UNIVERSIDAD DE TALCA
FACULTAD DE INGENIERÍA
ESCUELA DE INGENIERÍA CIVIL EN COMPUTACIÓN

Predicción de estabilidad de proteínas mutantes usando Vectores de Autocorrelación de Secuencia de Aminoácidos (AASA) y *Machine learning*

BENJAMÍN GUSTAVO PINO RAMÍREZ

Profesor Guía: CÉSAR ASTUDILLO

Profesor Co-guía: JULIO CABALLERO

Memoria para optar al título de
Ingeniero Civil en Computación

Curicó – Chile
Julio, 2020

CONSTANCIA

La Dirección del Sistema de Bibliotecas a través de su encargado Biblioteca Campus Curicó certifica que el autor del siguiente trabajo de titulación ha firmado su autorización para la reproducción en forma total o parcial e ilimitada del mismo.

Curicó, 2022

Dedicado a mi familia y amigos.

AGRADECIMIENTOS

Quiero partir agradeciendo a mi profesor guía, César Astudillo, quien me orientó en el desarrollo de este trabajo y de quien aprendí un montón de cosas; a mi profesor co-guía, Julio Caballero, quien tuvo siempre buena disposición para asistirnos; y a la profesora del Taller de Ciencia de Datos, Yamisleydi Salgueiro, cuyos conocimientos entregados en clases me han servido mucho. Agradezco, además, a todos los profesores de los cuales tuve la oportunidad de aprender durante mi paso por la Universidad.

Quiero dar especiales agradecimientos a quienes son mis compañeros y amigos, Ariel Cornejo, Diego Iturriaga, Diego Matus, Felipe Milla, José Núñez, Gabriel Sanhueza y Roberto Ureta, con quienes compartí muchos buenos momentos que recordaré por mucho tiempo. Me siento realmente afortunado de haberlos conocido.

Finalmente, debo agradecer a quienes son las personas más importantes en mi vida y por quienes intento crecer día a día, mi familia. Agradezco a mi papá, César Pino, y a mi mamá, María Victoria Ramírez, quienes se han preocupado mucho por mi, especialmente durante este período. Agradezco a mi abuelo, Antonio Ramírez y a mi tío, Esteban Ramírez quienes siempre me han apoyado. Agradezco a mis hermanos, Agustina, Gabriela, Rafaela y Raimundo, quienes me dan las ganas de soñar. Finalmente, quiero agradecer a quien me crió y educó, a quien soportó mis malos humores, a quién me ha entregado su cariño sin importar las circunstancias, y a quien por tanto debo mucho, a mi abuela. Todos ellos son el núcleo de la persona que soy hoy. A puertas de cerrar un ciclo importante en mi vida, no hago sino ansiar el día en que pueda retribuir todo lo que han hecho por mi, dándoles la tranquilidad y el bienestar que merecen. Todo mi esfuerzo es por ellos. Desde el fondo de mi corazón, muchas gracias.

TABLA DE CONTENIDOS

página

ÍNDICE DE FIGURAS

página

ÍNDICE DE TABLAS

página

RESUMEN

Contexto: Las proteínas son moléculas con una gran diversidad de funciones en la naturaleza. Éstas poseen una propiedad medible, que es la *estabilidad conformacional*, la cual se relaciona con su resistencia a altas temperaturas. Proteínas con alta estabilidad tienen muchas aplicaciones, pero desarrollarlas en un proceso costoso. Para asistir al experto en este proceso, se ha estudiado el uso de técnicas de *machine learning* para la predicción de estabilidad de proteínas mutantes.

Los vectores AASA son una representación cuantitativa de las proteínas que ha demostrado ser útil para modelar la estabilidad en investigaciones anteriores. Sin embargo, solo se ha aplicado en conjunto con técnicas de modelado no lineal. Si modelos más simples, como lo son los lineales, tienen un desempeño de predicción bueno o, al menos, similar al de modelos más complejos, los primeros son más deseables pues son más interpretables y útiles para análisis posteriores.

Problema: Desarrollar un método para modelar la estabilidad de las mutantes y que permita comparar modelos lineales y no lineales.

Solución propuesta: Se propone una metodología para determinar, empíricamente, si es que las técnicas de modelado lineal tienen un buen desempeño para la predicción de la estabilidad a partir de vectores AASA, y cómo su desempeño se compara con el de técnicas de modelado no lineal. Esta metodología es aplicada a cuatro conjuntos de datos distintos, evaluando y comparando el desempeño de cuatro técnicas de modelado lineal y una de modelado no lineal, usando tres variantes de *Cross-validation* (CV), *Nested* CV, *5-Fold* CV y 5x2 CV.

Resultados: Se observa que *5-Fold* CV y 5x2 CV producen estimaciones de desempeño con menos variabilidad que *Nested* CV, por lo que son más fiables. Se observa, además, que el desempeño de las técnicas de modelado lineal es consistentemente bajo a través de los distintos conjuntos de datos abordados y, en la mayoría de los casos, inferior al desempeño de la técnica de modelado no lineal.

Conclusiones: Se observa una superioridad de los métodos no lineales, en particular de SVR con *kernel* RBF, en el reconocimiento de patrones en los datos. Adicionalmente, se obtiene una metodología reproducible, útil para el análisis de conjuntos de datos disponibles a futuro, ya que es independiente de las técnicas de modelado y la implementación realizada en este trabajo es de libre acceso.

1. Introducción

El presente capítulo parte con una descripción a grandes rasgos del trabajo a realizar. Luego, se detallan aspectos del contexto de la investigación y la problemática a resolver. Finalmente, se describe la propuesta de solución y se definen las preguntas de investigación, objetivos y alcances del trabajo.

1.1. Descripción de la propuesta

Dentro del área de la bioinformática existen diversos campos de investigación. Uno de ellos, en el cual se enmarca este trabajo, contempla el estudio de los efectos que producen las mutaciones en una proteína sobre su estabilidad conformacional. Desarrollar proteínas de alta estabilidad es de gran interés, pues su desarrollo puede contribuir a áreas como la medicina [7] y la biotecnología [20], entre otras. Sin embargo, los métodos convencionales de laboratorio para encontrar mutaciones con alta estabilidad son sumamente costosos [16]. A raíz de ésto, una de las soluciones que se ha explorado a lo largo de los años es el uso técnicas de *machine learning* para la predicción de los cambios de estabilidad que produce una mutación en particular.

En este trabajo se aplican cinco técnicas de *machine learning* para entrenar modelos de predicción de variación de la estabilidad inducida por mutaciones. Esta investigación se enfoca en una manera en particular de codificar cuantitativamente la información de una proteína, que son los Vectores de Autocorrelación de Secuencias de Aminoácidos (AASA por su sigla en inglés), con los cuales ya se han realizado investigaciones anteriormente [5, 8, 9, 10, 16].

El objetivo de este trabajo es documentar, a partir de evidencia empírica, si es que los algoritmos de *machine learning* que modelan relaciones lineales entre atributos y variable de respuesta permiten modelar la relación entre vectores AASA y

estabilidad de las proteínas. Con esta idea en mente, se busca proponer una metodología reproducible de modelado predictivo que pueda ser útil para investigaciones futuras.

1.2. Contexto del proyecto

Las proteínas son las moléculas más versátiles presentes en organismos vivos [13], pues cumplen una diversidad de funciones. Éstas evolucionan naturalmente en medios biológicos. Sin embargo, profesionales como genetistas crean mutaciones de ellas en entornos de laboratorio con fines de investigación. A partir de una proteína salvaje (*wild-type* en inglés), se crean proteínas mutantes, las cuales son resultado de determinadas modificaciones en la cadena de aminoácidos de la proteína original. Las mutaciones alteran una propiedad medible de las proteínas, denominada estabilidad conformacional. La estabilidad está relacionada con la resistencia de la molécula a altas temperaturas; más estabilidad se traduce en una mayor resistencia. Así, las proteínas mutantes pueden tener una estabilidad mayor o menor con respecto a la original.

Lograr desarrollar mutaciones con mayor estabilidad es de gran interés, pues éstas tienen diversas aplicaciones, como por ejemplo en el desarrollo de medicamentos [7] o en el estudio de aplicaciones biotecnológicas e industriales de enzimas [20]. La estabilidad se puede medir en laboratorios utilizando varios métodos [13]. Sin embargo, éstos pueden ser muy costosos en términos de insumos químicos o especialistas [16]. Sumado a ésto, la cantidad de posibles mutaciones de una proteína es alta [16], lo que eleva la cantidad de intentos que se deben realizar. Por ello, a lo largo de los años se ha trabajado en variados métodos computacionales que le permitan a un experto saber de antemano cuáles mutaciones podrían presentar alta estabilidad.

En [9], sus autores trabajaron una forma de representar cuantitativamente información de proteínas y sus mutaciones, llamada Vectores de Autocorrelación de Secuencias de Aminoácidos (AASA por su sigla en inglés). Los vectores AASA codifican un conjuntos de datos definiendo atributos, llamados *descriptores*, utilizados para modelar la estabilidad conformacional. Existen varias otras maneras de representar una proteína cuantitativamente. Sin embargo, como se señala en [9], los vectores AASA tienen ciertas ventajas por sobre otras representaciones. Es por ello que ya se han realizado investigaciones utilizándola en conjunto con técnicas de *machine*

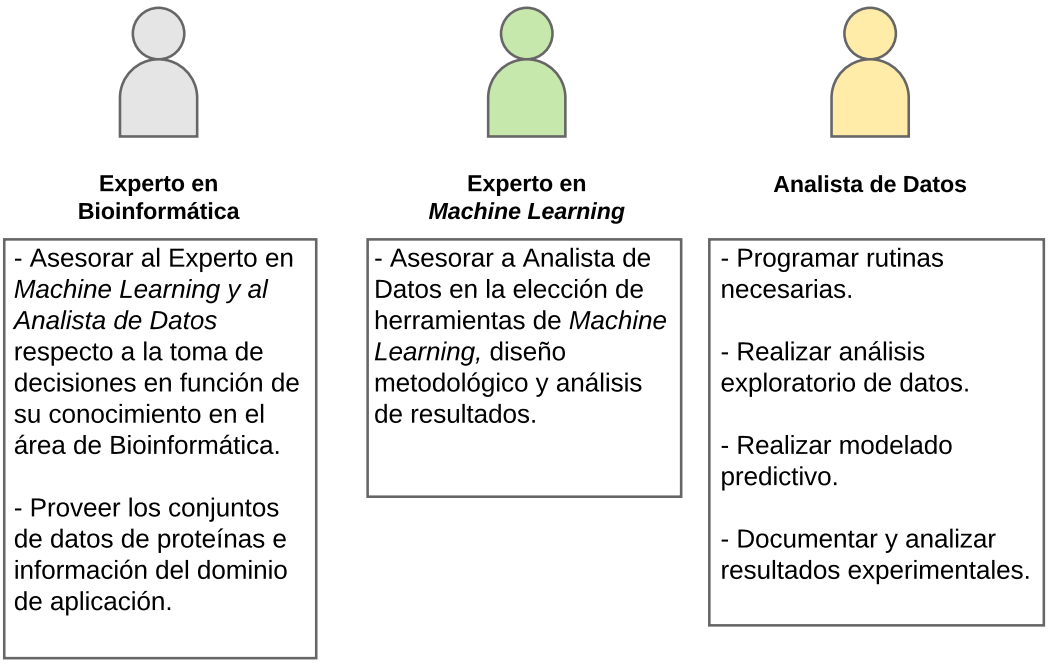


Figura 1.1: Equipo de trabajo y labores de investigación.

learning [5, 8, 10, 16].

En esta investigación se evalúa el desempeño de determinadas técnicas de *machine learning* para entrenar modelos de predicción usando datos de proteínas codificados como vectores AASA. El trabajo es realizado por un equipo interdisciplinario compuesto por un Experto en Bioinformática, un Experto en *machine learning* y un Analista de Datos. Las tareas de cada miembro se detallan en la Figura 1.1.

1.3. Definición del problema

Una de las áreas de investigación de la Bioinformática involucra el estudio de la creación de proteínas de alta estabilidad a partir de mutaciones hechas a una determinada proteína original. La problemática general a resolver se hace presente en el proceso de medición de la estabilidad por métodos convencionales de laboratorio. Por cada proteína existe una gran cantidad de mutaciones posibles, de las cuales solo una pequeña parte resultan en una proteína más estable que la original. A causa de ello, el proceso para encontrar proteínas más estables involucra una gran cantidad

de ensayo y error, lo que es altamente costoso producto de la demanda de tiempo, insumos químicos y de personal especializado para realizar los experimentos.

En las investigaciones [5, 8, 9, 10, 16], se ha abordado esta problemática utilizando técnicas de *machine learning* para entrenar modelos de predicción usando los descriptores AASA. Entre dichas investigaciones, los trabajos que han abordado el problema de predecir la variación de la estabilidad inducida por una mutación documentan únicamente el uso de redes neuronales y del algoritmo *Suppor Vector Regression* con *kernel Radial Basis Function* (RBF por su sigla en inglés). Éstas técnicas modelan relaciones no lineales entre descriptores y estabilidad. Hasta el momento no se ha documentado la aplicación de técnicas de modelado lineal. Considerando lo anterior, surgen las siguientes preguntas:

- ¿Será que modelos entrenados con algoritmos de aprendizaje que modelan relaciones lineales entre predictores y respuesta, tienen un buen desempeño en la predicción de la estabilidad de proteínas a partir de vectores AASA?

- ¿Será que los modelos de predicción entrenados con dichos algoritmos tienen un desempeño similar al de modelos más complejos como lo son aquellos entrenados por *SVR* con *kernels* no lineales?

Esta investigación tiene como principal objetivo responder estas interrogantes de manera empírica.

Resultados de investigaciones anteriores, muestran la dificultad de extraer patrones de los datos cuando se trata de entrenar modelos para predecir la estabilidad como una variable cuantitativa. No se sabe aún qué métodos pueden resultar útiles, por lo que es un problema abierto.

Es necesario considerar, además, que las investigaciones anteriores tiene ciertos aspectos que pueden ser mejorados. Uno de ellos, es que las metodologías de modelado predictivo tienen una complejidad alta producto de las técnicas de *machine learning* utilizadas. Otro aspecto, es que los códigos fuente que implementan dichas metodologías son de difícil acceso producto de la antigüedad de las publicaciones, de las cuales la más reciente es del año 2008. Éstos dos aspectos pueden dificultar, en cierto grado, la replicabilidad y reproducibilidad de dichas metodologías, ya que su codificación y aplicación demanda un esfuerzo adicional. Es por ello que es necesario utilizar una metodología de modelado lo más simple posible, con el objetivo de

generar modelos de predicción no solamente precisos, sino que también simples, que puedan ser analizados posteriormente por un experto.

1.4. Propuesta de solución

Para responder a las interrogantes planteadas, en este trabajo se propone una metodología de modelado predictivo, cuya implementación sea de libre acceso, que contemple la aplicación y evaluación de cuatro técnicas de *machine learning* de modelado lineal utilizadas sobre conjuntos de datos de proteínas codificados como vectores AASA.

En la primera etapa del trabajo se contempla llevar a cabo un análisis exploratorio de datos, con los objetivos de determinar la presencia de patrones y anomalías en los datos, y de extraer conocimiento acerca de los descriptores y sus relaciones. En particular, se contempla el uso de técnicas de visualización como lo son gráficos de correlación y mapas de calor, entre otros. Además, se utilizan algoritmos de *clustering* ampliamente conocidos como *clustering jerárquico* y *K-means*, con el objetivo de formar agrupaciones de observaciones que den información sobre algún patrón presente en los datos.

En la segunda etapa se contempla entrenar los modelos de predicción. En los trabajos anteriores, se ha documentado únicamente el uso de redes neuronales y del algoritmo SVR con *kernel* RBF, generando modelos no lineales y de alta complejidad. Ésto es sustentado por la premisa de que los fenómenos biológicos son complejos por naturaleza y, usualmente, las técnicas de modelado lineal son superadas por otras que son capaces de modelar relaciones más complejas [5]. No se ha documentado la aplicación de algoritmos de modelado lineal para generar modelos de predicción usando vectores AASA. Se considera que existe un valor en documentar la aplicación de métodos más simples para intentar resolver el problema. Es por esto que, con los objetivos de saber si las técnicas de modelado lineal son útiles para la predicción de estabilidad usando vectores AASA, y de saber si su desempeño es comparable a alguna técnica de modelado no lineal, se contempla aplicar cuatro técnicas de modelado lineal, *Ordinary Least Square Regression* (OLS), *Partial Leasts Square Regression* (PLS), *Ridge Regression*, *Lasso Regression*, y una técnica de modelado no lineal, SVR con *kernel* RBF y *kernel* polinomial.

La metodología se aplica de manera genérica e independiente, con la idea de

reportar los resultados obtenidos en cada caso.

Se espera que la metodología propuesta pueda ser generalizada para investigaciones futuras que requieran análisis sobre otros conjuntos de datos no abordados en este trabajo.

1.5. Preguntas de investigación

- **RQ1**: ¿Será que modelos entrenados con algoritmos de aprendizaje que modelan relaciones lineales entre predictores y respuesta, tienen un buen desempeño en la predicción de la estabilidad de proteínas a partir de vectores AASA?
- **RQ2**: ¿Será que los modelos de predicción entrenados con dichos algoritmos tienen un desempeño similar al de modelos más complejos como lo son aquellos entrenados por *SVR* con *kernels* no lineales?

1.6. Objetivos

Objetivo general

Proponer y validar una metodología de entrenamiento y evaluación de modelos para la predicción de estabilidad de proteínas mutantes partir de vectores AASA, que sea reproducible e independiente al tipo de proteína.

Objetivos específicos

1. Llevar a cabo un análisis exploratorio de datos con el fin de extraer conocimiento sobre las características de los datos.
2. Realizar un preprocesamiento de los datos, contemplando la aplicación un algoritmo de selección de atributos.
3. Entrenar y evaluar modelos de predicción de estabilidad que describan tanto relaciones lineales como no lineales entre descriptores y estabilidad.
4. Comparar el desempeño de modelos lineales y no lineales.
5. Validar la reproducibilidad de la metodología propuesta aplicándola sobre, al menos, tres conjuntos de datos distintos.

1.7. Alcances

- Este trabajo contempla la implementación de las rutinas necesarias para ejecutar cada uno de los pasos de la metodología propuesta, así como también para la realización de los experimentos y validaciones.
- Se contempla utilizar solo implementaciones de técnicas de *machine learning* ya existentes.

1.8. Descripción de contenido

El resto del documento está estructurado en cuatro capítulos. En el Capítulo 2, se habla sobre antecedentes relacionados tanto con el dominio de aplicación como con técnicas de *machine learning* en general. En el Capítulo 3 se habla sobre la metodología utilizada para realizar la investigación. En el Capítulo 4, se presenta el desarrollo de la investigación y el análisis de los resultados. Finalmente, en el Capítulo 5 se plantean conclusiones y alternativas para trabajos futuros.

2. Marco teórico

En este capítulo se definen y describen de manera genérica los conceptos que se utilizan en el trabajo. En primer lugar, se describen conceptos relacionados a *machine learning* en general. En segundo lugar, se presentan conceptos relacionados al dominio de los datos y al contexto del desarrollo del trabajo. Finalmente, se describen conceptos relacionados directamente con el desarrollo, como la metodología genérica y las tecnologías a utilizar.

2.1. *Machine Learning*

En escenarios del mundo real, existen problemas que son altamente complejos para resolverse con un enfoque tradicional o para los cuales simplemente no existe un algoritmo evidente que los solucione. Un ejemplo de ésto es el determinar si un correo electrónico es *spam*. Particularmente para éste tipo de problemas, las técnicas de *machine learning* tienen ventajas por sobre otros enfoques.

A grandes rasgos, el *machine learning*, conocido también como *aprendizaje automático* en español, es la ciencia de darle a los computadores la habilidad de aprender los datos.

Alpaydin en [1, págs. 1-4], describe el concepto de aprendizaje como el extraer patrones a partir de un conjunto de datos disponible con el objetivo de generar una solución aproximada que permita realizar predicciones sobre nuevos datos.

Durante una etapa de aprendizaje, o *entrenamiento*, el computador busca regularidades presentes en los datos y genera una aproximación de la solución real. Al ser problemas complejos, en muchos casos lo mejor a lo que se puede optar es a una solución parcial o estimada, pero que sea lo *suficientemente* buena y útil.

En el campo del *machine learning*, se distinguen cuatro grandes tipos de aprendizaje: supervisado, no supervisado, semisupervisado y de reforzamiento. En este trabajo se aplican los primeros dos.

2.2. Aprendizaje supervisado

Este tipo de aprendizaje involucra estimar una solución a partir de ejemplos, u *observaciones*, de las cuales cada una posee un conjunto de atributos y una correspondiente *respuesta* o *etiqueta* asociada [12, pág. 105]. El objetivo es predecir una variable Y, denominada comúnmente variable *objetivo*, en función de un conjunto de p variables independientes $X_1, ..., X_p$, denominadas *predictores* [15, pág. 15].

Aplicando un algoritmo de aprendizaje, se entrena un modelo de predicción a partir un conjunto de n observaciones $(x_1, y_1), (x_2, y_2)..., (x_n, y_n)$, donde $x_i = (x_{i1}, x_{i2}, ..., x_{ip})$ es un vector con los valores de cada predictor para la i-ésima observación, e y_i es su correspondiente valor observado para la variable objetivo.

Los problemas de aprendizaje supervisado, a su vez, se pueden dividir en dos grandes grupos. Si la variable respuesta Y es *cuantitativa*, se denominan problemas de *regresión*, mientras que si es *cualitativa*, se denominan problemas de *clasificación* [15, pág. 28].

2.2.1. Regresión

Un problema de regresión es aquel donde la variable objetivo Y es cuantivativa, y está descrita por una función f desconocida, la cual explica la relación *real* entre ésta y un vector de entradas X, denominadas predictores. Comúnmente, sin embargo, no es posible conocer dicha función, por lo que la mejor solución a la que se puede optar es a una aproximación de ella, una función $\hat{f}$, derivada de un proceso de entrenamiento. La función $\hat{f}$, al ser una estimación de la función verdadera, las predicciones que hagamos con ella no serán siempre correctas. Así, la predicción $\hat{Y}$ está definida como

$$\hat{Y} = \hat{f}(X) \tag{2.1}$$

El proceso de entrenamiento consiste en resolver un problema de optimización en el cual, a partir de un conjunto de datos de entrenamiento, se estima una función $\hat{f}$

que aproxime lo suficiente la función real f, minimizando una cierta función de *error*, también llamada en inglés *loss function* u *objective function*, la cual cuantifica el error de predicción del modelo $\hat{f}$ en función de los valores predichos y los valores reales observados de la variable de respuesta [3, págs. 41,46]. Para ésto, es necesario disponer de una muestra de datos que sea lo más representativa posible de la población. El cómo se encuentra una función $\hat{f}$ depende del algoritmo de entrenamiento que se utilice. No existe un algoritmo que sea mejor que otros en todos los escenarios [12, pág. 116], pues cada uno tiene sus propias ventajas y desventajas frente a distintos problemas. La elección de cuál utilizar depende del contexto.

2.2.2. Regresión lineal

La regresión lineal es una técnica de aprendizaje que asume de antemano que la *verdadera* relación entre la variable objetivo y el conjunto de predictores es aproximadamente lineal [15, pág. 63]. Cuando el modelo contempla un solo predictor, se denomina regresión lineal *simple* [15, pág. 61], mientras que cuando contempla más de un predictor, se denomina regresión lineal *múltiple* [15, pág. 71]. Dado que se asume que f es aproximada por una función lineal, la relación entre variable objetivo y predictores se puede escribir como

$$Y \approx \beta_0 + \left(\sum_{j=1}^{p} \beta_j X_j \right) \tag{2.2}$$

donde X_j representa el j-ésimo predictor y β_j cuantifica la relación entre dicho predictor y la variable de objetivo. Cada coficiente β_j es interpretado como el cambio promedio en el valor de Y correspondiente a un cambio de una unidad en el valor del predictor X_j, cuando todos los demás predictores permanecen constantes.

De acuerdo a lo señalado en [15, pág. 63], se puede decir que el modelo dado por la Ecuación 2.2 es la mejor aproximación lineal de la verdadera relación entre Y y los predictores, y usualmente desconocida.

Dado que se desconocen los coeficientes del modelo definido por la Ecuación 2.2, éstos pueden ser estimados usando un conjunto de datos de entrenamiento. Así, el modelo de predicción ajustado usando los datos de entrenamiento se puede escribir como

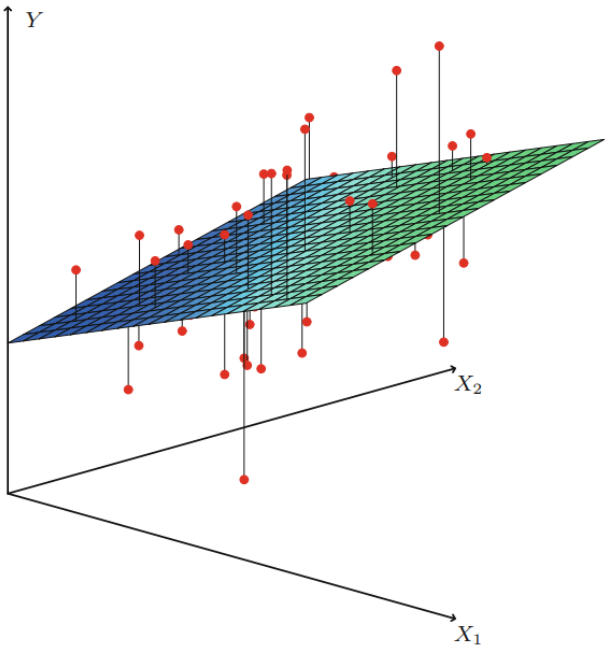


Figura 2.1: Ejemplo de hiperplano de dos dimensiones para una variable de respuesta Y, formado dos predictores, X_1 y X_2. Figura extraída de [15, pág. 73].

$$\hat{Y} = \hat{\beta}_0 + \left(\sum_{j=1}^{p} \hat{\beta}_j X_j \right) \tag{2.3}$$

donde $\hat{\beta}_0$, ... , $\hat{\beta}_p$ son *estimaciones* de β_0, ... , β_p. Así, la predicción para una observación está definida como

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \hat{\beta}_2 x_{i2} + ... + \hat{\beta}_p x_{ip} \tag{2.4}$$

donde $\hat{y}_i$ es el valor *predicho* para de la variable objetivo correspondiente a una observación x_i. El modelo definido por la Ecuación 2.3 es representado gráficamente como un hiperplano de p dimensiones. En la Figura 2.1, se muestra un ejemplo de un hiperplano bidimensional que describe la relación entre una variable de respuesta Y y dos predictores X_1 y X_2. Si fuera un solo predictor, ese plano pasaría a ser una recta.

El modelo estimado definido por la Ecuación 2.4, al ser una aproximación, puede generar predicciones distintas a los valores observados. Así, se puede establecer que

$$y_i = \hat{y}_i + \epsilon_i \tag{2.5}$$

donde ϵ_i corresponde a la diferencia entre el valor observado y_i y el valor predicho por el modelo $\hat{y}_i$; a dicha diferencia se le conoce también como *residual*. Los coeficientes se estiman de manera que estas diferencias sean lo más bajas posibles. Distintas técnicas de regresión tienen distintas maneras de estimar los coeficientes.

2.2.3. *Ordinary Least Squares Regression*

Ordinary Least Squares Regression (OLS por su sigla en inglés) es una técnica de regresión lineal que busca aquellos coeficientes que minimizan la suma del cuadrado de los errores de predicción para un conjunto de n observaciones de entrenamiento. Gráficamente, significa buscar un hiperplano que minimice las distancias entre éste y las observaciones. Para ajustar los coeficientes se define la función de error *residual sum of squares* como

$$\begin{aligned} RSS &= \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \\ &= \sum_{i=1}^{n} (y_i - \hat{\beta}_0 - \hat{\beta}_1 x_{i1} - ... - \hat{\beta}_p x_{ip})^2 \end{aligned} \tag{2.6}$$

donde y_i - $\hat{y}_i$ es la diferencia entre la i-ésima respuesta observada y_i, y la predicción arrojada por el modelo, $\hat{y}_i$. Esta diferencia es el *error* de predicción para la i-ésima observación. Cada diferencia es elevada al cuadrado como una manera de dejarlas positivas.

De acuerdo a [15, pág. 61], *least squares* es por lejos el método más común determinar $\hat{\beta}_0, ..., \hat{\beta}_p$. Sin embargo, éste puede tener problemas de *overfitting* cuando modelos muy complejos son entrenados usando conjuntos de datos de tamaño limitado [3, pág. 147], en particular cuando $p > n$ o $p \approx n$ [15, pág. 240]. El *overfitting*, o sobreajuste en español, es un fenómeno que se presenta cuando los modelos se ajustan demasiado a los datos de entrenamiento, perdiendo capacidad de *generalización* [1, pág. 39]. Los métodos de regresión *Ridge* y *Lasso* son alternativas que lidian con este fenómeno.

2.2.4. *Ridge Regression*

Ridge Regression es una técnica de regresión lineal similar a OLS, que no solo busca minimizar el error del modelo, sino que también lidiar con el *overfitting*. Ésto lo logra aplicando el concepto de *regularización*, o *contracción*, el que involucra introducir un término de penalización a la función de error con el objetivo de restringir el valor de los coeficientes [1, pág. 80][12, pág. 120]. La función de error extendida está definida por la expresión

$$RSS + \lambda \sum_{j=1}^{p} \hat{\beta}_j{}^2 \tag{2.7}$$

donde $\lambda >= 0$ es un parámetro de ajuste. El primer término, RSS, corresponde a la suma de los cuadrados de los errores de predicción definida anteriormente. El segundo término $\lambda \sum_{j=1}^{p} \hat{\beta}_j{}^2$ es un término de penalización, y tiene un efecto de disminuir hacia cero los coeficientes β_j [15, pág. 215]. Es importante notar que β_0 no es incluido en el término de penalización. La regularización penaliza la complejidad del modelo; a mayor valor de λ, más pequeños se vuelven los coeficientes [14, pág. 63] y el modelo se vuelve más simple [1, pág. 80].

Es aquí donde se introduce un nuevo problema. Los coeficientes estimados son distintos para cada valor de λ, por lo que es crítico encontrar el valor adecuado. Si es un valor muy pequeño, la penalización no será tan fuerte, y tendremos un modelo casi igual de complejo que el determinado por el método de *least squares*. Por el contrario, si es un valor muy grande, los coeficientes serán tan pequeños que el modelo se volverá extremadamente simple, y habrá un *underfitting*, que es cuando el modelo no logra modelar los datos de entrenamiento, pero tampoco es capaz de generalizar. Para determinar el valor de λ, usualmente se utiliza *cross-validation* para seleccionar aquel valor que lleva a una mejor generalización.

Los coeficientes estimados usando *Ridge* son dependientes de la escala en la que están los predictores, por lo que comúnmente los datos deben ser normalizados [14, pág. 63]. Es importante señalar que éstos nunca serán cero [15, pág. 219], por lo que los modelos generados con *Ridge* siempre contiene todos los p predictores y, desde cierto punto de vista, ésto puede ser considerado una desventaja.

2.2.5. *Lasso Regression*

Lasso Regression es una técnica de regresión lineal muy similar a *ridge*, pero que difiere en el término de regularización introducido en la función de error. En este caso, se busca minimizar la expresión

$$RSS + \lambda \sum_{j=1}^{p} |\hat{\beta}_j| \tag{2.8}$$

En lugar de $\hat{\beta}_j{}^2$, se introduce $|\hat{\beta}_j|$, haciendo que la regularización tenga un efecto distinto al que tiene en *Ridge*. Cuando el valor de λ es lo suficientemente alto, algunos de los coeficientes pueden volverse exactamente cero [3, pág. 145] [15, pág. 219]. Ésto se debe a que el método de *Lasso* prefiere asignar un valor mayor a unos coeficientes y desechar los otros en lugar de distribuirlo [1, pág. 352], contrario a lo que sucede con *Ridge*. Ésto implica que existe una *selección de atributos*, haciendo que el modelo resultante pueda estar compuesto por un subconjunto de predictores. Desde luego, el tener un modelo con menos predictores contribuye a una mayor interpretabilidad, sobre todo cuando se trata de un problema de alta dimensionalidad. Ésta característica representa una ventaja por sobre *Ridge*.

2.2.6. *Partial Least Squares*

Partial Least Squares (PLS) es una técnica de regresión lineal que internamente hace una reducción de dimensionalidad antes de ajustar un modelo lineal. Para algún $M \leq p$, PLS construye M nuevos predictores Z_m, de los cuales cada uno es una combinación lineal de los p predictores originales $X_1, ..., X_p$. A estos nuevos predictores se les denomina *componentes principales*. De acuerdo a [14, pág. 81], PLS busca las componentes principales que tienen más varianza y correlación con la variable de respuesta. Este proceso de reducción de la dimensionalidad constituye una transformación del espacio original. En la Figura 2.2, se muestra gráficamente la tranformación de los datos que ocurre cuando se calculan las componentes principales.

