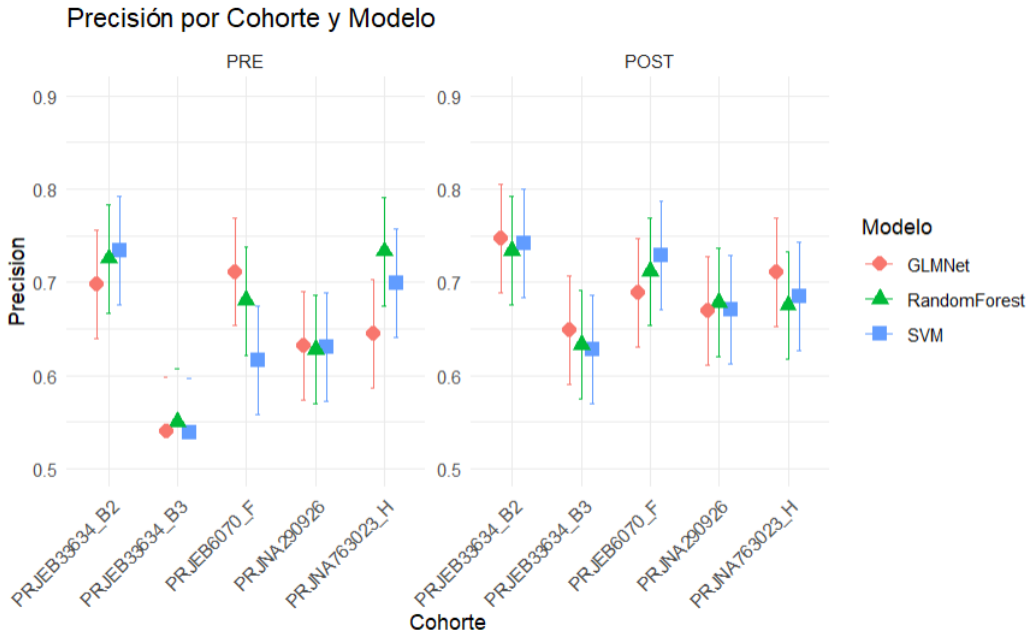


Figura 5.16: Resultados gráficos de Precisión por cohorte, modelo y fase

De nuevo, la Figura 5.16, se presentan resultados muy similares a los de la Figura 5.14, con un leve aumento de la precisión de los modelos empleados entre los datos previos y posteriores a MOBER. En este análisis, la precisión de los datos previos oscila entre 0.55 y 0.75, mientras que la precisión de los datos posteriores varía entre 0.65 y 0.75. Este aumento en la precisión es más leve que el observado anteriormente. Nuevamente, se destaca el mal rendimiento de la cohorte PRJEB33634_B3.

Estas observaciones indican nuevamente que el rendimiento individual de las cohortes no afecta significativamente al rendimiento global del modelo, ya que este bajo rendimiento está simplemente relacionado con el número de muestras de cada cohorte. Sabiendo esto, resulta lógico concluir que la eliminación de las cohortes con menor desempeño no mejorare sustancialmente el rendimiento global.

5.11.3 Estudio del funcionamiento global del conjunto de datos mediante CV

Otra estrategia para estudiar el rendimiento de las cohortes en conjunto fue mediante la validación cruzada estándar. Esta técnica proporciona una visión más global del comportamiento de las cohortes como un todo, ya que en este caso, los datos de todas las cohortes se mezclan tanto en el entrenamiento como en el testeo, ofreciendo así una perspectiva integrada del rendimiento del modelo en el conjunto completo de datos.

De la misma manera que en el estudio anterior, este se realizará de dos maneras: incluyendo todas las cohortes y excluyendo las cohortes PRJEB33634_B1 y Zackular, dado que son las que mostraron el peor rendimiento.

Conjunto de datos completo

Como en el anterior estudio, inicialmente se realizó el estudio para el conjunto de datos completo.

