para fin retornar

Universidad de Santiago de Chile

Facultad de Ciencia

Departamento de Matemática y Ciencia de la computación

Ingeniería Estadística · Examen de Grado

Clasificación de objetos astronómicos utilizando características derivadas de path signature

Catalina Piutrín

Profesor Guía:
Felipe Elorrieta

Comisión:
Andrés Iturriaga
Lucas Osses

Lunes 06 de Abril del 2026

A mi mamá,

por su amor infinito, su apoyo incondicional y
por sostenerme en cada etapa de mi vida.

Te amo con todo mi corazón.

Índice

1 Introducción

Desde tiempos antiguos, el ser humano ha intentado comprender los fenómenos que observa en el cielo. A lo largo de la historia, figuras como Copérnico, Kepler, Galileo, Newton y Einstein han aportado modelos fundamentales que han transformado nuestra comprensión del universo. En la actualidad, la astronomía ha entrado en la era de los datos, impulsada por los surveys, que generan millones de observaciones disponibles para el análisis automático.

En la última generación de astronomía, los datos se reciben de forma secuencial y son procesados por distintos brokers distribuidos a lo largo del mundo. Uno de ellos, ALeRCE, actualmente trabaja con alertas provenientes del Zwicky Transient Facility (ZTF), procesando alrededor de 200.000 eventos por noche. Este sistema ya se encuentra preparándose para el próximo gran desafío: el Observatorio Vera C. Rubin (LSST), que generará aproximadamente 10 millones de alertas cada noche. Dada esta magnitud en la generación de información astronómica en tiempo real, resulta fundamental contar con métodos de clasificación automática que permitan identificar, filtrar y analizar eficientemente los distintos tipos de objetos transitorios y variables, facilitando así la toma de decisiones científicas.

Este proyecto de investigación se enfoca en mejorar los modelos actuales de clasificación automática de objetos astronómicos. Tradicionalmente, la clasificación se ha basado en características extraídas de las curvas de luz, como el período o la amplitud. Sin embargo, los modelos existentes, como el clasificador de curvas de luz del sistema ALeRCE, enfrentan limitaciones al clasificar objetos con variabilidad más compleja o menos regular [1], como los núcleos activos de galaxias (AGN), blazares y objetos transitorios.

Para abordar este desafío, se propone incorporar el uso de path signature, una técnica que permite capturar relaciones no lineales y multivariadas en series temporales. El objetivo general es desarrollar un sistema de clasificación automática de objetos astronómicos basado en características derivadas de

path signature. Se espera que esta metodología contribuya a la creación de herramientas de clasificación automatizadas que operen en tiempo real, optimizando el análisis de los grandes volúmenes de datos generados por los surveys astronómicos modernos.

1.1. Formulación del problema

Los modelos actuales de clasificación automática presentan limitaciones cuando deben identificar objetos astronómicos cuya variabilidad es compleja. Esto ocurre porque las técnicas tradicionales no logran representar bien la estructura no lineal de sus curvas de luz, lo que reduce la precisión de los resultados. Frente a este desafío, el problema que se plantea es cómo mejorar la capacidad de clasificación automática en estos casos, incorporando path signature como una herramienta que permita a los modelos diferenciar de manera más efectiva los patrones de variabilidad con estructuras temporales irregulares y difíciles de predecir.

1.2. Objetivos de la investigación

1.2.1. Objetivo general

Desarrollar un sistema de clasificación automática de objetos astronómicos basado en características derivadas de path signature.

1.2.2. Objetivos específicos

- ⋆ Investigar la teoría de path signature, enfocándose en sus fundamentos teóricos y su aplicación en el análisis de series temporales.
- ⋆ Estudiar sobre las técnicas de clasificación usadas en curvas de luz de objetos astronómicos para extraer características no lineales que representen la variabilidad de estos.
- ⋆ Diseñar un conjunto de características basadas en path signature y llevar a cabo un análisis descriptivo de su comportamiento.
- ⋆ Implementar modelos de clasificación utilizando aprendizaje automático para clasificar los objetos astronómicos, utilizando las características extraídas mediante path signature.
- ⋆ Determinar la importancia de las características derivadas de path signature en la clasificación astronómica.
- ⋆ Calcular el rendimiento en la clasificación basada en path signature y comparar con los métodos de clasificación actuales.
- ⋆ Explorar la capacidad del clasificador para manejar el desbalance de clases en el conjunto de datos astronómicos.

1.3. Metodología

Figura 1.1: Metodología para clasificar curvas de luz astronómicas.

Adquisición y preprocesamiento de datos astronómicos

En este trabajo se plantea utilizar curvas de luz como representación principal de la información temporal de los objetos astronómicos. Estas curvas corresponden a secuencias de mediciones de brillo tomadas en distintos instantes de tiempo, típicamente en una o más bandas fotométricas.

Se considera como fuente de datos a surveys astronómicos modernos como ZTF o VVV, y eventualmente datos simulados en el marco de futuros proyectos como LSST. Estas curvas de luz pueden incluir observaciones en múltiples bandas, por ejemplo, g y r en ZTF, o ZYJHKs en VVV. En todos los casos, se prevé realizar un preprocesamiento que incluya la limpieza de datos atípicos, la imputación o eliminación de observaciones faltantes y la corrección de artefactos instrumentales.

Cada curva de luz será representada inicialmente como una secuencia de vectores (t_i, y_i) en el caso de una sola banda, o como $(t_i, y_{i,1}, y_{i,2}, \dots)$ en el caso multibanda, formando así puntos en un espacio de dimensión d correspondiente al número de bandas más uno (si se incluye el tiempo explícitamente).

Representación de datos como path (camino)

El método signature opera sobre caminos o paths en un espacio euclidiano multidimensional. Una curva de luz, siendo una secuencia ordenada de puntos con valores de tiempo y brillo, debe convertirse en una trayectoria continua (un camino) para poder aplicar el método signature.

Se evaluarán distintas formas de construir estos caminos a partir de los datos discretos. Una opción básica será la interpolación lineal entre puntos consecutivos ordenador por tiempo. También se considera

aplicar transformaciones adicionales como la suma acumulada, que puede facilitar la interpretación geométrica de las iteraciones, o la incrustación lead-lag, la cual duplica las dimensiones del path y permite capturar la información sobre la variación cuadrática del proceso.

La elección de la técnica de representación se hará en función de su capacidad para preservar o resaltar patrones relevantes en las curvas de luz, y podrá ser ajustada durante la validación del modelo.

Cálculo del path signature

Para cada curva de luz representada como un camino $\mathbb{R}^d$, se calculará su signature truncada hasta un nivel L, la cual se utilizará como vector de características para la clasificación. Esta representación permite extraer información secuencial relevante, preservando la estructura temporal y la magnitud de las variaciones observadas en la serie.

El cálculo se realizará a partir de los puntos discretos del camino. El resultado será un vector de dimensión finita, donde cada componente se interpreta como una característica escalar. El nivel de truncamiento de L se seleccionará de acuerdo con el balance entre capacidad descriptiva y complejidad computacional del conjunto de características resultante.

Formación del conjunto de características

Las características derivadas del path signature constituyen un vector numérico fijo por curva de luz, que resume su estructura temporal y geométrica. Estas características serán utilizadas como entrada para los modelos de clasificación.

Selección de características

Dado que el conjunto total de características (especialmente si se utilizan niveles de truncamiento altos en la signature o se combinan muchos tipos de características) puede ser de muy alta dimensión, por ende, se propone la selección de características. Esto ayuda a identificar las características más discriminatorias, reducir el ruido, mejorar la eficiencia computacional y potencialmente evitar el sobreajuste.

Entrenamiento del clasificador

Con las características extraídas y un conjunto de datos etiquetado, se entrenará un modelo de aprendizaje automático. Algoritmos comunes son Random Forest y AdaBoost, siendo AdaBoost ligeramente más estable y preciso en algunos casos. También se evaluarán otros métodos como LASSO.

Se explorará también la posibilidad de realizar clasificación jerárquica, en caso de que las clases tengan estructura multinivel, así como estrategias para manejar el desbalance de clases, por ejemplo, Balanced Random Forest.

Evaluación y validación

Para evaluar el desempeño del modelo, se utilizarán métricas estándar como precisión, recall, F1-score, matriz de confusión y área bajo la curva ROC (AUC).

Se empleará validación cruzada para estimar el rendimiento general, ajustar hiperparámetros y comparar distintas configuraciones del modelo, incluyendo la representación del path y el nivel de truncamiento del signature. Se contará con un conjunto de prueba que se utilizará únicamente para la evaluación final del modelo, sin participar en las etapas de validación o ajuste.

Optimización

Se contempla la posibilidad de realizar ajustes en distintas etapas, especialmente si los resultados iniciales no alcanzan el rendimiento esperado. Algunas de las estrategias consideradas incluyen:

- Probar diferentes formas de representar las curvas de luz como paths.
- Modificar el nivel de truncamiento L del path signature para capturar más o menos información.
- Incorporar nuevas características adicionales que complementen las derivadas del signature.
- Comparar el desempeño de distintos algoritmos de clasificación.
- Ajustar el preprocesamiento o el muestreo para mejorar la calidad de las curvas de luz

1.4. Resultados esperados

Se espera encontrar características derivadas del path signature que permitan separar de manera efectiva distintos tipos de objetos astronómicos, en particular aquellos con comportamientos complejos en sus curvas de luz. Estas características, al capturar la estructura dinámica y temporal de las señales, podrán ser útiles no solo para mejorar la clasificación de objetos en sistemas existentes como el algoritmo ALeRCE, sino también para resolver otros problemas de clasificación astronómica, como la detección de rarezas, el agrupamiento de clases desconocidas o la mejora de modelos supervisados y no supervisados. De lograr este objetivo, se tendrán nuevas características que ofrezcan una representación más precisa de los objetos astronómicos, optimizando así el proceso de clasificación y ampliando las capacidades de análisis de grandes volúmenes de datos en astronomía.

2

Marco Teórico

2.1. Conceptos astronómicos

Dado que gran parte del análisis se centrará en las curvas de luz, se debe comprender estos conceptos para interpretar los resultados posteriores. Las curvas de luz utilizadas provienen de surveys astronómicos principalmente obtenidas de OGLE, HIPPARCOS y VVV*, que se detallan más adelante.

2.1.1. Objetos astronómicos

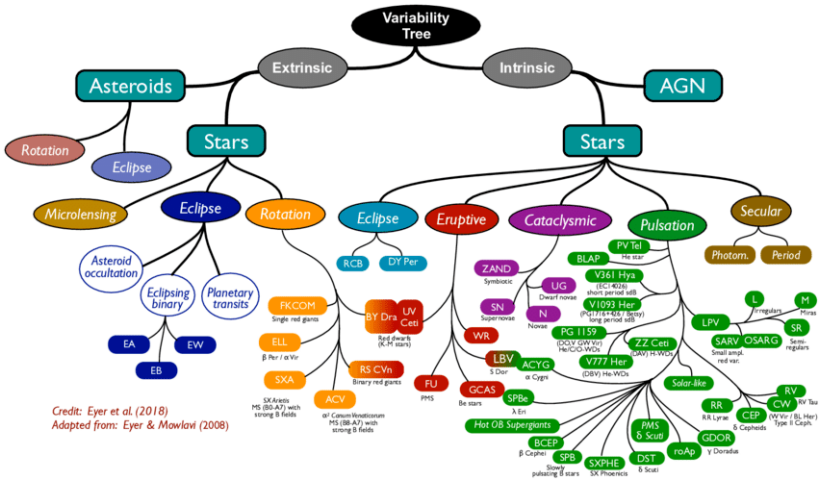


Figura 2.1: Árbol de variabilidad, que muestra los muchos tipos diferentes de fenómenos estelares (y no estelares) que se encuentran en la astronomía (tomado de Eyer y Mowlavi, 2008) [2].

*Optical Gravitational Lensing Experiment, HIgh Precision PARallax COllecting Satellite y VISTA Variables in the Via Lactea, respectivamente.

2.1.1.1. Estrellas variables

Una estrella variable es aquella cuya magnitud de brillo varía con el tiempo. Históricamente, estas estrellas han sido clave para estudiar la estructura y contenido del universo. Su variabilidad puede ser intrínseca (propia del objeto) o extrínseca (causada por factores externos). El General Catalogue of Variable Stars [3] registra más de 110 clases y subclases.

El árbol de variabilidad de Eyer & Mowlavi [2] organiza estos fenómenos en cuatro niveles jerárquicos. El primero distingue entre variabilidad intrínseca y extrínseca; el segundo clasifica por tipo de objeto (estrella, galaxia, asteroide); el tercero identifica el fenómeno causante (rotación, microlente, eclipses, pulsaciones, etc.).

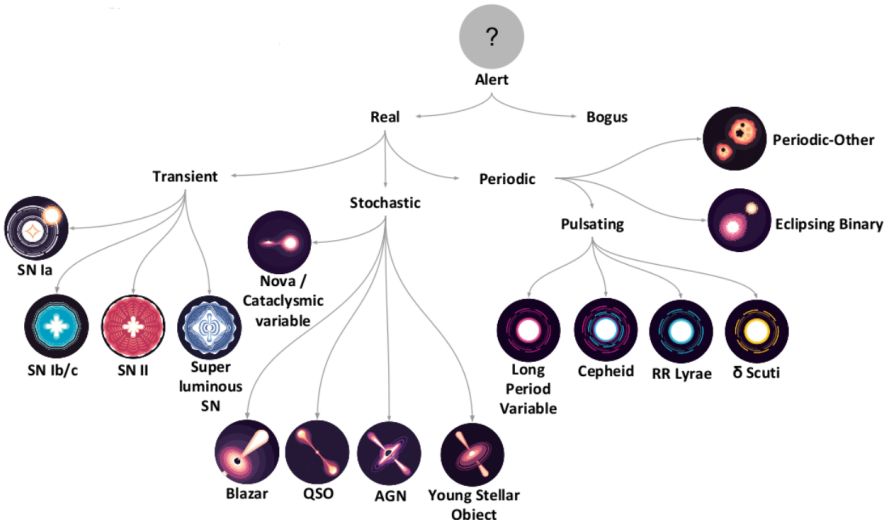


Figura 2.2: Clasificación jerárquica de alertas astronómicas reales y falsas realizada por el broker ALeRCE. Incluye supernovas, AGN, quásares y estrellas variables. Förster (2020) [4].

Entre las variables extrínsecas, destacan las binarias eclipsantes, que incluyen clases como ELL, EA, EB y EW[**]. Entre las intrínsecas, sobresalen las pulsantes, como las RR Lyrae y las Cefeidas, que permiten estimar distancias gracias a su relación período-luminosidad [5].

Las RR Lyrae se subdividen en RRab y RRc, diferenciadas por el modo de pulsación. Las Cefeidas también se clasifican en Tipo I (más brillantes) y Tipo II, con relaciones período-luminosidad distintas. Además, ambas presentan subtipos multiperiódicos como Cefeidas de doble modo (DMCEP) y RR Lyrae de doble modo (RRD) [6]. La siguiente tabla [3] resume las clases más comunes de estrellas pulsantes, incluyendo períodos y amplitudes típicas.

[**] Elipsoidales, Algol-type eclipsing binaries (Tipo Algol), Beta Lyrae-type eclipsing binaries (Tipo Beta Lirae) y W Ursae Majoris-type eclipsing binaries (Tipo W UMa), respectivamente.

Clase	Período (días)	Amplitud (mag)
Cefeidas	2–70	0.1–1.5
RR Lyrae	0.2–1.1	0.2–2
SR-Mira	50–1000	hasta 8
SPB	0.5–5	hasta 0.03
RV Tau	30–150	1–3
δ Scuti	0.02–0.25	hasta 0.1

Tabla 2.1: Resumen de las principales clases de estrellas pulsantes, indicando sus rangos típicos de período y amplitud fotométrica [3].

2.1.1.2. AGN

Un núcleo galáctico activo (AGN) es una región muy brillante en el centro de algunas galaxias, cuya luminosidad no se explica por las estrellas, sino por un agujero negro supermasivo que consume materia cercana y emite radiación en todo el espectro electromagnético. Los AGN son las fuentes persistentes más luminosas del universo y permiten detectar objetos muy lejanos. Según sus características, se clasifican en distintos tipos, siendo los más potentes los cuásares, y los blazares aquellos cuyos chorros de energía apuntan hacia la Tierra [7].

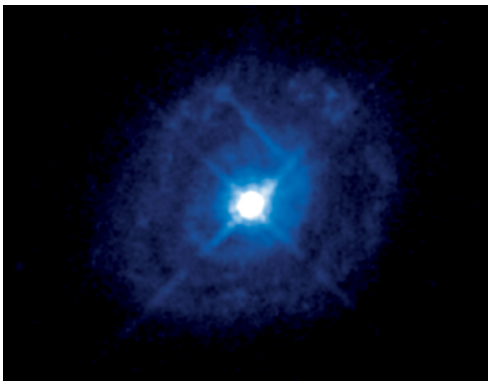


Figura 2.3: Esta imagen del Telescopio Espacial Hubble de la NASA/ESA muestra la galaxia Markarian 509. El objeto brillante en el centro de la galaxia, que parece una estrella, es un núcleo galáctico activo. Se trata de un fenómeno celeste brillante causado por la materia que brilla al caer en un agujero negro supermasivo en el corazón de la galaxia [7].