Finalmente, usando los nuevos predictores $Z_1, ..., Z_M$, ajusta un modelo lineal usando *least squares*, donde en lugar de estimar $p+1$ coeficientes, se estiman $M+1$. El modelo de predicción se expresa como

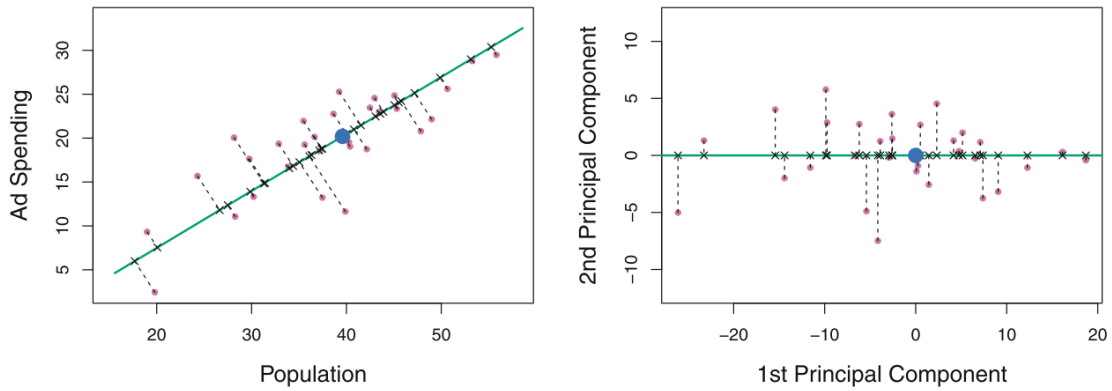


Figura 2.2: Ejemplo de la transformación realizada por PLS antes de ajustar el modelo. A la izquierda se grafican observaciones en función de dos predictores, *Population* y *Ad Spending*. A la derecha se grafican las observaciones en función de sus componentes principales calculadas a partir de dichos predictores. Figura extraída de [15, pág. 232].

$$y_i = \theta_0 + \sum_{m=1}^{M} \theta_m z_{im} + \epsilon_i \tag{2.9}$$

donde $\theta_0, ..., \theta_M$ corresponden a los coeficientes asociados a las componentes principales y z_{im} corresponde al valor de la m-ésima componente principal para la i-ésima observación.

PLS no es invariante a las escalas de los predictores [15, pág. 236][14, pág. 80], por lo que es recomendable normalizarlos cuando éstos son medidos en distintas escalas. El valor de M se suele determinar usando *cross-validation* [15, pág. 238].

2.2.7. Regresión no lineal

Los métodos de regresión lineal tienden a generar modelos menos complejos y más interpretables. Sin embargo, al ser tan restrictivos, tienen problemas cuando la relación entre la variable objetivo y los predictores no es lineal. Es aquí donde aparecen métodos de mayor flexibilidad, y que permiten definir relaciones no lineales entre predictores y variable de respuesta. En esencia, éstos métodos *extienden* el espacio original de los descriptores a uno que permita encontrar una estimación adecuada de la función f. En este trabajo, se utiliza *Support Vector Regression* (SVR).

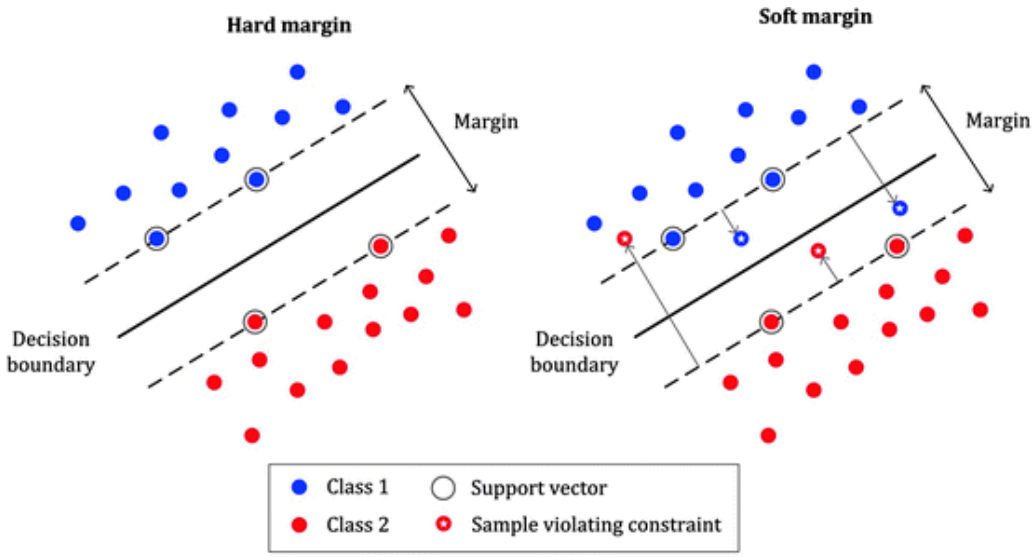
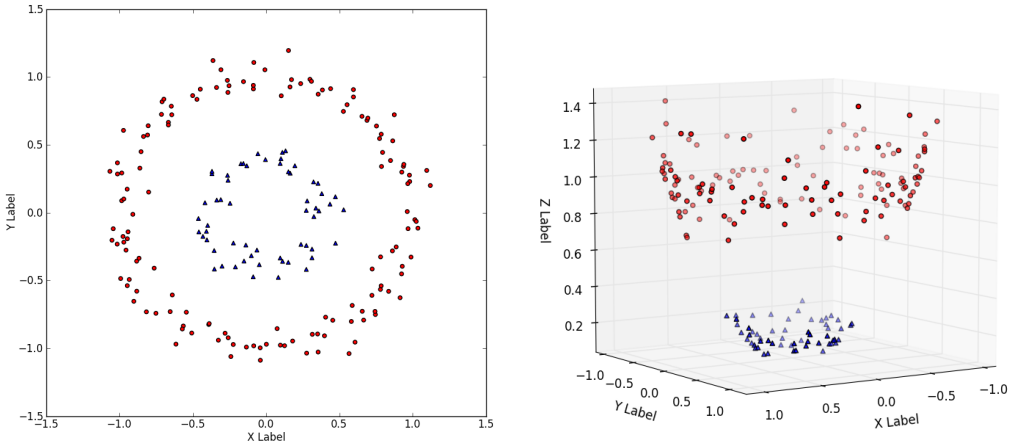


Figura 2.3: Diferencia entre un margenes suaves y rígidos para el caso de un hiperplano separador unidimensional

2.2.8. *Support Vector Machines*

Support Vector Machines (SVM) es una técnica de ampliamente utilizada para resolver problemas de clasificación. Para un problema que involucra dos clases, dado un conjunto de observaciones de entrenamiento, el algoritmo busca un *hiperplano* de manera que las observaciones de una clase queden separadas de las de la otra clase. Ésto lo hace maximizando el *margen*, que corresponde a la mínima distancia entre el hiperplano y las observaciones [1, pág. 311] [15, pág. 341]. En la Figura 2.3 se puede observar como el tamaño del margen define una frontera a cada lado del hiperplano; cada frontera corresponde a una de las clases.

En el caso en que las observaciones son linealmente separables, es decir, cuando *existe* un hiperplano que las separa, SVM lo encuentra y asegura que todas las observaciones están al menos a un margen de distancia [15, pág. 343]. En tal caso el margen es considerado *rígido*. Sin embargo, comúnmente no existe un hiperplano que logre separar las observaciones manera exacta. Para ésto, SVM permite otorgar cierto grado de tolerancia al error a través de un parámetro C, lo que posibilita que algunas de las observaciones de entrenamiento puedan estar del lado incorrecto de su correspondiente frontera e, incluso, del lado incorrecto del hiperplano [1, pág. 315][15, pág. 345]. En tal caso, el margen es considerado *suave*. La Figura 2.3 muestra la



(a) Datos no separables en espacio original.

(b) Proyecciones separables en espacio extendido.

Figura 2.4: Ejemplo de *kernel trick*. Los datos no son separables en $\mathbb{R}^2$. Sin embargo, sus proyecciones en $\mathbb{R}^3$ dadas por una función *kernel* sí lo son.

diferencia entre ambos tipos de márgenes.

Las observaciones que están del lado incorrecto de su correspondiente frontera se denominan *descriptores de soporte*, y de éstas depende qué tan amplio es el margen [15, pág. 347]. Así, el modelo solo está definido por un subconjunto de las observaciones de entrenamiento, y es insensible al comportamiento de las demás [1, pág. 314] [3, pág. 327].

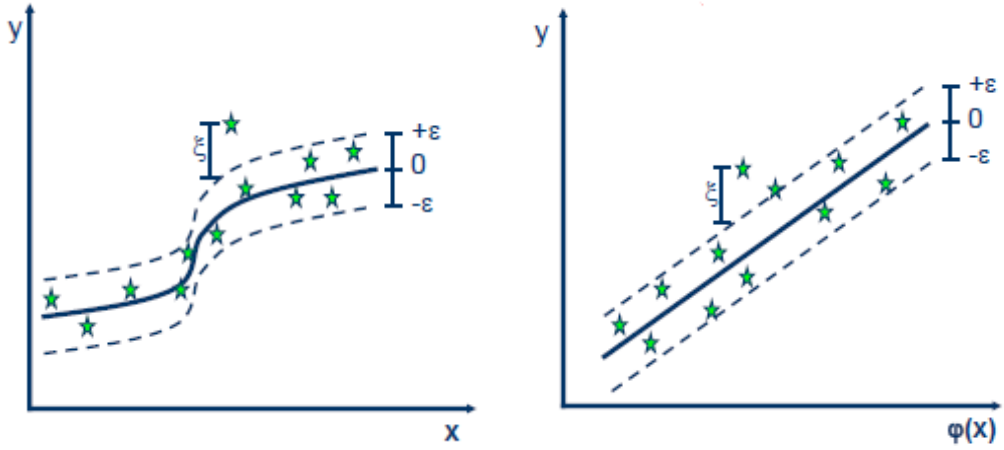
El parámetro C regula el balance entre minimización del error y la maximización del margen [1, pág. 317]. Un valor alto de C corresponde a una mayor penalización del error de clasificación en los datos de entrenamiento, resultando así en un margen menos amplio. Ésto puede producir un problema de *overfitting* [1, pág. 317], ya que provoca que el modelo se ajuste mejor a los datos de entrenamiento. Por el contrario, un valor bajo para C corresponde a una menor penalización del error, lo que permite ampliar el margen tolerando cierta cantidad de errores de clasificación. Ésto puede contribuir a una mejor generalización, pues hace que el modelo se ajuste menos a los datos de entrenamiento. En otras palabras, el parámetro C es un parámetro de regularización, similar a λ en los casos de *Lasso Regression* y *Ridge Regression*. Elegir el valor de C es importante y, comúnmente, se hace a través de técnicas como *Cross-validation*. De acuerdo a [1, pág. 318], se suele probar con valores de una escala logarítmica $(10^{-6}, 10^{-5}, ..., 10^{5}, 10^{6})$.

Los casos descritos hasta ahora asumen que las observaciones son, al menos, cercanas a ser linealmente separables. Sin embargo, hay casos como el que se muestra en la Figura 2.4a, para los cuales un hiperplano definido directamente en el espacio original de los datos no es adecuado. Aquí es donde se introduce el concepto de *kernel trick*. Éste consiste en utilizar una función *kernel* que, en términos prácticos, permite trabajar con las observaciones como si estuvieran en un espacio extendido donde las clases sí son linealmente separables [1, pág. 319]. La Figura 2.4b muestra la proyección en un espacio extendido de los datos mostrados en 2.4a. El espacio extendido lo define la función *kernel* que se utiliza. Existen varias alternativas entre las cuales se encuentran los *kernels* lineal, polinomial, RBF y sigmoideo. Típicamente no se sabe de antemano cuál de todos es mejor dado un determinado problema. Es por ello que usualmente se prueba más de uno, y se elige el que permita generar un mejor modelo.

2.2.9. *Support Vector Regression*

El algoritmo de SVM tiene una variante que extiende el algoritmo original adaptándolo para resolver problemas de regresión. A dicha variante se le conoce como *Support Vector Regression* (SVR). La mayor parte de las características y funcionamiento de SVR es igual al de SVM, con unas ligeras diferencias.

El objetivo de SVR es encontrar un hiperplano que describa los datos. En caso de que ello no sea posible en el espacio original, es posible aplicar el *kernel trick* para llevarlos a un espacio en donde sí puedan ser ajustados por un modelo lineal. El modelo lineal encontrado en el espacio extendido corresponde a un modelo no lineal en el espacio original [1, pág. 319]. La Figura 2.5 muestra un ejemplo del *kernel trick* aplicado en SVR. Para el caso de regresión, en lugar de un *margen*, se define ϵ (*epsilon*), un parámetro que determina una región alrededor del hiperplano, denominada *tubo* ϵ. Todas las observaciones que están a lo más a una distancia ϵ del hiperplano, es decir, aquellas que están dentro del tubo, son ignoradas pues su error es considerado bajo [14, pág. 435]. Por el contrario, las observaciones que están fuera del tubo se consideran lejanas, y contribuyen al error del modelo. Éstas últimas son, en el caso de SVR, los vectores de soporte. El parámetro de regularización C tiene los mismos efectos que en SVM, pues controla la cantidad de error permitido para los vectores de soporte.



(a) Datos ajustados por una función no lineal en el espacio original.

(b) Datos ajustados por un modelo lineal en el nuevo espacio.

Figura 2.5: Ejemplo de *kernel trick* en SVR. El modelo no lineal en el espacio original corresponde al modelo modelo lineal en el espacio extendido.

2.3. Aprendizaje no supervisado

En un problema de aprendizaje no supervisado, al contrario del aprendizaje supervisado, no es de interés predecir una variable de respuesta Y, sino descubrir regularidades en las observaciones a partir únicamente de las p variables independientes $X_1, ..., X_p$ [15, pág. 26]. En este trabajo se aplica *clustering*, una de las técnicas de aprendizaje no supervisado.

2.3.1. *Clustering*

El *clustering* es una técnica que busca encontrar subgrupos de observaciones en los datos [15, pág. 385]. Las observaciones dentro de un subgrupo deberían ser aquellas que son más similares entre ellas. De manera similar, observaciones en grupos distintos deberían ser disimilares entre ellas. Ésta técnica es útil para la visualización de patrones en los datos, por lo que es una herramienta importante en el análisis exploratorio de datos. De a acuerdo a [15, pág. 386], los dos algoritmos más conocidos de *clustering* son *K-means* y *clustering jerárquico*.

2.3.2. K-means

Es un algoritmo de agrupamiento que separa las observaciones en K *clusters* distintos. La cantidad de *clusters* K debe ser definida de antemano. Una i-ésima observación pertenece únicamente a un *cluster*, por lo que la unión de todos los grupos es el conjunto de datos completo. Los pasos del algoritmo son los siguientes:

1. Aleatoriamente, asignar a cada observación un número de 1 a K. Ésto resulta en configuración inicial de *clusters*.

2. Iterar hasta que las asignaciones de *clusters* no cambien:

   a) Para cada *cluster* C_k, calcular el *centroide*, que es el vector de las medias de los p predictores, calculado a partir de las observaciones pertenecientes al *cluster*.

   b) Asignar cada observación al *cluster* cuyo centroide es el más cercano. La cercanía está determinada por la distancia euclidiana.

La Figura 2.6 muestra un ejemplo de cómo funciona *K-means*. La idea detrás de *K-means* es agrupar las observaciones de manera que la variación *intra-cluster* sea la menor posible [15, pág. 386]. Dicha variación es una medida de qué tan cercanas, o *similares*, son las observaciones de un mismo *cluster* entre ellas. La variacion *intra-cluster* de un cluster C_k está definida como

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 \tag{2.10}$$

donde i e i' representan cada par de observaciones distintas pertenecientes a C_k. El algoritmo *K-means* provee una solución óptima local al problema de encontrar una *clusterización* cuya suma de variaciones *intra-cluster* sea mínima [15, págs. 387-388]. Dicha solución es dependiente de la asignación aleatoria inicial [15, pág. 388], por lo que usualmente se genera más de una *clusterización* con distintas configuraciones iniciales, escogiendo aquella que resulta en la menor variación *intra-cluster* total.

2.3.3. *Clustering* jerárquico

Este algoritmo se caracteriza por producir una jerarquía con respecto a la cercanía entre las observaciones. Dicha jerarquía se representa gráficamente como un

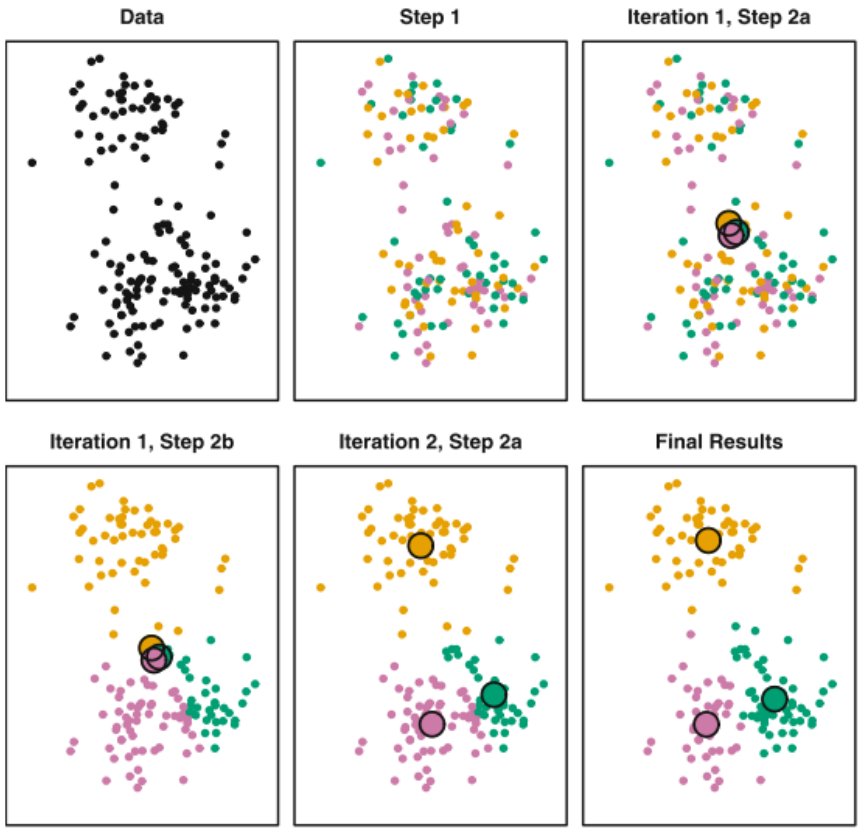


Figura 2.6: Ejemplo de funcionamiento de *K-means* con $K = 3$. Figura extraída de [15, pág. 389].

dendrograma, una representación similar a un *árbol*. Un dendrograma, como el mostrado en la Figura 2.7, grafica de abajo hacia arriba como las observaciones van formando *clusters* hasta conformar una sola agrupación. Dos *clusters* que se une a una menor altura en el árbol son más cercanos que dos *clusters* que se unen a una mayor altura. En la Figura 2.7 se muestra como un mismo dendrograma, *cortado* a distintas alturas, puede generar *clusterizaciones* con distintas cantidades de agrupaciones. Por ello, no es necesario conocer el número de *clusters* K de antemano, contrario a lo que sucede con *K-means*.

Análoga a la *distancia* definida en *K-means*, en *clustering* jerárquico es necesario definir una métrica de *disimilitud*. La métrica a usar depende de los datos con los que se está trabajando. Comúnmente se utiliza la distancia euclidiana, siempre que sea adecuada.

Existen dos tipos de *clustering* jerárquico. Uno es el aglomerativo, donde se parte

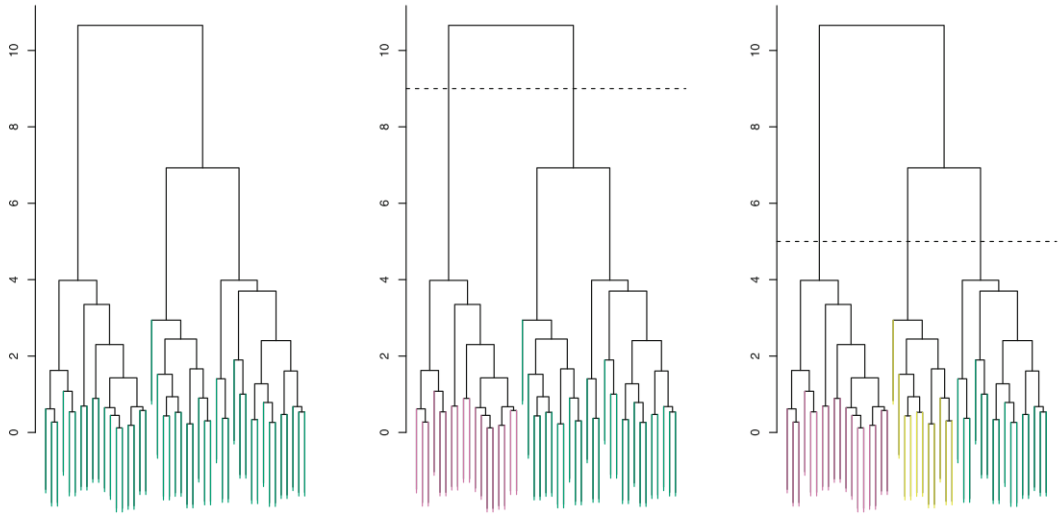


Figura 2.7: Ejemplo de resultado *clustering* jerárquico aglomerativo. De izquierda a derecha, se grafican las *clusterizaciones* correspondientes para un valor de k entre 1 y 3, respectivamente. Figura extraída de [15, pág. 392].

considerando cada elemento del conjunto como un *cluster*, y luego se van uniéndo en función de su similaridad. El otro es divisivo, que al contrario del aglomerativo, parte desde el conjunto completo como un solo *cluster* y va dividiendo en grupos en función de su disimilitud. El aglomerativo es el más común de los dos.

Para un conjunto de datos de n observaciones, los pasos del algoritmo de *clustering* jerárquico aglomerativo son los siguientes:

1. Inicialmente es necesario tener la distancia, o *disimilitud*, entre cada par observaciones del conjunto de datos. Cada observación se considera un *cluster* de un elemento.

2. Para $i = n, n-1, ..., 2$:

   a) Unir el par de *clusters* que tienen la menor disimilitud entre ellos, formando un único *cluster* a partir de ambos, y quedando con un total de $i-1$ *clusters*.

   b) Calcular la distancia entre cada par de los $i-1$ *clusters*.

La disimilitud entre dos *clusters* depende del método de *enlace* (*linkage* en inglés). De acuerdo a [15, pág. 394-395], existen cuatro métodos comunes de enlace: *complete*,

average, *single* y *centroid*, siendo los dos primeros los recomendados por el autor. A continuación se describen estos dos métodos de enlace:

- *Complete*: De todas las disimilitudes entre elementos de un *cluster* A y elementos de un *cluster* B, aquella que es *máxima* es considerada la disimilitud entre ambos *clusters*.
- *Average*: De todas las disimilitudes entre elementos de un *cluster* A y elementos de un *cluster* B, el *promedio* de todas ellas es la disimilitud entre ambos *clusters*.

2.4. Evaluación de modelos

A continuación se definen las métricas y mecanismos de evaluación de modelos de predicción a utilizar.

2.4.1. Métricas de desempeño

Las métricas de desempeño indican, en palabras simples, qué tan bien se ajusta el modelo a la distribución de un conjunto de datos; en otras palabras, qué tan buenas son las predicciones del modelo sobre dicho conjunto. En el contexto de los problemas de regresión existen varias métricas. Entre las más comunes se encuentran R^2 y RMSE.

R^2 es una métrica que indica la proporción de la variación en la variable dependiente que es explicada por el modelo [15, pág. 70]. Para un conjunto de datos de n observaciones, esta métrica se define matemáticamente como:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2} \tag{2.11}$$

donde $\overline{y}$ es la media de las respuestas observadas, y_i, e $\hat{y}_i$ es la predicción asociada a la i-ésima instancia. Normalmente, R^2 toma valores en un rango $[0, 1]$. Generalmente, valores cercanos a uno indican un buen desempeño del modelo, y valores cercanos a cero indican un mal desempeño. Sin embargo, cuando el modelo está muy alejado de la tendencia de los datos observados, R^2 también puede tomar valores negativos.

RMSE (*Root Mean Squared Error* en inglés) es una métrica que mide qué tan alejados están los valores observados del las predicciones hechas por el modelo [15, pág. 29]. A más distancia, mayor es el valor de RMSE. Esta métrica se define como:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}} \tag{2.12}$$

RMSE Se mide en las unidades de la variable de respuesta. Un valor más bajo indica un buen desempeño del modelo.

2.4.2. *k-Fold Cross-validation*

La evaluación de un determinado método de aprendizaje o modelo consiste en estimar su *desempeño de generalización*, que indica qué tan buenas son las predicciones para observaciones que el modelo no ha "visto" [14, pág. 219]. Usar el desempeño que tiene un modelo sobre los mismos datos de entrenamiento como una estimación del desempeño de generalización puede introducir un sesgo optimista [1, pág. 475]. Esto se debe a que los algoritmos funcionan minimizando el error de entrenamiento, y si se fuesen a realizar predicciones sobre esos mismos datos, el error, teóricamente, debería ser bajo. Por esta razón, es una buena práctica evaluar el desempeño de un modelo sobre un conjunto de datos distinto al usado para entrenamiento. Además, para combatir los efectos que podrían tener las particularidades tanto del conjunto de entrenamiento como el de pruebas, idealmente, la estimación del desempeño debería hacerse sobre distintos pares de conjuntos.

En un escenario donde se tuviera una cantidad considerable de datos a disposición, bastaría con extraer múltiples pares de conjuntos de entrenamiento y validación. Sin embargo, ésto rara vez sucede en la práctica; por el contrario, los datos suelen ser escasos. Aquí es donde entra *k-Fold Cross-validation*, un método de remuestreo que permite simular lo anterior mediante la extracción de múltiples conjuntos de datos a partir de un único conjunto disponible.

k-Fold Cross-validation consiste en dividir *aleatoriamente* un conjunto de datos en k partes de aproximadamente el mismo tamaño, denominadas *folds*. Para generar cada par, se aparta una de las k particiones como conjunto de pruebas y las $k-1$ particiones restantes se combinan para formar un conjunto de entrenamiento. Ésto se repite k iteraciones, cada vez apartando una parte distinta como conjunto de pruebas.

Cada iteración se entrena y evalúa el modelo con un par distinto de conjuntos de entrenamiento y prueba. De ésta manera, se obtienen k puntajes de desempeño del modelo de acuerdo a alguna métrica definida. El promedio de los puntajes representa

Figura 2.8: Diagrama explicativo de k-Fold Cross-validation para un valor k igual a 10.

la estimación del desempeño de generalización. La Figura 2.8 muestra un esquema de cómo funciona un 10-Fold Cross-validation. La literatura indica que los valores de k usados comúnmente son 5 y 10 [1, pág. 487] [15, pág. 184].

2.4.3. *Nested Cross-validation*

Cada algoritmo de entrenamiento tiene sus propios parámetros de funcionamiento, llamados *hiperparámetros*. El desempeño de los modelos entrenados por un algoritmo depende, en gran parte, de la selección de dichos parámetros. El proceso por el cual se determina la mejor combinación de parámetros del algoritmo se denomina *optimización de hiperparámetros*. Como mejor combinación, usualmente, se selecciona aquella que tiene el mejor desempeño sobre algún conjunto de datos de prueba. En el caso de conjuntos de datos de pocas instancias, se suele seleccionar el que tiene un mejor desempeño de generalización en un proceso de *k-Fold Cross-validation*. Luego, el modelo es entrenado con los parámetros optimizados.

Un problema se hace presente cuando se utiliza el mismo conjunto de datos para el entrenamiento del modelo y la optimización de hiperparámetros. En los trabajos [23] y [6], se ha mostrado evidencia de que usar los mismos datos para ambas tareas puede resultar en estimaciones sesgadas del desempeño de generalización.

Es aquí donde *Nested Cross-validation* entra como una alternativa para combatir este problema. Consiste en utilizar dos ciclos de *k-Fold Cross-validation* de manera anidada, donde el interno se encarga de la optimización de hiperparámetros, y el

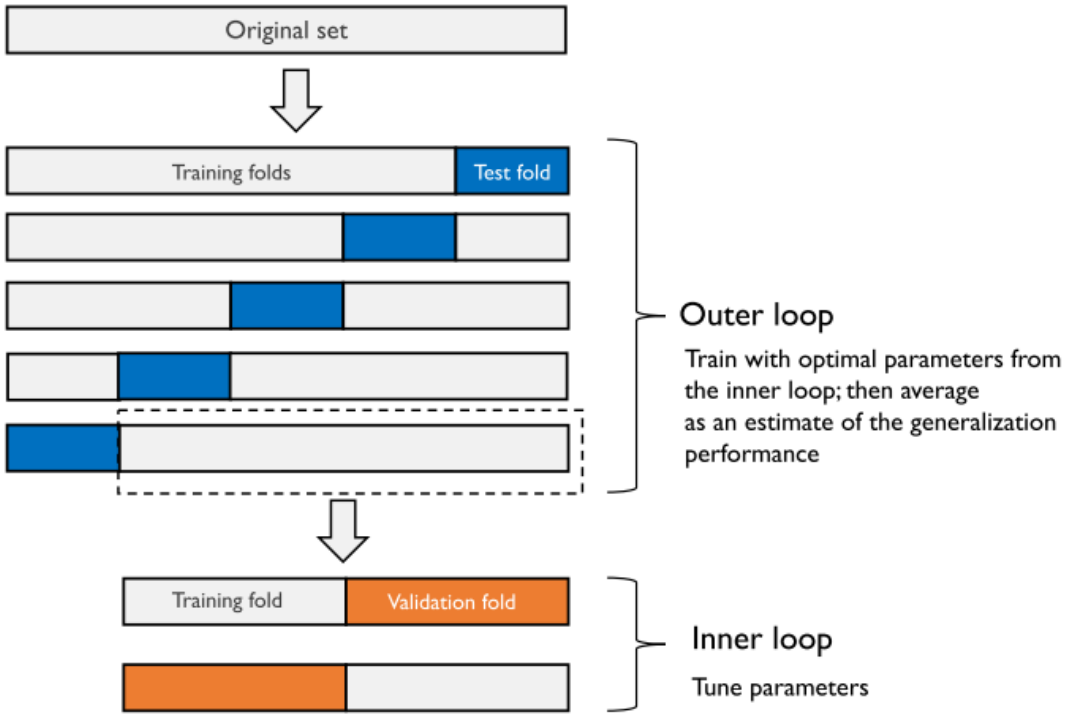


Figura 2.9: Diagrama de Nested Cross-validation. Figura extraída de [18].

externo de estimar el desempeño de generalización del algoritmo. De esta manera, se asegura que ambas tareas se hagan ocupando conjuntos de datos de prueba distintos.

Por cada iteración del ciclo externo, el conjunto de entrenamiento pasa previamente por un ciclo interno de *k-Fold Cross-validation*, obteniendo aquella combinación de hiperparámetros que tenga el mejor desempeño de generalización en promedio, y es con dicha combinación con la que se entrena el modelo sobre el conjunto de entrenamiento. Posteriormente, el modelo es evaluado en el conjunto de pruebas definido para esa iteración del ciclo externo. De ésta manera, como resultado del proceso completo se tiene una estimación del desempeño del algoritmo, basado en *los mejores modelos* que genera por cada iteración del ciclo externo. Los valores de k pueden ser distintos entre ciclo externo e interno. La Figura 2.9 ilustra el proceso completo con un 5-Fold Cross-validation para el ciclo externo y un 2-Fold Cross-validation para el ciclo interno.

2.4.4. 5x2 *Cross-validation*

5x2 *Cross-validation* es un método de validación similar a *k-Fold Cross-validation*. Este consiste en 5 iteraciones, donde en cada una el conjunto de datos es dividido aleatoriamente en dos partes de similar tamaño; primero se utiliza un *fold* para entrenamiento y el otro para pruebas, y luego sus roles se invierten. De ésta manera, por cada iteración se tienen dos puntajes. Para las 5 iteraciones, se tiene un total de 10 puntajes. La estimación del desempeño de generalización está dada por el promedio de los diez puntajes. Cabe mencionar que el muestreo aleatorio realizado en una iteración es independiente del muestreo hecho en las otras iteraciones. Este método de validación se define formalmente en [1, pág. 488].

2.5. Preprocesamiento de datos

En esta sección se detallan describen las técnicas utilizadas para el preprocesamiento de datos.

2.5.1. Normalización Z-Score

La fórmula de normalización Z-Score, definida en la Ecuación 2.13, se utiliza para transformar la distribución de una variable X en una cuya media y desviación estándar sean 0 y 1, respectivamente.

$$Z = \frac{x - \mu}{\sigma} \tag{2.13}$$

donde x es el valor de X a normalizar, μ y σ son la media y la desviación estándar de X, respectivamente.

2.5.2. Coeficiente de correlación de Pearson

El coeficiente de correlación de Pearson es un estadístico que indica la potencia y la dirección de una relación lineal entre dos variables. Varía entre -1 y 1. Un coeficiente de que tiende a -1, indica que cuando el valor de una variable incrementa, el de la otra decrementa. Un coeficiente que tiende a 1 indica que cuando el valor de una variable incrementa, el de la otra también lo hace. Un coeficiente igual a 0

indica que no existe una relación lineal entre las variables. El coeficiente para dos variables X e Y se calcula como

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{2.14}$$

2.5.3. Algoritmo de Selección de Atributos Relief

Relief es un algoritmo de selección de atributos que, a partir de un conjunto de datos, calcula un puntaje para cada uno de los atributos, el cual representa su *relevancia* para distinguir la variable objetivo. Existe una familia de algoritmos basados en *Relief* (*Relief-based Algorithms* (RBAs) en inglés). De acuerdo a [22], los RBAs logran detectar interacciones entre atributos sin la necesidad de hacer una búsqueda de subconjuntos de atributos.

Para un conjunto de datos con una cantidad de atributos a, el algoritmo funciona a grandes rasgos de la siguiente manera:

1. Inicializar la relevancia de todos los a atributos en 0.
2. Para m instancias R_i seleccionadas aleatoriamente:
   - *a*) Determinar la instancia más cercana a R_i que sea de la misma clase (H), y la más cercana de la clase contraria (M).
   - *b*) Por cada atributo A, actualizar su relevancia en función de la diferencia de los valores de dicho atributo entre R_i y M, y ente R_i y H.

Un atributo A incrementa su relevancia en el paso 2b, si es que la diferencia de los valores para dicho atributo entre instancias de la misma clase es baja y si la diferencia entre instancias de distinta clase es alta. En el Paso 1, para determinar las instancias más cercanas, *Relief* utiliza la distancia euclidiana. Ésto hace que, de manera implícita, esté teniendo en cuenta interacciones entre otros atributos al momento de calcular la relevancia. El parámetro m, determina la cantidad de instancias utilizadas para calcular la relevancia de atributos.