Modelo <chr>	Error <dbl>	Precision <dbl>
SVM_PRE	0.2910534	0.7082411
SVM_POST	0.3120109	0.6867820
RandomForest_PRE	0.2910260	0.7102023
RandomForest_POST	0.3248153	0.6767873
GLMNet_PRE	0.2817100	0.7171840
GLMNet_POST	0.3073735	0.6935877

Tabla 5.7: Resultados gráficos de Error y Precisión por modelo y fase

En la Tabla 5.7 se presentan los resultados obtenidos del procesamiento de los distintos modelos, entrenando y testeando con todas las cohortes. Se observa un rendimiento similar tanto entre los distintos modelos como entre las distintas fases.

El error de los modelos al entrenar con los datos previos a MOBER oscila entre 0,2817 y 0,2910, mientras que para los datos posteriores a MOBER, el error varía entre 0,3074 y 0,3248. La precisión para los datos previos a MOBER oscila entre 0,7082 y 0,7172, y para los datos posteriores a MOBER, la precisión se sitúa entre 0,6768 y 0,6936.

En este estudio, a diferencia del anterior, no se observa una mejora del rendimiento de los modelos cuando los datos han sido procesados con MOBER. En cambio, se aprecia un ligero empeoramiento. Aunque los resultados son muy similares, a nivel global, se observa que MO-

BER no ha logrado que se realice una mejor clasificación de los datos y, por ende, no ha sido capaz de capturar una mayor cantidad de señal biológica tras la eliminación de los efectos de lote y el procesamiento de los datos. La diferencia entre los rendimientos es muy pequeña, lo que sugiere que el impacto de MOBER en el procesamiento de los datos es mínimo en este contexto.

Como se acaba de mencionar, la diferencia entre los valores obtenidos para los modelos entrenados con los datos previos y posteriores a MOBER es mínima. Sin embargo, estos valores son aceptables en todos los casos. En el estudio anterior, esta diferencia era mayor, con resultados mejores para los datos procesados con MOBER, mientras que los resultados con los datos previos a MOBER eran más deficientes.

Estas observaciones indican que MOBER ha sido efectivo en el estudio donde se prueba con una cohorte completamente desconocida (no utilizada en el entrenamiento), pero no en el estudio donde las cohortes se mezclan y parte de todas ellas ya se han visto en el entrenamiento a la hora de realizar el test. En este último estudio, los propios datos integrados sin la necesidad de eliminar los efectos de lote consiguen buenos resultados, por lo que es difícil mejorarlos. Esto sugiere que MOBER es capaz de realizar su función pero, en un conjunto de datos tan homogéneo como el de este estudio (datos de ARN ribosomal 16S secuenciados de la misma manera), la diferencia en los resultados es casi imperceptible ya que los propios datos se integran bien sin necesidad de eliminar los efectos de lote.

Excluyendo las cohortes PRJEB33634_B1 y Zackular

Al igual que en el estudio anterior, una vez realizado el análisis sobre todas las cohortes, se realizó otro estudio excluyendo las dos cohortes que mostraron el peor rendimiento de forma individual con el fin de evaluar si esto afectaba en el rendimiento global.

Modelo <chr>	Error <dbl>	Precision <dbl>
SVM_PRE	0.2746512	0.7284208
SVM_POST	0.3143365	0.6873055
RandomForest_PRE	0.2957456	0.7049294
RandomForest_POST	0.3178523	0.6832435
GLMNet_PRE	0.2806293	0.7165519
GLMNet_POST	0.3084542	0.6926313

Tabla 5.8: Resultados gráficos de Error y Precisión por modelo y fase

En la Tabla 5.8, se presentan los resultados globales del estudio excluyendo las dos cohortes

con peor desempeño.
De la misma manera que en la Tabla 5.7, aquí también se aprecia un ligero empeoramiento del rendimiento de los modelos cuando los datos han sido procesados con MOBER. El error de los modelos procesados con los datos previos a MOBER oscila entre 0,2746 y 0,2947, mientras que con los datos posteriores a MOBER varía entre 0,3084 y 0,3178. En cuanto a la precisión, los modelos procesados con los datos previos a MOBER presentan valores entre 0,7049 y 0,7284, y con los datos posteriores a MOBER entre 0,6832 y 0,6926.

Al comparar las Tablas 5.7 y 5.8, se puede observar que, al igual que en el estudio anterior, no se detectan diferencias significativas en el rendimiento global de los modelos al eliminar las cohortes que mostraron un peor desempeño individual. Esto sugiere de nuevo que el mal rendimiento individual de las cohortes eliminadas viene derivado por el tamaño de las muestras y, por lo tanto, es lógico que los modelos no se vean sustancialmente afectados por el mal funcionamiento de estas cohortes.