Se muestran tres tipos de AGN:

- AGN: Son galaxias cuyo centro es extremadamente brillante porque en su núcleo hay un agujero negro supermasivo que está absorbiendo materia. Al caer, esa materia libera mucha energía y hace que el brillo de la galaxia cambie con el tiempo.
- QSO: O quásares, son básicamente AGN muy brillantes. No funcionan de manera distinta: tienen el mismo tipo de agujero negro y el mismo proceso físico. La diferencia principal es que los QSO son más luminosos y, en general, están más lejos. Por eso, AGN y QSO se parecen tanto astronómicamente y muestran comportamientos de variabilidad muy similares [8].
- Blazar: También son AGN, pero con una característica especial: expulsan chorros de energía extremadamente rápidos, y uno de estos chorros apunta casi directamente hacia la Tierra. Esto hace que su brillo cambie de forma más intensa y desordenada, por lo que su variabilidad suele ser distinta a la de AGN y QSO [9].

2.1.1.3. Supernova

Figura 2.4: Fotografía de NASA, la ESA, P. Challis y R. Kirshner (Centro de astrofísica Harvard-Smithsonian) [10].

Una supernova es una explosión estelar extremadamente energética que destruye la estrella original y puede originarse por dos procesos principales: el colapso gravitacional de estrellas muy masivas (tipos II, Ib, Ic), cuando ya no pueden sostenerse al producir elementos del grupo del hierro, o por una ignición termonuclear en una enana blanca dentro de un sistema binario al acumular materia de su compañera. Estas explosiones dejan remanentes como estrellas de neutrones o agujeros negros, y enriquecen el medio interestelar con elementos pesados esenciales para la vida [10].

2.1.2. Curvas de luz y observaciones fotométricas

Las curvas de luz son series temporales que muestran cómo varía el brillo aparente de un objeto astronómico con el tiempo. Se construyen a partir de observaciones fotométricas, que miden el flujo de luz recibido en diferentes momentos.

El brillo se expresa en magnitud aparente, que está relacionada con el flujo F mediante una escala logarítmica:

$$m = -2{,}5 \log_{10}(F) + C$$

donde C es una constante dependiente del sistema fotométrico. Una disminución de 1 en magnitud implica un aumento de 2.512 veces en el brillo.

En general, las curvas de luz se representan con el tiempo (usualmente en fecha juliana) en el eje x, y la magnitud en el eje y.

Cuando el objeto varía de forma periódica, sus cambios de brillo se repiten una y otra vez en ciclos. En lugar de usar el tiempo absoluto, es más útil representar cada observación según en qué parte del ciclo ocurre. A esto se le llama "fasear" la curva de luz. De este modo, aunque las observaciones provengan de distintos ciclos, se pueden alinear sobre un mismo patrón para analizar mejor la forma característica de esta variación.

La fase $\varphi \in [0, 1)$ de una observación se calcula con la siguiente fórmula:

$$\varphi = \left(\frac{t - t_o}{p}\right) - \left\lfloor \frac{t - t_0}{p} \right\rfloor$$

donde:

- t es el tiempo de la observación.
- t_0 es el tiempo de referencia (por ejemplo, el instante máximo del brillo).
- p es el período del objeto.
- $\lfloor \cdot \rfloor$ denota la parte entera.

Esta fórmula calcula en qué parte del ciclo se encuentra cada observación, es decir, indica si la observación ocurrió al principio, a la mitad o al final de un período. Al usar la fase en lugar del tiempo real, se pueden agrupar todas las observaciones de distintos ciclos en un mismo gráfico. Esto permite ver con más claridad la forma repetitiva de la curva de luz y, facilita su análisis y clasificación.

2.1.3. Surveys astronómicos

Los surveys astronómicos son proyectos de observación sistemática del cielo que registran información de millones de objetos en distintas regiones del espacio. Estos surveys recopilan datos como brillo, posición y variabilidad a lo largo del tiempo, lo que permite construir curvas de luz para su análisis. A continuación se describen tres surveys importantes cuyas bases de datos son frecuentemente utilizadas en el estudio de estrellas variables:

- **OGLE (Optical Gravitational Lensing Experiment):** Es un proyecto de observación terrestre que comenzó en 1992. Está enfocado principalmente en regiones densas del cielo como el bulbo de la Vía Láctea y las Nubes de Magallanes. OGLE ha detectado cientos de miles de estrellas variables y es especialmente útil para el estudio de microlentes gravitacionales y pulsantes clásicos. La mayoría de sus observaciones fueron tomadas en la banda I, con hasta 150 épocas por estrella [11].
- **HIPPARCOS:** Fue una misión espacial de la Agencia Espacial Europea (ESA), lanzada en 1989. Su objetivo principal fue medir con precisión la posición y el brillo de más de 100.000 estrellas. Además, de datos astrométricos, HIPPARCOS permitió identificar miles de estrellas variables, muchas de ellas desconocidas hasta entonces [12].
- **VVV (Vista Variables in the Vía Láctea):** Es un survey público en el infrarrojo cercano, operado desde el telescopio VISTA en Chile. Observó el bulbo y parte del disco de la galaxia entre 2010 y 2015. Utiliza filtros en bandas Z, Y, J, H y K_s, adecuados para estudiar regiones con mucho polvo interestelar donde la luz visible es absorbida. VVV ha generado millones de curvas de luz en regiones muy pobladas del cielo y ha sido clave para detectar variables en zonas poco accesibles para los surveys ópticos.

  Uno de sus objetivos clave es construir un mapa tridimensional del bulbo galáctico utilizando estrellas pulsantes como las RR Lyrae, que actúan como fósiles estelares por su antigüedad [13].

Gracias a estos surveys hoy existen catálogos masivos de curvas de luz, tanto de objetos variables como no variables.

2.2. Antecedentes de la investigación

En 2008, los primeros esfuerzos significativos para clasificar automáticamente las estrellas variables se materializaron con el trabajo de Debosscher et al. (2008) titulado **"Automated supervised classification of varible stars"** [14], quienes diseñaron un sistema automático de clasificación multiclase. Utilizaron 28 características estadísticas extraídas de las curvas de luz de las estrellas variables, y probaron diversos clasificadores como árboles de decisión, redes neuronales y algoritmos basados en reglas. Este sistema logró tasas de acierto superiores al 90 % en varias clases, destacando la importancia de un preprocesamiento de los datos para lidiar con el ruido y la cobertura temporal incompleta.

Años más tarde, en 2011, Richards et al. publicaron dos trabajos fundamentales. El primero, **"On Machine-Learned Classification of Variable Stars with Sparse and Noisy Time-Series Data"** [15], propuso un enfoque basado en mezclas gaussianas para clasificar las estrellas variables. A diferencia del trabajo anterior, su metodología era no supervisada y utilizaba el algoritmo de expectation-maximization para modelar las curvas de luz mediante una combinación de componentes gaussianos. Esta técnica permitió manejar datos incompletos y estimar la incertidumbre en las clasificaciones, lo que representaba un avance significativo para la clasificación de estrellas variables en series temporales incompletas.

Ese mismo año, Richards et al. también introdujeron el concepto de active learning para la clasificación de eventos transitorios en su trabajo **"Active Learning to Overcome Sample Selection Bias: Application to Photometric Variable Star Classification"** [16]. Su enfoque se centraba en la detección de variabilidad mediante métricas simples como la dispersión y los ajustes polinomiales a las curvas de luz. Aunque no utilizaron modelos complejos como las mezclas gaussianas, su propuesta estaba diseñada para ser escalable, permitiendo la creación de catálogos confiables de eventos transitorios, como novas y supernovas, sin necesidad de espectros ópticos costosos.

En 2016, Elorrieta et al. implementaron un sistema supervisado de clasificación automática específicamente para identificar estrellas RR Lyrae tipo ab en los datos del sondeo infrarrojo cercano VVV. Su trabajo, titulado **"A Machine Learned Classifier for RR Lyrae in the VVV Survey"** [17], se centró en la selección de variables clave utilizando el índice de Stetson, aplicando un filtrado con recortes sigma y la eliminación de observaciones con errores mayores a 5σ. De un conjunto de 68 características numéricas extraídas de las curvas de luz, seleccionaron 12 basadas en un análisis de importancia, con un enfoque en parámetros armónicos transformados. Luego, probaron varios clasificadores, como la regresión logística y el algoritmo AdaBoost.M1, obteniendo el mejor rendimiento con un F1-score de 0.933 y un AUC de 0.9937, lo que permitió una clasificación precisa en un entorno NIR (infrarrojo cercano) complejo.

Ese mismo año, Nun et al. publicaron **"Ensemble Learning Method for Outlier Detection and its Application to AStronomical Light Curves"** [18], en el cual propusieron un modelo estadístico para detectar outliers en grandes volúmenes de curvas de luz astronómicas. Utilizaron una red ponderada de cinco métodos de detección de anomalías, que incluían variantes de k-NN, Random Forest y modelos de distribución aprendida. Estos métodos asignaban puntajes de anomalía a cada curva, que luego se combinaban mediante una red de gating network entrenada con descenso estocástico del gradiente. Esta metodología permitió identificar eventos astronómicos raros, como novas y estrellas gigantes, y fue escalable gracias a técnicas de reducción de dimensionalidad.

Finalmente, en 2021, un gran avance se dio con el desarrollo del clasificador de curvas de luz para el sistema ALeRCE (Automatic Learning for the Rapid Classification of Events). En **"Alert Classification for te AleRCE Broker System: The Light Curve Classifier"** [19], presentado por Sánchez-Sáez et al., se diseñó un clasificador jerárquico de dos niveles para procesar el flujo de alertas del Zwicky Transient Facility (ZTF) y clasificar eventos astronómicos transitorios. El primer nivel clasifica los eventos en tres categorías principales: periódicos, estocásticos y transitorios. Luego, en un segundo nivel, los clasifica en 15 subclases, utilizando un total de 152 características derivadas de las curvas de luz en bandas g y r del ZTF, colores del catálogo AIIWISE, coordenadas galácticas y medidas morfométricas. El clasificador emplea un balanced random forest (BRF) para manejar el alto desequilibrio de clases en los datos, logrando un macro-averaged F1-score de 0.59 en el segundo nivel y 0.97 en el primero. Este avance permitió clasificar una amplia variedad de objetos, incluyendo núcleos activos galácticos, cuásares, blazares, objetos estelares jóvenes y variables cataclísmicas, en adición a las estrellas variables periódicas. El sistema está operativo y actualiza las clasificaciones diariamente para más de 800.000 objetos, proporcionando una herramienta escalable y eficiente para la clasificación automática de eventos astronómicos.

2.3. Bases conceptuales

2.3.1. iAR

El modelo iAR (Irregular Autoregressive) es un modelo autorregresivo diseñado para el análisis de series de tiempo discretas observadas irregularmente en el tiempo. Este modelo se utiliza a menudo para identificar y modelar la autocorrelación en los residuos de otros modelos, en este caso, curvas de luz astronómicas [20].

El modelo iAR se basa en la representación en tiempo discreto del modelo autorregresivo de tiempo continuo de orden 1 (CAR(1)). También se le conoce como CARMA(1,0). El modelo iAR es una extensión del modelo autorregresivo regular de orden 1 (AR(1)).

El proceso iAR relaciona una observación actual (y_{t_j}) con la observación anterior ($y_{t_{j-1}}$) ajustando el

parámetro φ (autocorrelación) según el intervalo de tiempo transcurrido $(t_j - t_{j-1})$. Se define formalmente como:

$$y_{t_j} = \varphi^{t_j - t_{j-1}} y_{t_{j-1}} + \sigma\sqrt{1 - \varphi^{2(t_j - t_{j-1})}}\varepsilon_{t_j}$$

donde ε_{t_j} son variables aleatorias independientes con media cero y varianza unitaria.

El proceso iAR es débilmente estacionario y, bajo ciertas condiciones, también es estacionario y ergódico. Esto proporciona un marco sólido para evaluar la autocorrelación en series de tiempo muestreadas irregularmente.

Este modelo ha demostrado ser útil en el campo de la astronomía. Por ejemplo, elmodelo iAR (o CiAR) fue el modelo más adecuado en la mayoría de los casos analizados con autocorrelación positiva para curvas de luz de estrellas variables y para objetos estocásticos del sondeo ZTF [21].

Un aspecto importante a tener en cuenta es que el modelo iAR solo permite estimar autocorrelación positiva; el parámetro φ está restringido a valores no negativos.

2.3.2. Path signature

El path signature (o firma de un camino) es una manera de representar completamente la forma y el orden en que se mueve una trayectoria dentro de un espacio con varias dimensiones. Un camino es una función continua que a cada instante t del intervalo $[a, b]$ le asigna un punto en un espacio euclidiano de d dimensiones, es decir,

$$X : [a, b] \to \mathbb{R}^d, \quad t \mapsto (X_t^1, X_t^2, \ldots, X_t^d)$$

Para definir la signature de un camino, se calculan todas las integrales iteradas posibles sobre las coordenadas del camino. La integral iterada más simple es la integral de primer nivel, que corresponde a los incrementos de cada dimensión individualmente:

$$S^{(i)}(X) = \int_a^b dX_t^i = X_b^i - X_a^i$$

donde $i = 1, \ldots, d$. Esta integral mide cuánto cambia el camino en la dimensión i entre los extremos del intervalo.

Luego, existen integrales de nivel superior, que capturan la interacción entre las distintas dimensiones a lo largo del camino. Por ejemplo, la integral iterada de segundo nivel para dos dimensiones i y j es $S^{(i,j)}(X) = \int_a^b \int_a^{t_2} dX_{t_1}^i dX_{t_2}^j$. Estas integrales miden cantidades más complejas, como áreas firmadas que el camino genera al moverse en esas dos dimensiones.

En general, para un multi-índice $(i_1, i_2, \ldots, i_k)$, la integral iterada de nivel k es:

$$S^{(i_1,\ldots,i_k)}(X) = \int_{a<t_1<\ldots<t_k<b} dX_{t_1}^{i_1} \ldots dX_{t_k}^{i_k},$$

La colección completa de todas estas integrales iteradas para todos los niveles $k = 0, 1, 2, \ldots$ conforma la signature del camino $S(X)$. Por convención, el término de nivel cero es igual a uno (*Chevyrev & Kormilitzin* [22], *2016, p. 5)*:

$$S^{(0)}(X) = 1$$

Estas integrales forman una estructura algebraica llamada álgebra tensorial, que tiene reglas para combinar y manipular los términos de la signature. Una propiedad fundamental de esta estructura es la identidad de Chen (*Chevyrev & Kormilitzin* [22], *2016, p. 13)*, que establece que la signature de la concatenación de dos caminos es igual al producto tensorial ($\otimes$) de las signatures de cada camino por separado. Esta propiedad permite descomponer y construir signatures de caminos complejos a partir de sus partes.

$$S(X * Y) = S(X) \otimes S(Y)$$

Otra propiedad algebraica importante es el Shuffle Product, que describe cómo se combinan productos de términos de la signature para obtener otros términos de mayor orden.

Desde un punto de vista práctico, la signature es invariante bajo reparametrizaciones del tiempo. Esto significa que si se recorre el mismo camino a diferentes velocidades, siempre y cuando se mantenga el mismo orden de puntos visitados, la signature permanece igual. Así, la signature captura la forma y el orden del camino, pero no la velocidad ni los tiempos específicos en que se recorren los puntos.

Matemáticamente, el interés en la signature también surge de su conexión con las ecuaciones diferenciales ordinarias (ODEs). La signature de un camino que actúa como función de control de una ODE determina de manera única la solución de dicha ecuación. Este concepto se extiende a caminos más irregulares a través de la teoría de Rough Paths, desarrollada por Terry Lyons (*Chevyrev & Kormilitzin* [22], *2016, p. 3)*, que amplía la aplicación de la integración y las ODEs controladas a casos con menor suavidad.

Aunque la signature es una representación muy completa del camino, no siempre determina el camino original de forma única, ya que caminos reparametrizados o con movimientos que se anulan pueden tener la misma signature. Sin embargo, bajo ciertas condiciones matemáticas se ha demostrado que la unicidad puede garantizarse.

Desde un enfoque computacional, los caminos utilizados provienen de datos observacionales discretos. Si se tiene un conjunto de puntos $\{X_0, X_1, \ldots, X_n\} \subset \mathbb{R}^d$, se considera la trayectoria formada por

conectar puntos secuencialmente. A partir de estos, se calculan los incrementos:

$$\triangle X_i = X_{i+1} - X_i$$

Estos incrementos se combinan para generar los términos del signature truncado. Por ejemplo:

- **Nivel 1 (orden 1):** Suma total de incrementos en cada dimensión.

$$S^{(1)}(X) = \sum_{i=0}^{n-1} \triangle X_i$$

- **Nivel 2 (orden 2):** Suma de productos ordenados de pares de incrementos.

$$S^{(2)}(X) = \sum_{i<j} \triangle X_i \cdot \triangle X_j$$

- **Niveles superiores:** Incluyen combinaciones anidadas de mayor longitud que capturan relaciones de orden superior entre dimensiones y tiempo.

En aplicaciones prácticas, especialmente en aprendizaje automático, no es posible calcular la serie infinita completa de integrales iteradas, por lo que se usa una signature truncada hasta un nivel L

$$S^{\leq L}(X) = \{1, S^{(i)}, S^{(i,j)}, \ldots, S^{(i_1,\ldots,i_L)}\}$$

Los términos de esta signature truncada se usan como características numéricas estadísticamente significativas y no paramétricas que resumen la información clave sobre el camino. Los niveles bajos capturan aspectos simples como el desplazamiento total y el área (área de Lévy), mientras que los niveles superiores capturan momentos estadísticos y correlaciones más complejas. Este método ha demostrado ser eficaz para transformar datos secuenciales en conjuntos estructurados de características útiles para tareas de aprendizaje automático.