De acuerdo a [22], *Relief* es rara vez utilizado, al contrario que su variación más conocida, llamada *ReliefF*. A diferencia de *Relief*, *ReliefF* utiliza $m = n$, lo que quiere decir que utiliza todo el conjunto de datos, generando estimaciones más robustas sobre la relevancia de los atributos [22]. Otra diferencia, es que *ReliefF*

determina los k vecinos más cercanos de la misma clase y los k vecinos más cercanos de distinta clase, en lugar de únicamente un vecino por cada clase.

Relief y *ReliefF* están pensados originalmente para problemas de clasificación. Sin embargo, *ReliefF* tiene una extensión para problemas de regresión, llamada *RReliefF*, cuyos detalles pueden ser encontrado en [19].

Como se menciona en [22], una de las desventajas de RBAs es que no descarta atributos que son redundantes. Por ello, es necesario lidiar con la redundancia de datos previamente.

Más detalles sobre éstos algoritmos pueden ser encontrados en [19], [21] y [22].

2.6. Antecendentes del dominio de aplicación

En esta sección se describen algunos de los conceptos relacionados con el dominio de aplicación, tales como qué son las proteínas, qué es la estabilidad conformacional, qué son los vectores AASA y cómo se han aplicado en conjunto con técnicas de *machine learning* en investigaciones anteriores.

2.6.1. Estabilidad conformacional

Las proteínas son moléculas complejas que tienen muchas funcionalidades en los organismos. Éstas están formadas como cadenas de aminoácidos, y pueden ser representadas como una secuencia de letras, como por ejemplo ASTCGFHCSD.

Las proteínas poseen una propiedad medible llamada *estabilidad conformacional*, y que tiene que relación con el proceso de *desnaturalización por calor* de una proteína [16], que es el proceso en el cual una proteína pierde su estructura producto de su exposición a altas temperaturas. Una alta estabilidad indica que una proteína tiene una alta resistencia a altas temperaturas. De manera similar, una estabilidad baja indica que la proteína es poco resistente a altas temperaturas.

2.6.2. Mutaciones de proteínas y estabilidad

A partir de una proteína salvaje (*wild-type* en inglés) se pueden generar múltiples mutaciones, las cuales consisten en intercambiar un aminoácido por otro, como por ejemplo de A**S**TCGFHCSD a A**D**TCGFHCSD. Las mutaciones pueden inducir un

cambio en la estabilidad con respecto a la de la proteína original. De ésta manera, una mutante puede presentar uno de los siguientes casos:

- Tiene una estabilidad igual a la de la proteína salvaje. Ésto indica el aminoácidos que es reemplazado no influye en la estabilidad.
- Tiene una estabilidad menor al de la proteína salvaje. Ésto indica que el aminoácido es esencial para mantener su estabilidad.
- Tiene una estabilidad mayor de la proteína salvaje. Éstos son los casos de interés.

2.6.3. Vectores de Autocorrelación de Secuencia de Aminoácidos

Los vectores Autocorrelación de Secuencia de Aminoácidos (AASA por su sigla en inglés) son una representación cuantitativa de información estructural de proteínas. Ésta representación es introducida y explicada detalladamente en los trabajos [5] y [8].

Los vectores de autocorrelación calculados codifican información estructural acerca de toda la proteína. Los vectores AASA se calculan como

$$AASAlp_k = \frac{1}{L}\sum_i \delta_{ij} p_{ki} p_{kj} \tag{2.15}$$

donde $AASAlp_k$ es la correlación de secuencia de aminoácidos a una distancia l ponderada por una propiedad p_k; l corresponde a la distancia lineal en la secuencia de aminoácidos [16] y L corresponde a la cantidad de elementos de la suma distintos de cero; p_{ki} y p_{kj} son los valores de la propiedad p_k de los aminoácidos i y j en la secuencia, respectivamente; δ_{ij} es una función Dirac-delta [10] que toma un valor de 1 para aquellos pares de aminoácidos que tienen una distancia l en la secuencia, y un valor de 0 para los que tienen una distancia distinta de l.

Como ejemplo, para un decapéptido cuya secuencia de aminoácidos es ASTCGFHCSD, los vectores AASA con un l igual a 1 e igual a 5 se calcula como

$$\begin{aligned} AASA1p_k =& \frac{1}{9}(p_{kA} * p_{kS} + p_{kS} * p_{kT} + p_{kT} * p_{kC} + p_{kC} * p_{kG} + p_{kG} * p_{kF} + \\ & p_{kF} * p_{kH} + p_{kH} * p_{kC} + p_{kC} * p_{kS} + p_{kS} * p_{kD}) \end{aligned} \tag{2.16}$$

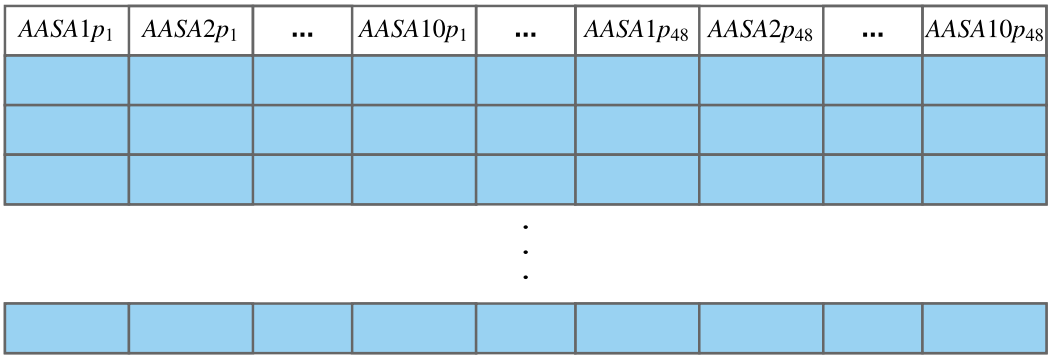


Figura 2.10: Ejemplo de estructura de un conjunto de datos codificado como vectores AASA.

$$AASA1p_k = \frac{1}{5}(p_{kA} * p_{kF} + p_{kS} * p_{kH} + p_{kT} * p_{kC} + p_{kC} * p_{kS} + p_{kG} * p_{kD}) \quad (2.17)$$

En la práctica, los datos son codificados tomando los valores de 48 propiedades p_k distintas. A su vez, por cada propiedad, se calculan vectores usando 10 distancias l distintas. Así, finalmente un conjunto de datos está codificado por 480 vectores AASA. La Figura 2.10 muestra un ejemplo de la estructura de un conjunto de datos de proteínas codificado usando vectores AASA. Cada columna es un vector AASA, donde las componentes corresponden a los valores calculados para cada una de las mutaciones. En este contexto, los atributos construidos usando los vectores AASA se denominan *descriptores*.

La Figura 2.11 muestra como se calcula el vector o descriptor $AASA5p_k$ para un conjunto de datos de mutaciones; la primera instancia corresponde a la proteína salvaje, mientras que las que le siguen son las mutantes generadas a partir de ella. Cada mutante difiere en un aminoácido, el cual es marcado en color rojo. Finalmente, cada componente del vector es calculado en función de la secuencia de aminoácidos definida por cada mutación.

Cada instancia del conjunto de datos tiene asociado un valor corrrespondiente a la variación de estabilidad inducida por la mutación. Una variación de estabilidad mayor a 0 indica que su estabilidad mejora respecto a la de la proteína original,

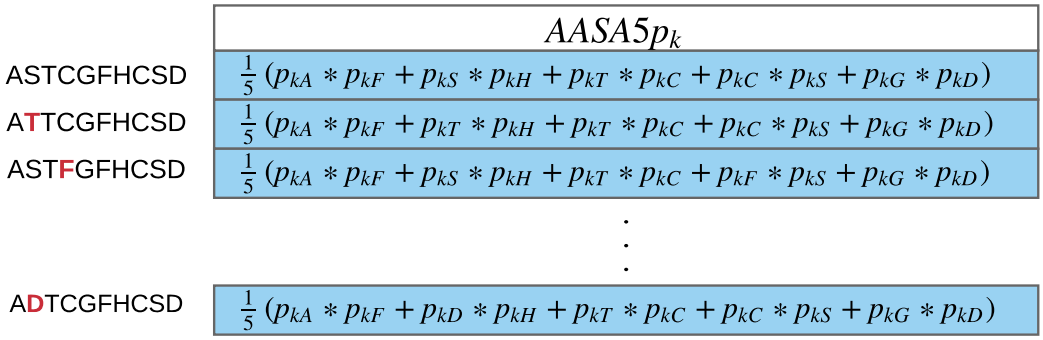


Figura 2.11: Cálculo de un vector AASA con un $l = 5$ para la propiedad p_k.

Cuadro 2.1: Ejemplo de conjuntos de datos real codificado usando vectores AASA.

Mutation	**AASA1K0**	**AASA2K0**	**AASA3K0**	**... AASA10f**	**Stability**
WT	945,88	949,46	948,32	... 4,18	0
L36G	944,01	947,31	946,31	... 4,11	-5,4
Y91S	943,91	947,14	946,49	... 4,12	-5,3

una variación menor a 0 indica que su estabilidad empeora y una variación igual a 0 indica que la estabilidad se mantuvo. El Cuadro 2.1 muestra un ejemplo de la estructura real de un conjunto de datos codificado usando vectores AASA. La primera instancia corresponde a la proteína salvaje, mientras que las otras son mutaciones. A la proteína salvaje se le asocia una variación de la estabilidad igual a 0 por defecto.

2.6.4. Investigaciones anteriores relacionadas con *machine learning* y Vectores AASA

En el contexto más acotado pertinente a este trabajo, se han realizado cinco investigaciones que utilizan datos de proteínas codificados usando vectores AASA para generar modelos capaces de predecir la estabilidad conformacional. En los trabajos [5] y [8], los autores introducen el uso de los vectores de autocorrelación como una nueva manera de codificar información de las proteínas y sus mutaciones. En dichos trabajos, utilizando los vectores AASA, se entrenaron modelos de redes neuronales capaces de predecir la estabilidad de lisozimas mutantes, logrando un R^2 de 0.66 sobre conjuntos de datos de prueba. Así, demostraron que los vectores AASA constituyen una alternativa eficaz de representar una proteína.

En [9], se comparan modelos de predicción de variación de estabilidad utilizando redes neuronales y vectores AASA y AA3DA (*Amino acid 3D autocorrelation* en inglés). Los vectores AA3DA son una variación de los vectores AASA, que incorporan información de la estructura tridimensional de las proteína. Los modelos óptimos entrenados alcanzaron un R^2 de 0.58.

Otro trabajo es [10], en el cual los autores emplearon exitosamente *Support Vector Machines* (SVMs) con *kernel* RBF, tanto para regresión como clasificación, logrando resultados con un conjunto reducido de descriptores. En el caso de regresión, alcanzaron un R^2 de 0.42 y un RMSE de 1.139.

En el trabajo más reciente [16], sus autores proponen una metodología basada en el paradigma de aprendizaje semi-supervisado, motivados por la escasez y dificultad de obtención de datos en este campo de investigación. Utilizando el algoritmo *Tree-based Topology-Oriented Self-organized Map* (TTOSOM) propuesto en [2], los autores lograron generar un modelo capaz de clasificar instancias de proteínas mutantes entre más estables y menos estables respecto a una proteína salvaje.

2.7. Conceptos para el desarrollo de la investigación

A continuación se describen conceptos relacionados directamente con el desarrollo de la investigación, particularmente la metodología de investigación y las tecnologías utilizadas para implementar las rutinas necesarias.

2.7.1. Metodología de Minería de Datos CRISP-DM

Cross Industry Standard Process for Data Mining (CRISP-DM) es una metodología de minería de datos la cual describe un marco de referencia para proyectos relacionados con ciencia de datos en general. Esta metodología define un ciclo de seis etapas, el cual se ilustra en la Figura 2.12. Dichas etapas se describen a continuación en base a como se definen en [24]:

1. **Entendimiento del negocio** (*Business Understanding* en inglés): Se enfoca entender los objetivos de un proyecto y requerimientos del negocio, para luego formularlos como tareas de proyecto de minería de datos. También se crea una planificación con las tareas para cumplir dichos objetivos.

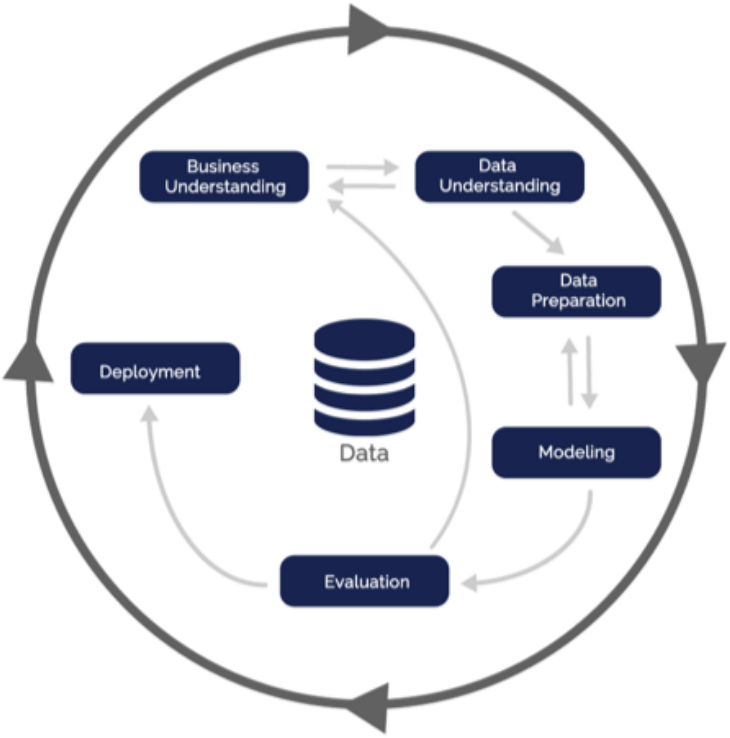


Figura 2.12: Ilustración de metodología CRISP-DM.

2. **Entendimiento de los datos** (*Data Understanding* en inglés): Contempla la recolección de los datos, actividades para determinar la calidad de los datos y actividades de exploración, con el objetivo de obtener información preliminar, encontrar patrones y establecer hipótesis.

3. **Preparación de los datos** (*Data Preparation* en inglés): Incluye actividades que modifican el conjunto de datos crudos previamente a utilizarlo para entrenamiento en la etapa de modelado. Algunas actividades de ejemplo son selección o extracción de atributos, limpieza y transformación de datos, entre otras.

4. **Modelado** (*Modeling* en inglés): En esta etapa se escogen las técnicas de aprendizaje a utilizar, así como también sus hiperparámetros. Se suele iterar frecuentemente entre ésta y la etapa de Preparación de los datos.

5. **Evaluación** (*Evaluation* en inglés): En esta etapa el objetivo es evaluar si es que los modelos entrenados cumplen con los objetivos del proyecto y el negocio.

Figura 2.13: Tecnologías a usar para el desarrollo de la investigación.

Además, se hace una revisión de los pasos hechos anteriormente con el fin de saber si es que hubo algún objetivo importante que no se haya tenido en cuenta lo suficiente.

6. **Despliegue** (Deployment en inglés): El conocimiento adquirido durante el proceso se presenta al usuario para que lo pueda usar. Contempla todas las actividades relacionadas con la implantación del proceso o los modelos en conjunto con el usuario, además de la comunicación de resultados.

2.7.2. Tecnologías

En este apartado se mencionan y definen las tecnologías a utilizar para realizar este trabajo. La Figura 2.13 muestra un resumen gráfico de las tecnologías a utilizar.

R

R es un lenguaje y un ambiente para el cómputo estadístico. Provee múltiples implementaciones de técnicas para el análisis y modelado estadístico. Se utiliza R para aplicar técnicas de *clustering* pues, en comparación a *Scikit-learn*, que es librería de

machine learning en Python, R posee librerías con mayor variedad de funcionalidades para la visualización de *clusters*, ya implementadas. Más información se puede encontrar en el sitio web de R[1].

Python

Python es un lenguaje de programación interpretado y orientado a objetos, que es conocido principalmente por la simplicidad y legibilidad de su sintaxis. Es uno de los lenguajes más utilizados para realizar tareas relacionadas a la ciencia de datos, pues ofrece acceso a variados *frameworks* y librerías para manipulación y análisis de datos.

Por las características mencionadas es que se utiliza el lenguaje Python para implementar gran parte de las rutinas necesarias para llevar a cabo la investigación. La versión de Python que se utiliza en este trabajo es la 3.7.7, una de las más recientes. Más detalles pueden ser encontrados en el sitio web[2] oficial de Python.

Scikit-learn

Scikit-learn es una librería de código abierto que implementa múltiples herramientas, tanto de aprendizaje supervisado como no supervisado, listas para usar. Todas las herramientas poseen interfaces estándar y definidas, lo que otorga flexibilidad, rapidez, y facilidad para implementar *pipelines* de procesamiento de datos y modelado predictivo.

Entre tecnologías similares, existen otras muy populares como Keras, Tensorflow y Pytorch. Sin embargo, éstas se orientan al modelado predictivo usando redes neuronales. Por su parte, Scikit-learn es una librería con un propósito más general, que implementa no solamente herramientas para el modelado predictivo, sino que también para el preprocesamiento de datos, selección de modelos y evaluación. El lenguaje de programación R ofrece una variedad de herramientas muy similar a la de Scikit-learn. Sin embargo, éstas están implementadas en múltiples librerías; cada una con sus propias interfaces y documentaciones.

En este trabajo se utiliza Scikit-learn para la etapa de modelado y evaluación, debido a las ventajas que tiene sobre otras tecnologías. La versión que se utiliza es

[1]Sitio web oficial de R: `https://www.r-project.org`
[2]Sitio web de Python: `https://www.python.org`

0.22.2, una de las más recientes. Más detalles pueden ser encontrados en el sitio web[3] oficial.

Otras librerías adicionales de Python

- **Pandas**: Librería para el manejo y procesamiento de datos. Se utiliza la versión 1.0.3. Más información puede ser encontrada en el sitio oficial[4].
- **Scikit ReBATE**: Librería de Python que implementa algoritmos de selección de atributos basados en *Relief* (RBAs), cuya interfaz es compatible con la librería Scikit-learn. Se utiliza la versión 0.6. Los detalles teorícos de su implementación están definidos en [21] y la documentación de la librería está en su sitio web oficial[5].

Jupyter Notebook

Jupyter Notebook es una aplicación web de código abierto para crear documentos interactivos, los cuales pueden contener texto, visualizaciones y código de diversos lenguajes de programación. Es ampliamente utilizado para compartir resultados experimentales de procesos de ciencia de datos, pues los documentos creados pueden ser compartidos fácilmente a través de plataformas como Google Collab o Github, lo que facilita la reproducibilidad. Con el fin de que que las rutinas implementadas en esta investigación estén disponibles y sean reproducibles, se utiliza Jupyter Notebook para realizar las implementaciones en este trabajo. Más detalles pueden ser encontrados en la página oficial[6].

Git/Github

Git es un sistema de control de versiones de código abierto, el cual permite crear y administrar repositorios de código de proyectos de *software*. Su principal característica es que otorga la capacidad de manejar, de manera simple, un historial de los cambios hechos en el código fuente, permitiendo gestionar dichos cambios libremente. Git es de las alternativas más populares pues posee integración con diversos

[3]Sitio web de Scikit-learn: `https://scikit-learn.org`
[4]Sitio web de Pandas `https://pandas.pydata.org`
[5]Sitio oficial de ReBATE: `https://epistasislab.github.io/scikit-rebate/`
[6]Sitio web de Jupyter Notebook: `https://jupyter.org`

servicios de alojamiento remoto de repositorios. Debido a sus características, se utiliza Git en este trabajo para manejar los cambios hechos en las rutinas implementadas.

Con el objetivo de mantener un respaldo en la nube de las rutinas implementadas y sus cambios, en este trabajo se hace uso de un servicio para alojar y administrar repositorios de manera remota llamado GitHub. Existen alternativas a GitHub, como Bitbucket y GitLab. Sin embargo, éstas son muy similares en términos de funcionalidades, y la investigación no tiene requisitos de privacidad que posicionen una alternativa por sobre la otra. Se elige GitHub, pues tiene integración con una gran variedad de otros servicios y herramientas.

Más detalles sobre éstas tecnologías pueden ser encontrados en los sitios web oficiales de Git[7] y GitHub[8]

[7]Sitio web de Git: `https://git-scm.com`
[8]Sitio web de GitHub: `https://github.com`

3. Metodología

En función de los antecedentes presentados en el capítulo anterior, en este capítulo se plantean las justificaciones detrás del diseño metodológico. A continuación se describen cada una de las etapas de la metodología que se utiliza para desarrollar este trabajo.

3.1. Metodología experimental

La metodología a ocupar en este trabajo es una metodología ad hoc definida en base a la metodología de minería de datos CRISP-DM descrita en la Sección 2.7.1. SEMMA[1], es otra metodología para la minería de datos. Sin embargo, SEMMA no contempla tareas relacionadas a la concepción del proyecto, sino que asume que los objetivos están planteados desde un inicio. CRISP-DM, por su parte, sí contempla dichas tareas. Por esta razón, es que la metodología se define bajo el marco de referencia de CRISP-DM. La Figura 3.1 ilustra el esquema de la metodología definida.

La metodología utilizada en este trabajo presenta una ligera diferencia con respecto a CRISP-DM, que es la etapa del Entendimiento del Negocio. Dicha etapa se redefine como Entendimiento del Dominio, orientándola al planteamiento de los objetivos de una investigación en lugar de a los objetivos de un negocio o empresa. El detalle de cómo se lleva a cabo cada una de las etapas de la metodología en esta investigación se presenta a continuación.

[1]Documentación de SEMMA: `https://documentation.sas.com/?docsetId=emref&docsetTarget=n061bzurmej4j3n1jnj8bbjjm1a2.htm&docsetVersion=14.3&locale=en`

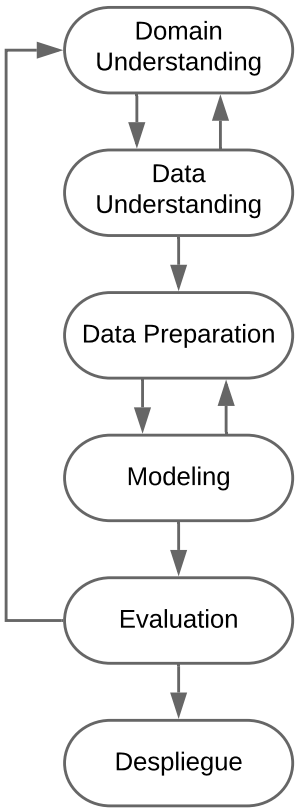


Figura 3.1: Esquema de metodología de investigación.

3.2. Entendimiento del dominio

En esta etapa se realizan actividades orientadas a entender el contexto y definir los objetivos de la investigación. Involucra principalmente la revisión de la literatura y conversaciones con los expertos en bioinformática y *machine learning*. Además, se definen y planifican las tareas a realizar para cumplir con los objetivos.

3.3. Entendimiento de los datos

Los conjuntos de datos son otorgados por el experto en bioinformática, quien es el encargado tanto de su recolección como de su transformación a vectores AASA. Utilizando los conjuntos de datos recolectados, esta etapa contempla principalmente realizar un análisis exploratorio de datos.

Las tareas principales definidas para esta etapa son observar las distribuciones y características tanto de descriptores como de la estabilidad. Detectar patrones, anomalías, y agrupaciones en los datos mediante *clustering*. Todo ésto, aplicando principalmente técnicas de visualización.

3.3.1. Análisis Exploratorio de Datos

El análisis exploratorio de datos (EDA por su sigla en inglés) es un proceso que tiene como objetivo obtener conocimiento sobre las características de los datos con los que se trabajan, con la idea de orientar la elección de las herramientas a utilizar en las etapas posteriores.

Algunas de las tareas definidas para el análisis exploratorio son:

- Observar correlaciones entre descriptores y estabilidad.
- Observar correlaciones entre descriptores.
- Visualizar las distribuciones de la estabilidad.
- Detectar observaciones posiblemente anómalas usando *boxplots*.
- Detectar la presencia de agrupaciones en los datos de manera no supervisada usando los algoritmos de *clustering* K-means y jerárquico.

Cabe mencionar que, durante el análisis exploratorio, también se realizan parte de las tareas de preprocesamiento descritas en la siguiente sección.

3.4. Preparación de los Datos

En esta etapa se consideran todas aquellas tareas relacionadas con la transformación de los datos que necesitan realizarse previamente a la etapa de modelado. En este caso, se hace una normalización de los datos y una reducción de la dimensionalidad disminuyendo la cantidad de descriptores.

3.4.1. Normalización de datos

Cuando los atributos de un conjunto de datos están medidos en escalas muy diferentes unas de otras, algunos algoritmos de entrenamiento pueden tener problemas

para estimar adecuadamente los parámetros de los modelos. Uno de los algoritmos a utilizar en este trabajo, PLS, es particularmente susceptible a la diferencia de escalas en los predictores, por lo que es necesario *normalizar* los datos antes de poder trabajar con ellos.

Los atributos se normalizan utilizando la función Z-Score, definida en la Ecuación 2.13. Ésta función convierte la distribución de un atributo en una cuyas media y desviación estándar sean 0 y 1 respectivamente. La Ecuación 3.1 muestra un ejemplo de como se normalizan los valores para el descriptor $AASA5K0$.

$$AASA5K0 = \frac{AASA5K0_i - \mu_{AASA5K0}}{\sigma_{AASA5K0}} \tag{3.1}$$

3.4.2. Reducción de la dimensionalidad

La reducción de la dimensionalidad es el proceso por el cual se reduce la cantidad de atributos de un conjunto de datos. Se suele hacer por diversos motivos, como por ejemplo disminuir el tiempo de procesamiento en el modelado y descartar la presencia de ruido o redundancia en los datos, entre otros. En este caso es necesario reducir la cantidad de descriptores principalmente para mejorar el desempeño de generalización. La cantidad de instancias de los conjuntos de datos con los que se trabajan suele ser menor que la cantidad de atributos. Ésto puede causar que los modelos tengan que tengan un mal desempeño de generalización.

Comúnmente, los métodos de reducción de dimensionalidad se pueden dividir en aquellos que seleccionan un subconjunto de los atributos (*feature selection* en inglés) y aquellos que extraen un nuevo conjunto de atributos calculados a partir de los originales (*feature extraction* en inglés). En el contexto de datos de proteínas, los datos son muy difíciles de conseguir, por lo que es ideal generar modelos de predicción que utilicen la menor cantidad de descriptores posible, sin comprometer tanto el desempeño. Considerando ésto, se descarta el uso de métodos de extracción de atributos como PCA, ya que estas técnicas utilizan *todos* los atributos.

Se decide entonces usar algún método de selección de atributos. De acuerdo a [11, págs. 173-176], los métodos de selección de atributos se pueden clasificar, a su vez, en tres tipos: métodos de *filtro* (*Filter Methods* en inglés), métodos de envoltorio (*Wrapper Methods*) y métodos incrustados (*Embedded Methods*). Los métodos de filtro seleccionan atributos de manera independiente a las técnicas de modelado a

utilizar posteriormente, mientras que métodos de los otros dos tipos seleccionan un conjunto de atributos basándose en un determinado algoritmo de aprendizaje. Considerando que se desea hacer una comparación entre los algoritmos, es necesario que éstos se apliquen sobre conjuntos de datos preprocesados de la misma manera con el fin de que los resultados sean comparables. Por esta razón, se elige utilizar métodos de filtro, pues de esta manera la selección de atributos se hace de manera independiente a los algoritmos de modelado a utilizar.

Algunos de los métodos de filtro más comunes contemplan el uso de estadísticos como la correlación o la varianza. Sin embargo, estos métodos evalúan la relevancia de un atributo de manera individual. Esto constituye un problema considerando que las proteínas son estructuras químicas complejas, para las cuales podrían existir interacciones importantes entre sus propiedades y, en consecuencia, entre los descriptores. Dichas interacciones podrían no ser detectadas analizando cada atributo de manera individual. De acuerdo a lo señalado en [21] y [22], el algoritmo *ReliefF* podría ser adecuado para este caso, pues es un algoritmo de filtro que logra tener en cuenta interacciones entre atributos al momento calcular la relevancia algún atributo en particular. El algoritmo es descrito con más detalle en la Sección 2.5.3. Considerando lo anterior, se decide utilizar *ReliefF*, particularmente una variación de éste llamada *SURF*, la cual se encuentra implementada en la librería Scikit-ReBATE en Python. Se utiliza dicha implementación porque emplea una heurística que elimina la necesidad de pasarle la cantidad de vecinos k como parámetro. Los detalles del algoritmo y su implementación pueden ser encontrados en [21].

Como se menciona en [22], los RBAs no logran descartar atributos que son redundantes, por lo que descartar descriptores que son redundantes previamente puede beneficiar la selección de atributos. En este caso, se considera razonable establecer que descriptores que están correlacionados linealmente con algún otro descriptor son redundantes. Por ello, antes de hacer la selección de atributos, se descartan aquellos descriptores que tienen un coeficiente de correlación de Pearson absoluto superior a 0.99 con algún otro descriptor.

Finalmente, se escoge un subconjunto descriptores de los descriptores más relevantes de acuerdo a un *ranking* generado por el algoritmo *SURF*.

3.5. Modelado

Esta etapa contempla principalmente la optimización de hiperparámetros y el entrenamiento y evaluación de modelos.

3.5.1. Algoritmos de entrenamiento

Los conjuntos de datos con los que se trabajan tienen una alta cantidad de predictores en comparación a la cantidad de instancias. Por esta razón, es altamente probable que los modelos tengan poca capacidad de generalización producto de la presencia de *overfitting*.

Los métodos de regresión a utilizar en este trabajo son OLS, *Ridge Regression*, *Lasso Regression*, PLS y SVR (con *kernels* polinomial y RBF). Una razón del por qué se utilizan, es que son métodos clásicos, bien documentados, y que están implementados en una buena parte de librerías utilizadas para *machine learning*. La segunda razón es que éstos algoritmos incorporan mecanismos para prevenir presencia de *overfitting*; en el caso de SVR, Ridge y Lasso, es la regularización, mientras que en el caso de PLS es la reducción de la dimensionalidad a través de la extracción de atributos.

3.5.2. Métricas de evaluación

Para evaluar el desempeño de un modelo entrenado, es necesario utilizar una métrica de evaluación. Con el fin de poder comparar los resultados de este trabajo con los de investigaciones anteriores, se utilizan dos métricas, *RMSE* y R^2. Otra razón por la que se escogen éstas es que ambas son métricas comúnmente usadas para evaluar el desempeño de modelos de regresión.

La Figura 3.2 ilustra cómo se evalúa el desempeño de un modelo sobre un determinado conjunto de datos. El conjunto se separa en atributos, en este caso los descriptores (cuadros de color azul), y los correspondientes valores de estabilidad reales (cuadros de color naranja). Las instancias son pasadas al modelo y éste, usando los valores de los descriptores, retorna la predicción de estabilidad (cuadros de color rojo) para cada una de las observaciones. Finalmente, se calcula el desempeño del modelo usando una métrica, que es una función de los valores de estabilidad reales y de los valores de estabilidad predichos por el modelo.

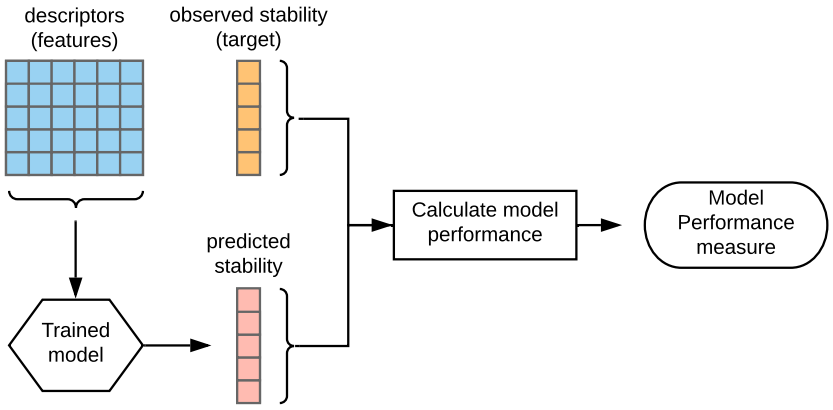


Figura 3.2: Esquema de evaluación de desempeño de modelos entrenados.

Para un conjunto de datos de prueba de n instancias, las métricas de evaluación de desempeño, de acuerdo a su definición en las Ecuaciones 2.11 y 2.12, se aplican como

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(estabilidad\ real_i - estabilidad\ predicha_i)^2}{\sum_{i=1}^{n}(estabilidad\ real_i - estabilidad\ promedio)^2} \tag{3.2}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(estabilidad\ real_i - estabilidad\ predicha_i)^2}{n}} \tag{3.3}$$

donde valor de la estabilidad real corresponde a y_i, la estabilidad predicha por el modelo corresponde a $\hat{y}_i$ y el promedio de la estabilidad observada corresponde a $\overline{y}$.

3.5.3. Optimización de hiperparámetros

Cada uno de los algoritmos tiene sus propios parámetros de funcionamiento, denominados hiperparámetros. Dependiendo de la elección de éstos, el desempeño de los modelos entrenados por un algoritmo puede variar bastante. Es por ésto que, por cada algoritmo, es necesario determinar cuál es la combinación de hiperparámetros que mejores modelos genera.

Para ésto existen varias maneras. Entre ellas, dos alternativas destacan por su amplio uso y simplicidad. Una de ellas es *Grid Search*, que corresponde una búsqueda exhaustiva dentro de espacio predefinido de posibles valores para los hiperparámetros.