5.11.4 Conclusiones

Tras la realización de estos tres estudios para evaluar el rendimiento de los modelos en la tarea de clasificación entre sanos y enfermos y comparar los rendimientos obtenidos entre los datos previos y posteriores a MOBER, se pueden extraer varias conclusiones importantes.

En primer lugar, se observa que el rendimiento individual de cada cohorte no está afectando significativamente al rendimiento global de los datos cuando se consideran en conjunto. Como se ha mencionado, esto sugiere que esta deficiencia en el rendimiento viene dada por el bajo número de muestras a procesar por el modelo y, por lo tanto, que la variabilidad entre cohortes no tiene un impacto considerable en la eficacia de los modelos al evaluar los datos combinados.

Por otro lado, al analizar el objetivo principal del estudio, se observa que la señal biológica captada por los datos presenta un aumento significativo en los datos posteriores al procesamiento de MOBER en comparación con los datos previos para el experimento realizado con LCO-CV. Sin embargo, para el experimento realizado con CV estándar, la señal prácticamente no presenta diferencias. Esto indica que el procesamiento de los datos con MOBER resulta útil en un experimento de mayor complejidad, como ha sido el de LCO-CV, donde antes de aplicar MOBER los datos se clasificaban de manera más deficiente.

En contraste, MOBER no consigue resultados destacables en un experimento más sencillo, como ha sido el de CV estándar, donde los datos previos a MOBER ya lograban buenos re-

sultados. Debido a la homogeneidad general de los datos, simplemente con su normalización ya se consigue captar gran parte de la señal biológica, la cual es difícilmente mejorable con el procesamiento de MOBER.

A partir de estas observaciones, se concluye que, aunque MOBER parece no estar funcionando de manera óptima en este caso, esto podría deberse a la sencillez y similitud de los datos y no a un mal funcionamiento de la herramienta. MOBER logra una notable eliminación de los efectos de lote, lo que se traduce en una mayor capturación de características biológicas relevantes en los estudios donde, antes de procesar los datos con MOBER, se captaba muy poca señal biológica.

5.12 SPRINT 13 - Creación de contenedor en Singularity

Como se explicó en el Sprint 3, sección 5.3, surgieron complicaciones en el manejo de Singularity, lo que dificultó completar la tarea de creación y puesta en marcha del contenedor. Por esta razón, en este Sprint se retomaron esos objetivos. Se llevó a cabo un estudio más exhaustivo del funcionamiento de Singularity, así como de la creación de archivos ”recipe” y contenedores. Una vez completado este estudio, se procedió a seguir los mismos pasos que en el Sprint anterior para la creación del contenedor, pero esta vez empleando un nuevo archivo ”recipe” en el cual se incorporaron los cambios y mejoras estudiadas, así como todas las funcionalidades del proyecto.

Además, la decisión de posponer la creación del contenedor no se debió solo a las dificultades técnicas encontradas, sino también a la necesidad de disponer de todos los archivos de entrada necesarios para el entrenamiento de MOBER. En el intento inicial de crear el contenedor, no se contaba con estos archivos, fundamentales tanto para la ejecución del contenedor como para las pruebas de su funcionamiento.

Para ello, se realizaron las siguientes acciones:

- Se editaron los archivos en la carpeta utilizada para la creación del contenedor, añadiendo los diversos archivos de entrada necesarios para MOBER (sin normalización y ALR, ya que fue el que se utilizó a lo largo del estudio), y se eliminó todo lo referente a MOBER, dado que su clonado se incluyó en el recipe.
- Se procedió a modificar el recipe inicial conforme a los errores identificados.

```
Bootstrap: docker
From: python:3.8

%labels
    Maintainer Carla Rodriguez
    Version 1.0

%help
    Singularity image for MOBER

%post
    # Actualizar e instalar dependencias necesarias
    apt-get update && apt-get install -y \
        git \
        && rm -rf /var/lib/apt/lists/*

    # Clonar el repositorio de mober
    git clone https://github.com/Novartis/mober.git /opt/mober

    # Instalar mober
    pip install --upgrade pip
    pip install torch
    pip install -e /opt/mober
    pip install numexpr

%runscript
    exec "$@"

%startscript
    exec "$@"
```