El número total de términos en $S^L(X)$ crece con $\sum_{k=1}^{L} d^k$, lo que implica que niveles más altos aumentan exponencialmente la dimensión del vector de características. Por ello, la elección de L debe responder respecto a complejidad computacional, riesgo de sobreajuste y capacidad del modelo para capturar estructuras relevantes en los datos.

2.3.2.1. Log-firma

La log-firma de un camino se define como el logaritmo de su signature. En particular, si $S(X)$ denota la signature de un camino X, entonces la log-firma se define como (*Chevyrev & Kormilitzin* [22]):

$$\log S(X) \;:=\; \log\big(S(X)\big).$$

Si se escribe $S(X) = 1 + \widetilde{S}(X)$, donde $\widetilde{S}(X)$ contiene todos los términos de nivel ≥ 1, el logaritmo se entiende como:

$$\log\left(1 + \widetilde{S}(X)\right) = \sum_{m=1}^{\infty} \frac{(-1)^{m+1}}{m} \widetilde{S}(X)^{\otimes m},$$

la cual está bien definida a nivel truncado (profundidad finita), que es el caso usado en aplicaciones computacionales [23].

Desde un punto de vista práctico, esto implica dos ventajas:

1. La log-firma entrega una parametrización más compacta, o sea, menos redundante del mismo contenido geométrico que la signature truncada.

2. Para concatenación de caminos, mientras la signature satisface la identidad de Chen $S(X * Y) = S(X) \otimes S(Y)$, la log-firma combina concatenaciones mediante la fórmula de Baker-Campbell-Hausdorff, lo que refleja que el espacio natural de la log-firma es una estructura de Lie[***] [24].

2.3.3. Random forest

El algoritmo de random forest propuesto por Breiman (2001) es una mejora del método bagging (bootstrap aggregating). Mientras que bagging genera múltiples árboles de decisión a partir de muestras bootstrap del conjunto de entrenamiento y promedia sus predicciones para reducir la varianza, random forest introduce una aleatorización adicional en la selección de variables que reduce la correlación entre los árboles individuales, incrementando así la precisión del ensamble sin aumentar significativamente la varianza total [25].

La idea de este algoritmo consiste en reducir la correlación entre los árboles construidos mediante bagging, sin comprometer la estabilidad del modelo. Esto se logra seleccionando aleatoriamente un subconjunto de variables en cada punto de división del árbol. Específicamente, en cada nodo se eligen $m \leq p$ variables (siendo p el número total de predictores), y de ese subconjunto se selecciona la mejor división posible.

Procedimiento del Algoritmo

Se describe el procedimiento para construir un Random Forest con B árboles, tomando como entrada un conjunto de entrenamiento (x_{ij}, y_i), donde $i = 1, \ldots, N$ y $j = 1, \ldots, P$.

[***] Forma de describir un sistema algebraico donde el orden en que se combinan los elementos importa. Incluye una operación que mide cómo dos elementos interactúan entre sí y se usa para modelar simetrías y relaciones no conmutativas, como las que aparecen en la log-firma de los paths.

Algoritmo 2.1 Random Forest

Datos de entrenamiento (x_{ij}, y_i), con $i = 1, \ldots, N, j = 1, \ldots, P$.

Número de árboles B, número de variables por división $m \leq P$.

1: Inicializar los pesos $w_m(i) = \frac{1}{N}$, para todo $i = 1, \ldots, N$.

2: **para** $b = 1$ **hasta** B

3: Tomar una muestra bootstrap T_b del conjunto de entrenamiento T_N.

4: Construir un árbol sobre T_b aplicando recursivamente en cada nodo terminal:

a. Seleccionar m variables al azar entre las P disponibles.

b. Escoger la mejor variable y punto de corte entre las m seleccionadas.

c. Dividir el nodo en dos nodos hijo.

5: **fin para**

6: **retornar** Conjunto de árboles entrenados $\{C_b(x)\}_{b=1}^{B}$.

2.3.4. Support vector machine

Son métodos de aprendizaje supervisado utilizados principalmente en problemas de clasificación y regresión. El objetivo de una SVM es encontrar una frontera de decisión que separe las observaciones de distintas clases maximizando el margen entre ellas. El modelo se define a partir de un subconjunto de observaciones llamadas vectores de soporte, que determinan completamente la frontera de decisión [26]. A diferencia de otros clasificadores que se centran únicamente en minimizar el error empírico, las SVM buscan un equilibrio entre capacidad de generalización y complejidad del modelo.

Una SVM construye un hiperplano que separa linealmente los datos en el espacio de características original. No obstante, mediante el uso de funciones kernel[****], es posible proyectar los datos a un espacio de mayor dimensión donde la separación lineal resulta factible, permitiendo modelar fronteras de decisión no lineales sin realizar explícitamente dicha proyección [27].

Procedimiento del Algoritmo

Sea un conjunto de entrenamiento $T_n = (x_1, y_1), \ldots, (x_n, y_n)$, con $y_i \in \{-1, 1\}$.

[****] Función que mide la similitud entre dos observaciones y permite trabajar implícitamente en un espacio de mayor dimensión, sin necesidad de calcular explícitamente la transformación de los datos. Gracias a esto, los modelos pueden aprender relaciones no lineales utilizando solo productos internos.

Algoritmo 2.2 Support Vector Machine

Datos de entrenamiento (x_i, y_i), $i = 1, \ldots, N$
Parámetro de regularización C
Función kernel $K(\cdot, \cdot)$

1: Inicializar multiplicadores de Lagrange $\alpha_i = 0$ para todo i
2: Inicializar sesgo $b = 0$
3: **para** $m = 1, \ldots, M$
4: **para** $i = 1, \ldots, N$
5: Calcular la salida:

$$f(x_i) = \sum_{j=1}^{N} \alpha_j y_j K(x_j, x_i) + b$$

6: Actualizar α_i resolviendo localmente el problema de optimización
7: Proyectar α_i al intervalo $[0, C]$
8: **fin para**
9: Actualizar el sesgo b usando las condiciones de optimalidad
10: **fin para**
11: Identificar vectores de soporte: $SV = \{x_i : \alpha_i > 0\}$
12: Definir la función de decisión final:

$$\hat{y}(x) = \text{sign}\left(\sum_{i \in SV} \alpha_i y_i K(x_i, x) + b\right)$$

2.3.5. Boosting

El término boosting hace referencia a una familia de métodos de aprendizaje ensemble que buscan mejorar el rendimiento predictivo mediante la combinación secuencial de modelos simples. A diferencia de enfoques que ajustan modelos de forma independiente, los métodos de boosting construyen cada nuevo modelo considerando los errores cometidos por los modelos anteriores, con el objetivo de corregirlos progresivamente.

Uno de los más avanzados es Extreme Gradient Boosting (XGBoost), propuesto por Chen y Guestrin, el cual extiende el enfoque clásico de boosting incorporando optimización basada en gradientes, regularización explícita y técnicas computacionales eficientes [28]. XGBoost se ha convertido en uno de los algoritmos más utilizados en problemas de clasificación y regresión por su rendimiento y robustez.

Procedimiento del Algoritmo

Sea un conjunto de entrenamiento $T_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, con $y_i \in \{1, \ldots, k\}$.

XGBoost construye un modelo aditivo de la forma:

$$\hat{y}_i = \sum_{m=1}^{M} f_m(x_i),$$

donde cada f_m corresponde a un árbol de decisión. En cada iteración, el nuevo árbol se obtiene minimizando una función objetivo que combina la pérdida empírica y un término de regularización.

Algoritmo 2.3 eXtreme Gradient Boosting

Datos de entrenamiento (x_i, y_i), con $i = 1, \ldots, N$.
Número de iteraciones M.
Función de pérdida $L(y, \hat{y})$.

1: Inicializar las predicciones $\hat{y}_i^{(0)} = 0$ para todo i.
2: **para** $m = 1$ **hasta** M
3: Calcular los gradientes y hessianos:

$$g_i = \frac{\partial L(y_i, \hat{y}_i^{(m-1)})}{\partial \hat{y}_i^{(m-1)}}, \quad h_i = \frac{\partial^2 L(y_i, \hat{y}_i^{(m-1)})}{\partial (\hat{y}_i^{(m-1)})^2}$$

4: Entrenar un árbol de decisión $f_m(x)$ usando (g_i, h_i).
5: Actualizar las predicciones:

$$\hat{y}_i^{(m)} = \hat{y}_i^{(m-1)} + \eta f_m(x_i)$$

6: **fin para**
7: **retornar** Modelo final:

$$\hat{y}(x) = \arg \max_{j \in \{1,\ldots,k\}} \hat{y}_j(x)$$

3
Paquetes de Python

3.1. esig

La librería `esig` es un paquete de Python orientado al cálculo de path signatures a partir de integrales iteradas [29], construido sobre la biblioteca C++ `libalgebra`. Prioriza una implementación de la formulación algebraica de la teoría de firmas, ofreciendo herramientas explícitas para el cálculo, la dimensión y la estructura de la firma y la log-firma truncadas. En este sentido, `esig` se concibe como una librería adecuada para análisis metodológicos y exploratorios donde la interpretabilidad del objeto resulta relevante.

stream2sig

`stream2sig` es una función de la librería `esig` de Python [29]. Su propósito es transformar una trayectoria discreta multidimensional (también llamada stream) en su path signature hasta un orden k determinado.

```
from esig import stream2sig
signature = stream2sig(stream, depth)
```

- ⋆ `stream`: secuencia ordenada de puntos en $\mathbb{R}^d$, es decir, un arreglo de forma `(n, d)` donde `n` es la cantidad de observaciones y `d` la dimensión; en este caso, tiempo y magnitud.
- ⋆ `depth`: número entero k, representa el orden máximo de integrales iteradas que serán calculadas.
- ⋆ `output`: vector en $\mathbb{R}^m$, donde $m =, \sum_{i=1}^{k} d^i$, que representa la path signature hasta el orden k.

Internamente, `esig.stream2sig` convierte el stream en diferencias incrementales $\triangle x_i = x_{i+1} - x_i$, calcula recursivamente las integrales iteradas de orden 1 hasta `depth` y devuelve un vector aplanado con todos los términos: primero orden 1 (incrementos), luego orden 2 (interacciones), etc.

A continuación, se muestra el código fuente extraído del archivo `__init__.py` de la librería ya mencionada:

```
@_verify_stream_arg
def stream2sig(stream, depth):
    """
    Compute the signature of a stream
    """
    if depth <= 0:
        raise ValueError("Depth must be at least 1")
    elif depth == 1:
        return numpy.concatenate([[1.0], numpy.sum(numpy.diff(stream, axis=0),
                                                   axis=0)])

    _verify_valid_depth(stream.shape[1], depth)

    backend = get_backend()
    return backend.compute_signature(stream, depth)
```

Esta función se asegura de que el stream tenga el formato correcto (una lista de puntos con la misma dimensión) y que el orden sea válido. Si se entrega un orden menor a 1, lanza un error. Si el orden es 1, no se calculan integrales complejas, solo se suman los cambios (diferencias) entre los puntos del path, y se agrega un `1.0` al inicio, dando como resultado un vector que contiene el desplazamiento total del path. Si el orden es mayor a 1, se llama a un backend*, que realiza el cálculo completo de la signature. Este motor está optimizado para obtener todas las integrales necesarias que describen la forma del path.

3.2. IISIGNATURE

Por otro lado, `iisignature` [30] es un paquete de Python que implementa el mismo concepto matemático que `esig`, pero con un énfasis explícito en la eficiencia algorítmica. Su implementación está optimizada para reducir el costo computacional del cálculo de firmas y log-firmas, especialmente en escenarios con grandes volúmenes de trayectorias o altos niveles de truncamiento.

*Parte interna de un sistema que se encarga de realizar los cálculos o procesos complejos. En el caso de la función `stream2sig`, el backend es un motor matemático optimizado que calcula las integrales necesarias para construir la firma de camino.

IISIG

`iisig` es la función principal de la librería `iisignature` en Python [30], utilizada para calcular la path signature de una trayectoria discreta multidimensional hasta un orden k determinado. Al igual que `stream2sig`, esta función transforma un stream representado como una secuencia de puntos en $\mathbb{R}^d$ en un vector de coeficientes que corresponden a integrales iteradas truncadas.

```
import iisignature
signature = iisignature.sig(stream, depth)
```

- ⋆ `stream`: arreglo de forma `(n, d)` que representa una trayectoria discreta en $\mathbb{R}^d$, donde `n` es el número de observaciones y `d` la dimensión del path.
- ⋆ `depth`: entero positivo k que indica el orden máximo de las integrales iteradas a calcular.
- ⋆ `output`: vector en $\mathbb{R}^m$, donde $m = \sum_{i=1}^{k} d^i$, que representa la path signature truncada hasta orden k.

Internamente, `iisignature.sig` modela el stream como un camino lineal por tramos[**] y calcula las integrales iteradas asociadas mediante algoritmos optimizados implementados en C++. A diferencia de `esig`, la librería `iisignature` no prioriza la exposición explícita de la estructura algebraica subyacente, sino que concentra su diseño en la eficiencia del cálculo, lo que permite obtener firmas de mayor profundidad o procesar un gran número de trayectorias con un menor costo computacional.

Desde el punto de vista funcional, `iisig` cumple el mismo rol que `stream2sig`: ambas funciones producen una representación vectorial de la firma de un camino. Sin embargo, mientras `stream2sig` se integra dentro de una librería orientada a la exploración algebraica y estructural de la firma, `iisig` se concibe como una herramienta optimizada para su uso en pipelines de análisis de datos y aprendizaje automático, donde la velocidad y escalabilidad del cálculo resultan determinantes.

[**] Función continua $x(t)$ definida en un intervalo $[t_0, t_n]$. tal que para cada intervalo $[t_i, t_{i+1}]$, el camino se interpola linealmente entre los puntos x_i y x_{i+1}. Es decir, entre dos observaciones consecutivas el camino es un camino recto.

4
Resultados

En primer lugar, los experimentos fueron ejecutados en un servidor remoto al que se accedió mediante la VPN institucional de la universidad. El entorno de ejecución correpondió a un sistema Ubuntu 20.04.2 LTS con kernel Linux 5.4.0-216-generic y arquitectura x86_64. El servidor dispone de un procesador Intel Xeon E-2224 de 3.40 GHz, con 4 núcleos y 4 hilos de ejecución, además de 31 GiB de memoria RAM y 11 GiB de memoria swap.

Se tienen 802249 curvas de luz correspondientes a 1887 núcleos galácticos activos estudiados en distintos tiempos; en cada punto hay un tiempo y una magnitud medida, con intervalos irregulares y ruido. Primero se ordena el tiempo de forma creciente y se limpian duplicados o valores atípicos. Después se añade dos puntos: uno al inicio y otro al final, determinados por los tiempos mínimo y máximo de todas las curvas de luz, asignando en ambos puntos la magnitud media de cada una de ellas. De este modo, todas las curvas comparten el mismo intervalo temporal, es decir, poseen el mismo dominio.

Luego, se ajusta un modelo iAR, el cual permite modelar el comportamiento de cada curva de luz considerando que las observaciones no están igualmente separadas en el tiempo. A partir de este ajuste, se generan curvas simuladas que presentan variaciones similares a las observadas en los datos reales. De esta forma, se obtiene un conjunto de series que mantiene las características temporales de las curvas originales, pero incorporando nuevas curvas con un comportamiento similar.

Cuando se calcula la path signature, esta resume la forma y el orden de los cambios de un camino 2D formado por (tiempo, magnitud) en un vector de números. No necesita que las mediciones estén igualmente espaciadas, pero sí que el tiempo vaya creciendo. Por eso es importante que todas las curvas tengan el mismo dominio temporal y que el inicio y el final estén estabilizados con la media; así, cuando se comparan signatures entre curvas, las diferencias reflejan su comportamiento real y no que una curva cubra un tramo de tiempo distinto a la otra.