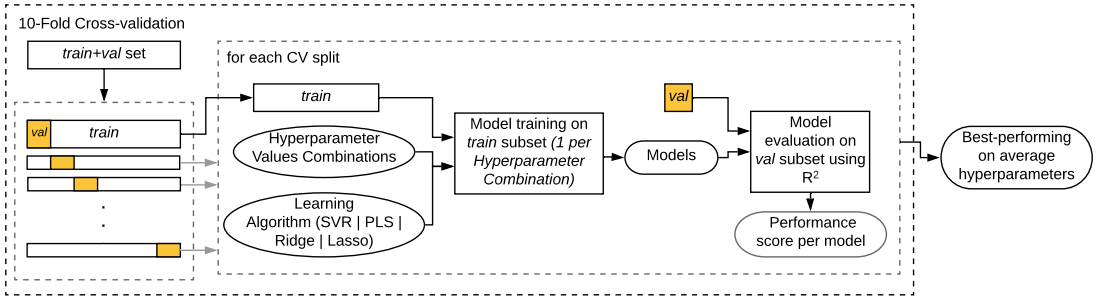


Figura 3.3: Esquema del ciclo interno del proceso de *Nested Cross-validation*, correspondiente a la optimización de hiperparámetros.

La otra es *Random Search*, donde se hace un muestreo aleatorio de una distribución predefinida para cada hiperparámetro. Cada alternativa tiene sus ventajas. Sin embargo, se decide utilizar por *Grid Search* pues, para los algoritmos utilizados en este trabajo, se tiene un conocimiento provisto por la literatura sobre qué combinaciones de valores comúnmente producen buenos resultados.

La mejor combinación de hiperparámetros es aquella que genera los modelos con mejor desempeño de generalización. Mediante un proceso de *10-Fold Cross-validation*, se selecciona la combinación que genere los modelos con mejor desempeño promedio en las 10 iteraciones. El desempeño de los modelos entrenados, se mide usando la métrica *RMSE*. La Figura 3.3 ilustra cómo se hace la optimización de hiperparámetros.

3.5.4. Entrenamiento y evaluación

Existe una gran variedad de procedimientos para la etapa de evaluación y selección de modelos. La idea importante a tener en cuenta cuando se estima el desempeño un modelo, es que los mismos datos de entrenamiento no deberían ser utilizados para evaluarlo pues, como se menciona en [1, pág. 475], el desempeño sobre los datos de entrenamiento tiende a ser mayor que el desempeño sobre un conjunto datos de prueba que el modelo no ha visto. En otras palabras, el desempeño de entrenamiento es una estimación sesgada del desempeño de generalización.

La selección del procedimiento depende, en primera instancia, de la cantidad de datos que se tienen a disposición. Si se tuvieran conjuntos de datos con una muchas instancias, se podría simplemente apartar un conjunto de datos de prueba (método

llamado *hold-out* en inglés) y entrenar con el resto. Sin embargo, los conjuntos de datos de proteínas disponibles son muy pequeños. Es por esto que se tiene que recurrir a procedimientos basados en remuestreo, que son adecuados especialmente para casos como el de este trabajo, donde la cantidad de observaciones en los conjuntos de datos es baja. Dentro de los métodos de remuestreo utilizados para la estimación del desempeño de generalización, destacan *Boostrapping* y *k-Fold Cross-validation*. Sin embargo, teniendo en cuenta que *k-Fold Cross-Validation* es considerado un estándar *de facto* para la evaluación de modelos de predicción, y que tiene una amplia integración a librerías usadas para ciencia de datos como Scikit-learn, se decide descartar *Boostrapping*.

Existen variantes del proceso de *Cross-validation*. En este trabajo se utiliza *Nested Cross-validation* por dos razones. Una es que, de acuerdo a lo señalado en [23] y [6], es una alternativa robusta para prevenir el sesgo positivo en la estimación del desempeño de generalización. La otra, es que contempla la tarea de optimización de hiperparámetros de los algoritmos, tarea que es necesario realizar. Más detalle sobre el razonamiento detrás de *Nested Cross-validation* se describe en la Sección 2.4.3.

Es necesario seleccionar los valores de k para los ciclos interno y externo de *Nested Cross-validation*. De acuerdo a [1, pág. 487] y [18], lo recomendado para conjuntos de datos extremadamente pequeños es un $k = n$, conocido también como *Leave-one-out Cross-validation* (LOOCV). Sin embargo, ésto elevaría demasiado la complejidad computacional. Los valores $k = 5$ y $k = 10$ son generalmente usados. Se decide usar $k = 10$ en ambos ciclos pues, en vista de que los conjuntos de datos se dividen dos veces, una en el ciclo externo y otra en el ciclo interno, y que los conjuntos de datos son pequeños, de ésta manera se logra que los conjuntos de entrenamiento no sean tengan un tamaño tan reducido. Ésto es, sin embargo, a cambio de tener conjuntos de prueba más pequeños, lo que agrega mayor variabilidad a las estimaciones de desempeño.

Las Figuras 3.3 y 3.4 ilustran con más detalle cómo se llevan a cabo los ciclos interno y externo de *Nested Cross-validation*. El ciclo interno se encarga de la optimización de hiperparámetros usando *Grid Search* y el ciclo externo se encarga de estimar el desempeño de generalización del algoritmo de entrenamiento. Ésto se realiza para cada uno de los algoritmos, OLS, Ridge, Lasso, PLS y SVR.

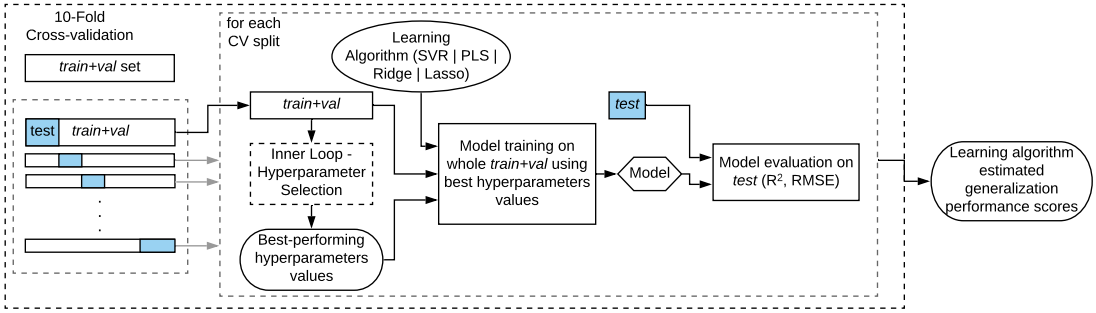


Figura 3.4: Esquema del ciclo externo del proceso de *Nested Cross-validation*, correspondiente a la estimación del desempeño de generalización de un algoritmo de entrenamiento.

3.6. Evaluación

Esta etapa se enfoca en verificar si es que los resultados obtenidos en la etapa de modelado permiten sacar algún tipo de conclusión en relación a las preguntas de investigación planteadas, o si es que se puede extraer alguna retroalimentación útil para corregir o refinar el proceso realizado en las etapas anteriores.

3.6.1. Criterios de evaluación de modelado predictivo

Los criterios de evaluación para el modelado predictivo se definen directamente en función de las preguntas de investigación planteadas al inicio del documento en la Sección 1.5.

En relación a la pregunta de investigación RQ1, para poder concluir si es que un algoritmo de modelado lineal tiene un buen desempeño en la predicción de la estabilidad se debe cumplir, consistentemente a través de los distintos conjuntos de datos abordados, que su desempeño de generalización estimado a partir del método de validación indique que los modelos entrenados incurren en errores de predicción bajos para conjuntos de prueba. Ésto, en términos de las métricas de evaluación, corresponde a un $R^2 > 0{,}8$ y un $RMSE < 0{,}5$.

En relación a la pregunta de investigación RQ2, para poder concluir que un algoritmo de modelado lineal tiene un desempeño similar con el de SVR, se debe cumplir, consistentemente a través de los distintos conjuntos de datos abordados, que el desempeño de generalización estimado para el algoritmo de modelado lineal

sea cercano al desempeño estimado para SVR con *kernels* no lineales. Ésto es, en términos de las métricas de evaluación, $R^2_{\text{Modelo lineal}} \approx R^2_{SVR}$ y $RMSE_{\text{Modelo lineal}} \approx RMSE_{SVR}$.

3.6.2. Criterio de evaluación de reproducibilidad

Para asegurar la reproducibilidad de la metodología, se procura establecer semillas para toda tarea que involucre un proceso aleatorio. Para validar la reproducibilidad, se aplica la misma implementación de la metodología sobre más de un conjunto de datos de proteínas. Se considera que la metodología es reproducible si es que no existen inconvenientes al aplicarla sobre los distintos conjuntos de datos abordados.

3.6.3. Interpretación y análisis de resultados

Para las tareas pertenecientes a las etapas de Modelado y Evaluación, se realiza una interpretación y un análisis de los resultados, con el fin de determinar si es que los criterios definidos anteriormente se cumplen. En caso de que no se cumplan, se busca encontrar información relevante en ellos que permita corregir o refinar el proceso metodológico.

3.7. Despliegue

Esta etapa contempla la estructuración del conocimiento obtenido a partir del desarrollo de la investigación, y su comunicación al Experto en Bioinformática. Para ésto, se elabora una presentación que resume el desarrollo del proceso, así como también sus resultados.

4. Desarrollo y análisis de resultados

En este capítulo se detalla cómo se realiza el desarrollo de la investigación en función de los resultados que se van obteniendo. Se parte explicando el desarrollo del análisis exploratorio de datos. Luego, se explica el desarrollo la Preparación de los Datos. Luego, se documentan y analizan los resultados de la etapa de Modelado y Evaluación. Finalmente, el capítulo cierra con una discusión de los resultados observados a lo largo del desarrollo.

4.1. Entendimiento de los datos

4.1.1. Conjuntos de datos

Se trabaja con cuatro conjuntos de datos de mutaciones correspondientes a cuatro de proteínas distintas, 1STN, P4LYZ y 1BPI, HLYZ. El Cuadro 4.1 muestra las dimensiones de los conjuntos de datos. Todos poseen los mismos atributos. 480 atributos corresponden a los descriptores AASA y éstos toman valores numéricos y continuos en distintas escalas. Otro atributo es la variación de estabilidad, la cual está representada a través números continuos; una variación negativa de la estabilidad indica que una proteína mutante es menos estable con respecto a la original, mientras que una variación positiva indica que la mutante es más estable. Una mutante con una estabilidad igual a cero indica que ésta es igual de estable que la original. La proteína salvaje tiene, por defecto, una estabilidad igual a cero. El atributo restante corresponde a un identificador descriptivo para cada instancia del conjunto.

Cuadro 4.1: Dimensiones de los conjuntos de datos abordados.

Datasets		
Name	**Attributes**	**Instances**
1STN	482	42 (1 Wild-type + 41 Mutants)
4LYZ	482	51 (1 Wild-type + 50 Mutants)
1BPI	482	53 (1 Wild-type + 52 Mutants)
HLYZ	482	123 (1 Wild-type + 122 Mutants)

Cuadro 4.2: Número de descriptores intercorrelacionados para umbrales de r de Pearson entre 0.90 y 0.99, para cada conjunto de datos.

Pearsons's r Threshold	**1STN**	**4LYZ**	**1BPI**	**HLYZ**
0.99	78	89	34	77
0.98	109	129	83	128
0.96	211	205	157	190
0.95	245	236	183	221
0.94	266	255	211	249
0.93	285	277	237	278
0.92	305	298	254	303
0.91	328	315	273	318
0.90	339	335	296	337

4.1.2. Correlación entre descriptores

El Cuadro 4.2 muestra la cantidad de descriptores que tienen una correlación de Pearson mayor a determinado umbral con al menos algún otro descriptor. Se puede notar que los descriptores están altamente correlacionados entre ellos. Para el conjunto HLYZ, aproximadamente el 16 % (78 de 480) de los descriptores tiene una correlación mayor a 0.99 y cerca del 70 % (337 de 480) tienen una correlación mayor a 0.90 con algún otro descriptor.

4.1.3. Valores distintos de la estabilidad

La estabilidad está definida como una variable numérica continua. Sin embargo, podrían haber muchos valores repetidos, lo que indicaría la presencia de algún patrón o, por lo menos, una forma de discretizar los valores. Ésto sería en el caso en que la cantidad de valores distintos sea baja en comparación a la cantidad de instancias. El

Cuadro 4.3: Porcentaje de valor distintos para la estabilidad en cada uno de los conjuntos de datos.

Dataset	Stability distinct values (%)
1STN	78.6 %
4LYZ	80.4 %
1BPI	64.1 %
HLYZ	70.7 %

Cuadro 4.3 muestra el porcentaje de valores distintos con respecto a la cantidad de instancias por cada conjunto de datos. Los porcentajes son relativamente altos, por lo que es posible continuar tratándola como una variable continua.

4.1.4. Distribución de la estabilidad

La Figura 4.1 muestra *boxplots* con las distribuciones de estabilidad para cada conjunto de datos. En general, se puede notar que hay un gran cantidad de observaciones cuya estabilidad es cercana a 0. El conjunto de datos HLYZ alcanza valores más extremos de estabilidad que los demás conjuntos. Para los conjuntos 4LYZ, 1BPI y HLYZ se puede notar un fuerte sesgo hacia el extremo negativo, producto de la presencia de observaciones con una variación de la estabilidad muy negativa. La distribución de 1STN no presenta anormalidades evidentes. Gráficos individuales para cada conjunto de datos se presentan en el Anexo A.1.

Cuadro 4.4: Cantidad de observaciones según su variación de estabilidad, por cada conjunto de datos.

Dataset	Negative variation	Positive variation	No variation
1STN	36	3	3
4LYZ	29	19	3
1BPI	46	2	5
HLYZ	89	31	3

El Cuadro 4.4 muestra la cantidad de instancias según su variación de estabilidad, para cada conjunto de datos. En los conjuntos 1STN y 1BPI hay muy pocas instancias con una variación positiva. Ésto representa un problema, pues los modelos entrenados probablemente tengan un mal desempeño prediciendo estabilidad de mutantes con

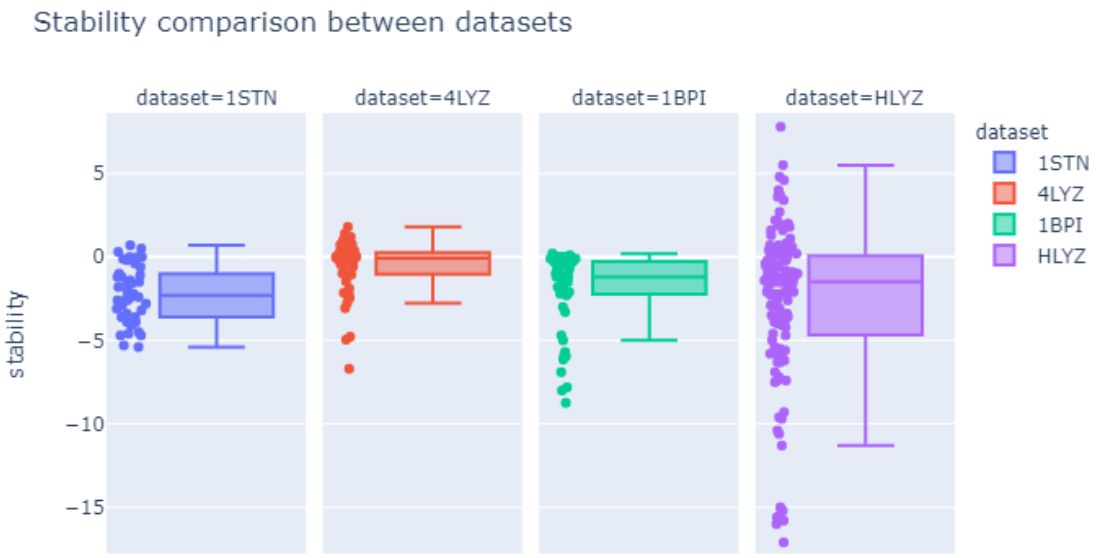


Figura 4.1: *Boxplots* de las distribuciones de estabilidad para cada conjunto.

estabilidad positiva. El conjunto 4LYZ, por su parte, no presenta un desbalance de casos tan pronunciado.

4.1.5. Detección de observaciones atípicas

A simple vista, a partir de los *boxplots* mostrados en la Figura 4.1, se pueden establecer que existen observaciones variaciones de estabilidad extremas, particularmente negativas. Para el conjunto de datos 4LYZ, hay 3 mutantes muy inestables, con valores entre -6.7 y -4.78. Para el conjunto de datos 1BPI, hay 8 observaciones muy inestables, con valores entre -8.74 y -5.7. En el caso del conjunto de datos HLYZ, hay 7 observaciones con una variación de la estabilidad muy negativa entre -17.1 y -15, mientras que hay 1 observación con una variación de la estabilidad muy positiva, con un valor de 7.8.

Éstas observaciones con valores extremos de estabilidad podrían ser, posiblemente, observaciones atípicas. Sin embargo, de acuerdo a lo revisado en trabajos anteriores, dentro del dominio de aplicación tales valores para la estabilidad no son tan anormales. Otro aspecto a considerar es que las muestras tienen pocas instancias,

Cuadro 4.5: Los 10 descriptores más correlacionados con la estabilidad en el conjunto HLYZ.

Descriptor	Pearsons's r
$AASA3DCph$	0.46
$AASA5Rf$	0.44
$AASA5DCph$	0.43
$AASA5Ht$	0.43
$AASA8Ht$	0.43
$AASA3Ht$	0.43
$AASA3Ra$	0.43
$AASA3DASA$	0.42
$AASA2pK$	0.42
$AASA5pK$	0.41

por lo que no es adecuado descartar las observaciones con valores extremos pues se estarían perdiendo muchos datos.

4.1.6. Correlación lineal entre descriptores y estabilidad

Como se muestra en el Cuadro 4.5, para el conjunto HLYZ las correlaciones individuales entre descriptores y estabilidad son muy bajas, de la misma manera que para el conjunto 4LYZ, cuya correlación más alta es de 0,54. Sin embargo, para el conjunto 1STN y 1BPI las correlaciones más altas son de 0,72 y 0,73, respectivamente; el Cuadro 4.6 muestra los descriptores más correlacionados con la estabilidad del conjunto 1STN. Los demás resultados están en el Anexo A.2.

4.1.7. *Clustering*

Se aplican técnicas de *clustering* con el fin de visualizar los datos de manera general en búsqueda de algún patrón en las observaciones, particularmente sobre aquellas con estabilidad positiva, que son los casos de interés.

Se emplean dos algoritmos, *K-means* y *clustering* jerárquico con enlace *completo*. Se prueban configuraciones de 3 *clusters*, para las cuales lo esperado es que las mutantes con una variación de estabilidad negativa pertenezcan a un grupo distinto del de aquellas con estabilidad positiva, mientras que aquellas mutantes con variación igual a 0 se espera que estén junto a la proteína salvaje. Las visualizaciones del

Cuadro 4.6: Los 10 descriptores más correlacionados con la estabilidad en el conjunto 1STN.

Descriptor	**Pearsons's r**
$AASA2ASAD$	0.72
$AASA10V0$	0.72
$AASA2V0$	0.71
$AASA10ASAD$	0.71
$AASA6ASAD$	0.70
$AASA10DASA$	0.70
$AASA8V0$	0.70
$AASA10Ca$	0.69
$AASA10Mw$	0.69
$AASA2DASA$	0.69

clustering retornado por *K-means* se generan usando PCA para graficar las observaciones en dos dimensiones. Sin embargo, la varianza explicada por las dos primeras componentes es baja, por lo que la distancia en los gráficos no refleja la similitud. Por ésto, es necesario fijarse en las en las agrupaciones

La Figura 4.2 muestra la clusterización retornada por *K-means* para el conjunto de datos 4LYZ. Se puede observar que una de las instancias, D101F, está considerada en un grupo aparte. Sin embargo, en los otros dos grupos, las proteínas de estabilidad positiva están mezcladas con las de estabilidad negativa. En la clusterización arrojada por *clustering* jerárquico, mostrada en la Figura 4.3, se puede observar algo similar. La instancia G102R, cuya estabilidad es de 0.38, es un *cluster* por sí misma, mientras que en los otros dos conjuntos nuevamente están mezcladas observaciones con estabilidad negativa y positiva. Este comportamiento se observa también en el resto todos los conjuntos de datos, cuyas *clusterizaciones* se adjuntan en el Anexo A.3.

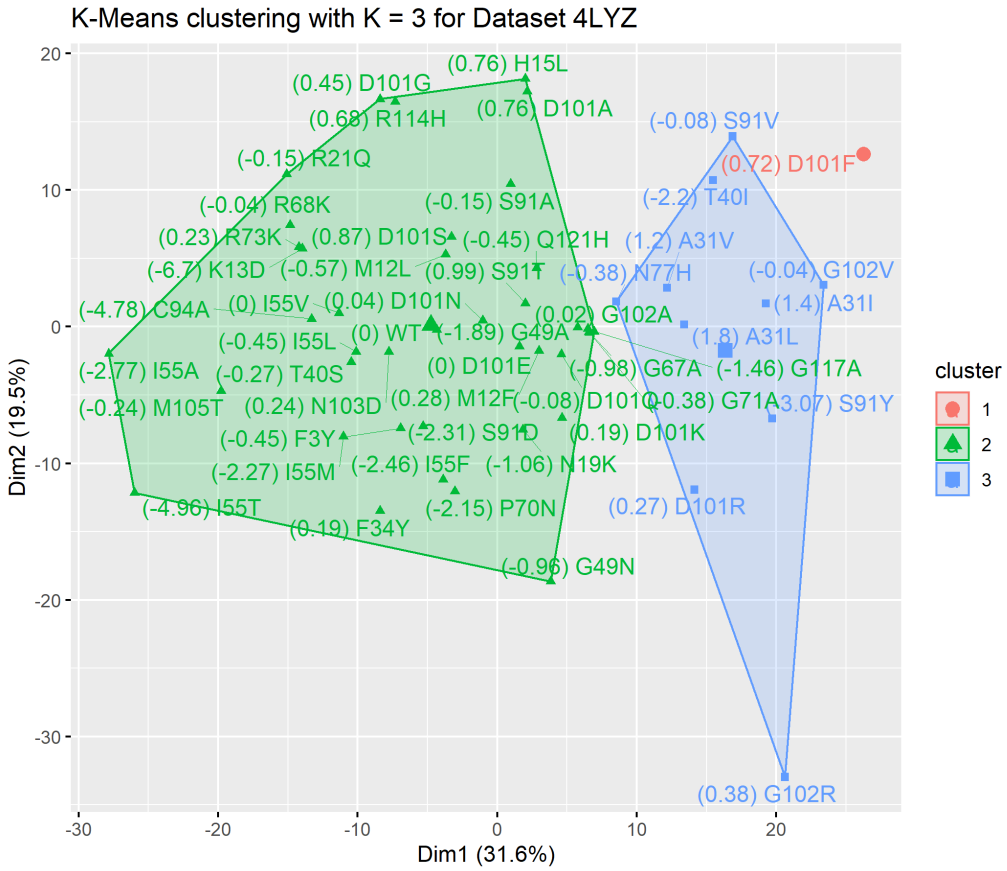


Figura 4.2: Agrupaciones arrojadas por *K-means* con 3 *clusters* para el conjunto de datos 4LYZ

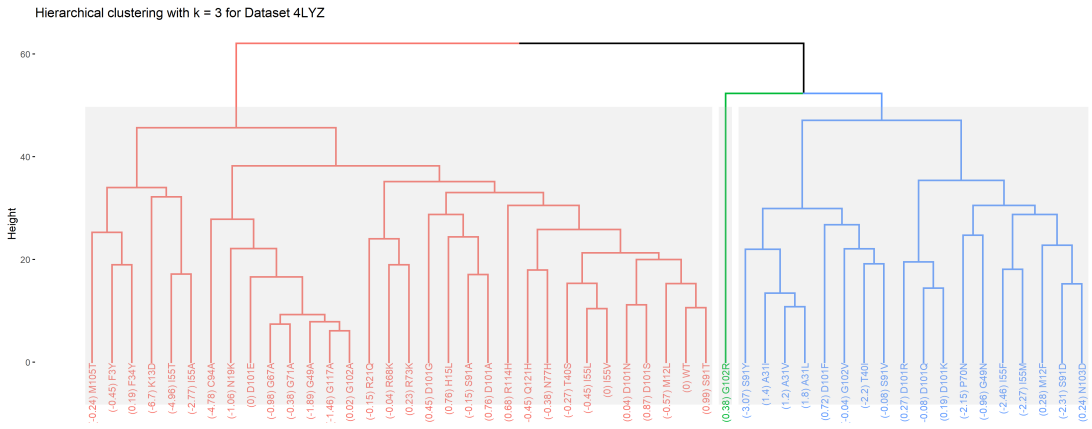


Figura 4.3: Agrupaciones arrojadas por *clustering* jerárquico con 3 *clusters* para el conjunto de datos 4LYZ.

La Figura 4.4 representa la matriz de distancia euclidiana como un mapa de calor. Se puede observar que hay casos en que una proteína de estabilidad positiva es muy similar a una con estabilidad negativa, por lo que la distancia euclidiana no es del todo adecuada. Lo mismo se puede observar en el resto de los conjuntos de datos. Se prueba con otras métricas de distancia, pero ninguna de ellas logra alejar completamente a las mutaciones con estabilidad positiva de las que tienen estabilidad negativa. Un ejemplo es la distancia *Manhattan*, cuya matriz de distancia se muestra en la Figura 4.5, para la cual se repite lo observado con la distancia euclidiana. Ésto es observado también en el resto de los conjuntos de datos cuyas matrices de distancia se adjuntan en el Anexo A.3.

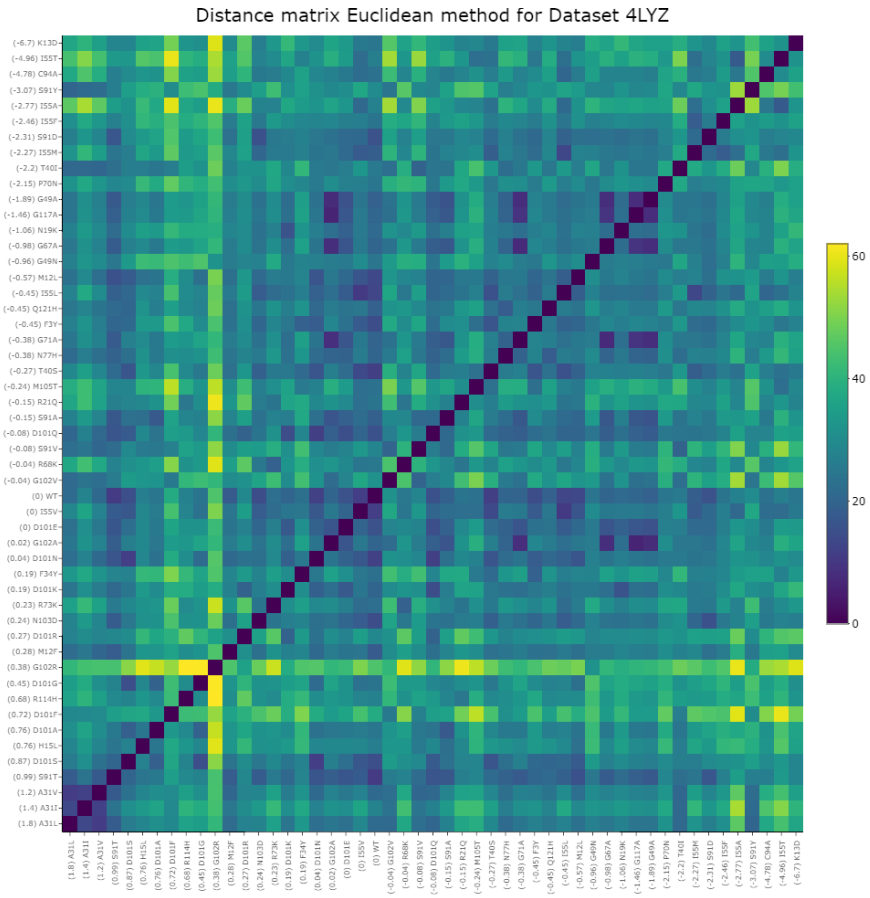


Figura 4.4: Mapa de calor de la matriz de distancia euclidiana calculada para el conjunto de datos 4LYZ.

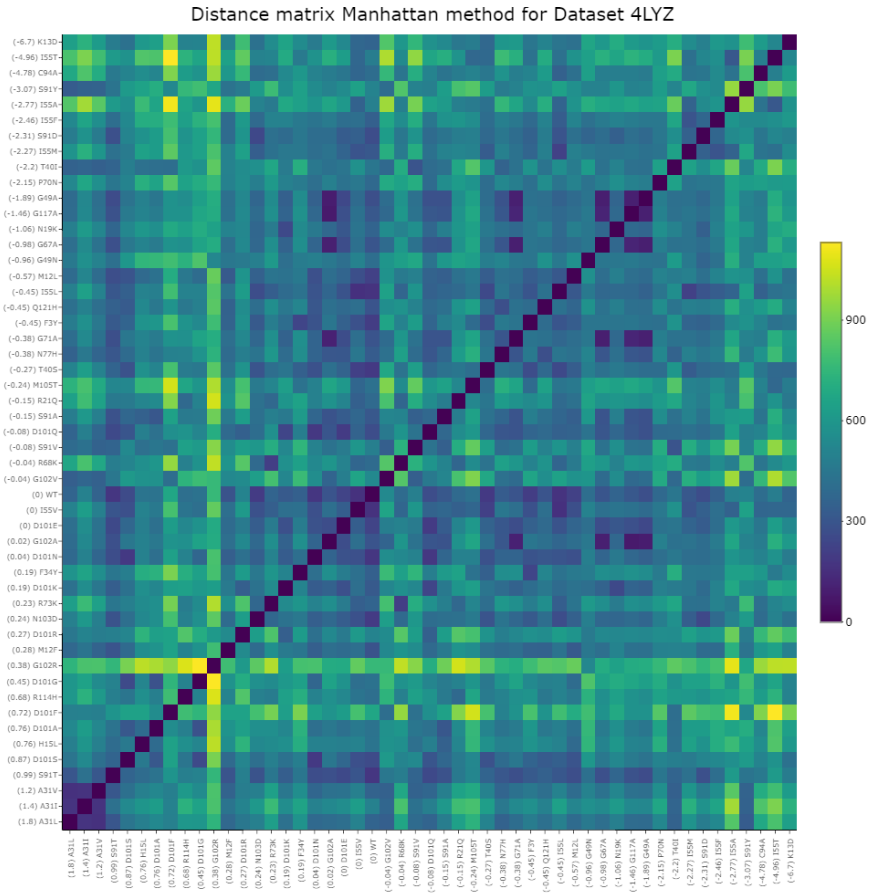


Figura 4.5: Mapa de calor de la matriz de distancia *Manhattan* calculada para el conjunto de datos 4LYZ.

En general, ninguna de las distancias probadas logra distinguir completamente las observaciones más estables de las menos estables. Una razón puede ser la alta dimensionalidad de los datos.

4.2. Preparación de los datos

4.2.1. Limpieza e integridad de los datos

Esta tarea no presenta mayores desafíos. Ninguno de los conjuntos de datos tiene datos faltantes. En el caso de los conjuntos 1STN, 4LYZ y 1BPI, los nombres de los descriptores no siguen el mismo formato que los del conjunto HLYZ; los descriptores son los mismos, por lo que se hace una estandarización de los nombres.

Cuadro 4.7: Los 40 descriptores más relevantes seleccionados por SURF para el conjunto HLYZ.

Ranking	Descriptor	Ranking	Descriptor
1	$AASA1pK$	21	$AASA6DASA$
2	$AASA2Ht$	22	$AASA6Hnc$
3	$AASA6f$	23	$AASA6V$
4	$AASA6DGc$	24	$AASA6GhN$
5	$AASA2TDSc$	25	$AASA2s$
6	$AASA2TDSh$	26	$AASA6TDS$
7	$AASA8Ht$	27	$AASA6Ht$
8	$AASA2V$	28	$AASA6s$
9	$AASA2DHc$	29	$AASA6Mw$
10	$AASA2DASA$	30	$AASA7DGc$
11	$AASA2DGc$	31	$AASA6ASAD$
12	$AASA2DCph$	32	$AASA5Ht$
13	$AASA8TDSc$	33	$AASA1f$
14	$AASA10f$	34	$AASA6Ca$
15	$AASA3TDSh$	35	$AASA8DHc$
16	$AASA6V0$	36	$AASA6DH$
17	$AASA2m$	37	$AASA8DCph$
18	$AASA6TDSh$	38	$AASA5TDSc$
19	$AASA2V0$	39	$AASA6m$
20	$AASA6DGh$	40	$AASA7an$

4.2.2. Descarte de descriptores intercorrelacionados

Previamente a aplicar el algoritmo de selección de atributos, se descartan aquellos descriptores que tienen una correlación de Pearson absoluta superior a 0.99 con algún otro. De acuerdo a lo mostrado en el Cuadro 4.2, la cantidad de descriptores descartados es 78, 89, 34 y 77 para los conjuntos 1STN, 4LYZ, 1BPI y HLYZ, respectivamente.

4.2.3. Selección de atributos

Para la selección de atributos, se seleccionan los 40 descriptores más relevantes de acuerdo a un *ranking* generado por el algoritmo *SURF*. Los Cuadro 4.7 y 4.8 muestran los *ranking* con los 40 descriptores más relevantes seleccionados para los conjuntos de datos HLYZ y 1BPI, respectivamente. El número 1 en el *ranking* es el

Cuadro 4.8: Los 40 descriptores más relevantes seleccionados por SURF para el conjunto 1BPI.