Figura 5.17: Estructura del archivo ”recipe”

En la Figura 5.17 se muestra el archivo usado como ”recipe” en la creación del contenedor Singularity. Contiene los siguiente campos:

- **Bootstrap: docker**: Indica que el contenedor se construirá a partir de una imagen Docker.
- **From: ubuntu:20.04**: Especifica la imagen base, en este caso Ubuntu 20.04, a partir de la cual se creará el contenedor.
- **%labels**: Define los metadatos del contenedor, proporcionando información adicional como el nombre, la versión y el autor.
- **%help**: Proporciona una descripción breve del contenedor y su propósito, ayudando a los usuarios a entender qué hace y cómo usarlo.
- **%post**: Contiene los comandos que se ejecutarán dentro del contenedor durante su proceso de creación. Estos comandos suelen estar relacionados con la configuración del entorno y la instalación de paquetes necesarios. En este caso, también incluye la clonación del repositorio de MOBER para integrarlo dentro del contenedor.

- **%runscript**: Define el/los scripts que se ejecutarán cuando se inicie el contenedor. El valor ”$@” se utiliza para tomar cualquier comando proporcionado al iniciar el contenedor y ejecutarlo dentro del mismo.
- **%startscript**: Define el/los comandos que se ejecutarán cuando se inicie el contenedor. Similar al apartado anterior, el valor ”$@” permite la ejecución de comandos específicos proporcionados al iniciar el contenedor, facilitando así la interacción con el contenedor y permitiendo al usuario ejecutar comandos de manera flexible.

Una vez que se tuvo el archivo ”recipe” con todo lo necesario para la creación del contenedor, se procedió a construirlo utilizando los comandos específicos. Posteriormente, se verificó su funcionamiento tanto en el CESGA como en una máquina virtual.

5.13 SPRINTS 14, 15 y 16 - Desarrollo de la memoria

Los Sprints 14, 15 y 16 (que abarcaron un período de 6 semanas) fueron dedicados por la alumna al desarrollo de esta memoria, en la que se realiza una revisión detallada de todos los aspectos sobre los que se ha trabajado en este TFG.

5.14 SPRINT 17 - Revisión final del proyecto

El último Sprint del proyecto implicó una revisión integral por parte de los tutores de todo lo realizado durante el desarrollo del proyecto y lo escrito en esta memoria. A través de esta revisión, la alumna recibió comentarios y feedback detallado sobre cada aspecto del trabajo. Basándose en esta retroalimentación, se dedicó a realizar modificaciones, mejoras y ajustes según las indicaciones de los tutores. Esta fase final fue crucial y supuso el broche final para asegurar la calidad y coherencia de este TFG antes de su presentación final.

Capítulo 6

Pruebas

Tanto en la metodología SCRUM como en otras, es habitual llevar a cabo una serie de pruebas a lo largo del desarrollo de un proyecto para garantizar su calidad y funcionalidad. En este caso concreto se han realizado pruebas de unidad, de integración, funcionales y de aceptación.

6.1 Pruebas de unidad

Estas pruebas se centran en verificar el correcto funcionamiento de componentes individuales del software, como funciones, métodos o clases, de manera independiente. Por lo general, son realizadas por los propios desarrolladores para asegurar la calidad del código que están escribiendo. Estas se realizan siempre a muy bajo nivel.

En este proyecto, las pruebas unitarias se llevaron a cabo conforme se desarrollaba la solución. Utilizando datos concretos o datos aleatorios de prueba, se evaluó el funcionamiento adecuado de las funciones y herramientas desarrolladas, como MOBER o el cálculo de los PCAs. Este enfoque permitió identificar y corregir posibles errores en cada componente del sistema de manera temprana, contribuyendo así a la calidad del producto final.

6.2 Pruebas de integración

Las pruebas de integración son esenciales para verificar que los diferentes componentes del sistema funcionen correctamente cuando se integran entre sí. Su objetivo principal es asegurar que no haya conflictos o errores de comunicación entre los distintos módulos del software. Estas pruebas se realizan a medida que se suman funciones al proyecto, y siempre después de las pruebas de unidad.