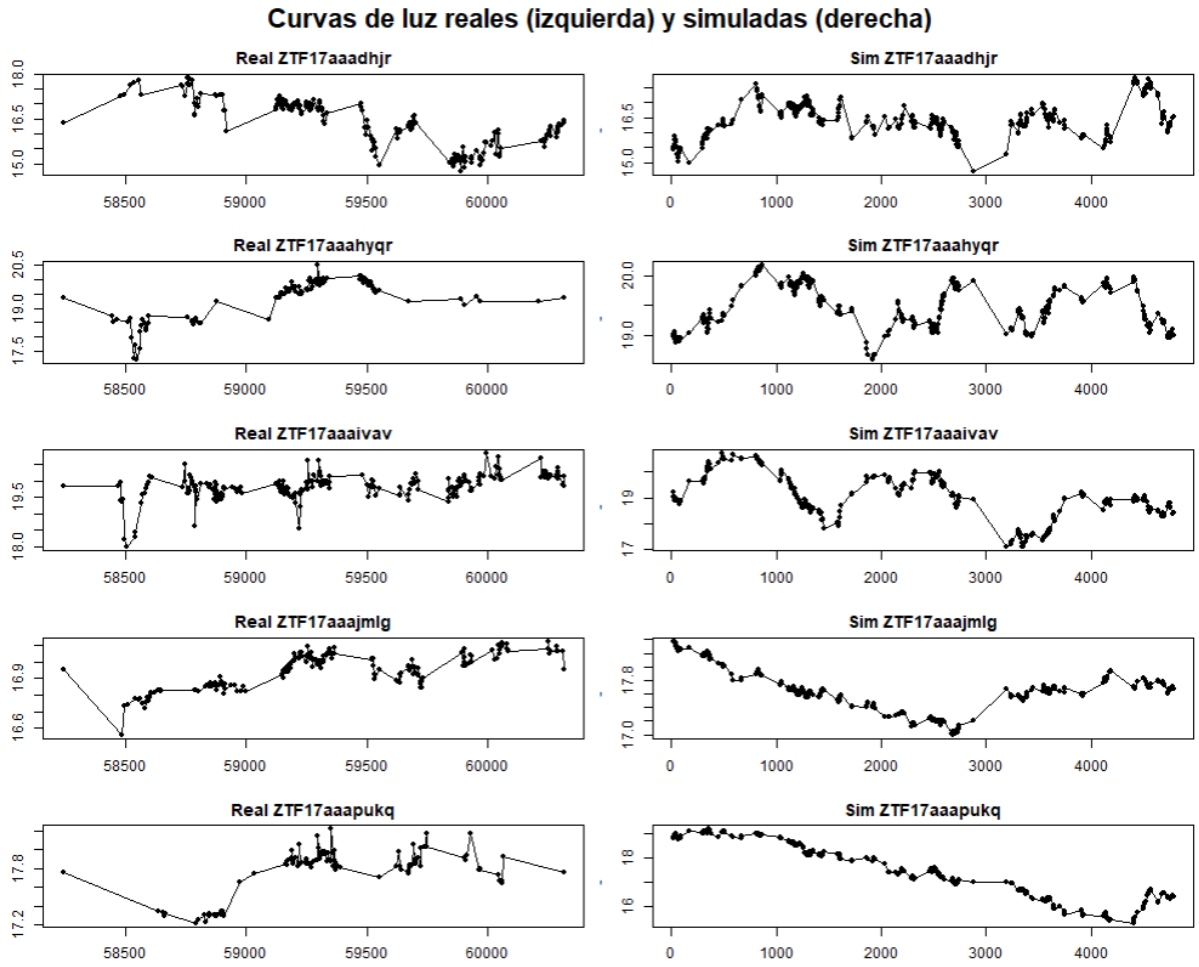


Figura 4.1: A la izquierda, las primeras cinco curvas de luz de los datos reales con la media agregada al inicio y al final. A la derecha, las primeras cinco curvas de luz con datos simulados basados en el modelo iAR.

4.1. Datos simulados

El enfoque de la simulación se basa en el modelo iAR. Este es un modelo estadístico diseñado específicamente para series de tiempo donde las observaciones no están espaciadas uniformemente, que es lo que comúnmente ocurre con los datos astronómicos. El proceso se divide en dos etapas:

1. Estimación de parámetros:

   Primero, el modelo iAR se aplica a la curva de luz real observada (una vez limpia y preprocesada). El propósito de esta etapa es aprender la estructura de dependencia temporal intrínseca del objeto. A diferencia de un modelo autorregresivo estándar (AR) que asume intervalos de tiempo fijos, el modelo iAR cuantifica cómo la correlación entre dos puntos de datos disminuye a medida que la separación temporal Δt entre ellos aumenta.

Estadísticamente, el modelo ajusta un coeficiente base de autocorrelación (análogo al φ de un modelo AR(1)). Este coeficiente representa la "memoria" del proceso. La correlación entre dos puntos no es constante, sino que decae exponencialmente en función del tiempo transcurrido entre ellos. El resultado de esta primera etapa es la estimación de este coeficiente que mejor describe la variabilidad observada del objeto real.

2. Generación de la serie simulada:

   En esta etapa se utiliza el coeficiente de autocorrelación estimado previamente, pero se descartan los tiempos y valores de la curva de luz original. En su lugar, se genera una nueva secuencia sintética de 400 tiempos irregulares a partir de una mezcla de distribuciones exponenciales, diseñada para reproducir patrones de observación realistas con intervalos cortos y largos.

   Sobre esta secuencia temporal, el modelo iAR genera de forma autorregresiva una nueva serie de magnitudes, asegurando que la correlación entre puntos consecutivos respete tanto el coeficiente de memoria estimado como los intervalos de tiempo correspondientes. Finalmente, a la serie simulada (centrada en cero) se le añade la magnitud promedio observada en los datos reales, garantizando que la variabilidad y el rango de brillo sean consistentes con la curva de luz original.

Se muestran a continuación las primeras observaciones de, en este caso, cinco curvas de luz, simuladas usando únicamente la banda fotométrica lcg[*]. Solo se va a usar esta banda porque cada una de ellas capta un rango distinto de luz, así se evita mezclar datos de diferentes longitudes de onda, lo que permite analizar las variaciones de brillo de forma apropiada.

	id	tiempo	banda	magnitud	error
0	ZTF17aaadhjr	50.691083	1	16.632125	0
1	ZTF17aaadhjr	54.173140	1	16.673578	0
2	ZTF17aaadhjr	57.155599	1	16.705209	0
3	ZTF17aaadhjr	59.682992	1	16.856806	0
4	ZTF17aaadhjr	61.442856	1	16.778805	0
...	...	...	...	...	...
400	ZTF17aaahyqr	50.691083	1	20.733629	0
401	ZTF17aaahyqr	54.173140	1	20.699563	0
402	ZTF17aaahyqr	57.155599	1	20.661081	0
403	ZTF17aaahyqr	59.682992	1	20.606604	0
404	ZTF17aaahyqr	61.442856	1	20.564259	0
...	...	...	...	...	...

Luego, se presenta un gráfico de cajas de los coeficientes de autocorrelación, obtenidos del análisis individual de cada curva de luz real y agrupados según su clase astronómica (AGN, Blazar o QSO). El

[*] Curva de luz en la banda g (luz verde-azulada). También existe la banda r, que hace referencia a la luz rojiza.

objetivo es comparar visualmente sus distribuciones y evaluar posibles diferencias en la estructura de variabilidad entre las tres clases.

Dicho coeficiente es el parámetro que el modelo iAR estima a partir de la curva real. Este valor cuantifica la variabilidad del objeto y es este mismo el que se utiliza posteriormente como parámetro de entrada para generar la curva simulada, garantizando así que ambas curvas compartan su estructura (aunque sus valores numéricos sean nuevos y aleatorios, se sigue el mismo comportamiento: fluctúan alrededor de la misma media, con la misma varianza y la misma autocorrelación).

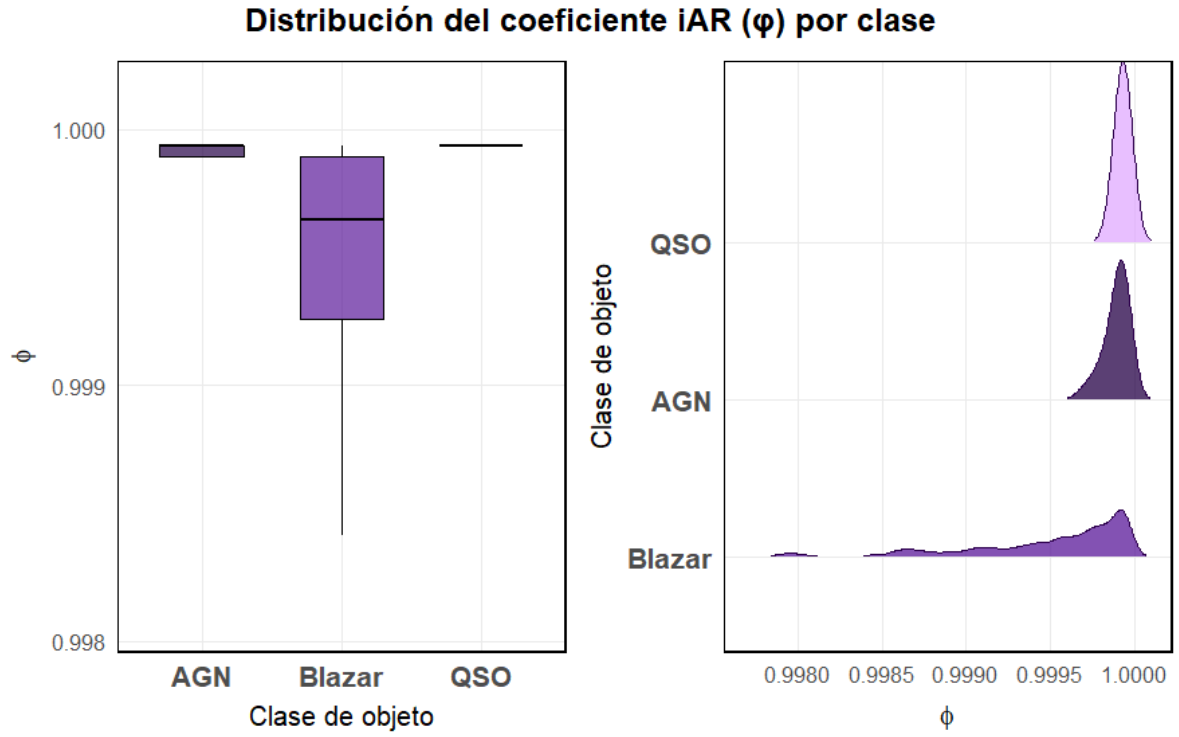


Figura 4.2: Gráfico de cajas y de distribución de los coeficientes de autocorrelación.

El primer gráfico indica que no existen diferencias significativas en el coeficiente de variabilidad entre AGN, Blazar y QSO, lo que sugiere un comportamiento estadísticamente homogéneo. Dado que φ es cercano a 1, el brillo presenta una fuerte dependencia temporal, con cambios suaves que conservan memoria del estado previo. Por su parte, el gráfico de distribución muestra que AGN y QSO tienen comportamientos muy similares, con densidades concentradas en torno a $\varphi \approx 1$, mientras que Blazar exhibe una distribución más amplia y desplazada hacia valores menores, evidenciando un patrón distinto.

Para calcular path signature, se toma la curva como un camino bidimensional (tiempo, magnitud) ordenado en el tiempo y se transforma en un vector de tamaño fijo que resume su forma y secuencia de cambios: 2 dimensiones y nivel 9, ese vector tiene 1022 componentes ($2^1 + 2^2 + \cdots + 2^9$); su tamaño depende solo del número de dimensiones y del nivel, no del número de puntos en la curva.

Usar 9 niveles en path signature significa que la firma examina patrones de la curva hechos de secuencias de hasta nueve pasos entre tiempo y magnitud: a mayor nivel, más detalle sobre el ritmo de subidas y bajadas, pero también mayor complejidad computacional. Los primeros valores miden cambios globales (cuánto avanzó el tiempo y cuánto cambió la magnitud), los siguientes distinguen el orden de los cambios (si primero avanza el tiempo y luego cambia la magnitud, o al revés), y los de niveles altos recogen secuencias más largas. Cada número indica cuánto aparece ese patrón en la curva; en conjunto, el vector sintetiza la dinámica de la curva y permite compararla o clasificarla. Tal y como se examina en la siguiente salida:

```
     0            1         2             3            4            5  \
0  1.0  4749.699999  1.592561  1.127983e+07  5439.755762  2124.431281
1  1.0  4749.699999 -0.235436  1.127983e+07 -5112.286139  3994.037394
2  1.0  4749.699999 -1.534474  1.127983e+07 -3344.054745 -3944.234892
3  1.0  4749.699999  0.127918  1.127983e+07   469.262479   138.307571
4  1.0  4749.699999 -0.790126  1.127983e+07  2388.843813 -6141.705790

          6             7       ...              1017           1018  \
0  1.268125  1.785859e+10       ...       2005.878223      11.115354
1  0.027715  1.785859e+10       ...    -276021.625362     316.087543
2  1.177305  1.785859e+10       ...        979.363188      34.659804
3  0.008181  1.785859e+10       ...          1.239267       0.000308
4  0.312150  1.785859e+10       ...     176669.486772     114.729292

             1019        1020       1021           1022             id
0    3735.928190   -3.808765   2.323217   1.815941e-04   ZTF17aaadhjr
1   72300.783368  -81.287206   9.147342  -6.124732e-12   ZTF17aaahyqr
2  -11493.916757  -26.050765   9.757281  -1.299786e-04   ZTF17aaaivav
3      -0.069388   -0.000082   0.000011   2.527019e-14   ZTF17aaajmlg
4   -76871.04066  -51.098399  10.083065  -3.307537e-07   ZTF17aaapukq
```

Considerando lo anterior, se lleva a cabo un análisis exploratorio de la base de datos previo a la aplicación de modelos de clasificación. Se enseña que el conjunto de datos está compuesto por 1130 objetos tipo QSO, 520 AGN y 237 Blazar. Esto indica que las clases no están balanceadas, ya que la mayoría de los registros corresponden a cuásares (QSO), mientras que los blazars son los menos representados. Un desbalance puede influir en el desempeño de los modelos de clasificación, haciendo que tiendan a predecir con mayor precisión la clase dominante.

Una vez extraídas las características mediante path signature de nivel 9, se procede a la clasificación. El objetivo es entrenar un clasificador que prediga la clase del objeto basándose en dicho vector de características.

```
Clases únicas: ['Blazar' 'QSO' 'AGN']

Cantidad de ejemplos por clase:
 tipo
QSO       1130
AGN        520
Blazar     237
Name: count, dtype: int64
```

4.1.1. Random forest

Para construir el modelo, el conjunto de datos se dividió en un 80 % para entrenamiento y un 20 % para prueba, utilizando una semilla fija para asegurar la reproducibilidad. La variable objetivo se codificó numéricamente mediante Label Encoding[**]. Por el desbalance entre clases, se aplicó ponderación para favorecer un desempeño más equilibrado entre categorías. La selección de hiperparámetros se realizó mediante validación cruzada estratificada con cuatro particiones, utilizando como criterio principal el AUC multiclase OVR ponderado, ya que esta métrica resume la capacidad discriminativa del modelo considerando simultáneamente las tres clases y ponderando según su soporte. Para explorar el espacio de hiperparámetros se empleó una búsqueda aleatoria de 200 configuraciones, evaluando en total 800 ajustes. El espacio de búsqueda incluyó hiperparámetros asociados a la complejidad del bosque, la regularización y el grado de aleatoriedad en la construcción de los árboles: En particular, el número de árboles se evaluó entre 100 y 1000, mientras que la profundidad máxima se exploró entre 5 y 50 niveles, incluyendo también crecimiento sin restricción. Asimismo, el número mínimo de muestras para dividir nodos y formar hojas se varió en distintos rangos. Para la selección de variables en cada división se evaluaron estrategias basadas en la raíz cuadrada, el logaritmo en base 2 y distintas fracciones del total de características, con el fin de introducir aleatoriedad y reducir la correlación entre árboles. Además, se consideraron configuraciones con y sin remuestreo bootstrap, junto con distintas ponderaciones de clases para mitigar el desbalance entre categorías.

esig

Inicialmente, el estudio se realiza con la librería `esig`, incorporando posteriormente `iisignature` para evaluar la consistencia de los resultados obtenidos bajo una implementación alternativa. Una vez concluida la búsqueda, se seleccionaron las cinco configuraciones con mejor desempeño promedio en validación cruzada. Para cada una de ellas se calculó la brecha entre el rendimiento en entrenamiento y validación cruzada, utilizada como indicador de posible sobreajuste. Después, cada uno de estos cinco modelos se reentrenó sobre el conjunto completo de entrenamiento y se evaluó en el conjunto de

[**]Técnica de preprocesamiento en machine learning que convierte variables categóricas en numéricas asignando un número entero único a cada categoría.

prueba mediante accuracy[***], precision [****], recall[*****] y F1-score ponderado[******].

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
RF_1	0.6047	0.0261	0.3393	0.6025	0.0224	0.3412	0.8223	0.8265
RF_2	0.6033	0.0244	0.3222	0.6015	0.0219	0.3233	0.7958	0.7980
RF_3	0.6033	0.0318	0.3310	0.6017	0.0244	0.3316	0.7931	0.7956
RF_4	0.6030	0.0268	0.2782	0.6002	0.0217	0.2792	0.7745	0.7761
RF_5	0.6029	0.0308	0.3519	0.6026	0.0248	0.3515	0.8170	0.8181

Tabla 4.1: Desempeño de los cinco mejores modelos RF utilizando firma (`esig`) sobre datos simulados.

- AUC_{CV} y SD_{CV}: promedio y desviación estándar del AUC obtenidos durante la búsqueda de hiperparámetros usando validación cruzada estratificada con 4 particiones.
- AUC_{rep} y SD_{rep}: promedio y desviación estándar del AUC obtenidos mediante validación cruzada repetida (4 particiones $\times$ 5 repeticiones) aplicada sobre el conjunto de entrenamiento.

El proceso de compilación tuvo una duración total de 14 horas, 11 minutos y 36 segundos.

A partir de esta comparación, se observa que el modelo RF_1 obtuvo el mejor desempeño global, alcanzando un AUC promedio de 0.6047 en validación cruzada, junto con un accuracy de prueba de 0.8223 y un F1-score ponderado de 0.8265. Aunque algunos modelos presentaron menor sobreajuste, ninguno igualó simultáneamente su desempeño en validación y prueba. Por este motivo, RF_1 fue seleccionado como modelo final de esta etapa.