Ranking	**Descriptor**	**Ranking**	**Descriptor**
1	$AASA10m$	21	$AASA9DGh$
2	$AASA5DH$	22	$AASA5TDSc$
3	$AASA5m$	23	$AASA9am$
4	$AASA1m$	24	$AASA8am$
5	$AASA8DH$	25	$AASA5am$
6	$AASA9m$	26	$AASA10am$
7	$AASA8m$	27	$AASA9TDSc$
8	$AASA3m$	28	$AASA4am$
9	$AASA8DG$	29	$AASA9pK$
10	$AASA3DH$	30	$AASA10DHc$
11	$AASA8DHc$	31	$AASA3am$
12	$AASA10DH$	32	$AASA1Hgm$
13	$AASA4m$	33	$AASA5DHc$
14	$AASA7DH$	34	$AASA1Nm$
15	$AASA7DG$	35	$AASA9DGc$
16	$AASA5DG$	36	$AASA5Nm$
17	$AASA3DG$	37	$AASA2am$
18	$AASA7m$	38	$AASA4DG$
19	$AASA8TDSc$	39	$AASA8Nm$
20	$AASA7TDSc$	40	$AASA9DHc$

descriptor más relevante, y el número 40 el menos relevante.

Se puede notar observar que los descriptores considerados más relevantes son distintos entre conjuntos de datos, con casi ninguna coincidencia entre ellos. Ésto ocurre para los cuatro conjuntos de datos de manera similar.

Otra observación que se puede hacer es que hay casos de descriptores asociados a una misma propiedad que son considerados más relevantes. Ésto se observa de manera más evidente en el conjunto de datos 1BPI, en cuyo *ranking* están presentes varios descriptores asociados a las propiedades propiedades m, am y DG, entre otras. Sucede algo similar en los demás conjuntos de datos pero no de manera tan prominente, con descriptores asociados a otras propiedades. Los listados de los descriptores seleccionados para los otros conjuntos se encuentran en el Anexo B.1.

Cuadro 4.9: Valores posibles para los hiperparámetros de cada algoritmo probados en la optimización de hiperparámetros con *Grid Search*.

Method	Hyperparameters	Values	Total
Ridge	λ	$10^{-6}, 10^{-5}, 10^{-4}, ..., 10^{6}$	13
Lasso	λ	$10^{-6}, 10^{-5}, 10^{-4}, ..., 10^{6}$	13
PLS	*n. components*	$2, 3, 4, ..., 25$	24
SVR	C	$10^{-6}, 10^{-5}, 10^{-4}, ..., 10^{6}$	13
	ϵ (*epsilon*)	1,0, 1,1, 1,2, ..., 2,5	16
	kernel	$rbf, polynomial$	2
	γ (*gamma*)	$10^{-6}, 10^{-5}, 10^{-4}, ..., 10^{6}$	13
	degree (only polynomial)	$2, 3$	2

4.3. Modelado y evaluación de modelos

La etapa de modelado se detalla la realización 3 experimentos de modelado predictivo. Por cada uno, se presentan sus características y los resultados con su respectivo análisis. Es necesario mencionar que la metodología de reducción de la dimensionalidad descrita en el la Sección 3.4.2 se aplica solamente en el Experimento 3. Los experimentos 1 y 2 son realizados en las primeras iteraciones de la metodología del trabajo.

4.3.1. Grilla de valores para hiperparámetros

Por cada uno de los algoritmos de entrenamiento se define un espacio, o *grilla*, de posibles valores que pueden tomar los hiperparámetros de cada algoritmo de aprendizaje. Los valores se muestran en el Cuadro 4.9.

4.3.2. Experimento 1 (E1): Modelado usando todos los descriptores

Para este experimento, se utilizan todos los descriptores de los conjuntos de datos. El único preprocesamiento que se les aplica es la normalización.

Los Cuadros 4.10 y 4.11 muestran un resumen de los puntajes de desempeño obtenidos sobre datos de prueba y entrenamiento, respectivamente, para el conjunto de datos HLYZ. Se puede notar que para todos los algoritmos el puntaje de R^2 de pruebas mínimo obtenido en las diez iteraciones es negativo; a la vez, el puntaje

Cuadro 4.10: Resumen de puntajes de prueba obtenidos en Experimento 1 para conjunto HLYZ.

	Test R^2					**Test RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	-0.34	-0.41	0.71	0.57	-1.58	4.69	4.57	1.03	6.30	3.46
RIDGE	0.38	0.46	0.28	0.79	-0.09	3.28	3.00	0.90	5.19	2.45
LASSO	0.25	0.38	0.38	0.66	-0.45	3.65	3.56	1.02	5.33	2.38
PLS	0.29	0.37	0.35	0.72	-0.40	3.54	3.25	1.02	5.81	2.53
SVR	0.32	0.49	0.40	0.69	-0.42	3.41	3.28	0.97	5.23	2.29

Cuadro 4.11: Resumen de puntajes de entrenamiento obtenidos en Experimento 1 para conjunto HLYZ.

	Train R^2					**Train RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	1.00	1.00	0.00	1.00	1.00	0.00	0.00	0.00	0.00	0.00
RIDGE	0.92	0.94	0.07	0.95	0.72	1.24	1.13	0.39	2.34	1.05
LASSO	0.82	0.83	0.02	0.84	0.77	1.97	1.96	0.06	2.09	1.90
PLS	0.88	0.91	0.14	0.98	0.49	1.47	1.38	0.68	3.12	0.65
SVR	0.95	0.95	0.01	0.96	0.93	1.08	1.09	0.13	1.27	0.92

máximo no llega a ser tan alto. Los promedios también son bajos. De entre todos los algoritmos, OLS presenta el peor desempeño, teniendo el promedio y la mediana más baja. Ésto tiene sentido si se observan los puntajes de entrenamiento. Para OLS, el desempeño sobre los datos de entrenamiento es perfecto, lo que muestra un *overfitting* severo. Ésto se debe a que OLS, al contrario que el resto de los algoritmos, no incorpora un mecanismo para combatir el *overfitting*, como por ejemplo la regularización. Resultados similares son observados en los demás conjuntos de datos.

Los puntajes de RMSE de prueba para el conjunto HLYZ son bastante más altos con respecto a los del resto de los conjuntos, como por ejemplo 1BPI, cuyos resultados se muestran en el Cuadro 4.12.

En general, los puntajes de R^2 de prueba son muy variables, llegando a valores muy negativos, lo que sucede cuando los modelos siguen una tendencia muy distinta de los datos. Los resultados de los demás conjuntos de datos se encuentran en el Anexo C.1.

Cuadro 4.12: Resumen de puntajes de prueba obtenidos en Experimento 1 para conjunto 1BPI.

	Test R^2					**Test RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	-2.21	0.13	6.65	0.74	-20.91	1.92	1.83	0.53	2.58	0.87
RIDGE	0.16	0.23	0.87	0.93	-2.13	1.36	1.40	0.50	2.13	0.70
LASSO	0.12	0.17	0.62	0.82	-1.24	1.54	1.52	0.61	2.59	0.72
PLS	0.04	0.20	1.02	0.93	-2.56	1.37	1.37	0.51	2.21	0.71
SVR	-0.07	0.34	1.72	0.91	-4.91	1.37	1.43	0.45	2.12	0.75

El análisis da origen a la conjetura de que los mecanismos para lidiar con el *overfitting* de cada algoritmo no son completamente suficientes para lidiar con la alta dimensionalidad (con respecto a la cantidad de instancias) de los datos. También, se piensa que podría existir información redundante y/o ruido en los datos. Por ésto, hacer una reducción de la cantidad de atributos podría mejorar los resultados.

4.3.3. Experimento 2 (E2): Modelado usando datos agregados con media aritmética

Este experimento se realiza con los datos agregados usando la media aritmética. Por cada propiedad p_k, se agregan sus correspondientes diez descriptores utilizando la media. Así, los conjuntos de datos se reducen de 480 a 48 atributos.

Los Cuadros 4.13 y 4.14 presentan un resumen de los puntajes de prueba y entrenamiento, respectivamente, obtenidos en este experimento para el conjunto HLYZ. Los resultados en general, tanto para RMSE como R^2 de pruebas siguen teniendo una altísima variabilidad. Los puntajes de entrenamiento, por su parte, indican incluso un *underfitting*, que es cuando los modelos no son capaces captar la relación en los datos entrenamiento y el desempeño de generalización no es bueno. Lo anterior descrito también ocurre en los otros tres conjuntos, cuyos resultados están en el Anexo C.2.

En vista de los resultados, se genera la conjetura de que agregar los descriptores por propiedad usando la media podría, posiblemente, no ser lo adecuado. Por ello, se hace una revisión de los trabajos anteriores y se determina que existía un error en el entendimiento del dominio, particularmente sobre la definición de los descriptores

Cuadro 4.13: Resumen de puntajes de prueba obtenidos en Experimento 2 para conjunto HLYZ.

	Test R^2					**Test RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	-0.43	-0.35	0.85	0.63	-2.08	4.76	4.69	0.88	6.60	3.21
RIDGE	0.11	0.22	0.41	0.62	-0.60	4.02	3.88	1.19	5.78	2.21
LASSO	0.04	0.21	0.53	0.58	-1.08	4.09	4.23	1.08	5.61	2.25
PLS	0.05	0.23	0.54	0.58	-1.05	4.04	4.23	1.05	5.41	2.25
SVR	0.22	0.28	0.42	0.68	-0.67	3.68	3.96	0.93	4.74	1.97

Cuadro 4.14: Resumen de puntajes de entrenamiento obtenidos en Experimento 2 para conjunto HLYZ.

	Train R^2					**Train RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	0.72	0.72	0.02	0.75	0.69	2.48	2.48	0.07	2.58	2.40
RIDGE	0.41	0.39	0.06	0.51	0.33	3.60	3.62	0.19	3.86	3.28
LASSO	0.40	0.43	0.09	0.48	0.21	3.62	3.51	0.31	4.25	3.35
PLS	0.43	0.43	0.04	0.48	0.36	3.53	3.51	0.12	3.72	3.35
SVR	0.62	0.60	0.08	0.74	0.51	2.87	3.02	0.36	3.34	2.37

AASA. Los descriptores de una propiedad son calculados de manera distinta entre ellos, contrario a lo que se pensaba hasta este punto. Por lo tanto, agregarlos con la media no tiene sentido en relación al dominio. Por ello, para reducir la dimensionalidad se decide utilizar la estrategia descrita en la Sección 3.4.2 en el siguiente experimento.

4.3.4. Experimento 3 (E3): Modelado usando datos reducidos por selección de atributos

Para este experimento se hace una reducción la dimensionalidad descartando primero aquellos descriptores que tienen una correlación de Pearson absoluta superior a 0.99 con al menos un otro descriptor. Luego, se selecciona un subconjunto de 40 descriptores considerados los más relevantes de acuerdo a un *ranking* generado por el algoritmo *SURF*.

Los Cuadros 4.15 y 4.16 presentan un resumen de los puntajes de prueba y

Cuadro 4.15: Resumen de puntajes de prueba obtenidos en Experimento 3 para conjunto HLYZ usando validación Nested CV.

	Test R^2					**Test RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	0.12	0.20	0.33	0.50	-0.51	3.98	3.53	0.92	5.53	3.06
RIDGE	0.23	0.29	0.27	0.51	-0.17	3.73	3.46	0.81	4.85	2.44
LASSO	0.18	0.32	0.44	0.64	-0.56	3.75	3.82	0.92	5.43	2.42
PLS	0.19	0.22	0.27	0.56	-0.30	3.85	3.68	1.01	5.97	2.60
SVR	0.42	0.47	0.27	0.76	-0.06	3.13	3.16	0.43	3.98	2.56

Cuadro 4.16: Resumen de puntajes de entrenamiento obtenidos en Experimento 3 para conjunto HLYZ usando validación Nested CV.

	Train R^2					**Train RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	0.71	0.71	0.02	0.73	0.67	2.53	2.54	0.06	2.59	2.43
RIDGE	0.67	0.70	0.06	0.70	0.54	2.68	2.62	0.19	3.06	2.52
LASSO	0.61	0.59	0.04	0.71	0.56	2.92	2.98	0.20	3.12	2.56
PLS	0.64	0.68	0.13	0.70	0.28	2.78	2.68	0.34	3.71	2.52
SVR	0.84	0.85	0.04	0.87	0.74	1.87	1.80	0.21	2.42	1.71

entrenamiento, respectivamente, obtenidos en el Experimento 3 sobre el conjunto de datos HLYZ. La variabilidad de los puntajes de prueba, en el caso de HLYZ, disminuyó un poco. Pero sigue siendo alta. La variabilidad de los puntajes en el resto de los conjuntos también sigue siendo alta.

En este punto, se comienza a cuestionar si es que el proceso de validación de *Nested Cross-validation* es el adecuado. Por ésto, se decide probar métodos de validación alternativos. Para ello, se repite el Experimento 3, pero esta vez usando dos ciclos de *Cross-validation* independientes no anidados; uno para optimización de hiperparámetros y otro evaluar el modelo entrenado con los hiperparámetros optimizados. La diferencia con *Nested Cross-validation* es que la selección de hiperparámetros se hace una sola vez y sobre el conjuntos de datos completo. El experimento se repite dos veces, uno usando dos ciclos de *5-Fold Cross-validation* y otra usando dos ciclos de 5x2 *Cross-validation*. Con éstos métodos de validación se busca disminuir la variabilidad de los puntajes, particularmente para el caso de R^2.

Los Cuadros 4.17, 4.18, 4.19, 4.20 muestran el resumen de los puntajes prueba del Experimento 3 obtenidos con *Nested*, 5x2 y 5CV *Cross-validation*, para los conjuntos de datos 1STN, 4LYZ, 1BPI y HLYZ, respectivamente.

Para el conjunto de HLYZ, el método de validación 5-Fold CV produce puntajes con menor desviación estándar que los otros métodos de validación, es decir, puntajes menos variables. De acuerdo a los resultados obtenidos con 5-Fold CV para HLYZ, SVR tiene ventaja, con un R^2 de pruebas promedio de 0.48 y un RMSE de pruebas promedio de 3.25. El modelo con el desempeño más bajo es OLS, teniendo un R^2 y un RMSE de pruebas promedio de 0.11 y 4.24, respectivamente. *Ridge, Lasso* y PLS tienen un desempeño similar.

Para el conjunto 1STN, la validación 5x2 CV tiene una menor variabilidad de puntajes en comparación a los otros dos métodos de validación. De acuerdo a los resultados entregados por la validación 5x2 CV, *Ridge* y SVR tienen el mejor desempeño, con un RMSE de pruebas promedio de 1.28 y 1.27, respectivamente, y un R^2 de pruebas promedio de 0.39 para ambos. Siguen *Lasso* y PLS con un RMSE de 1.42 y 1.39, respectivamente, y un R^2 de 0.25 y 0.28, respectivamente. OLS tiene los peores puntajes.

Para el conjunto 4LYZ, la validación 5x2 CV tiene una menor variabilidad de puntajes en comparación a los otros dos métodos de validación. Sin embargo, los resultados siguen siendo altamente variables en el caso de R^2 de pruebas. El RMSE de prueba promedio es de 1.39 para SVR, siendo el más bajo, mientras que el resto tienen RMSE promedio de 1.50 hacia arriba. OLS nuevamente es el peor método, con un RMSE de pruebas promedio de 2.97.

Para el conjunto 1BPI, la validación 5x2 CV tiene la menor variabilidad de puntajes. SVR tiene el mejor desempeño con un R^2 y RMSE de prueba promedio de 0.62 y 1.42, respectivamente. Le siguen PLS y *Ridge* , y un RMSE de 1.67 y 1.69, respectivamente, y un R^2 de 0.46 para ambos. Sigue *Lasso*, con un R^2 de 0.35 y un RMSE de 1.82. Finalmente, OLS es el que tiene el peor desempeño.

En conclusión se observa que el algoritmo SVR tiene el mejor desempeño de generalización en la mayoría de los casos. Cuadros con los puntajes obtenidos en el Experimento 3 se adjuntan en el Anexo C.3.

Los hiperparámetros seleccionados en el Experimento 3 con métodos alternativos se muestra en el Cuadro 4.21.

Cuadro 4.17: Resumen de puntajes de prueba obtenidos en Experimento 3 para conjunto 1STN, usando *Nested*, 5-Fold y 5x2 *Cross-validation*.

		Test R^2					**Test RMSE**				
		$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
Nested CV	OLS	-37.83	-17.35	51.95	-1.58	-173.64	6.73	7.06	2.33	9.90	3.42
	RIDGE	0.06	0.30	0.91	0.76	-2.33	1.24	1.13	0.41	1.90	0.67
	LASSO	-0.06	0.15	1.00	0.73	-2.63	1.31	1.09	0.45	1.98	0.72
	PLS	-0.22	0.19	1.16	0.76	-2.88	1.38	1.18	0.54	2.27	0.73
	SVR	0.10	0.21	0.67	0.71	-1.49	1.27	1.16	0.49	2.26	0.61
5-Fold CV	OLS	-12.42	-11.27	10.02	-3.71	-28.10	5.27	5.23	1.47	7.04	3.68
	RIDGE	0.32	0.43	0.35	0.60	-0.28	1.25	1.25	0.25	1.50	0.89
	LASSO	0.23	0.29	0.43	0.56	-0.49	1.33	1.39	0.29	1.59	0.88
	PLS	0.27	0.35	0.35	0.56	-0.32	1.31	1.33	0.24	1.53	0.95
	SVR	0.39	0.38	0.23	0.69	0.11	1.20	1.23	0.24	1.44	0.90
5x2 CV	OLS	-3.68	-3.33	2.28	-0.78	-7.12	3.43	3.45	0.80	4.60	2.19
	RIDGE	0.39	0.40	0.07	0.50	0.29	1.28	1.30	0.15	1.49	1.01
	LASSO	0.25	0.23	0.14	0.45	-0.02	1.42	1.43	0.19	1.78	1.02
	PLS	0.28	0.34	0.16	0.45	-0.04	1.39	1.37	0.17	1.65	1.08
	SVR	0.39	0.44	0.18	0.62	0.05	1.27	1.29	0.21	1.57	0.85

Cuadro 4.18: Resumen de puntajes de prueba obtenidos en Experimento 3 para conjunto 4LYZ, usando *Nested*, 5-Fold y 5x2 *Cross-validation*.

		Test R^2					**Test RMSE**				
		$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
Nested CV	OLS	-71.58	-34.68	100.90	-4.79	-340.24	7.53	7.29	2.89	12.16	3.37
	RIDGE	-0.91	0.09	2.09	0.73	-6.21	1.44	1.34	0.64	2.50	0.62
	LASSO	-1.87	-0.35	3.23	-0.01	-10.32	1.79	1.77	0.64	2.86	0.83
	PLS	-1.26	-0.12	3.00	0.70	-9.04	1.43	1.31	0.62	2.36	0.65
	SVR	-0.53	0.18	1.71	0.73	-4.51	1.21	1.14	0.47	2.23	0.63
5-Fold CV	OLS	-1076.61	-182.69	2084.51	-36.38	-4802.95	30.84	19.42	29.44	81.97	9.66
	RIDGE	0.04	0.03	0.16	0.26	-0.14	1.46	1.49	0.35	1.82	0.98
	LASSO	-0.28	-0.21	0.16	-0.18	-0.56	1.71	1.76	0.50	2.33	1.00
	PLS	-0.03	-0.02	0.24	0.35	-0.30	1.49	1.59	0.31	1.79	1.05
	SVR	0.26	0.40	0.28	0.44	-0.22	1.24	1.18	0.26	1.64	1.01
5x2 CV	OLS	-3.59	-2.33	3.75	-1.02	-12.85	2.97	2.93	0.51	3.77	2.14
	RIDGE	0.06	0.08	0.17	0.32	-0.33	1.50	1.52	0.36	2.03	1.08
	LASSO	-0.18	-0.12	0.19	-0.00	-0.69	1.68	1.65	0.36	2.22	1.22
	PLS	-0.05	-0.04	0.24	0.36	-0.63	1.58	1.66	0.39	2.09	1.14
	SVR	0.18	0.21	0.22	0.40	-0.35	1.39	1.30	0.34	1.96	1.05

Cuadro 4.19: Resumen de puntajes de prueba obtenidos en Experimento 3 para conjunto 1BPI, usando *Nested*, 5-Fold y 5x2 *Cross-validation*.

		Test R^2					**Test RMSE**				
		$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
Nested CV	OLS	-6.36	-1.92	14.77	0.49	-47.99	2.90	2.75	0.89	4.38	1.53
	RIDGE	-0.21	0.12	1.03	0.82	-2.06	1.65	1.57	0.53	2.61	0.84
	LASSO	-0.20	0.18	1.01	0.88	-2.07	1.63	1.64	0.49	2.23	0.84
	PLS	-0.24	0.04	1.05	0.92	-2.21	1.65	1.62	0.62	2.64	0.69
	SVR	0.18	0.61	0.98	0.88	-1.92	1.24	1.16	0.41	1.87	0.71
5-Fold CV	OLS	-14.09	-11.83	11.54	-1.93	-29.18	7.72	6.86	4.71	15.68	3.38
	RIDGE	0.23	0.74	0.91	0.79	-1.35	1.45	1.44	0.32	1.74	0.94
	LASSO	0.18	0.72	0.99	0.82	-1.52	1.47	1.52	0.38	1.82	0.86
	PLS	0.20	0.70	0.96	0.86	-1.44	1.46	1.57	0.43	1.83	0.75
	SVR	0.42	0.75	0.66	0.82	-0.74	1.30	1.39	0.24	1.46	0.86
5x2 CV	OLS	-0.45	0.02	1.10	0.28	-3.06	2.63	2.31	0.83	4.66	1.93
	RIDGE	0.46	0.49	0.14	0.59	0.13	1.69	1.65	0.21	2.11	1.39
	LASSO	0.35	0.46	0.26	0.62	-0.12	1.82	1.77	0.27	2.40	1.53
	PLS	0.46	0.50	0.17	0.61	0.06	1.67	1.70	0.17	1.87	1.32
	SVR	0.62	0.61	0.11	0.80	0.40	1.42	1.44	0.21	1.67	1.09

Cuadro 4.20: Resumen de puntajes de prueba obtenidos en Experimento 3 para conjunto HLYZ, usando *Nested*, 5-Fold y 5x2 *Cross-validation*.

		Test R^2					**Test RMSE**				
		$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
Nested CV	OLS	0.12	0.20	0.33	0.50	-0.51	3.98	3.53	0.92	5.53	3.06
	RIDGE	0.23	0.29	0.27	0.51	-0.17	3.73	3.46	0.81	4.85	2.44
	LASSO	0.18	0.32	0.44	0.64	-0.56	3.75	3.82	0.92	5.43	2.42
	PLS	0.19	0.22	0.27	0.56	-0.30	3.85	3.68	1.01	5.97	2.60
	SVR	0.42	0.47	0.27	0.76	-0.06	3.13	3.16	0.43	3.98	2.56
5-Fold CV	OLS	0.11	-0.03	0.23	0.40	-0.08	4.24	4.11	0.55	4.90	3.49
	RIDGE	0.28	0.24	0.12	0.42	0.13	3.84	3.92	0.52	4.42	2.99
	LASSO	0.28	0.20	0.16	0.46	0.13	3.83	3.87	0.45	4.41	3.18
	PLS	0.29	0.33	0.24	0.60	-0.05	3.73	3.75	0.32	4.07	3.30
	SVR	0.48	0.49	0.13	0.67	0.35	3.25	3.13	0.50	3.80	2.72
5x2 CV	OLS	-0.78	-0.55	0.64	-0.15	-1.96	6.08	6.03	0.98	8.06	4.85
	RIDGE	0.09	0.06	0.23	0.38	-0.41	4.40	4.44	0.61	5.43	3.61
	LASSO	0.10	0.10	0.22	0.43	-0.31	4.37	4.34	0.66	5.55	3.50
	PLS	0.03	-0.00	0.25	0.42	-0.36	4.53	4.53	0.72	5.74	3.58
	SVR	0.34	0.35	0.10	0.46	0.10	3.77	3.78	0.44	4.35	3.13

Cuadro 4.21: Hiperparámetros seleccionados por cada algoritmo de aprendizaje y por cada conjunto de datos, a partir de los resultados del Experimento 3 con métodos de validación alternativos.

Method	Hyperparameter	1STN	4LYZ	1BPI	HLYZ
Ridge	λ	100,0	100,0	100,0	10,0
Lasso	λ	0,1	0,1	1,0	0,1
PLS	*n. components*	2	2	2	12
SVR	C	10,0	10,0	100,0	1000,0
	ϵ (*epsilon*)	1,3	1,0	1,0	1.3
	kernel	*rbf*	*rbf*	*rbf*	*rbf*
	γ (*gamma*)	0,01	0,01	0,01	0.001

4.4. Discusión

Los resultados obtenidos en el Experimento 3 de la etapa de modelado, en los cuáles se aplican tres métodos de validación distintos, tienen resultados interesantes. Se observa que la metodología de evaluación *Nested Cross-validation* con dos ciclos de *10-Fold Cross-validation* anidados, produce puntajes de desempeño de generalización que tienen una alta variabilidad. Se observa, además, que la aplicación de los métodos de validación alternativos, 5x2 *Cross-validation* y *5-Fold Cross-validation*, disminuye la variabilidad de los puntajes. Un ejemplo de ésto se observa en los resultados obtenidos para el conjunto de datos 1STN, los cuales se muestran en el Cuadro 4.17, donde los puntajes de R^2 obtenidos con 5x2 *Cross-validation* tienen desviaciones estándar más bajas que los obtenidos con *Nested Cross-validation*. Para los conjuntos de datos más pequeños, 1STN, 4LYZ y 1BPI, en la mayoría de los casos el método 5x2 *Cross-validation* produce la menor variabilidad en los puntajes de desempeño, mientras que para el conjunto más grande, HLYZ, en la mayoría de los casos el método *5-Fold Cross-validation* produce la menor variabilidad. Ésto se observa tanto para puntajes de R^2 como de RMSE. La disminución de la variabilidad de los puntajes observada se atribuye a que los métodos de validación alternativos apartan conjuntos de prueba con mayor cantidad de instancias, haciendo que los puntajes sean menos sensibles a qué observaciones queden en los conjuntos de prue-

ba.

La menor variabilidad de los puntajes hace que las estimaciones del desempeño de generalización de los algoritmos sea más fiable. Por esta razón, para señalar las conclusiones relacionadas a las preguntas de investigación, se consideran los resultados del modelado usando los 40 descriptores más relevantes de cada conjunto de datos (Experimento 3) y aplicando los métodos de validación alternativos. Para el conjunto de datos HLYZ, se consideran los resultados obtenidos con *5-Fold Cross-validation*, mientras que para los conjuntos de datos 1STN, 4LYZ y 1BPI se consideran los resultados obtenidos con 5x2 *Cross-validation*.

La Pregunta de Investigación 1 (RQ1) que se busca responder en este trabajo es:

- **RQ1**: ¿Será que modelos entrenados con algoritmos de aprendizaje que modelan relaciones lineales entre predictores y respuesta, tienen un buen desempeño en la predicción de la estabilidad de proteínas a partir de vectores AASA?

En relación a esta pregunta, un algoritmo de modelado lineal tiene un buen desempeño en la predicción de la estabilidad si es que, consistentemente a través de los distintos conjuntos de datos abordados, el desempeño de generalización de los modelos entrenados por éste es bueno. En los resultados se observa que el desempeño de generalización de las técnicas de modelado lineal es consistentemente bajo a través de todos los conjuntos de datos. El puntaje de R^2 de pruebas promedio más alto observado entre los algoritmos de modelado lineal es de 0.46, correspondiente a PLS y a *Ridge* en el conjunto de datos 1BPI, lo que está por debajo del umbral de 0.8, sobre el cual se consideraría un desempeño bueno. El puntaje de RMSE de pruebas promedio más bajo observado entre los algoritmos de modelado lineal es de 1.28, correspondiente a *Ridge* en el conjunto de datos 1STN, lo que está sobre el umbral de 0.5; un RMSE por debajo a 0.5 habría sido considerado un buen desempeño. Por lo tanto, de acuerdo a los resultados observados, los algoritmos de modelado lineal utilizados en este trabajo no tienen un desempeño bueno en la predicción de la estabilidad.

La Pregunta de Investigación 2 (RQ2) que se busca responder en este trabajo es:

- **RQ2**: ¿Será que los modelos de predicción entrenados con dichos algoritmos tienen un desempeño similar al de modelos más complejos como lo son aquellos entrenados por *SVR* con *kernels* no lineales?

En relación a esta pregunta, para poder concluir que un algoritmo de modelado lineal tiene un desempeño similar con el de SVR, se debe cumplir, consistentemente a través de los distintos conjuntos de datos abordados, que el desempeño de generalización estimado para el algoritmo de modelado lineal sea cercano al estimado para SVR. Se observa que el desempeño SVR con *kernels* no lineales es, en la mayoría de los casos, superior al de las técnicas de modelado lineal. Para el conjunto de datos 1STN, SVR tiene un R^2 y un RMSE de prueba promedio de 0.39 y 1.27, respectivamente, mientras que *Ridge*, cuyo desempeño es el mejor entre los algoritmos de modelado lineal, tiene un R^2 y un RMSE de prueba promedio de 0.39 y 1.28, respectivamente. Para el conjunto de datos 4LYZ, SVR tiene un R^2 y un RMSE de prueba promedio de 0.18 y 1.24, respectivamente, mientras que *Ridge*, cuyo desempeño es el mejor entre los algoritmos de modelado lineal, tiene un R^2 y un RMSE de prueba promedio de 0.06 y 1.52, respectivamente. Para el conjunto de datos 1BPI, SVR tiene un R^2 y un RMSE de prueba promedio de 0.62 y 1.42, respectivamente, mientras que *PLS*, cuyo desempeño es el mejor entre los algoritmos de modelado lineal, tiene un R^2 y un RMSE de prueba promedio de 0.42 y 1.67, respectivamente. Para el conjunto de datos HLYZ, SVR tiene un R^2 y un RMSE de prueba promedio de 0.48 y 3.25, respectivamente, mientras que *PLS*, cuyo desempeño es el mejor entre los algoritmos de modelado lineal, tiene un R^2 y un RMSE de prueba promedio de 0.29 y 3.73, respectivamente. Los resultados observados tienen concordancia con lo esperado. Los modelos lineales son más simples, pero a la vez menos flexibles. SVR, utilizado con *kernels* no lineales, es más flexible y permite modelar relaciones más complejas. Por lo tanto, si la relación entre los descriptores y estabilidad es más compleja que una relación lineal, como se esperaba, SVR tiene la ventaja.

Otras observaciones hechas en el desarrollo de la investigación, y que es importante destacar pensando en trabajos a futuros, son las siguientes:

- Para el conjunto 1BPI, el desempeño de los algoritmos, respecto a los puntajes R^2 de prueba, es mejor en comparación al observado en otros conjuntos de datos, alcanzando un R^2 de prueba promedio de 0.62 con SVR. El R^2 de prueba promedio más alto obtenido en el resto de los conjuntos es de 0.48 con SVR en el conjunto de datos HLYZ. Ésto sugiere la presencia de características distintivas del resto de los conjuntos. Una característica evidente que podría ser señalada como causa, es la poca presencia de mutaciones con una variación

positiva (2 instancias). Sin embargo, podrían haber otras razones, por lo que se recomienda explorarlo más en detalle en el futuro.

- En los resultados de la selección de atributos se observa que descriptores asociados a una misma propiedad son considerados dentro del *ranking* de descriptores más relevantes. Por ejemplo en el conjunto de datos 1BPI, 8 de los 10 descriptores asociados a la propiedad m fueron considerados dentro de los 20 descriptores más relevantes. Sucede algo similar en los otros conjuntos de datos, pero con otras propiedades. Ésto podría indicar que hay, no solo descriptores, sino que también propiedades que tienen una mayor relevancia. Además se observa que los descriptores considerados más relevantes en un conjunto de datos son muy distintos a los de otro conjunto. Ésto sería algo a tener en cuenta en el caso en que, por ejemplo, se hiciera en el futuro un trabajo que planteara la idea de fusionar los conjuntos de datos para conformar uno más grande, puesto que las propiedades relevantes en un conjunto de datos no necesariamente lo son para otro. Además, el que los descriptores más relevantes no sean los mismos, muestra la necesidad de aplicar una técnica selección de atributos de manera individual para cada uno de los conjuntos de datos.

4.5. Aspectos metodológicos mejorables

A continuación se presenta un listado de mejoras a ciertos aspectos metodológicos de este trabajo que podrían realizarse a futuro:

- Tener en cuenta la presencia de valores extremos en los descriptores en al momento de escoger el método de normalización. La media aritmética y, en consecuencia *Z-Score*, son sensibles a valores extremos. Ésto puede hacer la normalización de los datos empeore el desempeño de las demás técnicas utilizadas posteriormente.

- Un eslabón débil en la metodología propuesta es la reducción de la dimensionalidad. No se realiza algún tipo de validación al seleccionar un subconjunto de descriptores. Para mejorar ésto, una alternativa es aplicar el paradigma de *ensemble* para combinar distintos algoritmos de selección de atributos. De esta manera, se logra una selección más robusta, pues mezcla los criterios definidos

por distintos algoritmo. Más información sobre como aplicar el paradigma de *ensemble* con métodos de selección de atributos se puede encontrar en [17].

5. Conclusiones y trabajo futuro

En este capítulo se presentan las conclusiones del trabajo. Primero, se hace una revisión de las preguntas de investigación en función de los resultados observados. Finalmente, se cierra el documento planteando alternativas para trabajos futuros.

5.1. Conclusiones

El estudio y desarrollo de proteínas de alta estabilidad es de suma importancia para una gran diversidad de aplicaciones y áreas de investigación. Los vectores AASA, una representación cuantitativa de las proteínas, en conjunto con técnicas de *machine learning*, han demostrado ser útiles para la predicción de estabilidad de proteínas mutantes en investigaciones anteriores. Las metodologías de modelado predictivo documentadas hasta ahora aplican técnicas que entrenan modelos de alta complejidad. Sin embargo, si modelos más simples, como lo son los lineales, tienen un desempeño de predicción bueno o, al menos, similar al de modelos más complejos, los primeros son más deseables pues son más interpretables y más útiles para análisis posteriores. Con dicha motivación, en este trabajo se propone una metodología para determinar si es que las técnicas de modelado lineal tienen un buen desempeño para la predicción de la estabilidad a partir de vectores AASA.