En este caso específico, las pruebas de integración consistieron en el análisis conjunto de las funciones. Por ejemplo, se evaluó la interacción entre la aplicación de MOBER y las funciones PCA aplicadas a los datos procesados por MOBER. Posteriormente, se realizaron experimentos de aprendizaje automático con los mismos datos. El objetivo fue verificar que todos los componentes funcionaran correctamente en conjunto y, cuando se detectaron problemas de compatibilidad o comunicación, dedicarse a corregirlos.

6.3 Pruebas funcionales

Las pruebas funcionales se centran en verificar que el software cumpla con los requisitos funcionales especificados. Se llevan a cabo mediante la ejecución de casos de prueba diseñados para cubrir diferentes aspectos y escenarios de uso del software. Se relacionan tanto con las pruebas unitarias como con las pruebas de integración, ya que su propósito es el verificar el correcto funcionamiento de todos los componentes del proyecto en conjunto.

La diferencia con las pruebas unitarias es que las pruebas funcionales se basan en la verificación de todas las funcionalidades desarrolladas en el proyecto en conjunto, mientras que las unitarias lo hacen de cada componente individual.

La diferencia entre las pruebas de integración y las funcionales, aunque ambas requieren que varios componentes interactúen entre sí, es que una prueba de integración puede simplemente verificar que puedes hacer uso de las funcionalidades desarrolladas en conjunto, mientras que una prueba funcional esperaría obtener un valor específico de una ejecución, según dicten los requisitos del producto.

En el caso de este proyecto, se realizaron pruebas funcionales de manera similar a lo realizado para las pruebas unitarias, pero en este caso verificando mediante el conjunto de datos a utilizar en el proyecto tanto el correcto funcionamiento de la totalidad de funcionalidades desarrolladas como conjunto como la coherencia de los resultados obtenidos.

6.4 Pruebas de aceptación

Las pruebas de aceptación son una parte crucial del proceso de desarrollo de software que se centra en verificar que el sistema cumpla con los requisitos y expectativas del cliente o usuario final. Estas pruebas se realizan para validar que el software entregado satisface los criterios de aceptación definidos durante el proceso de desarrollo. A diferencia de las pruebas funcionales que se centran en validar las funciones del sistema, las pruebas de aceptación se

enfocan en la perspectiva del usuario y en garantizar que el software sea útil y cumpla con su propósito.

En el caso de este proyecto, las pruebas de aceptación se llevaron a cabo de manera colaborativa entre la alumna y los tutores. Esto implicó realizar pequeñas exposiciones en las que la alumna presentaba las funcionalidades desarrolladas, mientras que los tutores evaluaban su validez y determinaban si cumplían con los requisitos y expectativas establecidos. En los casos en que surgieron discrepancias o áreas de mejora, se realizaron ajustes y se volvieron a presentar las funcionalidades hasta alcanzar la conformidad deseada. Este enfoque iterativo y colaborativo permitió asegurar que el software desarrollado satisficiera las necesidades y expectativas del proyecto.

El código de este trabajo está disponible en GitHub: Repositorio de GitHub

Capítulo 7

Conclusiones

Una vez finalizado el proyecto, llegó el momento crucial de evaluar lo aprendido a lo largo del proceso y extraer de él las conclusiones más relevantes.

En cuanto a los conocimientos adquiridos durante el proyecto, destacan varios aspectos significativos. En primer lugar, se destaca el aprendizaje sobre la organización necesaria para llevar a cabo un proyecto de estas características. Además, se ha experimentado un primer contacto con una metodología de trabajo ampliamente aplicada en el sector profesional y empresarial, lo que será de gran utilidad en el futuro de la alumna.

En cuanto al tema del trabajo, se ha profundizado en la comprensión de la importancia de los datos en gran cantidad de ámbitos y estudios específicos. Además, se ha aprendido sobre la relevancia del correcto procesamiento de datos previo a los estudios para facilitar las tareas de análisis y obtener resultados más claros y significativos. También se ha ampliado el conocimiento previo adquirido durante el grado en herramientas como R y Python, entre otras.