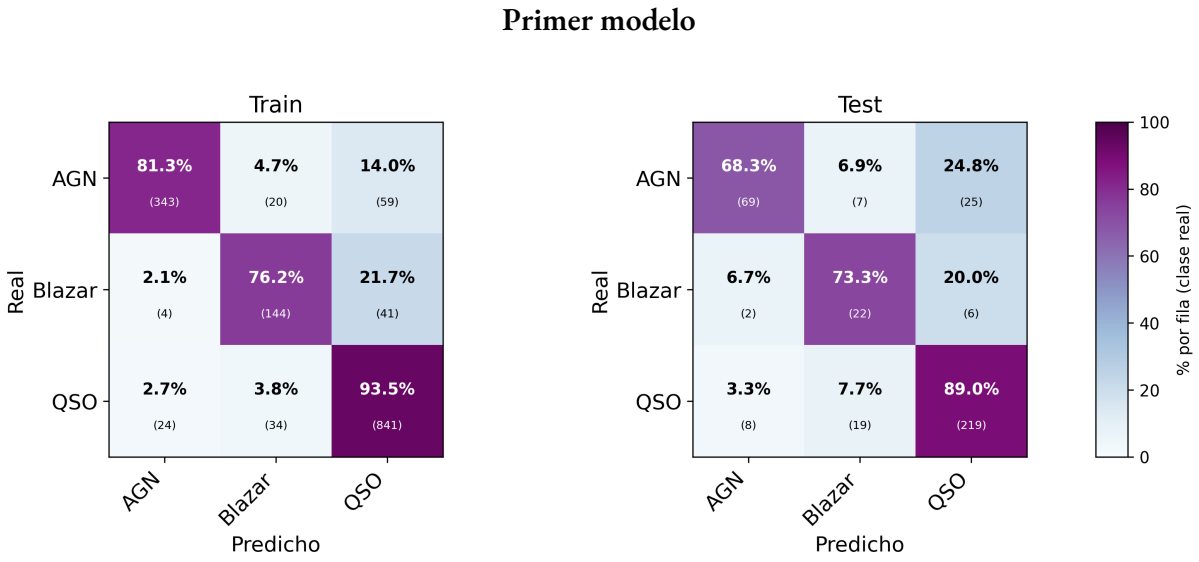


Figura 4.3: Matrices de confusión del entrenamiento y la prueba para el mejor modelo RF utilizando firma (`esig`) sobre datos simulados.

[***] Porcentaje total de predicciones correctas (positivas y negativas) sobre el total de casos.
[****] Mide qué tan preciso es el modelo al predecir la clase positiva.
[*****] Mide la capacidad del modelo para encontrar todos los casos positivos reales.
[******] Medida que combina precisión y recall, promediada entre clases según su frecuencia

```
Accuracy : 0.879
Precision: 0.882
Recall   : 0.879
F1-score : 0.879
```

```
Accuracy : 0.822
Precision: 0.842
Recall   : 0.822
F1-score : 0.826
```

Para complementar la evaluación, se analizó cómo se distribuye la importancia de las variables a lo largo de los niveles de la firma.. Para cada nivel se calcularon dos resúmenes: importancia promedio (media de sus variables) y la importancia mediana (mediana), con el objetivo de distinguir entre niveles que aportan importancia de manera relativamente homogénea versus niveles donde la señal se concentra en pocas variables.

Importancia promedio y mediana por nivel

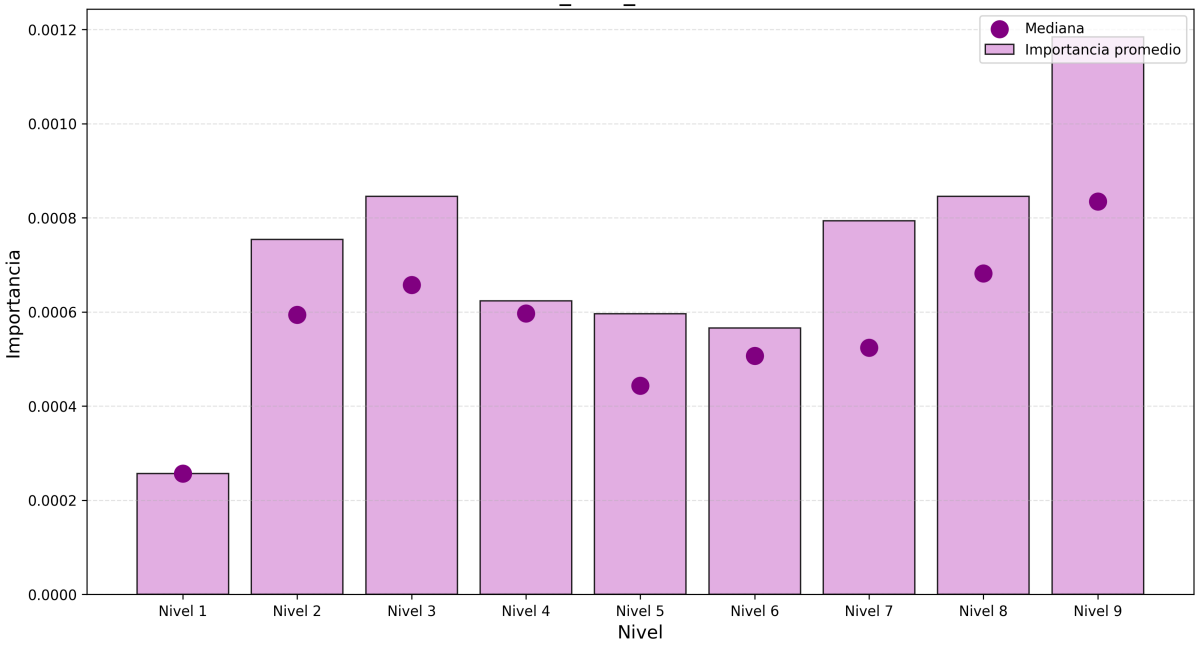


Figura 4.4: Importancia promedio y mediana por nivel para el modelo RF seleccionado con firma `esig` sobre datos simulados.

Se observa que el nivel 9 domina en importancia promedio, pero además aparece un aporte relativamente marcado de niveles intermedios (especialmente nivel 3, y luego 7–8). La comparación entre el promedio y la mediana muestra una diferencia clara en niveles altos: el promedio supera a la mediana, lo que sugiere una distribución asimétrica de importancias, o sea que pocas variables concentran gran parte del peso. En suma, en niveles como el 5 la mediana cae más que el promedio, lo que refuerza la idea de que ahí predominan variables poco informativas y el aporte del nivel proviene de un grupo reducido de características.

Finalmente, se evaluó si el conjunto completo de 1022 características correspondía a una representación adecuada o si era posible alcanzar un rendimiento competitivo con una representación más

simple. Por ello, se comparó el desempeño del modelo utilizando subconjuntos acumulativos de características hasta cada nivel de firma.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	Acc Test	F1 Test	AUC Test
1	2	0.579	0.571	0.594	0.585	0.636
2	6	0.632	0.631	0.584	0.591	0.689
3	14	0.715	0.710	0.655	0.653	0.741
4	30	0.752	0.747	0.682	0.686	0.773
5	62	0.797	0.794	0.719	0.726	0.802
6	126	0.825	0.823	0.748	0.752	0.823
7	254	0.846	0.845	0.761	0.768	0.840
8	510	0.868	0.867	0.785	0.791	0.857
9	1021	0.887	0.886	0.814	0.817	0.873

Tabla 4.2: Comparación del rendimiento del mejor modelo RF con firma `esig` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos simulados.

Los resultados muestran que el rendimiento mejora a medida que se incorporan características de mayor orden. En particular, el AUC de prueba aumenta desde 0.636 en el nivel 1 hasta 0.873 en el nivel 9, alcanzando en este último el mejor desempeño general, junto con una accuracy de prueba de 0.814 y un F1-score de 0.817. Por lo tanto, el nivel 9 corresponde al nivel más simple que maximiza el desempeño y respalda el uso de la representación completa de 1022 características.

Con el fin de comparar representaciones, el mismo procedimiento de entrenamiento, selección y evaluación se aplicó posteriormente utilizando como entrada las características derivadas de la log-signature, manteniendo inalterada la partición de los datos, la estrategia de validación cruzada, la métrica principal de comparación y el esquema de ponderación por clases. Los resultados mostraron nuevamente al modelo RF_1 como la mejor configuración bajo esta representación. Sin embargo, el desempeño obtenido fue inferior al alcanzado con la signature estándar.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
RF_1	0.5795	0.0174	0.3415	0.5771	0.0200	0.3436	0.7798	0.7775
RF_2	0.5784	0.0155	0.3235	0.5765	0.0187	0.3252	0.7507	0.7487
RF_3	0.5783	0.0176	0.2544	0.5746	0.0174	0.2575	0.6897	0.6924
RF_4	0.5781	0.0177	0.3141	0.5766	0.0198	0.3158	0.7427	0.7416
RF_5	0.5781	0.0164	0.2938	0.5765	0.0182	0.2957	0.7294	0.7328

Tabla 4.3: Desempeño de los cinco mejores modelos RF utilizando log-firma (`esig`) sobre datos simulados.

Primer modelo

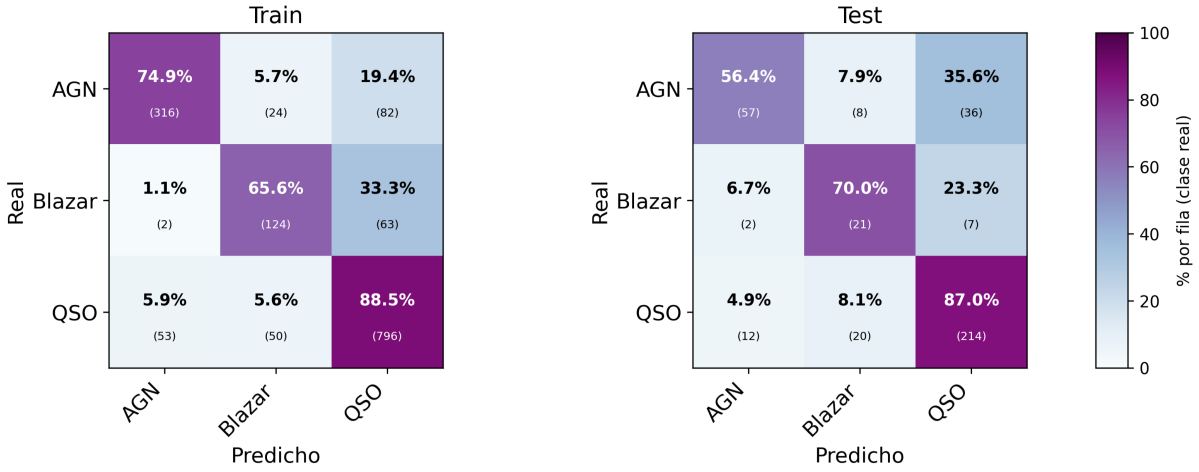


Figura 4.5: Matrices de confusión del entrenamiento y la prueba para el mejor modelo RF utilizando log-firma (`esig`) sobre datos simulados.

```
Accuracy : 0.831
Precision: 0.832
Recall   : 0.831
F1-score : 0.828
```

```
Accuracy : 0.780
Precision: 0.784
Recall   : 0.780
F1-score : 0.778
```

El mejor modelo con log-signature alcanzó un AUC promedio en validación cruzada de 0.5795, un AUC promedio en validación repetida de 0.5771, una accuracy de prueba de 0.7798 y un F1-score ponderado de 0.7775. Los cinco mejores modelos presentan desempeños muy similares, lo que sugiere estabilidad en el proceso de selección y en la capacidad predictiva del algoritmo bajo esta representación.

Al comparar estos resultados con los obtenidos mediante la signature estándar, se observa que la log-signature presenta un rendimiento levemente inferior, lo que indica que no preserva con la misma eficacia toda la información discriminante necesaria para separar las clases astronómicas. Sin embargo, esta diferencia no es drástica, por lo que la log-signature sigue siendo una alternativa válida cuando se busca reducir la dimensionalidad y el costo computacional.

En términos de clasificación por clase, el modelo continúa mostrando un mejor desempeño en la identificación de QSO, mientras que AGN y Blazar presentan mayor dificultad de separación, lo que es consistente con los resultados observados en otras representaciones. El proceso de entrenamiento con log-signature tuvo una duración total de 2 horas, 18 minutos y 30 segundos.

Para interpretar el comportamiento del modelo seleccionado, se analizó la importancia de las características agrupadas por nivel de la firma, considerando tanto la importancia promedio como la mediana.

Las barras muestran la importancia promedio de las variables en cada nivel y los puntos representan la mediana. Esto permite observar tanto el aporte general de cada nivel como el comportamiento central de sus variables. Los resultados muestran que la contribución de las características tiende a aumentar en los niveles superiores de la firma. En particular, el nivel 9 presenta tanto la mayor importancia promedio como la mediana más alta, mientras que el nivel 1 exhibe una contribución claramente menor. Asimismo, la diferencia observada entre promedio y mediana en varios niveles indica que dentro de esos grupos existen algunas variables especialmente influyentes, cuya importancia eleva el promedio por sobre el valor central representado por la mediana. En consecuencia, este resultado refuerza que los términos de orden superior contienen la información más discriminante para la clasificación. En consecuencia, este resultado refuerza que las características de orden superior son las que aportan mayor capacidad discriminante en la clasificación de AGN, Blazar y QSO.

Importancia promedio y mediana por nivel

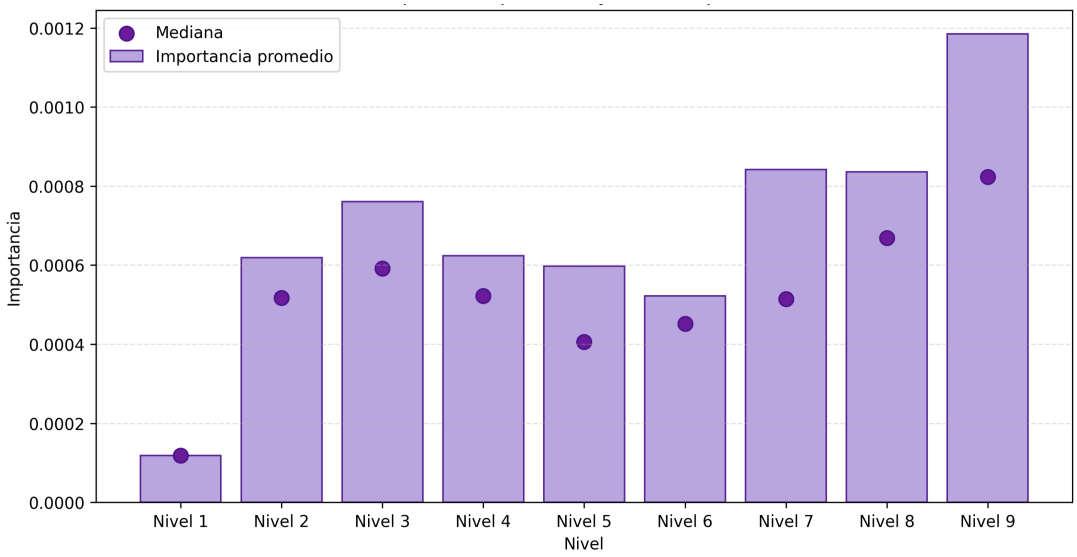


Figura 4.6: Importancia promedio y mediana por nivel para el modelo RF seleccionado con log-firma `esig` sobre datos simulados.

Finalmente, se evaluó si el conjunto completo de 1022 características correspondía a una representación adecuada o si era posible alcanzar un rendimiento competitivo con una representación más simple. Por ello, se comparó el desempeño del modelo utilizando subconjuntos acumulativos de características hasta cada nivel de firma.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	Acc Test	F1 Test	AUC Test
1	2	0.580	0.570	0.597	0.588	0.635
2	6	0.636	0.635	0.589	0.594	0.689
3	14	0.725	0.721	0.660	0.658	0.746
4	30	0.766	0.763	0.690	0.695	0.772
5	62	0.797	0.795	0.727	0.733	0.808
6	126	0.824	0.823	0.753	0.758	0.829
7	254	0.838	0.837	0.767	0.770	0.846
8	510	0.865	0.864	0.806	0.811	0.864
9	1022	0.879	0.879	0.820	0.823	0.877

Tabla 4.4: Comparación del rendimiento del mejor modelo RF con log-firma `esig` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos simulados.

Los resultados muestran que el rendimiento mejora a medida que se incorporan características de mayor orden. En particular, el AUC de prueba aumenta desde 0.635 hasta 0.877 en el nivel 9, alcanzando en este último el mejor desempeño general, junto con una accuracy de prueba de 0.820 y un F1-score de 0.823. Por lo tanto, el nivel 9 corresponde al nivel que maximiza el desempeño y respalda el uso de la representación completa de 1022 características.

iisignature

Dado que el procedimiento de entrenamiento, validación y selección de hiperparámetros ya fue descrito en la sección anterior, aquí se presentan únicamente los resultados obtenidos al utilizar características construidas con `iisignature`. En este caso, el tiempo total de ejecución para la representación fue de 14 horas, 50 minutos y 44 segundos.

Entre las cinco mejores configuraciones evaluadas con firma, el modelo RF_1 obtuvo el mayor AUC promedio en validación cruzada, con un valor de 0.6020. Además, alcanzó una accuracy de prueba de 0.8090 y un F1-score ponderado de 0.8114, por lo que fue seleccionado como modelo final bajo el criterio principal de mayor capacidad discriminativa estimada en validación cruzada.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
RF_1	0.6020	0.0196	0.3453	0.5970	0.0211	0.3500	0.8090	0.8114
RF_2	0.5996	0.0259	0.3585	0.5965	0.0239	0.3611	0.8276	0.8270
RF_3	0.5992	0.0225	0.2943	0.5950	0.0202	0.2973	0.7772	0.7793
RF_4	0.5988	0.0249	0.3476	0.5951	0.0237	0.3505	0.8037	0.8034
RF_5	0.5984	0.0196	0.3302	0.5958	0.0212	0.3318	0.8064	0.8098

Tabla 4.5: Desempeño de los cinco mejores modelos RF utilizando firma (`iisignature`) sobre datos simulados.