En esta investigación, para cuatro conjuntos de datos de mutaciones, correspondientes a cuatro proteínas distintas, se evalúa el desempeño de cuatro técnicas de modelado lineal, OLS, *Ridge*, *Lasso* y PLS, y el desempeño de una técnica de modelado no lineal, SVR. Para evaluar el desempeño, se emplean las métricas de evaluación R^2 y RMSE, aplicando tres variantes del método de validación *Cross-validation* (CV), *Nested* CV, *5-Fold* CV y 5x2 CV.

La primera pregunta de investigación que busca responder este trabajo es:

- **RQ1**: ¿Será que modelos entrenados con algoritmos de aprendizaje que modelan relaciones lineales entre predictores y respuesta, tienen un buen desempeño en la predicción de la estabilidad de proteínas a partir de vectores AASA?

En relación a esta interrogante, el desempeño de los modelos entrenados con las técnicas de modelado lineal es consistentemente bajo a través de los cuatro conjuntos de datos. Incluso SVR, que es un algoritmo que permite modelar relaciones más complejas, muestra dificultades para entrenar modelos con buen desempeño. Ésto habla de la dificultad del problema y la complejidad inherente a los datos.

La segunda pregunta de investigación que se busca responder en este trabajo es:

- **RQ2**: ¿Será que los modelos de predicción entrenados con dichos algoritmos tienen un desempeño similar al de modelos más complejos como lo son aquellos entrenados por *SVR* con *kernels* no lineales?

En relación a esta pregunta, se observa que el desempeño de SVR con *kernel* RBF es, en la mayoría de los casos, superior al de las técnicas de modelado lineal. Los modelos lineales son más simples, pero a la vez menos flexibles. SVR, utilizado con *kernels* no lineales, puede modelar relaciones más complejas. Por lo tanto, si la relación entre los descriptores y estabilidad es más compleja que una relación lineal, como se asume usualmente, SVR tiene la ventaja.

Los resultados muestran que el problema de intentar predecir la variación de la estabilidad inducida por una mutación puede ser complejo, pensando particularmente en los desafíos que plantea el trabajar con poca cantidad de datos. El método de *Nested* CV, si bien en la teoría parece un enfoque adecuado, la poca cantidad de datos hace que la variabilidad de los puntajes sea alta al punto de no poder obtener estimaciones fiables del desempeño de generalización. No obstante, métodos alternativos como 5x2 CV y *5-Fold* CV resultan en una menor variabilidad, generando estimaciones de desempeño más confiables.

A partir de lo observado durante el desarrollo de la investigación, se concluye que distintos conjuntos de datos pueden necesitar un análisis específico para cada uno. Sin embargo, la metodología propuesta en esta investigación es reproducible y puede servir como una base para el análisis de otros conjuntos de datos disponibles a futuro. Además, la metodología es independiente de las técnicas de modelado, por

lo que puede ser extendida. El proceso de generación de datos es altamente costoso, por lo que la escasez de éstos es considerable. Teniendo en cuenta ésto, y que los conjuntos de datos abordados en esta investigación no son fabricados, es posible que dichos datos lleguen a las manos de otro investigador en el futuro. Con ésto en mente, los códigos fuente programados para el desarrollo en esta investigación se encuentran alojados en un repositorio público[1], incluyendo conjuntos de datos tanto procesados como crudos.

5.2. Trabajos futuros

A partir del trabajo realizado en ésta investigación, y en función de los aprendizajes obtenidos durante su desarrollo, se proponen las siguientes alternativas de trabajo futuro:

- Diseñar e implementar una librería que haga el análisis realizado en este trabajo de manera automática, como un método para medir y comparar el desempeño, no solo de técnicas de modelado lineal, sino que también de técnicas más complejas, sobre conjuntos de datos de mutaciones de proteínas codificados con vectores AASA. Ésto sería útil pensando en que este es un trabajo interdisciplinario, por lo que podría haber interés por parte de personas que no necesariamente tengan conocimientos del área *machine learning*. Empaquetarlo como una herramienta evitaría que quienes quieran aplicar la metodología deban implementarlo desde cero.

- Extender el análisis realizado en este trabajo aplicando métodos de *oversampling* como un paso en el preprocesamiento de los datos. Ésto permitiría combatir una de las principales dificultades encontradas en este trabajo, que es la escasez de datos. De las técnicas de *oversampling* para regresión, SMOGN, al parecer, es la única empaquetada en una herramienta. Es preferible desarrollar sobre librerías o herramientas ya implementadas, pues éstas tienden a estar documentadas y probadas. La implementación del método de SMOGN se detalla en [4].

[1]Repositorio del proyecto: `https://github.com/BenjinP/aasa-stability-prediction`

Bibliografía

[1] Ethem Alpaydin. *Introduction to Machine Learning*. Second. The MIT Press, 2010.

[2] César A. Astudillo y B. John Oommen. "Imposing tree-based topologies onto self organizing maps". En: *Inf. Sci. (Ny)*. 181.18 (sep. de 2011), págs. 3798-3815.

[3] Christopher Bishop. *Pattern Recognition and Machine Learning*. First. Springer, 2006.

[4] Paula Branco y col. "SMOGN: a Pre-processing Approach for Imbalanced Regression". En: *Proc. Mach. Learn. Res.* 74 (2017), págs. 36-50.

[5] Julio Caballero y col. "Amino acid sequence autocorrelation vectors and ensembles of bayesian-regularized genetic neural networks for prediction of conformational stability of human lysozyme mutants". En: *J. Chem. Inf. Model.* 46.3 (mayo de 2006), págs. 1255-1268.

[6] Gavin C Cawley y Nicola L.C. Talbot. "On over-fitting in model selection and subsequent selection bias in performance evaluation". En: *J. Mach. Learn. Res.* 11 (2010), págs. 2079-2107.

[7] Naresh Chennamsetty y col. "Design of therapeutic proteins with enhanced stability". En: *Proc. Natl. Acad. Sci. U. S. A.* 106.29 (jul. de 2009), págs. 11937-11942.

[8] Leyden Fernández y col. "Amino acid sequence autocorrelation vectors and Bayesian-regularized genetic neural networks for modeling protein conformational stability: Gene V protein mutants". En: *Proteins Struct. Funct. Genet.* 67.4 (jun. de 2007), págs. 834-852.

[9] Michael Fernández y col. "Comparative modeling of the conformational stability of chymotrypsin inhibitor 2 protein mutants using amino acid sequence autocorrelation (AASA) and amino acid 3D autocorrelation (AA3DA) vectors and ensembles of Bayesian-regularized genetic neural networks". En: *Mol. Simul.* 33.13 (nov. de 2007), págs. 1045-1056.

[10] Michael Fernández y col. "Proteometric modelling of protein conformational stability using amino acid sequence autocorrelation vectors and genetic algorithm-optimised support vector machines". En: *Mol. Simul.* 34.9 (ago. de 2008), págs. 857-872.

[11] Salvador García, Julián Luengo y Francisco Herrera. *Data Preprocessing in Data Mining*. First. Springer International Publishing, 2015.

[12] Ian Goodfellow, Yoshua Bengio y Aaron Courville. *Deep Learning*. First. MIT Press, 2016.

[13] M. Michael Gromiha. *Protein Bioinformatics: From Sequence to Function*. Academic Press/Elsevier, 2010, pág. 320.

[14] Trevor Hastie, Tibshirani Robert y Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Second. Springer, 2009.

[15] Gareth James y col. *An Introduction to Statistical Learning: with Applications in R*. First. Springer, 2013.

[16] Gonzalo Maldonado y col. "Predicting the stability of human lysozyme mutants using the tree-based classifier TTOSOM". En: *Chemom. Intell. Lab. Syst.* 162 (mar. de 2017), págs. 65-72.

[17] Barbara Pes. "Ensemble feature selection for high-dimensional data: a stability analysis across multiple domains". En: *Neural Comput. Appl.* 32.10 (mayo de 2020), págs. 5951-5973.

[18] Sebastian Raschka. "Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning". En: (nov. de 2018).

[19] Marko Robnik-Šikonja e Igor Kononenko. "Theoretical and Empirical Analysis of ReliefF and RReliefF". En: *Mach. Learn.* 53.1-2 (oct. de 2003), págs. 23-69.

[20] Larry D. Unsworth, John Van Der Oost y Sotirios Koutsopoulos. "Hyperthermophilic enzymes - Stability, activity and implementation strategies for high temperature applications". En: *FEBS J.* 274.16 (ago. de 2007), págs. 4044-4056.

[21] Ryan J. Urbanowicz y col. "Benchmarking relief-based feature selection methods for bioinformatics data mining". En: *J. Biomed. Inform.* 85 (nov. de 2018), págs. 168-188.

[22] Ryan J. Urbanowicz y col. *Relief-based feature selection: Introduction and review*. Nov. de 2018.

[23] Sudhir Varma y Richard Simon. "Bias in error estimation when using cross-validation for model selection". En: *BMC Bioinformatics* 7.1 (feb. de 2006), pág. 91.

[24] Rüdiger Wirth e Hipp Jochen. "CRISP-DM : Towards a Standard Process Model for Data Mining". En: *Proc. Fourth Int. Conf. Pract. Appl. Knowl. Discov. Data Min.* 24959 (2000), págs. 29-39.

ANEXOS

A. Anexos Análisis Exploratorio de Datos

A.1. Distribución de estabilidad

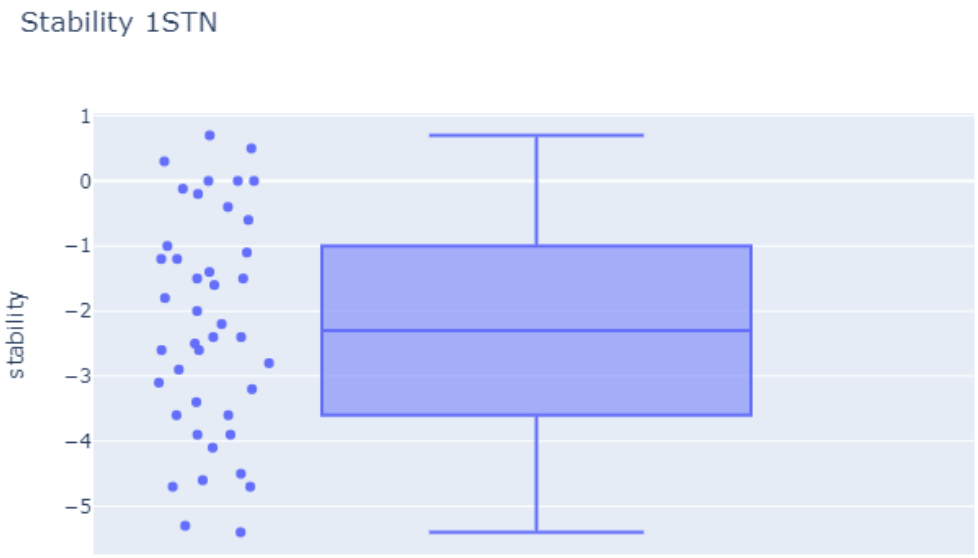


Figura A.1: *Boxplot* de la distribución de estabilidad para conjunto 1STN.

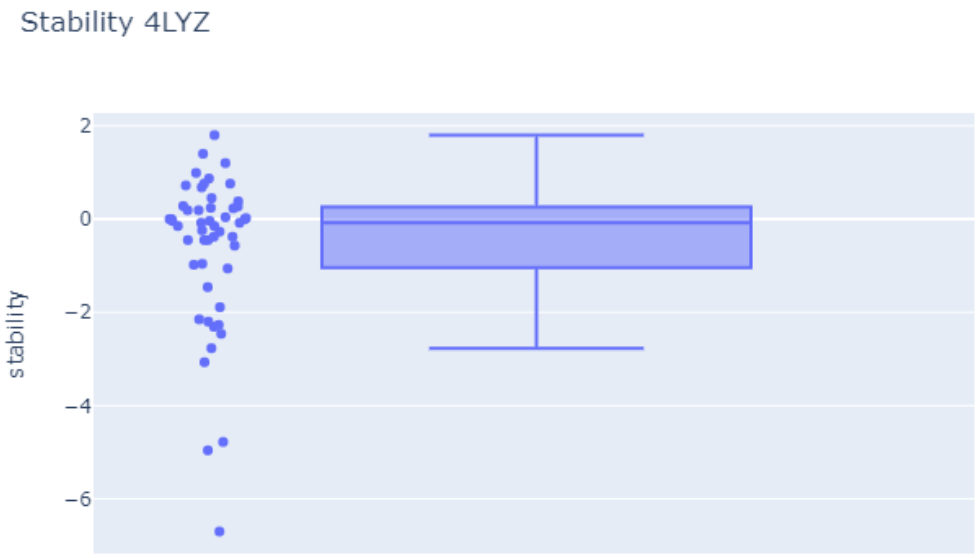


Figura A.2: *Boxplot* de la distribución de estabilidad para conjunto 4LYZ.

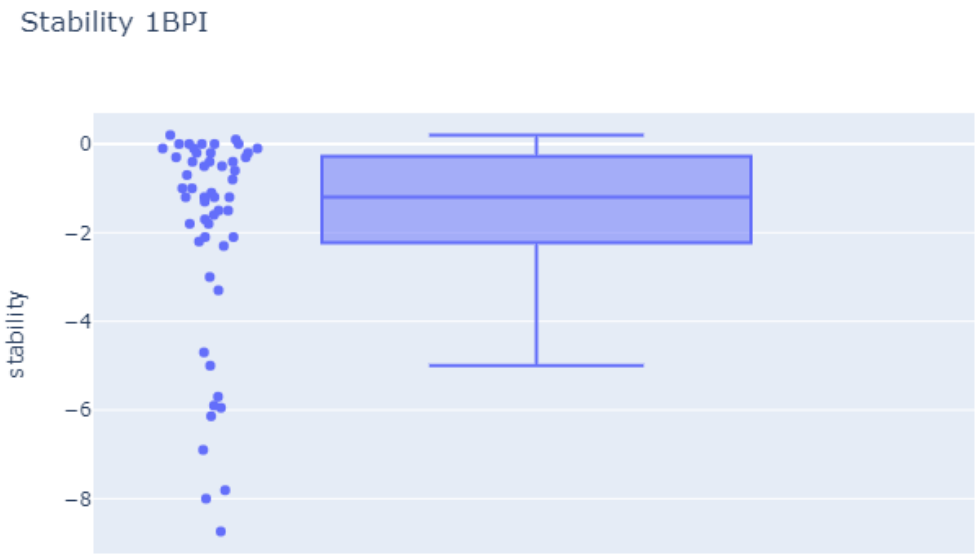


Figura A.3: *Boxplot* de la distribución de estabilidad para conjunto 1BPI.

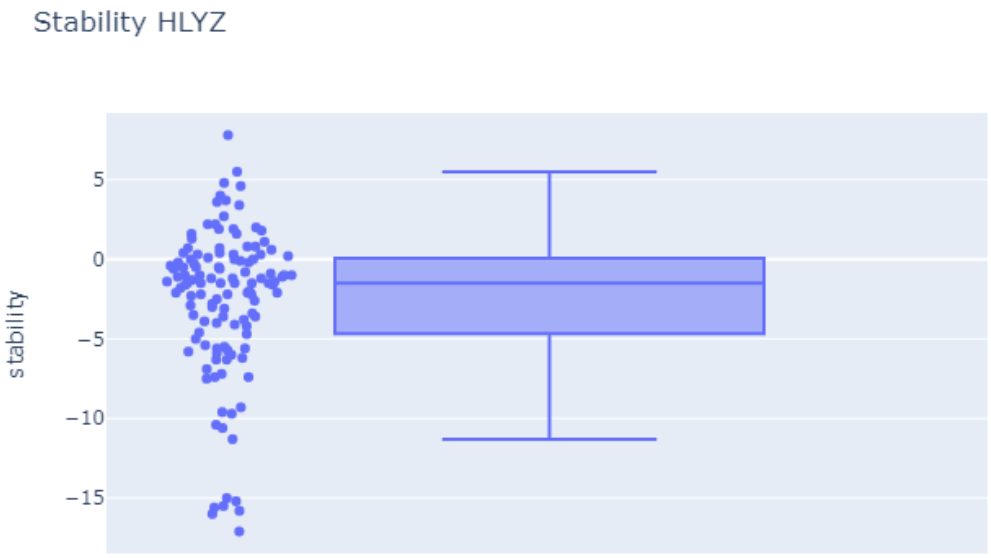


Figura A.4: *Boxplot* de la distribución de estabilidad para conjunto HLYZ.

A.2. Correlación lineal entre descriptores y estabilidad

Cuadro A.1: Los 10 descriptores más correlacionados con la estabilidad en el conjunto 4LYZ.

Descriptor	**Pearsons's r**
$AASA3ac$	0.54
$AASA10DCph$	0.52
$AASA1Ns$	0.52
$AASA7ac$	0.52
$AASA6Ns$	0.52
$AASA2Ns$	0.51
$AASA5Ns$	0.51
$AASA9Ns$	0.50
$AASA4Ns$	0.50
$AASA5ac$	0.49

Cuadro A.2: Los 10 descriptores más correlacionados con la estabilidad en el conjunto 1BPI.

Descriptor	**Pearsons's r**
$AASA8m$	0.73
$AASA10m$	0.72
$AASA9m$	0.70
$AASA8Hgm$	0.68
$AASA10Hgm$	0.67
$AASA3Hgm$	0.67
$AASA7Hgm$	0.67
$AASA1Hgm$	0.67
$AASA2Hgm$	0.66
$AASA9Hgm$	0.66

A.3. *Clustering*

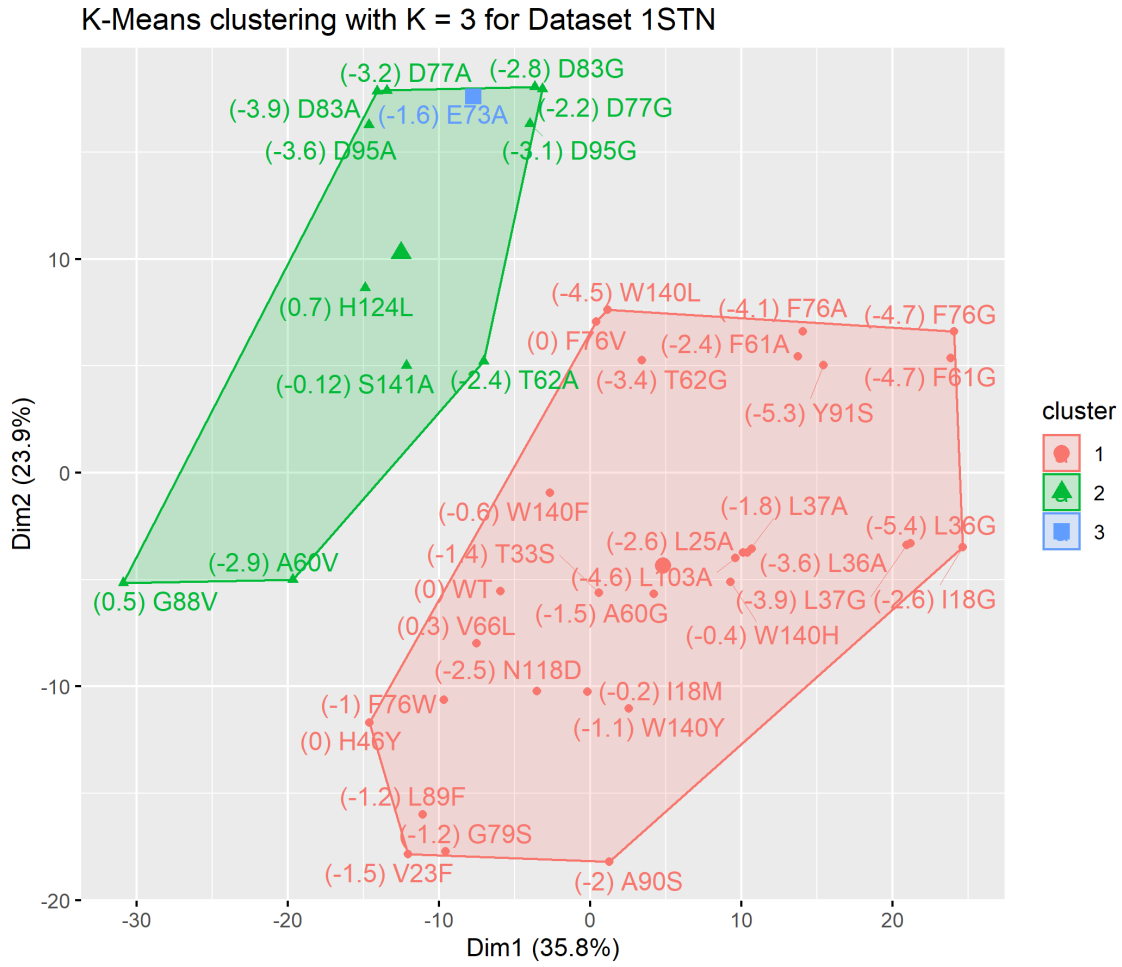


Figura A.5: Agrupaciones arrojadas por *K-means* con 3 *clusters* para el conjunto de datos 1STN.

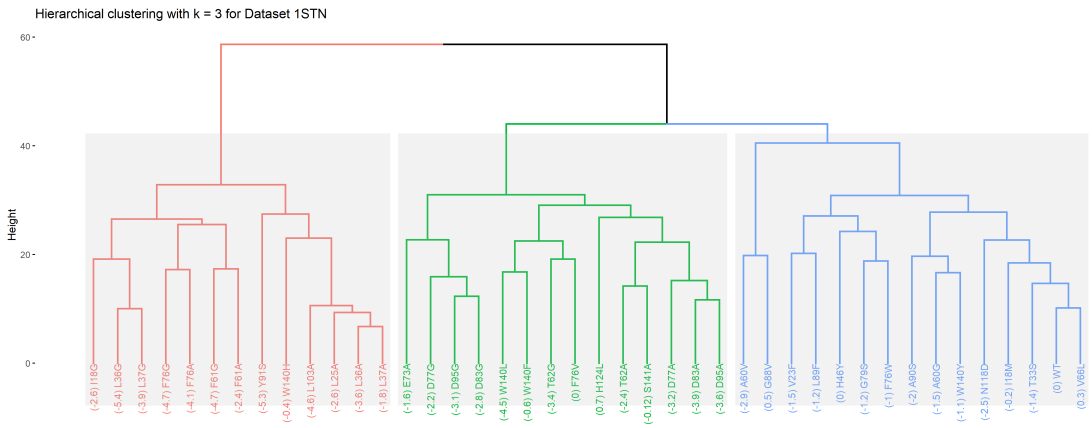


Figura A.6: Agrupaciones arrojadas por *clustering* jerárquico con 3 *clusters* para el conjunto de datos 1STN.

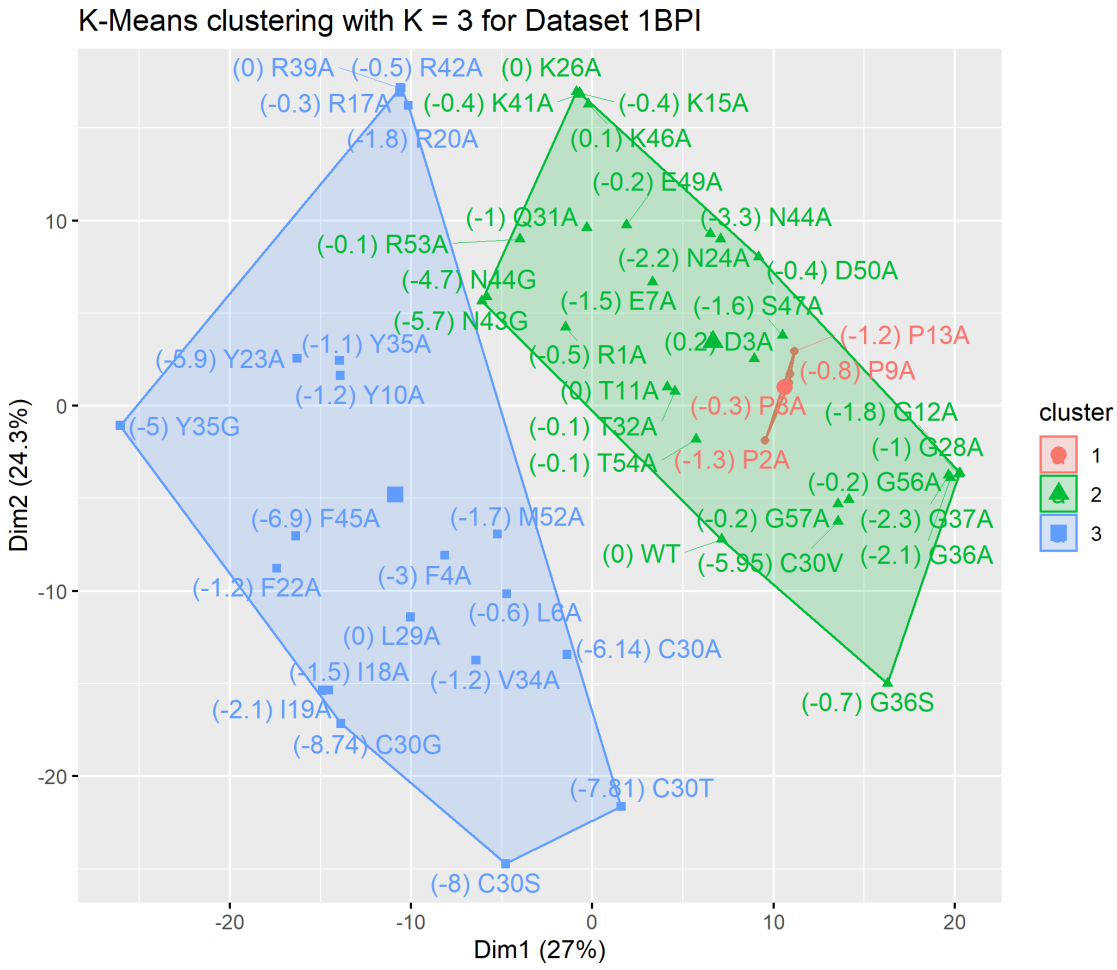


Figura A.7: Agrupaciones arrojadas por *K-means* con 3 *clusters* para el conjunto de datos 1BPI.

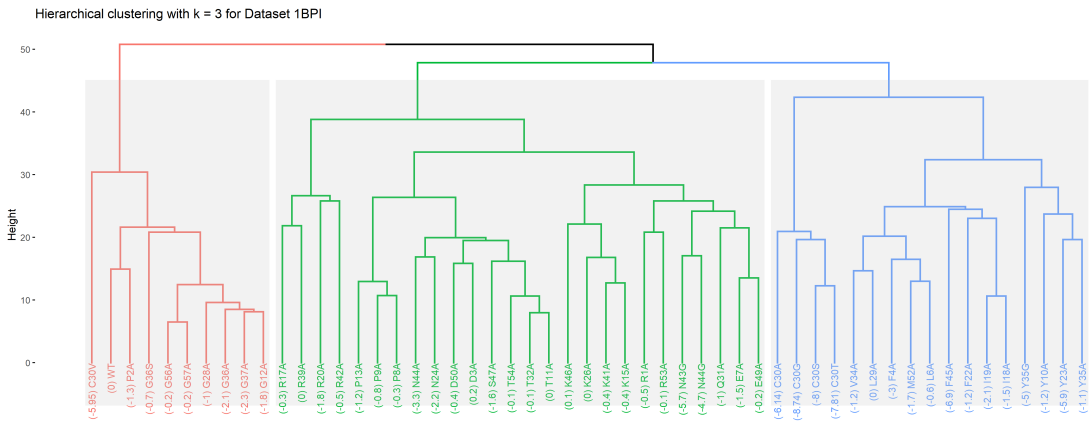


Figura A.8: Agrupaciones arrojadas por *clustering* jerárquico con 3 *clusters* para el conjunto de datos 1BPI.

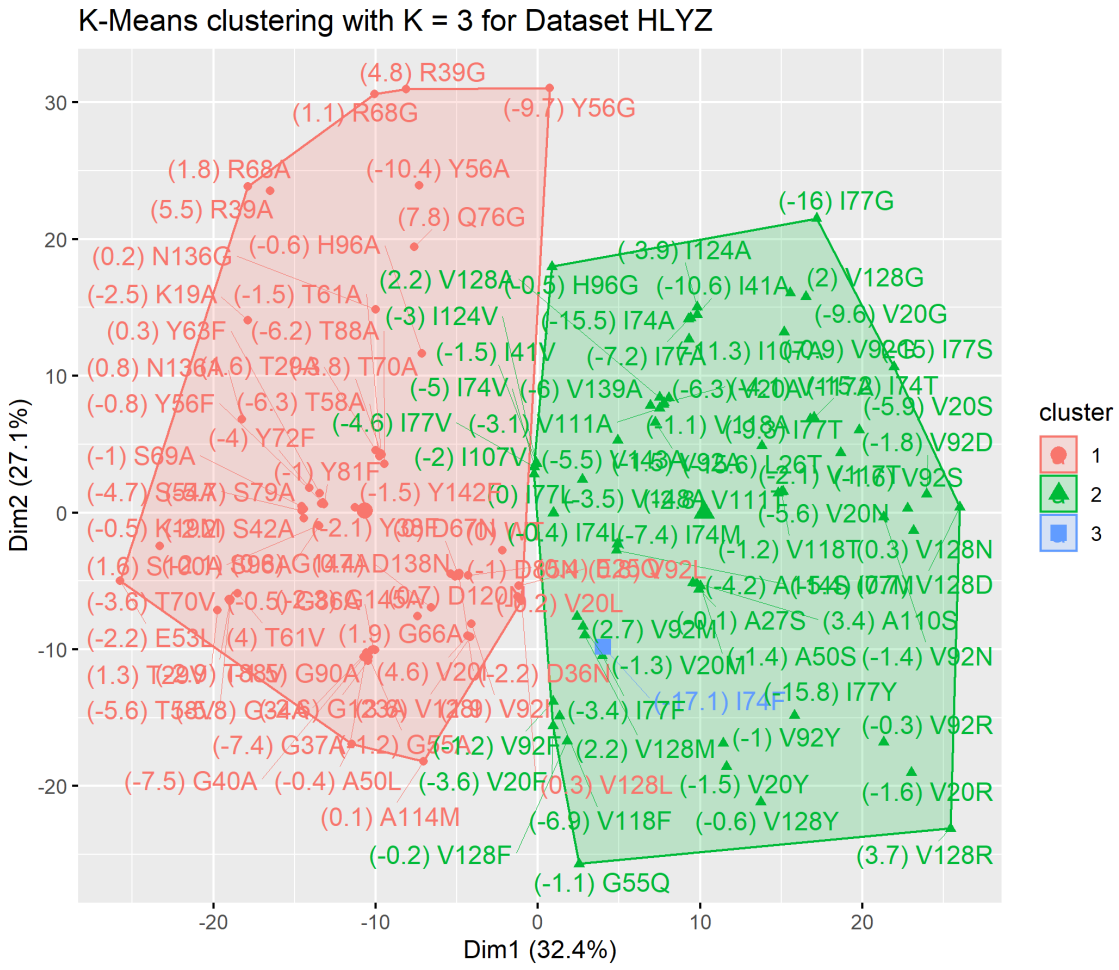


Figura A.9: Agrupaciones arrojadas por *K-means* con 3 *clusters* para el conjunto de datos HLYZ.

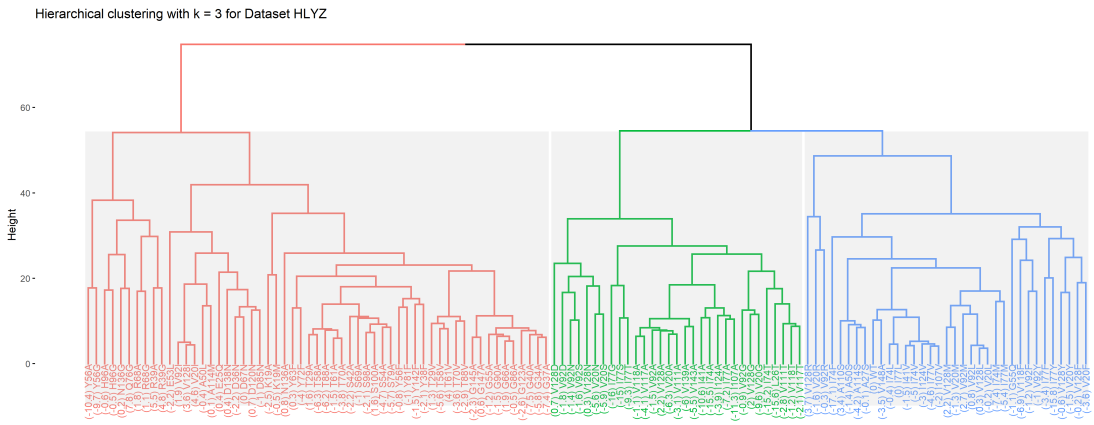


Figura A.10: Agrupaciones arrojadas por *clustering* jerárquico con 3 *clusters* para el conjunto de datos HLYZ.