Por último, se ha explorado con mayor profundidad la relación entre los datos y la salud, un tema que ya había sido abordado durante el grado pero que en el caso de este proyecto se ha estudiado de manera más detallada.

A mayores, resulta importante mencionar las conclusiones obtenidas y las líneas futuras de investigación derivadas de este estudio.

7.1 Conclusiones acerca de lo estudiado

En general, se determinó que los objetivos del trabajo se han cumplido satisfactoriamente, al haber logrado abordar el estudio de la eliminación de los efectos de lote en un conjunto

de cohortes de microbioma. Aunque la herramienta MOBER fue diseñada originalmente para trabajar con datos de transcriptoma, el objetivo de este proyecto fue poner a prueba la eficacia de la herramienta trabajando con datos de microbioma, con el objetivo de evaluar su fiabilidad y establecer un punto de partida para futuros estudios en este campo.

En detalle, respecto a la eficacia de MOBER en datos de microbioma, se concluye que la herramienta es efectiva. Aunque no se observaron mejoras significativas en la captación de la señal biológica presente en los datos y con ello en la clasificación de pacientes entre sanos y enfermos, se evidenció una notable reducción de los efectos de lote.

Como se mencionó, la aparente falta de eficacia en esta clasificación entre sanos y enfermos parece estar más relacionada con la homogeneidad y similitud de los datos que con deficiencias inherentes a la herramienta en sí misma.

A partir de estas reflexiones y todo lo estudiado a lo largo del proyecto, se concluye que MOBER demuestra una eficacia satisfactoria en el procesamiento de datos de microbioma.

7.2 Líneas futuras

Como se ha indicado, el objetivo de este proyecto se ha alcanzado satisfactoriamente. No obstante, a través del estudio realizado surgen varias líneas de investigación futuras orientadas a evaluar de manera más completa la eficacia de la herramienta:

- **Exploración con datos más diversos**: Actualmente, los datos utilizados provienen en su totalidad de cohortes de 16s-rRNA secuenciadas de la misma manera. Sería interesante investigar cómo MOBER se comporta en datos más diversos, con variaciones en la secuenciación y otras características, para evaluar su robustez y generalización.
- **Ampliación del alcance de aplicación**: Relacionado con el anterior punto, otra posible línea futura es la evaluación de la efectividad de MOBER en estudios de microbioma que presenten una mayor variabilidad y complejidad. Esto permitiría determinar si la herramienta es adecuada para diferentes tipos de datos biológicos y contextos experimentales.
- **Comparación con otros métodos**: Realizar comparativas exhaustivas entre la eficacia de MOBER y de otros métodos de eliminación de batch effect para determinar cuál es la opción más efectiva en el contexto del estudio de datos de microbioma.

Apéndices

Apéndice A

PCAs abundancias relativas

En este Apéndice, como en mencionó en la sección 5.9 se presentan las gráficas tanto previas como posteriores a la ejecución de MOBER para los datos normalizados mediante abundancias relativas. Estas gráficas confirman la falta de validez de este método de normalización, lo que llevó a su exclusión del estudio.