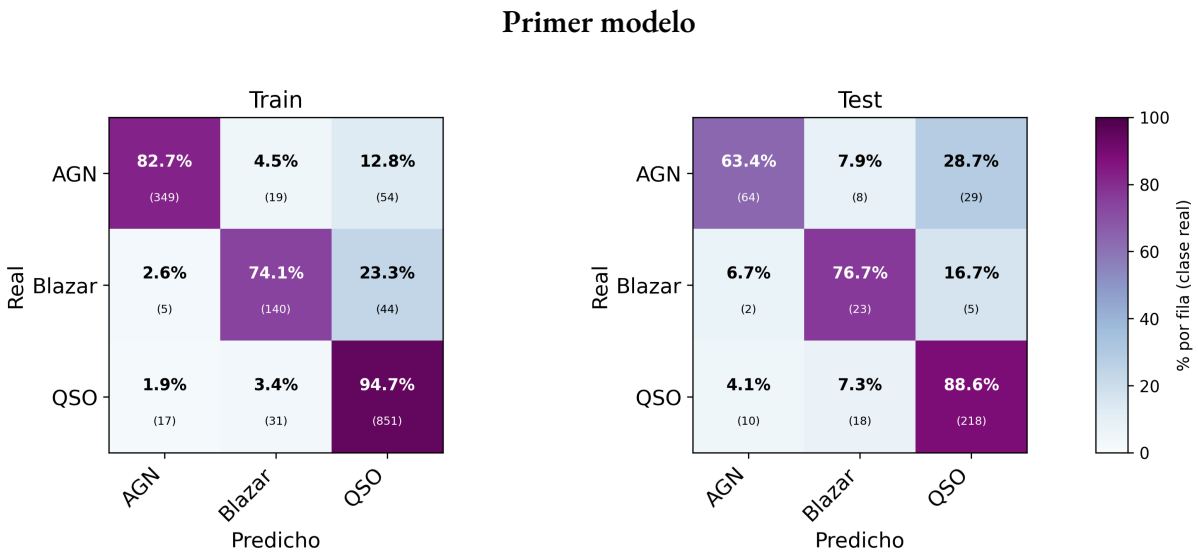


Figura 4.7: Matrices de confusión del entrenamiento y la prueba para el mejor modelo RF utilizando firma (`iisignature`) sobre datos simulados.

```
Accuracy : 0.887
Precision: 0.889
Recall   : 0.887
F1-score : 0.887
```

```
Accuracy : 0.809
Precision: 0.827
Recall   : 0.809
F1-score : 0.811
```

Para complementar la evaluación, se analizó cómo se distribuye la importancia de las variables a lo largo de los niveles de la firma.

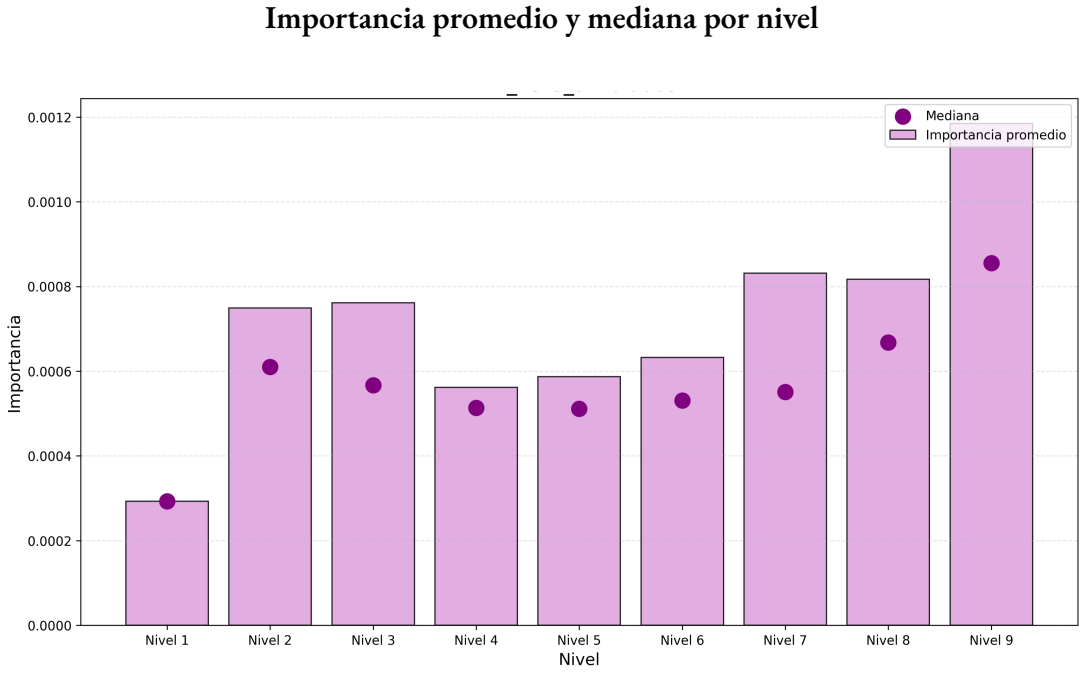


Figura 4.8: Importancia promedio y mediana por nivel para el modelo RF seleccionado con firma `iisignature` sobre datos simulados.

Nuevamente, las barras representan la importancia promedio por nivel y los puntos la mediana. El patrón global es consistente: los niveles altos tienden a concentrar mayor importancia típica que los niveles bajos, lo que sugiere que el modelo aprovecha principalmente términos de orden alto. Las diferencias entre promedio y mediana permiten además identificar niveles donde la señal se concentra en pocas variables particularmente influyentes, versus niveles donde la importancia está más repartida entre muchas variables.

Finalmente, se evaluó si el conjunto completo de 1022 características correspondía a una representación adecuada o si era posible alcanzar un rendimiento competitivo con una representación más simple. Por ello, se comparó el desempeño del modelo utilizando subconjuntos acumulativos de características hasta cada nivel de firma.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	Acc Test	F1 Test	AUC Test
1	2	0.579	0.571	0.594	0.585	0.636
2	6	0.632	0.631	0.584	0.591	0.689
3	14	0.715	0.710	0.655	0.653	0.741
4	30	0.752	0.747	0.682	0.686	0.773
5	62	0.797	0.794	0.719	0.726	0.802
6	126	0.825	0.823	0.748	0.752	0.823
7	254	0.846	0.845	0.761	0.768	0.840
8	510	0.868	0.867	0.785	0.791	0.857
9	1022	0.887	0.886	0.814	0.817	0.873

Tabla 4.6: Comparación del rendimiento del mejor modelo RF con firma `esig` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos simulados.

El rendimiento mejora a medida que se incorporan características de mayor orden. El AUC de prueba aumenta desde 0.636 hasta 0.873 en el nivel 9 alcanzando el mejor desempeño general, junto con una accuracy de prueba de 0.814 y un F1-score de 0.817. Por lo tanto, el nivel 9 corresponde al nivel más simple que maximiza el desempeño y respalda el uso de la representación de 1022 características.

Posteriormente, se evaluó la representación basada en log-firma. En este caso, el tiempo total de ejecución se redujo a 1 hora, 48 minutos y 23 segundos, lo que representa una ventaja computacional importante respecto de la firma estándar. Sin embargo, el rendimiento fue inferior. Aunque las diferencias en AUC promedio entre los cinco mejores modelos fueron pequeñas, el mejor comportamiento en el conjunto de prueba correspondió a RF_5, con un accuracy de 0.6260 y un F1-score ponderado de 0.6382.

Primer modelo

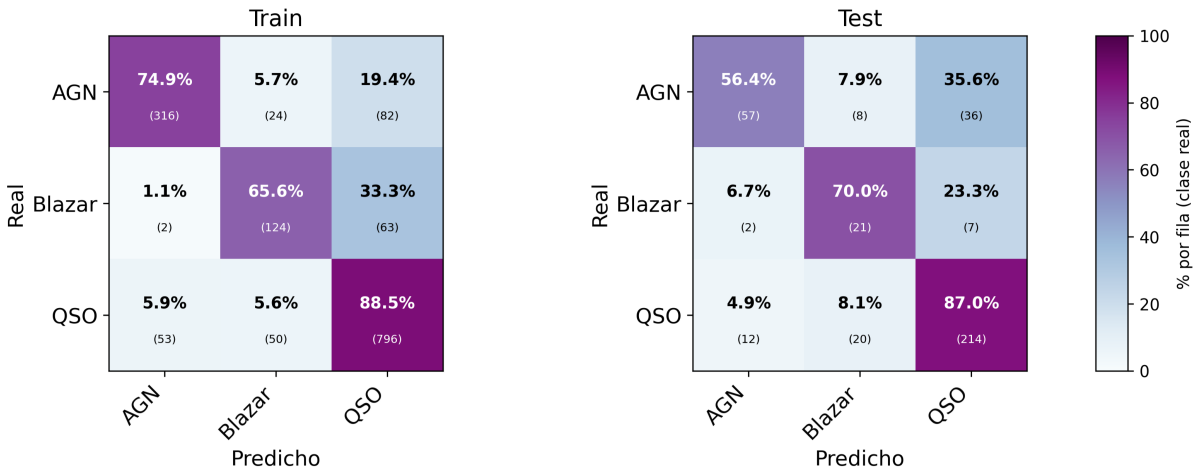


Figura 4.9: Matrices de confusión del entrenamiento y la prueba para el mejor modelo RF utilizando log-firma (`iisignature`) sobre datos simulados.

```
Accuracy : 0.831
Precision: 0.832
Recall   : 0.831
F1-score : 0.828
```

```
Accuracy : 0.780
Precision: 0.784
Recall   : 0.780
F1-score : 0.778
```

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
RF_1	0.5795	0.0174	0.3415	0.5771	0.0200	0.3436	0.7798	0.7775
RF_2	0.5784	0.0155	0.3235	0.5765	0.0187	0.3252	0.7507	0.7487
RF_3	0.5783	0.0176	0.2544	0.5746	0.0174	0.2575	0.6897	0.6924
RF_4	0.5781	0.0177	0.3141	0.5766	0.0198	0.3158	0.7427	0.7416
RF_5	0.5781	0.0164	0.2938	0.5765	0.0182	0.2957	0.7294	0.7328

Tabla 4.7: Desempeño de los cinco mejores modelos RF utilizando log-firma (`iisignature`) sobre datos simulados.

En conjunto, la comparación entre ambas representaciones muestra que la firma ofrece un mejor desempeño predictivo que la log-firma, aun cuando exige un mayor tiempo de cómputo. Por esta razón, dentro de los resultados obtenidos con `iisignature` se privilegia la representación basada en la firma.

Para complementar la interpretación del modelo final, se analizó la importancia de las características agrupadas por nivel de firma, considerando tanto la importancia promedio como la mediana.

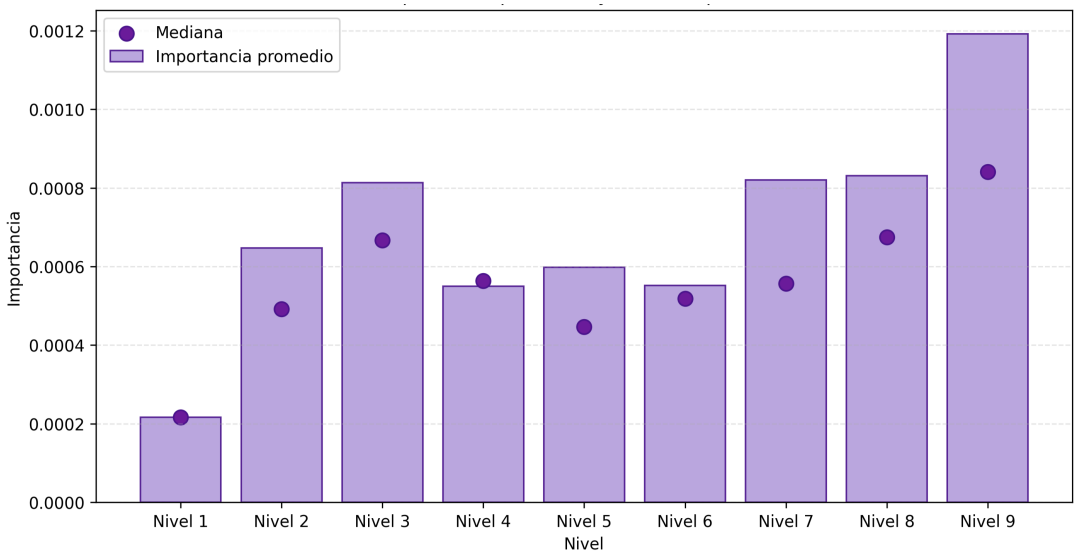


Figura 4.10: Importancia promedio y mediana por nivel para el modelo RF seleccionado con log-firma `iisignature` sobre datos simulados.

Los resultados muestran que las características más relevantes se concentran en los niveles superiores de la firma. En particular, el nivel 9 presenta la mayor importancia promedio y la mediana más alta, mientras que el nivel 1 aporta una contribución mucho menor. Además, la separación entre media y mediana en los niveles más altos sugiere la presencia de variables especialmente influyentes dentro de dichos niveles. Este resultado refuerza que los términos de orden superior contienen la información más discriminante para la clasificación.

Finalmente, se evaluó el efecto de utilizar subconjuntos acumulativos de características hasta cada nivel de firma. Los resultados muestran que el rendimiento mejora al incorporar características de mayor orden y que el mejor comportamiento se alcanza en el nivel 9. En particular, este nivel obtiene un AUC de prueba de 0.848, un accuracy de 0.793 y un F1-score de 0.792, lo que respalda el uso de la representación con nivel 9.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	Acc Test	F1 Test	AUC Test
1	2	0.549	0.542	0.562	0.576	0.619
2	6	0.621	0.615	0.592	0.596	0.670
3	14	0.677	0.670	0.637	0.634	0.712
4	30	0.730	0.723	0.674	0.677	0.743
5	62	0.756	0.749	0.695	0.701	0.772
6	126	0.785	0.779	0.732	0.734	0.797
7	254	0.803	0.800	0.743	0.747	0.816
8	510	0.831	0.828	0.767	0.770	0.831
9	1021	0.849	0.846	0.793	0.792	0.848

Tabla 4.8: Comparación del rendimiento del mejor modelo RF con log-firma `iisignature` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos simulados.

En síntesis, los resultados obtenidos con `iisignature` confirman la misma tendencia observada previamente: la firma supera a la log-firma en desempeño predictivo, aunque requiere un mayor tiempo de cómputo. Si bien la log-firma resulta más eficiente desde el punto de vista computacional, la pérdida observada en accuracy y F1-score hace preferible la representación basada en firma estándar para el modelo Random Forest.

Representación	Mejor modelo	AUC_{CV}	Acc_{train}	Acc_{test}	$F1_{w,test}$	Hiperparámetros							Tiempo total
						Crit.	Depth	Max feat.	Max samp.	Min leaf	Min split	Trees	
ESIG firma	RF_1	0.6047	0.879	0.8223	0.8265	entropy	15	0.8	0.8	13	41	3141	14:11:36
ESIG log-firma	RF_1	0.5795	0.831	0.7798	0.7775	gini	None	0.3	0.6	6	29	2465	02:18:30
IISIGNATURE firma	RF_1	0.6020	0.887	0.8090	0.8114	entropy	15	0.8	0.8	13	41	3141	14:50:44
IISIGNATURE log-firma	RF_1	0.5555	0.854	0.7558	0.7502	gini	None	0.3	0.6	6	29	2465	01:48:23

Tabla 4.9: Comparación de los mejores modelos RF obtenidos con `esig` e `iisignature`, usando firma y log-firma sobre datos simulados.

En conclusión, la comparación conjunta muestra que la representación basada en firma fue superior a la log-firma en ambas librerías, lo que confirma que los términos completos de la signature conservan mejor la información discriminante necesaria para la clasificación. Entre las cuatro combinaciones evaluadas, el mejor desempeño global se obtuvo con `esig` firma, que alcanzó el mayor AUC en validación cruzada, así como las mejores métricas en el conjunto de prueba, por lo que se selecciona como la configuración final del modelo. `iisignature` firma presentó un comportamiento cercano, aunque levemente inferior, mientras que ambas variantes basadas en log-firma mostraron una caída clara en desempeño. Desde el punto de vista computacional, las representaciones log-firma reducen de manera importante el tiempo de ejecución; sin embargo, esta ventaja se obtiene a costa de una pérdida apreciable en capacidad predictiva. En consecuencia, para este problema se prioriza la firma estándar, y en particular la construida con `esig`, como la alternativa que ofrece el mejor equilibrio entre capacidad discriminativa y robustez del modelo.

4.1.2. Support vector machine

Para la clasificación mediante SVM, se construyó un clasificador utilizando un kernel RBF, ya que este permite capturar relaciones no lineales entre las variables, lo cual resulta especialmente útil en datos astronómicos de alta dimensionalidad como las representaciones basadas en path signature. Previamente, se incorporó una etapa de estandarización para llevar todas las características a una escala comparable. Posteriormente, se aplicó una selección de variables con el objetivo de reducir ruido y conservar únicamente aquellas con mayor capacidad informativa; la cantidad de variables seleccionadas también se trató como hiperparámetro del modelo. Dado el desbalance entre las clases, se incorporó SMOTE dentro del pipeline de validación cruzada, de modo que la generación de muestras sintéticas para las clases minoritarias ocurriera únicamente sobre los datos de entrenamiento de cada fold, evitando así sesgos de evaluación. Adicionalmente, se utilizaron pesos de clase para penalizar en mayor medida los errores en AGN y Blazar, reforzando la capacidad del clasificador para distinguir estas clases menos representadas. Los principales hiperparámetros del modelo, en particular C, γ, los pesos de clase y el número de variables seleccionadas, fueron ajustados mediante validación cruzada estratificada de 5 folds, utilizando un criterio de selección orientado a privilegiar un rendimiento equilibrado entre clases.