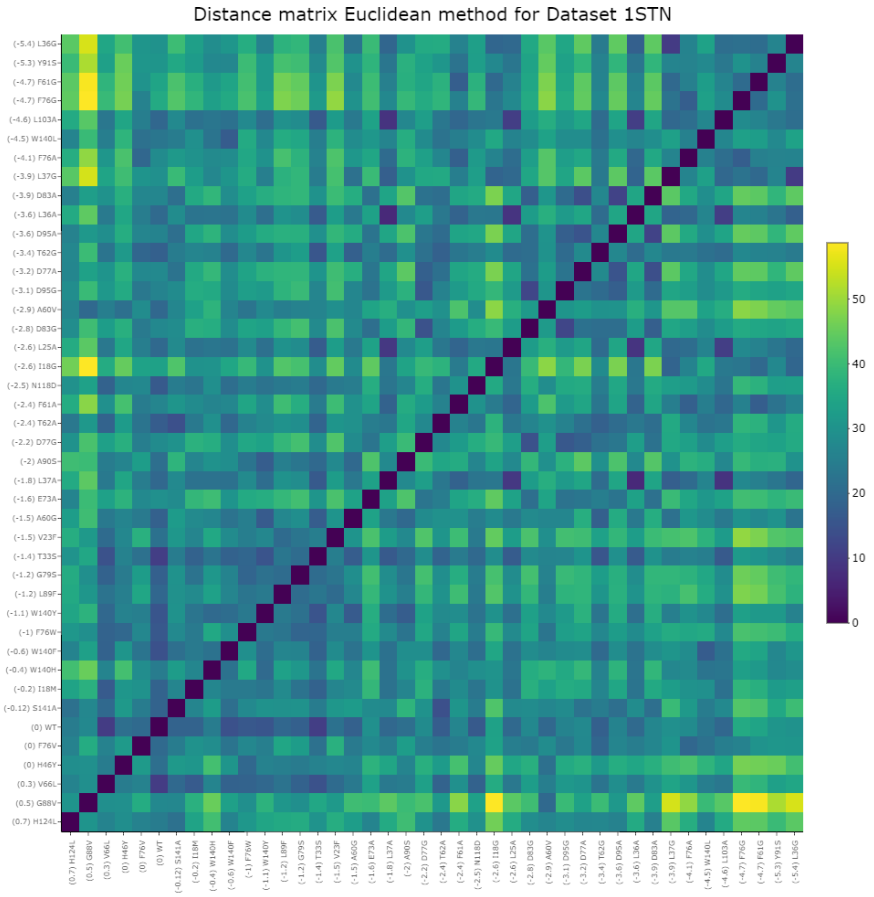


Figura A.11: Mapa de calor de la matriz de distancia euclidiana calculada para el conjunto de datos 1STN.

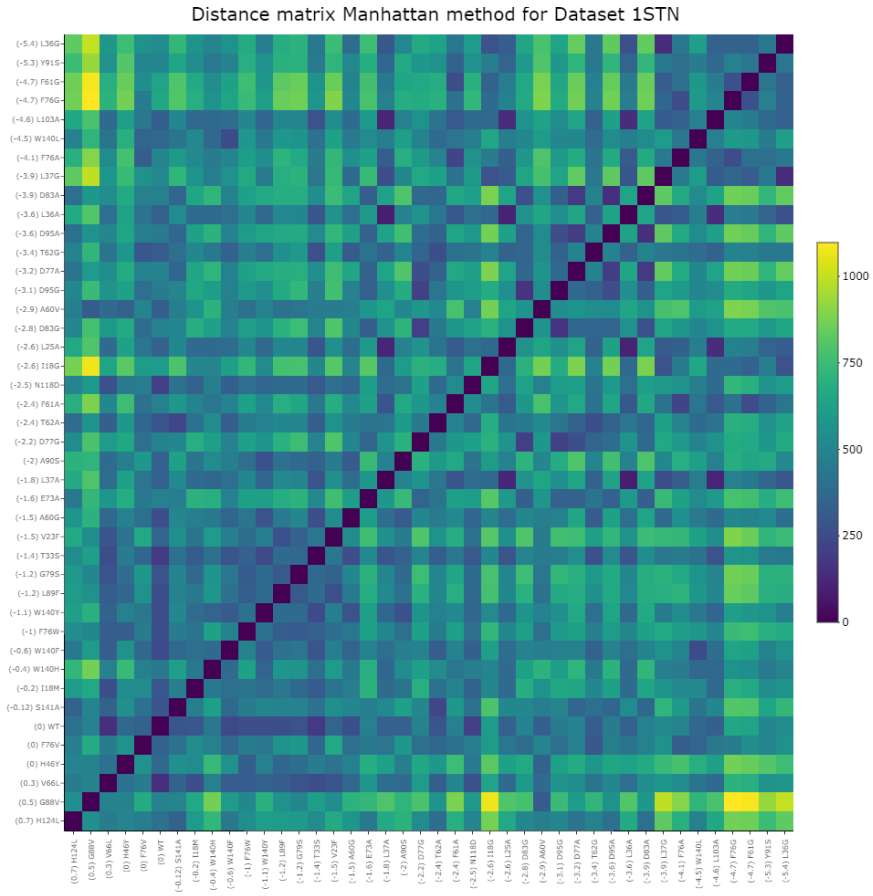


Figura A.12: Mapa de calor de la matriz de distancia *Manhattan* calculada para el conjunto de datos 1STN.

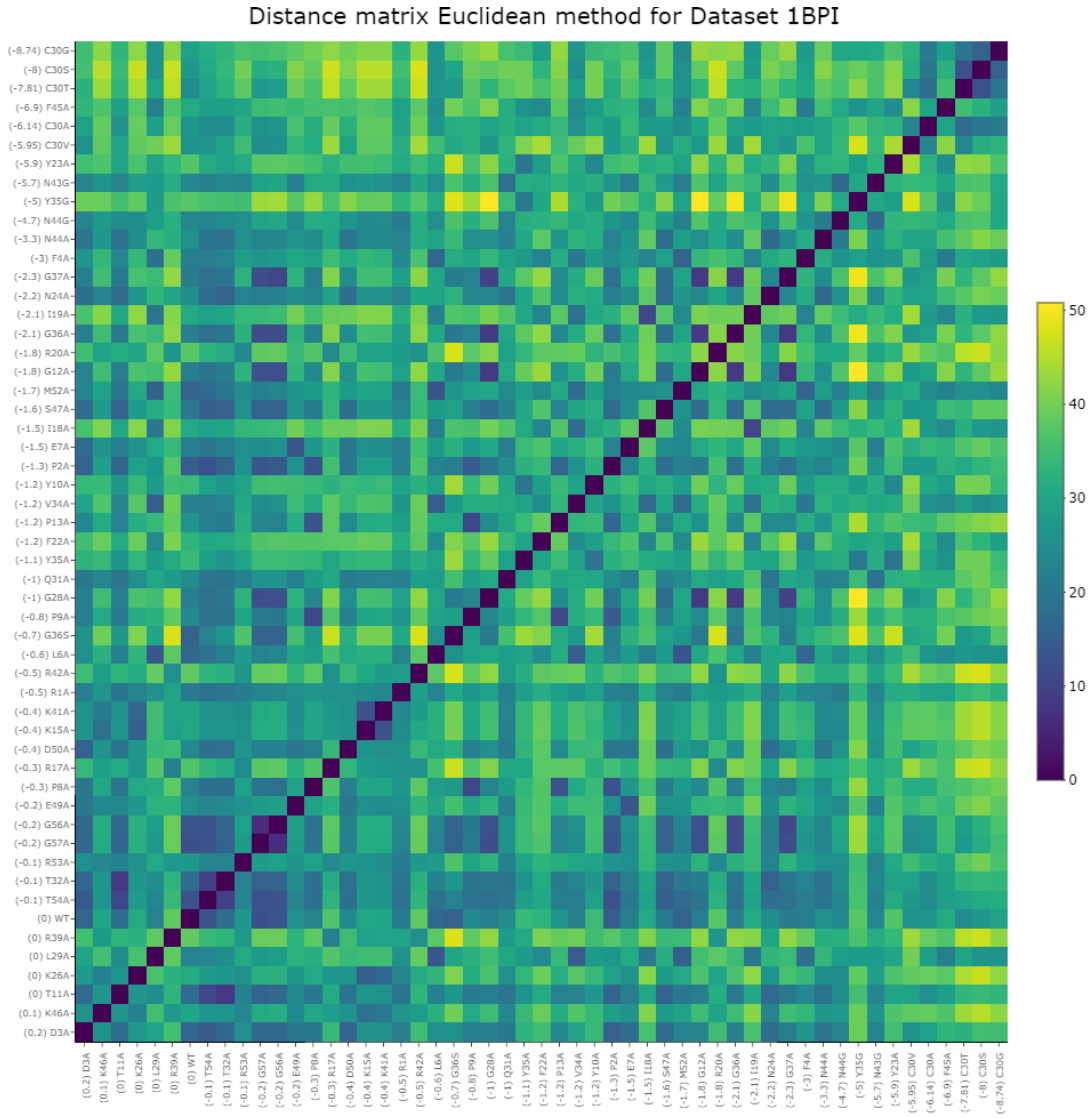


Figura A.13: Mapa de calor de la matriz de distancia euclidiana calculada para el conjunto de datos 1BPI.

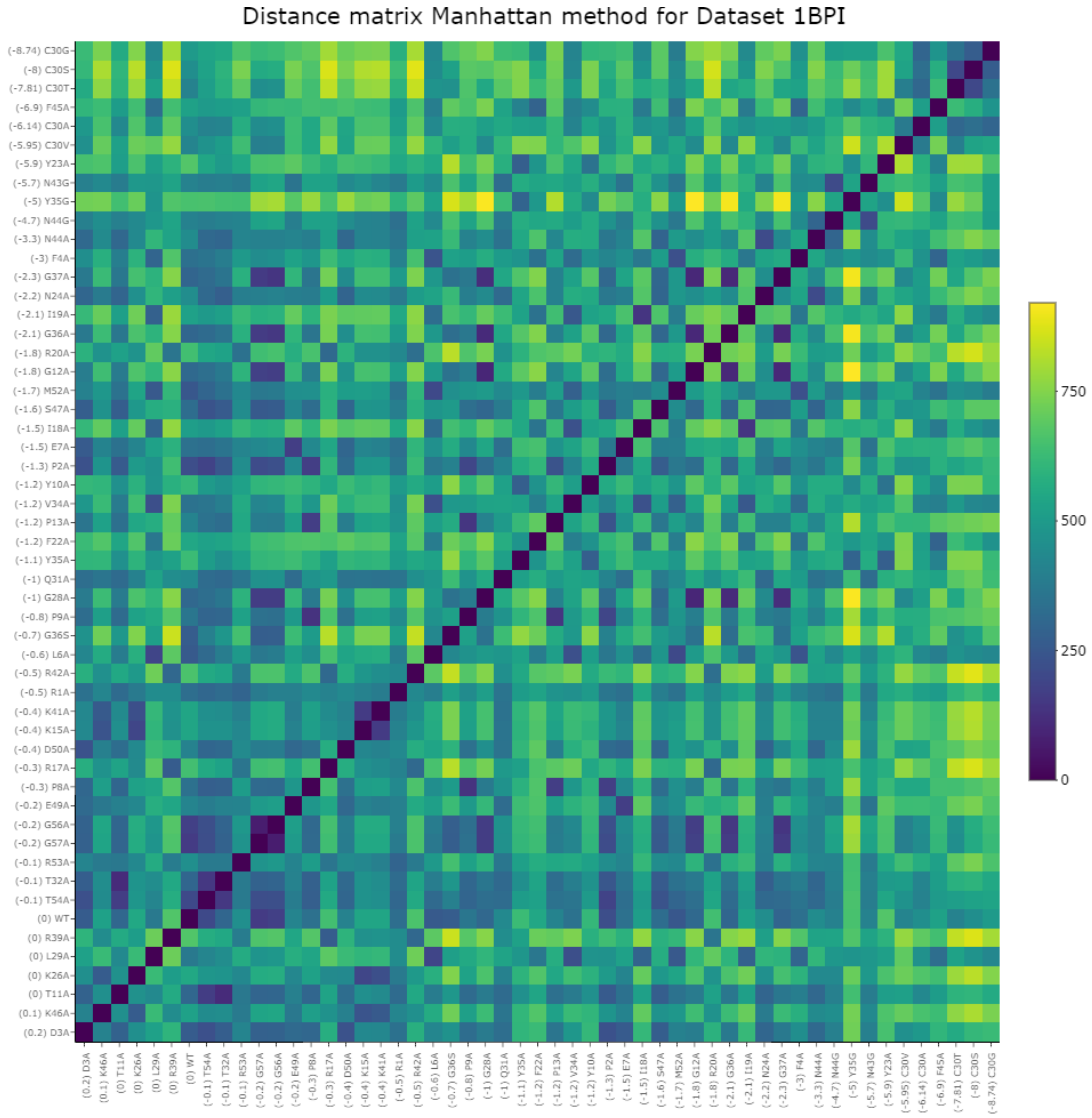


Figura A.14: Mapa de calor de la matriz de distancia *Manhattan* calculada para el conjunto de datos 1BPI.

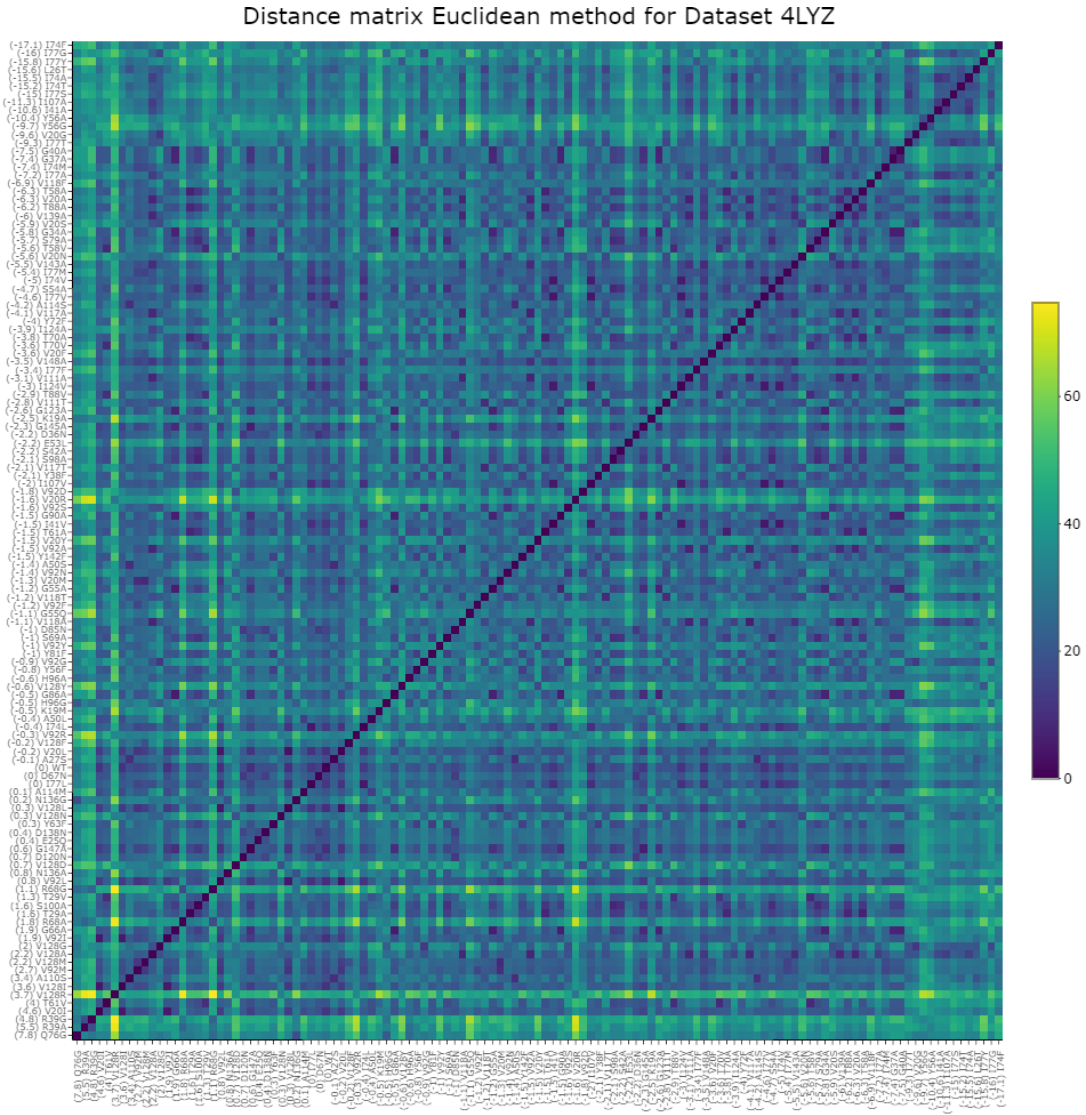


Figura A.15: Mapa de calor de la matriz de distancia euclidiana calculada para el conjunto de datos HLYZ.

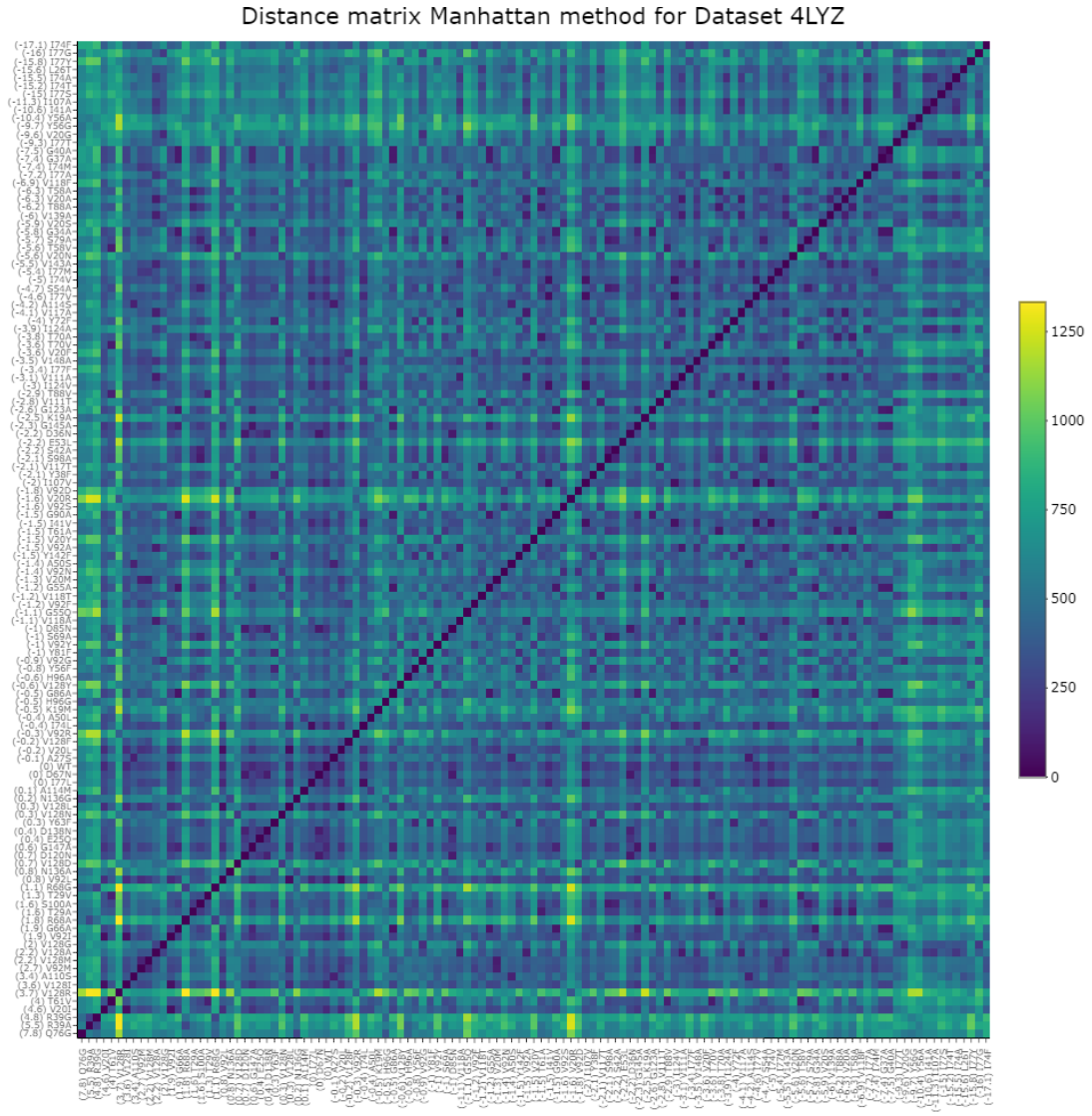


Figura A.16: Mapa de calor de la matriz de distancia *Manhattan* calculada para el conjunto de datos HLYZ.

B. Anexos Preparación de los datos

B.1. Selección de descriptores con algoritmo SURF

Cuadro B.1: Los 40 descriptores más relevantes seleccionados por SURF para el conjunto 1STN.

Ranking	Descriptor	Ranking	Descriptor
1	$AASA10K0$	21	$AASA9f$
2	$AASA10Mw$	22	$AASA6ac$
3	$AASA10f$	23	$AASA1ASAD$
4	$AASA1K0$	24	$AASA8TDSh$
5	$AASA10ASAD$	25	$AASA10s$
6	$AASA10V$	26	$AASA10DCph$
7	$AASA10V0$	27	$AASA6pHi$
8	$AASA6f$	28	$AASA3ASAD$
9	$AASA8P1$	29	$AASA10P1$
10	$AASA10DASA$	30	$AASA2ac$
11	$AASA8ASAD$	31	$AASA7ac$
12	$AASA8ac$	32	$AASA9E1$
13	$AASA2f$	33	$AASA2DASA$
14	$AASA10ac$	34	$AASA8DASA$
15	$AASA4ac$	35	$AASA9ac$
16	$AASA8V0$	36	$AASA10Rf$
17	$AASA10Ca$	37	$AASA6V0$
18	$AASA10TDSh$	38	$AASA6s$
19	$AASA6ASAD$	39	$AASA1DASA$
20	$AASA3ac$	40	$AASA5E1$

Cuadro B.2: Los 40 descriptores más relevantes seleccionados por SURF para el conjunto 4LYZ.

Ranking	**Descriptor**	**Ranking**	**Descriptor**
1	$AASA1Ns$	21	$AASA7Rf$
2	$AASA4N1$	22	$AASA2Ht$
3	$AASA1N1$	23	$AASA8TDSc$
4	$AASA2N1$	24	$AASA7Pb$
5	$AASA1Rf$	25	$AASA2DGc$
6	$AASA1Hnc$	26	$AASA3DCph$
7	$AASA8Hnc$	27	$AASA8Rf$
8	$AASA5Pb$	28	$AASA3Ht$
9	$AASA3ac$	29	$AASA2m$
10	$AASA2Pb$	30	$AASA1am$
11	$AASA8Pb$	31	$AASA5DG$
12	$AASA1pK$	32	$AASA2Ra$
13	$AASA7DGc$	33	$AASA8s$
14	$AASA4ac$	34	$AASA9Pb$
15	$AASA1m$	35	$AASA6Pb$
16	$AASA3Pb$	36	$AASA2DHc$
17	$AASA1Pb$	37	$AASA1DCph$
18	$AASA8DCph$	38	$AASA2Mw$
19	$AASA7ac$	39	$AASA8Ht$
20	$AASA2TDSc$	40	$AASA2am$

C. Anexos Modelado y Evaluación

C.1. Anexos Experimento 1

Cuadro C.1: Resumen de puntajes de prueba obtenidos en Experimento 1 para conjunto 1STN.

	Test R^2					**Test RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	-2.00	-1.02	2.75	0.58	-9.16	2.16	2.26	0.51	2.92	1.34
RIDGE	-0.05	0.27	0.91	0.60	-2.48	1.35	1.19	0.44	2.20	0.79
LASSO	-0.47	-0.01	1.27	0.21	-4.03	1.62	1.58	0.44	2.46	1.00
PLS	-0.37	0.17	1.42	0.65	-3.41	1.48	1.23	0.63	2.79	0.69
SVR	-0.13	0.04	0.60	0.41	-1.61	1.48	1.52	0.42	2.24	0.87

Cuadro C.2: Resumen de puntajes de entrenamiento obtenidos en Experimento 1 para conjunto 1STN.

	Train R^2					**Train RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	1.00	1.00	0.00	1.00	1.00	0.00	0.00	0.00	0.00	0.00
RIDGE	0.56	0.55	0.04	0.62	0.50	1.12	1.14	0.06	1.18	1.00
LASSO	0.23	0.18	0.21	0.83	0.10	1.45	1.54	0.28	1.57	0.66
PLS	0.56	0.53	0.08	0.78	0.48	1.11	1.16	0.13	1.21	0.78
SVR	0.57	0.57	0.10	0.76	0.43	1.09	1.07	0.14	1.28	0.81

Cuadro C.3: Resumen de puntajes de prueba obtenidos en Experimento 1 para conjunto 4LYZ.

	Test R^2					**Test RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	-5.76	-1.17	8.90	0.22	-26.67	2.35	2.41	0.82	3.60	1.18
RIDGE	-0.79	-0.08	1.50	0.52	-3.71	1.47	1.39	0.61	2.71	0.81
LASSO	-1.09	-0.21	1.78	-0.01	-4.50	1.66	1.47	0.68	2.86	0.75
PLS	-1.76	-0.07	2.93	0.74	-7.24	1.63	1.68	0.77	2.84	0.69
SVR	-0.57	0.04	1.30	0.51	-3.70	1.43	1.25	0.66	2.81	0.80

Cuadro C.4: Resumen de puntajes de entrenamiento obtenidos en Experimento 1 para conjunto 4LYZ.

	Train R^2					**Train RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	1.00	1.00	0.00	1.00	1.00	0.00	0.00	0.00	0.00	0.00
RIDGE	0.46	0.47	0.10	0.55	0.17	1.20	1.18	0.12	1.51	1.06
LASSO	0.09	0.00	0.27	0.86	0.00	1.52	1.65	0.33	1.71	0.62
PLS	0.54	0.50	0.12	0.86	0.44	1.09	1.15	0.18	1.21	0.62
SVR	0.41	0.37	0.10	0.69	0.35	1.25	1.29	0.16	1.36	0.80

Cuadro C.5: Resumen de puntajes de entrenamiento obtenidos en Experimento 1 para conjunto 1BPI.

	Train R^2					**Train RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	1.00	1.00	0.00	1.0	1.00	0.00	0.00	0.00	0.00	0.00
RIDGE	0.92	0.91	0.03	1.0	0.89	0.65	0.71	0.22	0.75	0.03
LASSO	0.93	0.92	0.04	1.0	0.90	0.55	0.68	0.29	0.70	0.00
PLS	0.85	0.85	0.07	1.0	0.75	0.89	0.91	0.29	1.14	0.16
SVR	0.88	0.89	0.02	0.9	0.85	0.81	0.81	0.03	0.87	0.77

C.2. Anexos Experimento 2

Cuadro C.6: Resumen de puntajes de prueba obtenidos en Experimento 2 para conjunto 1STN.

	Test R^2					**Test RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	-195.45	-1.41	566.20	0.82	-1803.79	6.42	3.00	7.49	22.88	0.47
RIDGE	-0.08	0.26	0.98	0.63	-2.69	1.37	1.18	0.50	2.45	0.74
LASSO	-0.53	0.27	2.06	0.72	-6.16	1.47	1.36	0.62	2.72	0.70
PLS	-0.20	0.18	1.17	0.68	-3.05	1.44	1.23	0.73	3.32	0.66
SVR	-0.16	-0.08	0.64	0.47	-1.75	1.50	1.52	0.45	2.34	0.81

Cuadro C.7: Resumen de puntajes de entrenamiento obtenidos en Experimento 2 para conjunto 1STN.

	Train R^2					**Train RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	1.00	1.00	0.00	1.00	1.00	0.00	0.00	0.00	0.00	0.00
RIDGE	0.52	0.51	0.05	0.62	0.46	1.17	1.20	0.07	1.24	1.00
LASSO	0.57	0.56	0.06	0.70	0.50	1.10	1.14	0.09	1.17	0.89
PLS	0.52	0.50	0.08	0.74	0.45	1.16	1.21	0.12	1.25	0.83
SVR	0.51	0.52	0.09	0.65	0.39	1.17	1.14	0.12	1.34	1.01

Cuadro C.8: Resumen de puntajes de prueba obtenidos en Experimento 2 para conjunto 4LYZ.

	Test R^2					**Test RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	-51.50	-5.65	76.96	-2.44	-236.92	5.95	4.41	4.42	14.28	2.58
RIDGE	-0.88	-0.06	1.64	0.48	-4.16	1.48	1.37	0.59	2.66	0.85
LASSO	-0.85	-0.23	1.45	0.14	-4.39	1.62	1.46	0.74	2.90	0.75
PLS	-1.18	0.04	2.24	0.69	-5.79	1.50	1.49	0.65	2.75	0.73
SVR	-0.60	-0.06	1.20	0.59	-3.03	1.46	1.24	0.72	3.03	0.75

Cuadro C.9: Resumen de puntajes de entrenamiento obtenidos en Experimento 2 para conjunto 4LYZ.

	Train R^2					**Train RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	1.00	1.00	0.00	1.00	1.00	0.00	0.00	0.00	0.00	0.00
RIDGE	0.38	0.39	0.08	0.48	0.17	1.28	1.28	0.10	1.51	1.13
LASSO	0.15	0.00	0.24	0.59	0.00	1.50	1.65	0.27	1.71	1.06
PLS	0.40	0.39	0.04	0.47	0.37	1.26	1.27	0.06	1.32	1.14
SVR	0.38	0.35	0.12	0.71	0.30	1.28	1.33	0.18	1.40	0.78

Cuadro C.10: Resumen de puntajes de prueba obtenidos en Experimento 2 para conjunto 1BPI.

	Test R^2					**Test RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	-163.40	-89.37	210.46	-3.64	-707.88	17.38	16.21	9.71	33.85	5.51
RIDGE	0.27	0.38	0.67	0.90	-1.34	1.27	1.12	0.46	2.11	0.73
LASSO	0.20	0.27	0.70	0.90	-1.25	1.31	1.18	0.46	2.11	0.72
PLS	-0.10	0.33	1.30	0.90	-3.27	1.40	1.30	0.44	2.13	0.82
SVR	0.29	0.48	0.90	0.92	-2.15	1.19	1.03	0.46	2.01	0.62

Cuadro C.11: Resumen de puntajes de entrenamiento obtenidos en Experimento 2 para conjunto 1BPI.