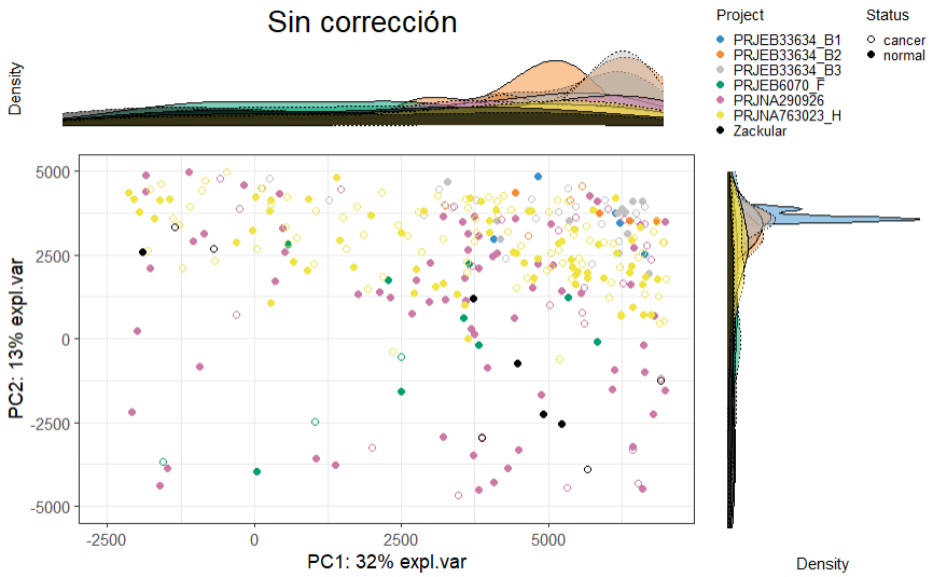


Figura A.1: Resultado de la PCA previa a MOBER con los datos normalizados por abundancias relativas

En la Figura A.1, no se aprecia agrupación por cohortes de los datos, pero tampoco una clara dispersión de los datos en el gráfico. En su lugar, se observan agrupaciones con baja uniformidad, con muchos datos concentrados en algunas zonas y muy pocos en otras.

Además, al comparar esta gráfica con las obtenidas para ALR y CLR, se puede observar que

los límites de esta gráfica son inmensamente mayores que los de las anteriores, lo que sugiere que los datos no se han escalado correctamente.

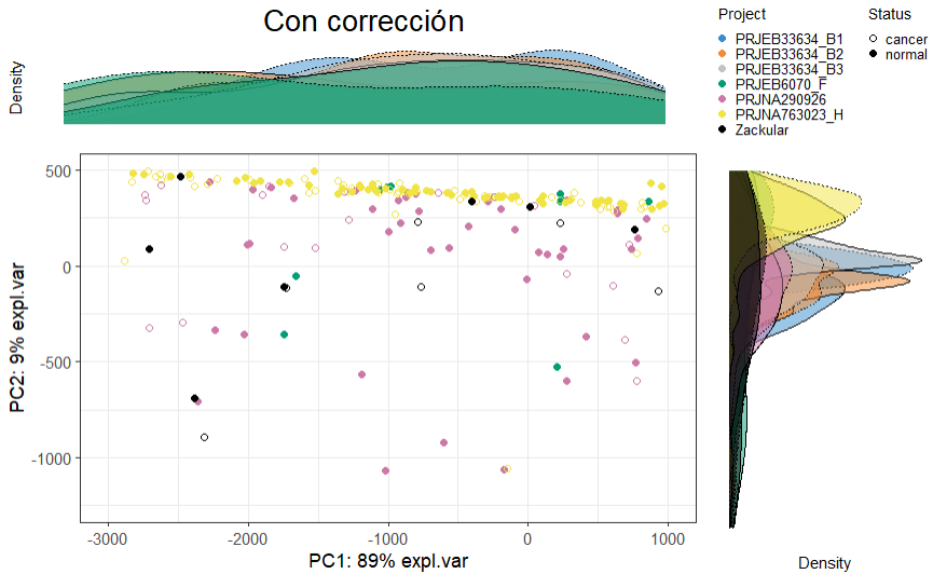


Figura A.2: Resultado de la PCA salida MOBER tras normalización por abundancias relativas

En la Figura A.2, se observa nuevamente una agrupación de los datos, pero de manera muy poco uniforme, con concentración de datos en algunas zonas de la gráfica y datos dispersos en otras zonas. A pesar de esta dispersión, el porcentaje de variabilidad explicada es considerablemente más alto en este caso (97,53%) en comparación con los casos anteriores. Este fenómeno resulta contradictorio, lo que demuestra una vez más que la variabilidad no está exclusivamente vinculada a la presencia o ausencia de efectos de lote, y que el método de normalización de abundancias relativas no es válido en este contexto.

Con estas observaciones y el valor tan alto obtenido para la función de pérdida se concluye, como se mencionó en la sección 5.9, que la normalización por abundancias relativas no es válida y que con esos datos MOBER no ha podido desempeñar su función de manera óptima. Esto resalta la importancia de evaluar la idoneidad de las técnicas aplicadas según el contexto específico de estudio y los datos a analizar. Los diferentes métodos de normalización y las características de los datos influyen en la interpretación de los resultados y en la eficacia de las herramientas utilizadas.