Además, se exploraron distintas configuraciones para el número de variables seleccionadas mediante SelectKBest, evaluando valores entre 50 y 500 características. Para los hiperparámetros del modelo se probaron valores de $C \in \{0{,}1, 1, 10, 50, 100\}$ y $\gamma \in \{\texttt{scale}, 0{,}1, 0{,}01, 0{,}001\}$.

esig

Una vez concluido el ajuste, se seleccionaron las cinco configuraciones con mejor desempeño global. Para cada una de ellas se reportaron métricas resumidas de validación cruzada y de prueba, incluyendo AUC promedio, desviación estándar, brechas de sobreajuste, accuracy y F1-score ponderado. El tiempo total de cómputo fue de 19 minutos y 22 segundos.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$\text{F1}_{\text{w,test}}$
SVM_1	0.5137	0.0449	0.2446	0.8458	0.0449	-0.0875	0.8117	0.8161
SVM_2	0.5087	0.0479	0.2677	0.8716	0.0479	-0.0952	0.8488	0.8506
SVM_3	0.5155	0.0455	0.2134	0.7965	0.0455	-0.0677	0.7109	0.7206
SVM_4	0.5120	0.0521	0.2758	0.8786	0.0521	-0.0908	0.7984	0.8025
SVM_5	0.5204	0.0566	0.2738	0.9101	0.0566	-0.1159	0.8462	0.8506

Tabla 4.10: Desempeño de los cinco mejores modelos SVM utilizando firma (`esig`) sobre datos simulados.

La comparación entre estos cinco modelos muestra que no existe una correspondencia directa entre el AUC promedio en validación cruzada y el mejor desempeño sobre el conjunto de prueba. Los valores de AUC_{CV} son similares entre sí, oscilando entre 0.5087 y 0.5204, mientras que los valores de AUC

en prueba presentan una mayor dispersión, con un rango entre 0.7965 y 0.9101.

Esta discrepancia se explica principalmente por el uso de SMOTE durante el entrenamiento. Al generar muestras sintéticas para balancear las clases, el clasificador enfrenta un problema más exigente en validación cruzada, lo que tiende a reducir el AUC en esta etapa. En contraste, al evaluarse sobre datos reales no balanceados, el modelo se beneficia de la mayor representatividad de la clase mayoritaria, lo que eleva el AUC en prueba.

Asimismo, todos los valores de Gap_{rep} son negativos, lo que indica que el AUC en prueba supera al AUC en entrenamiento. Este comportamiento es coherente con el cambio de distribución entre los datos de entrenamiento (balanceados artificialmente) y los datos de prueba (distribución original).

En particular, SVM_5 obtuvo el mejor desempeño global, alcanzando un AUC en prueba de 0.9101, junto con una accuracy de 0.8462 y un F1-score ponderado de 0.8506. Además, este modelo presenta el mejor equilibrio entre clases, destacando especialmente en la identificación de Blazar, que corresponde a la clase minoritaria. Por estas razones, SVM_5 fue seleccionado como modelo final.

Quinto modelo

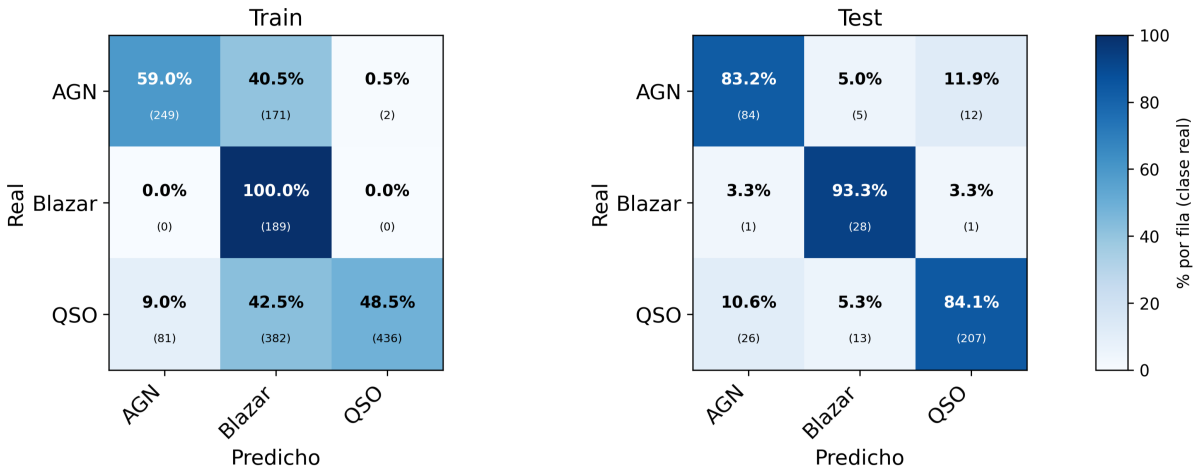


Figura 4.11: Matrices de confusión del entrenamiento y la prueba para el mejor modelo SVM utilizando firma (`esig`) sobre datos simulados.

```
Accuracy : 0.580
Precision: 0.840
Recall   : 0.580
F1-score : 0.620
```

```
Accuracy : 0.846
Precision: 0.870
Recall   : 0.846
F1-score : 0.851
```

Las métricas mostradas en los recuadros corresponden a medidas globales calculadas sobre la distribución original de los datos. En particular, se reportan la accuracy y promedios ponderados de precision, recall y F1-score, lo que permite mantener consistencia con las métricas resumidas en la tabla

de resultados. En entrenamiento, el bajo accuracy se explica porque el modelo fue optimizado sobre datos balanceados sintéticamente mediante SMOTE, mientras que estas métricas se calculan sobre la distribución real, altamente desbalanceada. En prueba, en cambio, se observa un desempeño considerablemente mejor, coherente con los valores resumidos en la tabla. Las métricas de prueba, en cambio, reflejan el desempeño real del modelo sobre datos no vistos con la distribución natural del problema, evidenciando una mejora significativa en todas las métricas globales.

Para entender mejor qué niveles de la path signature fueron más útiles para el modelo SVM, se calculó la importancia de las variables con permutation importance. Luego, esas importancias se agruparon según el nivel de la firma.

Este análisis se realizó únicamente sobre las variables que efectivamente sobrevivieron a las etapas de filtrado y selección, por lo que refleja de manera más fiel la información utilizada por el clasificador con kernel RBF.

Importancia promedio y mediana por nivel

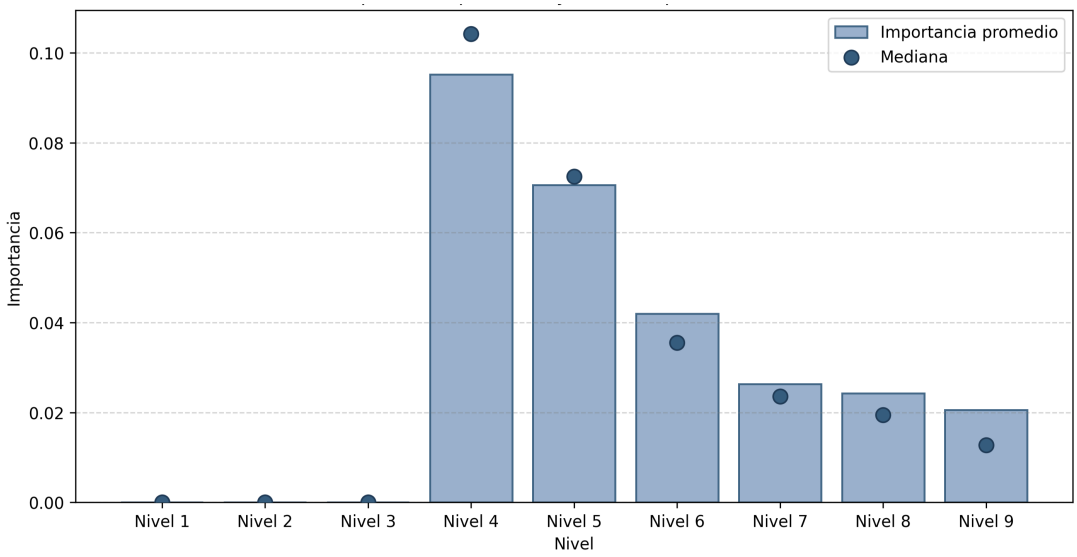


Figura 4.12: Importancia promedio y mediana por nivel para el modelo SVM seleccionado con firma `esig` sobre datos simulados.

Las barras muestran la importancia promedio de las variables en cada nivel y los puntos representan la mediana. Esto permite observar tanto el aporte general de cada nivel como el comportamiento central de sus variables. Los resultados muestran que los niveles 1, 2 y 3 no aportan al modelo final, mientras que la mayor importancia se concentra en los niveles 4 y 5. Desde el nivel 6 en adelante, la importancia disminuye de manera gradual, aunque sigue siendo positiva hasta el nivel 9. En conjunto, esto sugiere que el modelo SVM obtiene su mayor capacidad discriminante a partir de características de nivel intermedio y alto, más que de los términos más simples de la representación.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	Acc Test	F1 Test	AUC Test
1	2	0.125	0.028	0.080	0.012	0.474
2	6	0.137	0.053	0.090	0.033	0.529
3	14	0.285	0.264	0.398	0.400	0.710
4	30	0.438	0.456	0.642	0.657	0.805
5	62	0.543	0.575	0.769	0.776	0.861
6	126	0.611	0.646	0.857	0.858	0.894
7	254	0.635	0.672	0.878	0.878	0.918
8	510	0.648	0.685	0.891	0.891	0.928
9	1022	0.579	0.624	0.846	0.851	0.910

Tabla 4.11: Comparación del rendimiento del mejor modelo SVM con firma `esig` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos simulados.

Con el fin de estudiar con mayor detalle cómo influye el nivel de truncamiento de la path signature en el desempeño del clasificador, se evaluó el modelo SVM_5 desde el nivel 1 hasta el nivel 9.

Se observa una mejora progresiva en todas las métricas de prueba desde el nivel 1 hasta el nivel 8. En particular, el AUC_{rep} aumenta desde 0.4744 en el nivel 1 hasta 0.9278 en el nivel 8, acompañado también por un aumento en accuracy y F1-score ponderado. Esto indica que, a medida que se incorporan niveles más altos de la firma, el modelo dispone de una representación más exacta de la dinámica de las curvas de luz, lo que mejora su capacidad de discriminación entre las clases. Sin embargo, al incluir el nivel 9, el desempeño en prueba disminuye, con un AUC_{rep} de 0.9101, una accuracy de 0.8462 y un F1-score ponderado de 0.8506. Aunque este resultado sigue siendo bueno, es inferior al obtenido en el nivel 8. Por lo tanto, bajo el criterio de seleccionar el nivel más simple que maximiza el rendimiento en prueba, el nivel 8 corresponde a la mejor alternativa, ya que logra el mayor AUC_{rep} utilizando 510 variables. En conjunto, estos resultados sugieren que los niveles altos de la firma aportan información útil hasta cierto punto, pero que incorporar términos de orden aún mayor no necesariamente mejora la clasificación y puede afectar negativamente el rendimiento final.

En este sentido, el hecho de que el gráfico de importancia por nivel destaque principalmente a los niveles 4 y 5 no se contrapone con que el mejor rendimiento acumulado se alcance en el nivel 8. Más bien, ambos resultados se complementan: mientras el análisis de importancia muestra en qué niveles se concentran, en promedio, las variables más relevantes, el análisis por truncamiento permite ver que la incorporación acumulada de niveles superiores sigue aportando información útil al clasificador hasta el nivel 8.

Posteriormente, se evaluó la representación basada en log-firma, manteniendo la misma partición de los datos, la estrategia de validación cruzada y el esquema general de ajuste utilizados en la representa-

ción con firma estándar. Una vez concluido el proceso, se seleccionaron las cinco configuraciones con mejor desempeño y se resumieron sus métricas de validación cruzada y prueba.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
SVM_1	0.5562	0.0390	0.2359	0.8426	0.0390	-0.0505	0.7480	0.7551
SVM_2	0.5648	0.0316	0.2001	0.8159	0.0316	-0.0509	0.6923	0.7011
SVM_3	0.5152	0.0506	0.2306	0.7931	0.0506	-0.0473	0.7003	0.7101
SVM_4	0.5149	0.0550	0.2440	0.8110	0.0550	-0.0521	0.7321	0.7410
SVM_5	0.5082	0.0507	0.2442	0.8215	0.0507	-0.0692	0.7188	0.7285

Tabla 4.12: Desempeño de los cinco mejores modelos SVM utilizando log-firma (`esig`) sobre datos simulados.

A partir de esta comparación, se observa que el modelo SVM_2 obtuvo el mayor AUC promedio en validación cruzada, con un valor de 0.5648. Sin embargo, el mejor comportamiento sobre el conjunto de prueba correspondió a SVM_1, que alcanzó un AUC_{rep} de 0.8426, junto con una accuracy de 0.7480 y un F1-score ponderado de 0.7551. Esto indica que, a pesar de no ser el mejor en validación cruzada, SVM_1 presenta una mejor capacidad de generalización en datos no vistos, por lo que fue seleccionado como modelo final.

Además, todos los valores de Gap_{rep} son negativos, lo que indica que el AUC en prueba supera al AUC en entrenamiento. Este comportamiento es consistente con el uso de SMOTE durante el entrenamiento: el modelo es ajustado sobre datos balanceados artificialmente, mientras que en prueba es evaluado sobre la distribución original, donde la clase mayoritaria aporta mayor señal discriminante. El proceso completo de búsqueda y evaluación con log-firma tuvo una duración total de 3 minutos y 55 segundos.

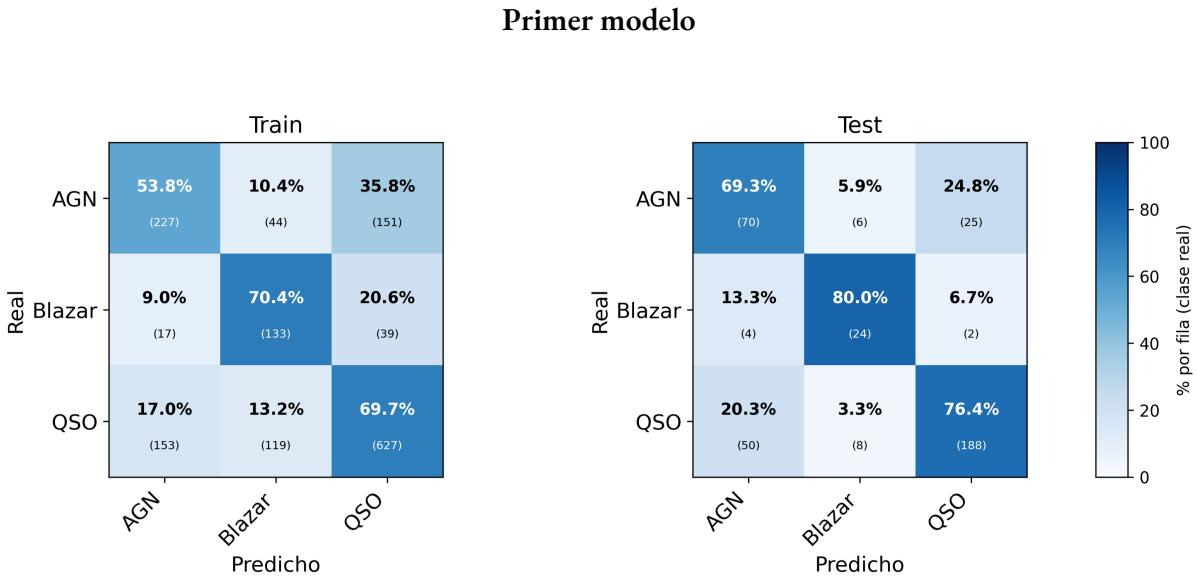


Figura 4.13: Matrices de confusión del entrenamiento y la prueba para el mejor modelo SVM utilizando log-firma (`esig`) sobre datos simulados.

```
Accuracy : 0.650
Precision: 0.670
Recall   : 0.650
F1-score : 0.660
```

```
Accuracy : 0.748
Precision: 0.770
Recall   : 0.748
F1-score : 0.755
```

Las métricas mostradas en los recuadros corresponden a promedios ponderados calculados sobre la distribución original de los datos, lo que permite mantener consistencia con las métricas resumidas en la tabla. En entrenamiento, el desempeño es menor debido a que el modelo fue optimizado sobre datos balanceados sintéticamente mediante SMOTE, mientras que estas métricas se calculan sobre la distribución real, desbalanceada. En prueba, en cambio, se observa una mejora consistente, reflejando el rendimiento real del modelo sobre datos no vistos.

A partir de la matriz de confusión de prueba, se observa que el modelo logra un buen reconocimiento de la clase Blazar, con 24 aciertos de un total de 30 observaciones, lo que equivale a un recall de 0.75. La clase AGN también presenta un desempeño aceptable, con 70 aciertos de 101 casos, mientras que QSO alcanza 188 clasificaciones correctas de 246 ejemplos. Sin embargo, la principal fuente de error sigue estando en la confusión entre AGN y QSO, especialmente en ejemplos de QSO que son clasificados como AGN. En conjunto, estos resultados muestran que, aunque la representación basada en log-firma permite distinguir razonablemente bien las tres clases, su capacidad discriminante sigue siendo inferior a la obtenida con la firma estándar.