	Train R^2					**Train RMSE**				
	$\overline{x}$	Med.	σ	Max.	Min.	$\overline{x}$	Med.	σ	Max.	Min.
OLS	1.00	1.00	0.00	1.00	1.00	0.00	0.00	0.00	0.00	0.00
RIDGE	0.79	0.80	0.02	0.84	0.76	1.07	1.09	0.06	1.13	0.95
LASSO	0.77	0.77	0.02	0.81	0.73	1.13	1.15	0.06	1.18	1.02
PLS	0.75	0.76	0.03	0.79	0.67	1.19	1.20	0.08	1.31	1.03
SVR	0.81	0.81	0.02	0.84	0.78	1.04	1.05	0.06	1.12	0.94

C.3. Anexos Experimento 3

Cuadro C.12: R^2s de prueba en Experimento 3 para conjunto 1STN, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	-12.42	-11.27	10.02	-3.71	-28.10	-15.08	-3.97	-3.71	-11.27	-28.10
RIDGE	0.32	0.43	0.35	0.60	-0.28	0.53	0.43	0.33	0.60	-0.28
LASSO	0.23	0.29	0.43	0.56	-0.49	0.54	0.29	0.26	0.56	-0.49
PLS	0.27	0.35	0.35	0.56	-0.32	0.47	0.35	0.30	0.56	-0.32
SVR	0.39	0.38	0.23	0.69	0.11	0.53	0.25	0.38	0.69	0.11

Cuadro C.13: R^2s de prueba en Experimento 3 para conjunto 1STN, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	-3.68	-3.33	2.28	-0.78	-7.12	-1.94	-3.23	-0.78	-6.18	-7.12	-1.02	-2.05	-5.12	-3.43	-5.93
RIDGE	0.39	0.40	0.07	0.50	0.29	0.37	0.32	0.35	0.50	0.47	0.45	0.29	0.32	0.42	0.45
LASSO	0.25	0.23	0.14	0.45	-0.02	0.22	0.21	0.17	0.23	0.30	0.38	-0.02	0.17	0.44	0.45
PLS	0.28	0.34	0.16	0.45	-0.04	0.41	0.32	0.22	0.41	0.18	0.45	0.12	-0.04	0.38	0.37
SVR	0.39	0.44	0.18	0.62	0.05	0.29	0.05	0.38	0.55	0.50	0.50	0.48	0.14	0.40	0.62

Cuadro C.14: R^2s de prueba en Experimento 3 para conjunto 1STN, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	-37.83	-17.35	51.95	-1.58	-173.64	-40.27	-69.29	-5.83	-12.83	-1.58	-10.11	-34.16	-21.87	-173.64	-8.71
RIDGE	0.06	0.30	0.91	0.76	-2.33	0.64	0.07	-0.21	0.76	0.25	0.54	0.36	0.71	-2.33	-0.24
LASSO	-0.06	0.15	1.00	0.73	-2.63	0.59	0.14	-0.56	0.67	0.14	0.52	0.16	0.73	-2.63	-0.32
PLS	-0.22	0.19	1.16	0.76	-2.88	0.58	-0.20	-1.39	0.76	0.23	0.51	0.15	0.74	-2.88	-0.64
SVR	0.10	0.21	0.67	0.71	-1.49	0.71	0.11	-0.30	0.59	-0.12	0.61	0.31	0.70	-1.49	-0.08

Cuadro C.15: RMSEs de prueba en Experimento 3 para conjunto 1STN, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	5.27	5.23	1.47	7.04	3.68	5.23	3.68	3.98	6.42	7.04
RIDGE	1.25	1.25	0.25	1.50	0.89	0.89	1.25	1.50	1.16	1.47
LASSO	1.33	1.39	0.29	1.59	0.88	0.88	1.39	1.57	1.22	1.59
PLS	1.31	1.33	0.24	1.53	0.95	0.95	1.33	1.53	1.22	1.50
SVR	1.20	1.23	0.24	1.44	0.90	0.90	1.43	1.44	1.02	1.23

Cuadro C.16: RMSEs de prueba en Experimento 3 para conjunto 1STN, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	3.43	3.45	0.80	4.60	2.19	2.75	3.30	2.19	4.60	4.39	2.57	3.07	3.78	4.05	3.59
RIDGE	1.28	1.30	0.15	1.49	1.01	1.27	1.33	1.33	1.22	1.12	1.35	1.49	1.26	1.46	1.01
LASSO	1.42	1.43	0.19	1.78	1.02	1.42	1.43	1.50	1.50	1.29	1.43	1.78	1.39	1.44	1.02
PLS	1.39	1.37	0.17	1.65	1.08	1.23	1.33	1.45	1.32	1.40	1.34	1.65	1.55	1.52	1.08
SVR	1.27	1.29	0.21	1.57	0.85	1.35	1.57	1.29	1.15	1.08	1.28	1.26	1.42	1.49	0.85

Cuadro C.17: RMSEs de prueba en Experimento 3 para conjunto 1STN, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	6.73	7.06	2.33	9.90	3.42	7.18	9.90	3.84	7.01	3.42	4.92	8.66	9.89	7.12	5.32
RIDGE	1.24	1.13	0.41	1.90	0.67	0.67	1.14	1.62	0.92	1.85	1.01	1.17	1.11	0.98	1.90
LASSO	1.31	1.09	0.45	1.98	0.72	0.72	1.09	1.84	1.09	1.98	1.02	1.34	1.08	1.03	1.96
PLS	1.38	1.18	0.54	2.27	0.73	0.73	1.29	2.27	0.93	1.87	1.04	1.34	1.06	1.06	2.19
SVR	1.27	1.16	0.49	2.26	0.61	0.61	1.11	1.68	1.21	2.26	0.93	1.21	1.12	0.85	1.77

Cuadro C.18: R^2s de entrenamiento en Experimento 3 para conjunto 1STN, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	1.00	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
RIDGE	0.55	0.53	0.05	0.62	0.50	0.53	0.52	0.58	0.50	0.62
LASSO	0.62	0.62	0.05	0.67	0.56	0.58	0.62	0.65	0.56	0.67
PLS	0.57	0.56	0.05	0.64	0.50	0.55	0.56	0.58	0.50	0.64
SVR	0.61	0.62	0.03	0.64	0.56	0.59	0.64	0.63	0.56	0.62

Cuadro C.19: R^2s de entrenamiento en Experimento 3 para conjunto 1STN, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	1.00	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
RIDGE	0.56	0.57	0.06	0.66	0.45	0.45	0.58	0.66	0.48	0.56	0.56	0.63	0.57	0.58	0.53
LASSO	0.69	0.67	0.08	0.83	0.58	0.58	0.68	0.82	0.62	0.67	0.70	0.83	0.66	0.70	0.61
PLS	0.61	0.61	0.06	0.70	0.51	0.51	0.61	0.70	0.56	0.62	0.58	0.69	0.64	0.67	0.55
SVR	0.62	0.60	0.05	0.70	0.56	0.64	0.56	0.70	0.57	0.61	0.60	0.60	0.70	0.58	0.62

Cuadro C.20: R^2s de entrenamiento en Experimento 3 para conjunto 1STN, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	1.00	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
RIDGE	0.55	0.55	0.04	0.61	0.50	0.54	0.55	0.57	0.50	0.58	0.54	0.55	0.50	0.55	0.61
LASSO	0.60	0.59	0.04	0.66	0.53	0.59	0.59	0.64	0.55	0.63	0.59	0.61	0.53	0.59	0.66
PLS	0.56	0.55	0.05	0.65	0.50	0.53	0.58	0.65	0.50	0.59	0.54	0.55	0.50	0.55	0.64
SVR	0.61	0.60	0.04	0.67	0.54	0.59	0.58	0.67	0.63	0.61	0.58	0.64	0.54	0.59	0.62

Cuadro C.21: RMSEs de entrenamiento en Experimento 3 para conjunto 1STN, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
RIDGE	1.12	1.13	0.06	1.20	1.06	1.20	1.13	1.06	1.16	1.06
LASSO	1.04	1.01	0.07	1.13	0.97	1.13	1.01	0.97	1.09	0.97
PLS	1.10	1.09	0.07	1.18	1.02	1.18	1.09	1.06	1.16	1.02
SVR	1.05	1.06	0.06	1.12	0.98	1.12	0.98	1.00	1.09	1.06

Cuadro C.22: RMSEs de entrenamiento en Experimento 3 para conjunto 1STN, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
RIDGE	1.09	1.10	0.14	1.32	0.89	1.19	1.04	1.00	1.19	1.20	1.02	0.93	1.15	0.89	1.32
LASSO	0.92	0.96	0.18	1.21	0.64	1.04	0.91	0.73	1.01	1.04	0.84	0.64	1.02	0.75	1.21
PLS	1.03	1.03	0.15	1.29	0.78	1.12	1.01	0.93	1.09	1.11	1.00	0.85	1.06	0.78	1.29
SVR	1.02	0.97	0.10	1.19	0.88	0.96	1.07	0.94	1.08	1.13	0.98	0.97	0.97	0.88	1.19

Cuadro C.23: RMSEs de entrenamiento en Experimento 3a para conjunto 1STN, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
RIDGE	1.13	1.15	0.05	1.18	1.04	1.18	1.14	1.08	1.17	1.05	1.15	1.14	1.15	1.15	1.04
LASSO	1.06	1.09	0.06	1.12	0.97	1.12	1.09	0.99	1.11	0.98	1.09	1.06	1.12	1.10	0.97
PLS	1.11	1.15	0.07	1.19	0.98	1.19	1.12	0.98	1.17	1.05	1.15	1.14	1.15	1.16	1.00
SVR	1.05	1.07	0.06	1.11	0.95	1.11	1.11	0.95	1.00	1.01	1.10	1.02	1.10	1.10	1.03

Cuadro C.24: R^2s de prueba en Experimento 3 para conjunto 4LYZ, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	-1076.61	-182.69	2084.51	-36.38	-4802.95	-36.38	-117.49	-182.69	-243.55	-4802.95
RIDGE	0.04	0.03	0.16	0.26	-0.14	0.11	0.03	0.26	-0.14	-0.08
LASSO	-0.28	-0.21	0.16	-0.18	-0.56	-0.24	-0.21	-0.21	-0.18	-0.56
PLS	-0.03	-0.02	0.24	0.35	-0.30	-0.02	-0.01	0.35	-0.30	-0.17
SVR	0.26	0.40	0.28	0.44	-0.22	0.44	0.43	0.40	-0.22	0.27

Cuadro C.25: R^2s de prueba en Experimento 3 para conjunto 4LYZ, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	-3.59	-2.33	3.75	-1.02	-12.85	-1.25	-7.36	-1.28	-3.41	-12.85	-1.02	-2.86	-2.27	-1.22	-2.39
RIDGE	0.06	0.08	0.17	0.32	-0.33	0.06	-0.01	0.32	0.09	-0.33	0.00	0.14	0.13	0.12	0.07
LASSO	-0.18	-0.12	0.19	-0.00	-0.69	-0.12	-0.23	-0.13	-0.08	-0.69	-0.20	-0.05	-0.00	-0.11	-0.20
PLS	-0.05	-0.04	0.24	0.36	-0.63	-0.03	-0.05	0.36	-0.05	-0.63	-0.06	0.02	-0.10	-0.03	0.04
SVR	0.18	0.21	0.22	0.40	-0.35	0.15	0.11	0.36	0.27	-0.35	0.07	0.40	0.34	0.22	0.19

Cuadro C.26: R^2s de prueba en Experimento 3 para conjunto 4LYZ, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	-71.58	-34.68	100.90	-4.79	-340.24	-340.24	-12.43	-4.79	-16.68	-19.39	-95.72	-95.30	-7.22	-49.96	-74.01
RIDGE	-0.91	0.09	2.09	0.73	-6.21	-6.21	0.09	0.22	-1.28	0.14	0.73	-1.45	0.09	0.62	-2.04
LASSO	-1.87	-0.35	3.23	-0.01	-10.32	-10.32	-0.11	-0.11	-2.05	-0.13	-0.01	-1.30	-0.41	-0.29	-4.02
PLS	-1.26	-0.12	3.00	0.70	-9.04	-9.04	0.01	0.63	-1.32	0.23	0.70	-0.91	-0.25	0.68	-3.38
SVR	-0.53	0.18	1.71	0.73	-4.51	-4.51	0.47	0.72	-0.29	0.32	0.72	-2.37	0.05	0.73	-1.10

Cuadro C.27: RMSEs de prueba en Experimento 3 para conjunto 4LYZ, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	30.84	19.42	29.44	81.97	9.66	9.66	19.42	28.70	14.44	81.97
RIDGE	1.46	1.49	0.35	1.82	0.98	1.49	1.76	1.82	0.98	1.23
LASSO	1.71	1.76	0.50	2.33	1.00	1.76	1.96	2.33	1.00	1.48
PLS	1.49	1.59	0.31	1.79	1.05	1.59	1.79	1.71	1.05	1.28
SVR	1.24	1.18	0.26	1.64	1.01	1.18	1.34	1.64	1.02	1.01

Cuadro C.28: RMSEs de prueba en Experimento 3 para conjunto 4LYZ, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	2.97	2.93	0.51	3.77	2.14	2.93	3.23	2.14	3.77	3.50	2.89	2.93	3.20	2.84	2.23
RIDGE	1.50	1.52	0.36	2.03	1.08	1.90	1.12	1.17	1.71	1.08	2.03	1.38	1.65	1.79	1.17
LASSO	1.68	1.65	0.36	2.22	1.22	2.07	1.24	1.51	1.87	1.22	2.22	1.53	1.77	2.01	1.32
PLS	1.58	1.66	0.39	2.09	1.14	1.99	1.14	1.14	1.84	1.20	2.09	1.47	1.86	1.93	1.18
SVR	1.39	1.30	0.34	1.96	1.05	1.80	1.05	1.14	1.53	1.09	1.96	1.15	1.44	1.68	1.09

Cuadro C.29: RMSEs de prueba en Experimento 3 para conjunto 4LYZ, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	7.53	7.29	2.89	12.16	3.37	6.91	7.67	5.15	5.60	12.16	11.74	5.38	3.37	9.57	7.80
RIDGE	1.44	1.34	0.64	2.50	0.62	1.00	2.00	1.88	2.01	2.50	0.62	0.86	1.12	0.82	1.57
LASSO	1.79	1.77	0.64	2.86	0.83	1.26	2.20	2.25	2.32	2.86	1.20	0.83	1.39	1.52	2.02
PLS	1.43	1.31	0.62	2.36	0.65	1.18	2.08	1.30	2.03	2.36	0.65	0.76	1.32	0.76	1.88
SVR	1.21	1.14	0.47	2.23	0.63	0.88	1.53	1.13	1.51	2.23	0.63	1.01	1.15	0.70	1.30

Cuadro C.30: R^2s de entrenamiento en Experimento 3 para conjunto 4LYZ, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	1.00	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
RIDGE	0.53	0.57	0.08	0.59	0.40	0.59	0.57	0.40	0.54	0.57
LASSO	0.65	0.69	0.08	0.71	0.52	0.70	0.71	0.52	0.63	0.69
PLS	0.58	0.61	0.09	0.64	0.42	0.64	0.63	0.42	0.59	0.61
SVR	0.80	0.81	0.03	0.83	0.75	0.81	0.79	0.75	0.83	0.81

Cuadro C.31: R^2s de entrenamiento en Experimento 3 para conjunto 4LYZ, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	1.00	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
RIDGE	0.55	0.56	0.07	0.68	0.43	0.53	0.59	0.52	0.59	0.60	0.43	0.68	0.51	0.47	0.59
LASSO	0.13	0.16	0.12	0.32	0.00	0.00	0.21	0.16	0.15	0.32	0.00	0.26	0.00	0.00	0.23
PLS	0.68	0.66	0.07	0.80	0.60	0.73	0.66	0.61	0.80	0.69	0.61	0.80	0.60	0.65	0.66
SVR	0.74	0.78	0.12	0.85	0.49	0.62	0.84	0.82	0.72	0.85	0.49	0.81	0.75	0.64	0.83

Cuadro C.32: R^2s de entrenamiento en Experimento 3 para conjunto 4LYZ, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	0.93	0.93	0.04	0.98	0.87	0.95	0.92	0.98	0.97	0.87	0.94	0.91	0.92	0.88	0.98
RIDGE	0.50	0.53	0.12	0.61	0.20	0.55	0.55	0.20	0.61	0.43	0.50	0.53	0.53	0.50	0.59
LASSO	0.27	0.00	0.34	0.71	0.00	0.64	0.00	0.00	0.71	0.00	0.00	0.61	0.00	0.00	0.69
PLS	0.57	0.57	0.07	0.67	0.44	0.60	0.60	0.52	0.66	0.44	0.55	0.57	0.58	0.54	0.67
SVR	0.80	0.81	0.02	0.82	0.75	0.82	0.80	0.79	0.81	0.75	0.81	0.82	0.81	0.80	0.82

Cuadro C.33: RMSEs de entrenamiento en Experimento 3 para conjunto 4LYZ, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
RIDGE	1.11	1.13	0.06	1.18	1.03	1.06	1.03	1.14	1.18	1.13
LASSO	0.96	0.96	0.09	1.06	0.85	0.90	0.85	1.02	1.06	0.96
PLS	1.05	1.07	0.08	1.12	0.95	0.98	0.95	1.12	1.11	1.07
SVR	0.72	0.72	0.01	0.75	0.71	0.71	0.71	0.73	0.72	0.75

Cuadro C.34: RMSEs de entrenamiento en Experimento 3 para conjunto 4LYZ, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
RIDGE	1.03	1.03	0.21	1.28	0.71	0.76	1.26	1.25	0.91	1.28	0.71	1.00	1.05	0.88	1.23
LASSO	1.43	1.51	0.27	1.73	0.94	1.12	1.73	1.64	1.31	1.68	0.94	1.52	1.49	1.21	1.68
PLS	0.88	0.87	0.24	1.14	0.58	0.58	1.14	1.12	0.63	1.14	0.59	0.80	0.95	0.71	1.11
SVR	0.75	0.76	0.04	0.79	0.67	0.69	0.79	0.76	0.75	0.79	0.67	0.77	0.75	0.73	0.78

Cuadro C.35: RMSEs de entrenamiento en Experimento 3a para conjunto 4LYZ, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	0.41	0.42	0.12	0.57	0.20	0.38	0.43	0.20	0.30	0.52	0.41	0.52	0.48	0.57	0.24
RIDGE	1.14	1.14	0.10	1.39	1.04	1.14	1.05	1.39	1.04	1.09	1.18	1.16	1.14	1.17	1.09
LASSO	1.35	1.50	0.32	1.68	0.89	1.03	1.56	1.55	0.89	1.44	1.68	1.06	1.66	1.65	0.94
PLS	1.06	1.08	0.06	1.13	0.97	1.09	0.99	1.07	0.97	1.08	1.13	1.12	1.08	1.12	0.97
SVR	0.72	0.72	0.01	0.74	0.70	0.73	0.70	0.72	0.72	0.72	0.72	0.72	0.72	0.74	0.72

Cuadro C.36: R^2s de prueba en Experimento 3 para conjunto 1BPI, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	-14.09	-11.83	11.54	-1.93	-29.18	-11.83	-5.04	-29.18	-22.46	-1.93
RIDGE	0.23	0.74	0.91	0.79	-1.35	0.79	0.74	0.75	-1.35	0.22
LASSO	0.18	0.72	0.99	0.82	-1.52	0.82	0.75	0.72	-1.52	0.15
PLS	0.20	0.70	0.96	0.86	-1.44	0.86	0.74	0.70	-1.44	0.14
SVR	0.42	0.75	0.66	0.82	-0.74	0.82	0.75	0.76	-0.74	0.51

Cuadro C.37: R^2s de prueba en Experimento 3 para conjunto 1BPI, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	-0.45	0.02	1.10	0.28	-3.06	0.28	-1.69	0.16	0.20	-3.06	0.26	-0.12	0.17	-0.55	-0.13
RIDGE	0.46	0.49	0.14	0.59	0.13	0.42	0.13	0.56	0.46	0.49	0.59	0.49	0.59	0.52	0.30
LASSO	0.35	0.46	0.26	0.62	-0.12	0.53	0.06	0.44	0.48	0.32	0.57	0.04	0.56	0.62	-0.12
PLS	0.46	0.50	0.17	0.61	0.06	0.61	0.06	0.60	0.47	0.46	0.60	0.41	0.54	0.55	0.31
SVR	0.62	0.61	0.11	0.80	0.40	0.63	0.40	0.66	0.59	0.58	0.80	0.59	0.74	0.66	0.51

Cuadro C.38: R^2s de prueba en Experimento 3 para conjunto 1BPI, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	-6.36	-1.92	14.77	0.49	-47.99	-1.80	-47.99	0.48	-0.54	0.45	-2.05	-3.71	-5.50	0.49	-3.39
RIDGE	-0.21	0.12	1.03	0.82	-2.06	0.01	-2.06	0.79	0.11	0.82	0.42	-1.21	-1.62	0.14	0.50
LASSO	-0.20	0.18	1.01	0.88	-2.07	0.28	-2.07	0.88	0.16	0.79	0.37	-1.53	-1.09	-0.01	0.21
PLS	-0.24	0.04	1.05	0.92	-2.21	-0.02	-1.09	0.92	-0.01	0.84	0.30	-1.60	-2.21	0.09	0.38
SVR	0.18	0.61	0.98	0.88	-1.92	0.88	-1.92	0.80	0.57	0.83	0.80	-1.30	0.24	0.24	0.66

Cuadro C.39: RMSEs de prueba en Experimento 3 para conjunto 1BPI, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	7.72	6.86	4.71	15.68	3.38	7.33	6.86	15.68	5.37	3.38
RIDGE	1.45	1.44	0.32	1.74	0.94	0.94	1.42	1.44	1.70	1.74
LASSO	1.47	1.52	0.38	1.82	0.86	0.86	1.40	1.52	1.76	1.82
PLS	1.46	1.57	0.43	1.83	0.75	0.75	1.43	1.57	1.73	1.83
SVR	1.30	1.39	0.24	1.46	0.86	0.86	1.39	1.39	1.46	1.38

Cuadro C.40: RMSEs de prueba en Experimento 3 para conjunto 1BPI, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	2.63	2.31	0.83	4.66	1.93	2.35	2.86	1.93	2.28	4.66	2.10	2.58	2.10	3.35	2.09
RIDGE	1.69	1.65	0.21	2.11	1.39	2.11	1.63	1.39	1.87	1.65	1.55	1.75	1.48	1.86	1.64
LASSO	1.82	1.77	0.27	2.40	1.53	1.89	1.69	1.57	1.85	1.91	1.60	2.40	1.53	1.67	2.08
PLS	1.67	1.70	0.17	1.87	1.32	1.73	1.69	1.32	1.86	1.71	1.55	1.87	1.56	1.81	1.63
SVR	1.42	1.44	0.21	1.67	1.09	1.67	1.35	1.22	1.64	1.50	1.09	1.57	1.17	1.57	1.37

Cuadro C.41: RMSEs de prueba en Experimento 3 para conjunto 1BPI, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	2.90	2.75	0.89	4.38	1.53	4.38	3.35	2.38	3.01	2.31	3.74	2.49	2.07	1.53	3.75
RIDGE	1.65	1.57	0.53	2.61	0.84	2.61	0.84	1.51	2.29	1.33	1.63	1.71	1.32	1.98	1.27
LASSO	1.63	1.64	0.49	2.23	0.84	2.22	0.84	1.15	2.23	1.41	1.69	1.83	1.18	2.15	1.59
PLS	1.65	1.62	0.62	2.64	0.69	2.64	0.69	0.95	2.45	1.23	1.79	1.85	1.45	2.04	1.41
SVR	1.24	1.16	0.41	1.87	0.71	0.90	0.82	1.50	1.59	1.27	0.95	1.74	0.71	1.87	1.05

Cuadro C.42: R^2s de entrenamiento en Experimento 3 para conjunto 1BPI, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	0.99	0.99	0.01	1.00	0.98	1.00	0.99	0.99	1.00	0.98
RIDGE	0.83	0.83	0.03	0.87	0.80	0.83	0.81	0.80	0.87	0.86
LASSO	0.79	0.76	0.04	0.84	0.75	0.76	0.76	0.75	0.84	0.82
PLS	0.77	0.75	0.05	0.82	0.72	0.75	0.72	0.73	0.81	0.82
SVR	0.88	0.89	0.02	0.91	0.86	0.89	0.87	0.86	0.91	0.89

Cuadro C.43: R^2s de entrenamiento en Experimento 3 para conjunto 1BPI, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	1.00	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
RIDGE	0.69	0.71	0.13	0.85	0.36	0.36	0.85	0.70	0.72	0.74	0.68	0.70	0.75	0.59	0.77
LASSO	0.85	0.87	0.07	0.93	0.69	0.69	0.93	0.89	0.82	0.87	0.83	0.89	0.87	0.78	0.89
PLS	0.75	0.76	0.11	0.89	0.48	0.48	0.89	0.76	0.76	0.79	0.73	0.76	0.82	0.66	0.81
SVR	0.87	0.89	0.04	0.92	0.78	0.78	0.92	0.89	0.87	0.88	0.89	0.89	0.88	0.84	0.91

Cuadro C.44: R^2s de entrenamiento en Experimento 3 para conjunto 1BPI, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	0.98	0.98	0.01	0.99	0.97	0.99	0.98	0.97	0.97	0.97	0.98	0.98	0.98	0.97	0.97
RIDGE	0.78	0.79	0.11	0.98	0.63	0.98	0.83	0.63	0.72	0.67	0.84	0.75	0.84	0.85	0.72
LASSO	0.79	0.79	0.07	0.97	0.70	0.97	0.77	0.70	0.79	0.75	0.79	0.81	0.78	0.80	0.77
PLS	0.76	0.76	0.08	0.97	0.65	0.97	0.76	0.65	0.73	0.69	0.78	0.75	0.77	0.80	0.72
SVR	0.89	0.88	0.02	0.91	0.86	0.88	0.90	0.86	0.88	0.87	0.89	0.91	0.88	0.89	0.89

Cuadro C.45: RMSEs de entrenamiento en Experimento 3 para conjunto 1BPI, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	0.15	0.19	0.12	0.30	0.01	0.01	0.19	0.20	0.06	0.30
RIDGE	0.96	0.97	0.04	1.02	0.92	1.02	0.97	0.97	0.92	0.92
LASSO	1.09	1.09	0.07	1.20	1.02	1.20	1.09	1.09	1.02	1.04
PLS	1.13	1.12	0.07	1.23	1.05	1.23	1.18	1.12	1.09	1.05
SVR	0.80	0.80	0.02	0.83	0.78	0.83	0.80	0.80	0.78	0.81

Cuadro C.46: RMSEs de entrenamiento en Experimento 3 para conjunto 1BPI, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
RIDGE	1.26	1.26	0.11	1.39	1.06	1.39	1.06	1.39	1.12	1.25	1.31	1.26	1.22	1.25	1.29
LASSO	0.87	0.89	0.08	0.97	0.73	0.97	0.73	0.84	0.89	0.88	0.96	0.77	0.87	0.93	0.91
PLS	1.12	1.14	0.11	1.26	0.92	1.26	0.92	1.25	1.03	1.12	1.19	1.13	1.02	1.14	1.17
SVR	0.80	0.80	0.03	0.85	0.76	0.81	0.79	0.85	0.76	0.83	0.76	0.77	0.83	0.79	0.83

Cuadro C.47: RMSEs de entrenamiento en Experimento 3a para conjunto 1BPI, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	0.36	0.37	0.06	0.44	0.23	0.23	0.34	0.36	0.41	0.41	0.37	0.38	0.32	0.44	0.39
RIDGE	1.06	1.12	0.31	1.32	0.29	0.29	1.01	1.32	1.23	1.30	0.96	1.24	0.98	0.93	1.29
LASSO	1.06	1.12	0.22	1.19	0.44	0.44	1.19	1.18	1.07	1.14	1.10	1.07	1.14	1.06	1.16
PLS	1.13	1.21	0.25	1.29	0.43	0.43	1.21	1.29	1.22	1.27	1.12	1.22	1.17	1.08	1.27
SVR	0.80	0.81	0.03	0.83	0.73	0.80	0.77	0.81	0.81	0.81	0.81	0.73	0.83	0.78	0.81

Cuadro C.48: R^2s de prueba en Experimento 3 para conjunto HLYZ, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	0.11	-0.03	0.23	0.40	-0.08	0.32	0.40	-0.07	-0.08	-0.03
RIDGE	0.28	0.24	0.12	0.42	0.13	0.40	0.42	0.13	0.23	0.24
LASSO	0.28	0.20	0.16	0.46	0.13	0.46	0.45	0.13	0.20	0.14
PLS	0.29	0.33	0.24	0.60	-0.05	0.33	0.60	0.37	0.21	-0.05
SVR	0.48	0.49	0.13	0.67	0.35	0.67	0.49	0.35	0.51	0.38

Cuadro C.49: R^2s de prueba en Experimento 3 para conjunto HLYZ, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	-0.78	-0.55	0.64	-0.15	-1.96	-0.58	-0.51	-0.31	-0.15	-0.79	-1.08	-0.20	-1.96	-1.79	-0.41
RIDGE	0.09	0.06	0.23	0.38	-0.41	0.05	0.16	0.38	0.23	0.04	-0.02	-0.02	-0.41	0.38	0.07
LASSO	0.10	0.10	0.22	0.43	-0.31	0.00	0.16	0.43	0.21	0.11	-0.01	-0.07	-0.31	0.38	0.08
PLS	0.03	-0.00	0.25	0.42	-0.36	-0.21	0.06	0.41	0.14	0.02	-0.08	-0.02	-0.36	0.42	-0.07
SVR	0.34	0.35	0.10	0.46	0.10	0.30	0.35	0.46	0.40	0.30	0.40	0.38	0.34	0.36	0.10

Cuadro C.50: R^2s de prueba en Experimento 3 para conjunto HLYZ, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	0.12	0.20	0.33	0.50	-0.51	0.42	0.29	0.46	0.50	-0.51	0.17	-0.01	-0.17	-0.19	0.23
RIDGE	0.23	0.29	0.27	0.51	-0.17	0.42	0.29	0.48	0.50	-0.08	0.28	0.23	-0.17	-0.14	0.51
LASSO	0.18	0.32	0.44	0.64	-0.56	0.34	0.45	0.53	0.64	-0.46	0.31	0.28	-0.56	-0.27	0.52
PLS	0.19	0.22	0.27	0.56	-0.30	0.43	0.16	0.56	0.24	0.05	0.34	0.20	-0.17	-0.30	0.44
SVR	0.42	0.47	0.27	0.76	-0.06	0.62	0.49	0.65	0.76	0.21	0.23	0.67	-0.06	0.13	0.46

Cuadro C.51: RMSEs de prueba en Experimento 3 para conjunto HLYZ, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	4.24	4.11	0.55	4.90	3.49	4.11	4.04	4.90	4.66	3.49
RIDGE	3.84	3.92	0.52	4.42	2.99	3.88	3.99	4.42	3.92	2.99
LASSO	3.83	3.87	0.45	4.41	3.18	3.68	3.87	4.41	4.00	3.18
PLS	3.73	3.75	0.32	4.07	3.30	4.07	3.30	3.75	3.98	3.53
SVR	3.25	3.13	0.50	3.80	2.72	2.86	3.72	3.80	3.13	2.72

Cuadro C.52: RMSEs de prueba en Experimento 3 para conjunto HLYZ, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	6.08	6.03	0.98	8.06	4.85	6.55	4.85	5.33	5.05	6.18	6.85	5.88	6.63	8.06	5.37
RIDGE	4.40	4.44	0.61	5.43	3.61	5.07	3.61	3.65	4.15	4.53	4.80	5.43	4.57	3.79	4.35
LASSO	4.37	4.34	0.66	5.55	3.50	5.20	3.62	3.50	4.19	4.36	4.78	5.55	4.41	3.79	4.33
PLS	4.53	4.53	0.72	5.74	3.58	5.74	3.82	3.58	4.37	4.57	4.94	5.43	4.49	3.66	4.67
SVR	3.77	3.78	0.44	4.35	3.13	4.35	3.19	3.42	3.65	3.88	3.68	4.24	3.13	3.87	4.29

Cuadro C.53: RMSEs de prueba en Experimento 3 para conjunto HLYZ, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	3.98	3.53	0.92	5.53	3.06	4.25	3.29	3.19	4.82	5.53	3.43	5.28	3.37	3.63	3.06
RIDGE	3.73	3.46	0.81	4.85	2.44	4.24	3.28	3.12	4.85	4.68	3.19	4.62	3.36	3.55	2.44
LASSO	3.75	3.82	0.92	5.43	2.42	4.54	2.88	2.95	4.08	5.43	3.12	4.45	3.88	3.76	2.42
PLS	3.85	3.68	1.01	5.97	2.60	4.22	3.57	2.86	5.97	4.38	3.05	4.71	3.36	3.80	2.60
SVR	3.13	3.16	0.43	3.98	2.56	3.44	2.79	2.56	3.34	3.98	3.30	3.04	3.20	3.11	2.56

Cuadro C.54: R^2s de entrenamiento en Experimento 3 para conjunto HLYZ, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	0.73	0.74	0.03	0.75	0.67	0.72	0.67	0.75	0.74	0.74
RIDGE	0.59	0.61	0.04	0.63	0.53	0.59	0.53	0.63	0.61	0.61
LASSO	0.60	0.60	0.04	0.63	0.53	0.60	0.53	0.60	0.63	0.63
PLS	0.70	0.71	0.03	0.72	0.65	0.71	0.65	0.71	0.72	0.72
SVR	0.86	0.85	0.02	0.88	0.85	0.85	0.85	0.88	0.85	0.87

Cuadro C.55: R^2s de entrenamiento en Experimento 3 para conjunto HLYZ, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	0.85	0.84	0.03	0.90	0.80	0.82	0.84	0.80	0.87	0.85	0.83	0.84	0.83	0.88	0.90
RIDGE	0.63	0.66	0.06	0.71	0.53	0.56	0.67	0.54	0.68	0.65	0.67	0.64	0.68	0.53	0.71
LASSO	0.67	0.69	0.06	0.73	0.57	0.57	0.72	0.61	0.70	0.72	0.68	0.63	0.72	0.66	0.73
PLS	0.57	0.57	0.08	0.69	0.43	0.56	0.58	0.43	0.63	0.56	0.64	0.56	0.63	0.44	0.69
SVR	0.90	0.91	0.02	0.93	0.86	0.87	0.93	0.91	0.91	0.91	0.90	0.86	0.92	0.90	0.91

Cuadro C.56: R^2s de entrenamiento en Experimento 3 para conjunto HLYZ, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	0.71	0.71	0.02	0.73	0.67	0.71	0.71	0.70	0.67	0.73	0.72	0.72	0.71	0.72	0.71
RIDGE	0.67	0.70	0.06	0.70	0.54	0.69	0.69	0.58	0.54	0.70	0.70	0.70	0.70	0.70	0.69
LASSO	0.61	0.59	0.04	0.71	0.56	0.59	0.59	0.56	0.66	0.59	0.71	0.60	0.60	0.61	0.59
PLS	0.64	0.68	0.13	0.70	0.28	0.68	0.69	0.65	0.28	0.69	0.70	0.70	0.66	0.68	0.66
SVR	0.84	0.85	0.04	0.87	0.74	0.84	0.86	0.85	0.84	0.82	0.87	0.83	0.87	0.74	0.87

Cuadro C.57: RMSEs de entrenamiento en Experimento 3 para conjunto HLYZ, usando 40 descriptores y 5-Fold CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5
OLS	2.45	2.42	0.10	2.58	2.31	2.42	2.58	2.31	2.40	2.52
RIDGE	2.98	2.94	0.10	3.08	2.85	2.94	3.07	2.85	2.93	3.08
LASSO	2.96	2.95	0.09	3.10	2.88	2.90	3.10	2.95	2.88	2.99
PLS	2.56	2.51	0.07	2.66	2.50	2.50	2.66	2.51	2.50	2.62
SVR	1.75	1.77	0.09	1.81	1.60	1.81	1.74	1.60	1.81	1.77

Cuadro C.58: RMSEs de entrenamiento en Experimento 3 para conjunto HLYZ, usando 40 descriptores y 5x2 CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	1.81	1.77	0.24	2.19	1.55	1.66	2.06	2.09	1.68	1.86	1.91	1.56	2.19	1.55	1.55
RIDGE	2.79	2.75	0.27	3.18	2.33	2.60	2.99	3.18	2.64	2.82	2.67	2.33	3.04	3.09	2.59
LASSO	2.63	2.61	0.18	2.93	2.34	2.60	2.74	2.93	2.54	2.53	2.61	2.34	2.85	2.62	2.48
PLS	3.02	2.98	0.37	3.56	2.56	2.63	3.39	3.56	2.81	3.15	2.77	2.56	3.28	3.39	2.69
SVR	1.44	1.43	0.02	1.47	1.41	1.43	1.43	1.41	1.43	1.42	1.43	1.42	1.47	1.44	1.47

Cuadro C.59: RMSEs de entrenamiento en Experimento 3a para conjunto HLYZ, usando 40 descriptores y Nested CV.

Method	$\overline{x}$	Med.	σ	Max.	Min.	I1	I2	I3	I4	I5	I6	I7	I8	I9	I10
OLS	2.53	2.54	0.06	2.59	2.43	2.46	2.57	2.59	2.53	2.43	2.54	2.44	2.57	2.54	2.59
RIDGE	2.68	2.62	0.19	3.06	2.52	2.53	2.63	3.06	2.98	2.55	2.62	2.52	2.63	2.60	2.68
LASSO	2.92	2.98	0.20	3.12	2.56	2.91	3.06	3.12	2.56	2.98	2.58	2.94	3.01	2.98	3.08
PLS	2.78	2.68	0.34	3.71	2.52	2.57	2.66	2.78	3.71	2.61	2.63	2.52	2.77	2.71	2.79
SVR	1.87	1.80	0.21	2.42	1.71	1.80	1.79	1.83	1.76	1.96	1.71	1.90	1.73	2.42	1.76