Bibliografía

[1] "Sociedad española de oncología médica (seom)." [En línea]. Disponible en: https://seom.org/

[2] "¿en qué consiste scrum?" [En línea]. Disponible en: https://www.itconsultors.com/metodologia-scrum

[3] "Metagenomic data analysis," 2022. [En línea]. Disponible en: https://server.t-bio.info/tutorials/52

[4] S. Dimitrieva, R. Janssens, G. Li, A. Szalata, R. Gopal, C. Parmar, A. Kauffmann, and E. Y. Durand, "Biologically relevant integration of transcriptomics profiles from cancer cell lines, patient-derived xenografts and clinical tumors using deep learning," 2022. [En línea]. Disponible en: https://www.biorxiv.org/content/10.1101/2022.09.07.506964v2,eprint={https://www.biorxiv.org/content/10.1101/2022.09.07.506964v2.full.pdf},journal={bioRxiv}

[5] "Scrum: conceptos clave y cómo se aplica en la gestión de proyectos." [En línea]. Disponible en: https://asana.com/es/resources/what-is-scrum

[6] "Human microbiome project (hmp)," 2024. [En línea]. Disponible en: https://commonfund.nih.gov/hmp

[7] "National institute of health (nih)." [En línea]. Disponible en: https://commonfund.nih.gov/hmp

[8] O. S. Soriano, "Chemical and microbial ecology and the demosponge aplysina aerophoba," Ph.D. dissertation, Universitat de Barcelona (UB), 2012.

[9] O. Chapleur, C. Madigou, R. Civade, Y. Rodolphe, L. Mazéas, and T. Bouchez, "Increasing concentrations of phenol progressively affect anaerobic digestion of cellulose and associated microbial communities," *NIH*, 2016.

[10] G. Kong, K.-A. L. Cao, L. M. Judd, S. Li, T. Renoir, and A. J. Hannan, "Microbiome profiling reveals gut dysbiosis in a transgenic mouse model of huntington's disease," *NIH*, 2020.

[11] "¿qué es r?" [En línea]. Disponible en: https://datademia.es/blog/que-es-r#:~:text=R%20es%20un%20entorno%20y,sin%20que%20pertenezca%20a%20nadie

[12] "El tutorial de python." [En línea]. Disponible en: https://docs.python.org/es/3/tutorial/

[13] "¿qué puede hacer el software r para resolver tus problemas?" [En línea]. Disponible en: https://www.revista.unam.mx/2019v20n3/que-puede-hacer-el-software-r-para-resolver-tus-problemas/#:~:text=R%20es%20una%20herramienta%20inform%C3%A1tica,y%20figuras%20de%20gran%20calidad

[14] "Spyder - the scientific python development environment." [En línea]. Disponible en: https://www.spyder-ide.org/

[15] "Visual studio code." [En línea]. Disponible en: https://code.visualstudio.com/

[16] "Git." [En línea]. Disponible en: https://git-scm.com/

[17] "Qué es github." [En línea]. Disponible en: https://www.hostinger.es/tutoriales/que-es-github

[18] "Singularity." [En línea]. Disponible en: https://wiki.nlhpc.cl/Singularity

[19] "Las 5 ceremonias scrum: claves para la gestión de procesos." [En línea]. Disponible en: https://www2.deloitte.com/es/es/pages/technology/articles/ceremonias-scrum.html

[20] "Artefactos scrum: las 3 herramientas clave de gestión." [En línea]. Disponible en: https://www2.deloitte.com/es/es/pages/technology/articles/artefactos-scrum.html

[21] "Roles en scrum." [En línea]. Disponible en: https://edu.gcfglobal.org/es/scrum/roles-en-scrum/1/

[22] G. de España, "El sistema universitario español," 2024.

[23] "Linkedin," 2024. [En línea]. Disponible en: https://es.linkedin.com

[24] "Infojobs," 2024. [En línea]. Disponible en: https://www.infojobs.net/

[25] S. Dimitrieva, R. Janssens, G. Li, A. Szalata, R. Gopal, C. Parmar, A. Kauffmann1, and E. Y. Durand, "Biologically relevant integration of transcriptomics profiles from cancer cell lines, patient-derived xenografts and clinical tumors using deep learning," *bioRxiv*, 2022.