Ahora se analiza la importancia de las variables agrupadas por nivel de la log-signature, con el objetivo de identificar qué órdenes de esta representación aportan más información al modelo SVM seleccionado.

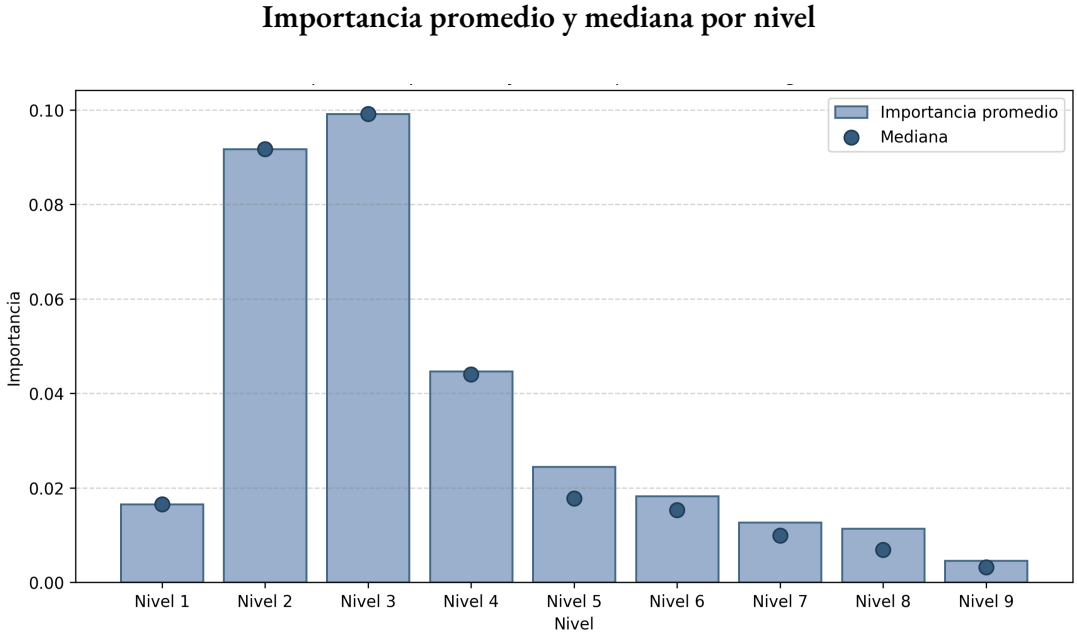


Figura 4.14: Importancia promedio y mediana por nivel para el modelo SVM seleccionado con firma `esig` sobre datos simulados.

Los resultados muestran que la mayor importancia se concentra en los niveles 2 y 3, que alcanzan los valores promedio y mediano más altos. Luego, a partir del nivel 4, la contribución disminuye de forma progresiva, aunque sigue siendo positiva hasta el nivel 9. Esto indica que, para la representación basada en log-firma, el modelo SVM obtiene la mayor parte de su capacidad discriminante a partir de niveles bajos e intermedios, mientras que los niveles más altos tienen un aporte menor. En otras palabras, la información más útil para clasificar las curvas de luz parece estar contenida principalmente en los primeros términos de la log-firma, y no en los de orden más alto.

Ahora se analiza el comportamiento del modelo SVM basado en log-firma al incorporar de manera acumulada los distintos niveles de truncamiento de la representación. A diferencia de lo observado con la firma estándar, en este caso se aprecia una mejora progresiva en todas las métricas de prueba desde el nivel 1 hasta el nivel 9. En particular, el AUC_{rep} aumenta desde 0.5397 en el nivel 1 hasta 0.8426 en el nivel 9, acompañado también por un incremento sostenido en accuracy y F1-score ponderado. Esto indica que, a medida que se agregan niveles superiores de la log-firma, el modelo dispone de una representación cada vez más informativa de la dinámica de las curvas de luz, lo que mejora su capacidad de discriminación entre las clases.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	Acc Test	F1 Test	AUC Test
1	2	0.3616	0.3883	0.5385	0.5425	0.5397
2	3	0.4344	0.4498	0.5225	0.5265	0.6012
3	5	0.5212	0.5263	0.5517	0.5714	0.6989
4	8	0.5457	0.5552	0.6021	0.6161	0.7223
5	14	0.5735	0.5808	0.6446	0.6577	0.7495
6	23	0.5907	0.5961	0.6472	0.6590	0.7782
7	41	0.6205	0.6251	0.6870	0.6959	0.8060
8	71	0.6391	0.6436	0.7268	0.7322	0.8249
9	127	0.6536	0.6586	0.7480	0.7551	0.8426

Tabla 4.13: Comparación del rendimiento del mejor modelo SVM con log-firma `esig` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos simulados.

Bajo este criterio, el mejor desempeño se alcanza en el nivel 9, con 127 variables, un AUC_{rep} de 0.8426, una accuracy de 0.7480 y un F1-score ponderado de 0.7551. Por lo tanto, para la representación basada en log-firma, el nivel 9 corresponde a la mejor alternativa entre los niveles evaluados. En conjunto, estos resultados sugieren que, en este caso, los niveles superiores no introducen un deterioro en el rendimiento, sino que aportan información complementaria que permite mejorar gradualmente la clasificación hasta el máximo nivel considerado.

Este resultado también ayuda a entender mejor el gráfico de importancia por nivel. Aunque los niveles 2 y 3 son los que muestran mayor importancia promedio por variable, eso no significa que por sí solos

produzcan el mejor resultado final. Al ir agregando los niveles siguientes, el modelo sigue mejorando su desempeño hasta llegar al nivel 9. En otras palabras, los primeros niveles parecen contener las variables más fuertes de manera individual, pero los niveles superiores también aportan información adicional que, al combinarse con las anteriores, permite mejorar la clasificación.

IISIGNATURE

Posteriormente, se evaluó la representación basada en firma utilizando `iisignature`, manteniendo la misma partición de los datos, la estrategia de validación cruzada y el esquema general de ajuste utilizados en los análisis anteriores. A partir de la búsqueda realizada, se seleccionaron las cinco configuraciones con mejor desempeño. En este caso, el mejor valor de la métrica robusta en validación cruzada fue 0.2631, con una configuración que utilizó kbest $=$ 1023, $C = 10{,}7722$, ponderación de clases balanced y $\gamma = 0{,}0583$.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
SVM_1	0.5081	0.0512	0.2635	0.8623	0.0512	-0.0907	0.8462	0.8480
SVM_2	0.5048	0.0570	0.2511	0.8422	0.0570	-0.0863	0.7958	0.8011
SVM_3	0.5102	0.0467	0.2108	0.7988	0.0467	-0.0778	0.7188	0.7290
SVM_4	0.5082	0.0524	0.2763	0.8724	0.0524	-0.0880	0.8223	0.8256
SVM_5	0.5335	0.0269	0.1846	0.7657	0.0269	-0.0476	0.7003	0.7070

Tabla 4.14: Desempeño de los cinco mejores modelos SVM utilizando firma (`iisignature`) sobre datos simulados.

El tiempo total de cómputo fue de 4 minutos y 1 segundo.

Al comparar los cinco modelos, se observa que el mejor desempeño sobre el conjunto de prueba correspondió a SVM_4, que alcanzó un AUC_{rep} de 0.8724, junto con una accuracy de 0.8223 y un F1-score ponderado de 0.8256. Aunque SVM_5 obtuvo el mayor AUC en validación cruzada (0.5335), su rendimiento en prueba fue considerablemente inferior. Esto indica que SVM_4 presenta una mejor capacidad de generalización en datos no vistos, por lo que fue seleccionado como modelo final.

Además, todos los valores de Gap_{rep} son negativos, lo que indica que el AUC en prueba supera al AUC en entrenamiento. Este comportamiento es consistente con el uso de SMOTE durante el entrenamiento, donde el modelo se ajusta sobre una distribución balanceada artificialmente y posteriormente es evaluado sobre la distribución original de los datos.

Cuarto modelo

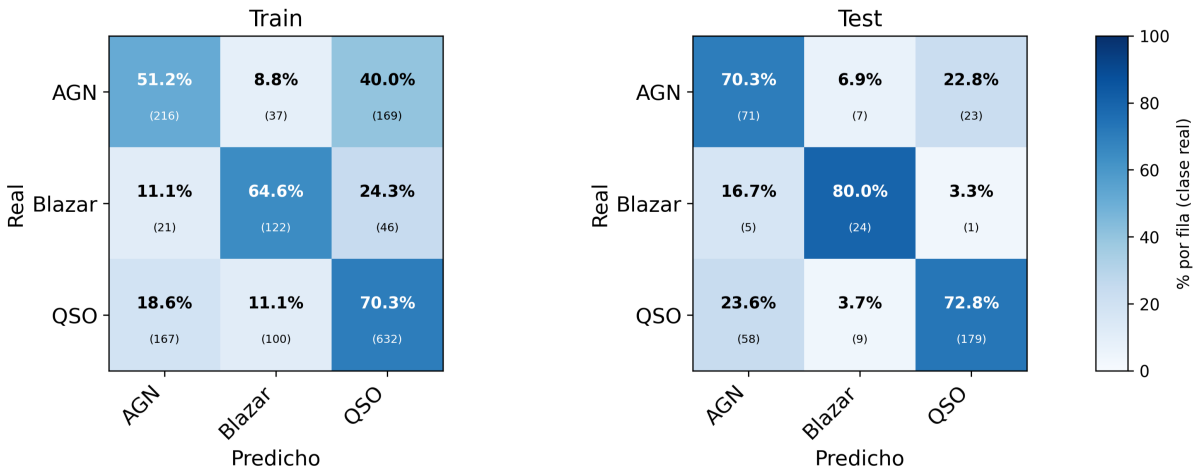


Figura 4.15: Matrices de confusión del entrenamiento y la prueba para el mejor modelo SVM utilizando firma (`iisignature`) sobre datos simulados.

```
Accuracy : 0.560
Precision: 0.810
Recall   : 0.560
F1-score : 0.590
```

```
Accuracy : 0.822
Precision: 0.830
Recall   : 0.822
F1-score : 0.826
```

A partir de la matriz de confusión de prueba, se observa que el modelo logra un buen reconocimiento de la clase Blazar, con 24 aciertos de 30 observaciones, lo que corresponde a un recall de 0.822. La clase AGN presenta 81 clasificaciones correctas de 101 casos, mientras que QSO alcanza 214 aciertos de 246 ejemplos. En conjunto, estos resultados muestran que el modelo logra una separación adecuada entre las tres clases, con un desempeño global sólido y particularmente favorable en la identificación de QSO y Blazar.

Ahora se analiza la importancia de las variables agrupadas por nivel de la path signature, con el fin de identificar qué órdenes de esta representación aportan más información al modelo SVM seleccionado con `iisignature`. Para ello, se calculó la importancia de las variables mediante permutation importance sobre el pipeline final y luego se agruparon según el nivel correspondiente. En la figura, las barras representan la importancia promedio de las variables en cada nivel y los puntos muestran la mediana.

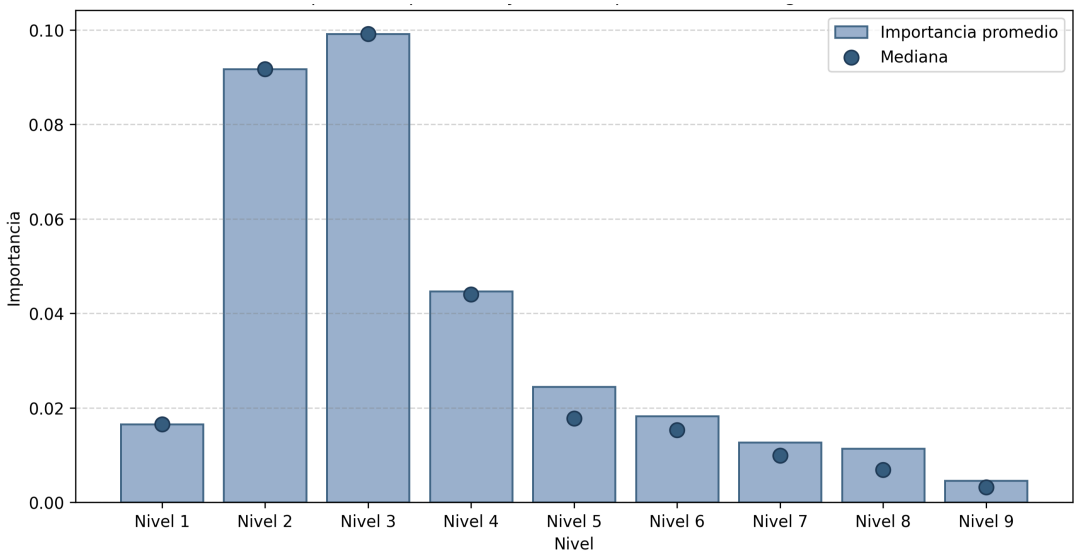


Figura 4.16: Importancia promedio y mediana por nivel para el modelo SVM seleccionado con firma `iisignature` sobre datos simulados.

Los resultados indican que la mayor importancia se concentra en el nivel 2, seguido por el nivel 3. A partir de ahí, el aporte disminuye de forma progresiva a medida que aumenta el nivel de la firma. Los niveles 4 y 5 todavía muestran una contribución positiva, pero menor que la observada en los niveles bajos, mientras que desde el nivel 6 en adelante la importancia es bastante reducida. En particular, los niveles 8 y 9 prácticamente no aportan al modelo final. En conjunto, esto sugiere que, para la representación basada en firma con `iisignature`, el modelo SVM obtiene la mayor parte de su capacidad discriminante a partir de niveles bajos e intermedios, mientras que los términos de orden más alto tienen un papel mucho menor.

Ahora se analiza el comportamiento del modelo SVM basado en log-firma al incorporar de manera acumulada los distintos niveles de truncamiento de la representación. Los resultados muestran una mejora progresiva en todas las métricas de prueba desde el nivel 1 hasta el nivel 9. En particular, el AUC_{rep} aumenta desde 0.5357 en el nivel 1 hasta 0.8418 en el nivel 9, mientras que el accuracy y el F1-score ponderado también crecen de forma sostenida, alcanzando valores de 0.7268 y 0.7367, respectivamente. Esto indica que, a medida que se incorporan niveles superiores de la log-firma, el modelo dispone de una representación más completa de la dinámica de las curvas de luz, lo que mejora su capacidad de discriminación entre las clases. Por ende, el mejor desempeño se alcanza en el nivel 9, con 127 variables, por lo que este corresponde al nivel más simple que maximiza el AUC en prueba. A diferencia de lo observado en otros casos, aquí no se aprecia una caída del rendimiento al incorporar el nivel más alto, sino una mejora continua hasta el final. En conjunto, esto sugiere que, para esta representación basada en log-firma, la información útil para la clasificación no se concentra solo en los primeros niveles, sino que sigue aumentando al agregar términos de orden superior.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	Acc Test	F1 Test	AUC Test
1	2	0.4305	0.4422	0.5252	0.5281	0.5357
2	3	0.5106	0.5064	0.5358	0.5344	0.5977
3	5	0.5073	0.5163	0.5544	0.5729	0.7018
4	8	0.5450	0.5497	0.5889	0.6041	0.7254
5	14	0.5795	0.5815	0.6366	0.6523	0.7456
6	23	0.5960	0.5988	0.6684	0.6787	0.7860
7	41	0.6132	0.6164	0.6976	0.7061	0.8110
8	71	0.6179	0.6215	0.6897	0.7004	0.8361
9	127	0.6424	0.6453	0.7268	0.7367	0.8418

Tabla 4.15: Comparación del rendimiento del mejor modelo SVM con firma `iisignature` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos simulados.

Este resultado complementa el análisis de importancia por nivel presentado anteriormente. Aunque los niveles bajos muestran una mayor importancia promedio por variable, el análisis acumulado indica que los niveles superiores siguen aportando información útil cuando se incorporan de manera conjunta. En otras palabras, las variables de los primeros niveles parecen ser las más influyentes de forma individual, pero la inclusión de los niveles siguientes permite mejorar gradualmente el rendimiento global del modelo hasta alcanzar su mejor resultado en el nivel 9.

Posteriormente, se evaluó la representación basada en log-firma utilizando `iisignature`, manteniendo la misma partición de los datos, la estrategia de validación cruzada y el esquema general de ajuste utilizados en los análisis anteriores. A partir de la búsqueda realizada, se seleccionaron las cinco configuraciones con mejor desempeño. En este caso, el mayor valor de la métrica robusta en validación cruzada fue 0.2631, con una configuración que utilizó kbest $= 1023$, $C = 10{,}7722$, ponderación de clases balanceada y $\gamma = 0{,}0583$.

Sin embargo, al comparar el comportamiento final de los cinco modelos sobre el conjunto de prueba, se observa que el mejor desempeño correspondió a SVM_1, con un AUC_{rep} de 0.8418, una accuracy de 0.7268 y un F1-score ponderado de 0.7367. Por esta razón, SVM_1 fue seleccionado como modelo final para la representación basada en firma con `iisignature`, privilegiando el rendimiento en datos no vistos por sobre pequeñas diferencias en validación cruzada.

Además, en los cinco modelos evaluados el valor de Gap_{rep} fue negativo, lo que indica que el AUC en prueba superó al AUC obtenido en entrenamiento.

El tiempo de cómputo fue de 19 minutos y 38 segundos.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
SVM_1	0.5491	0.0332	0.2257	0.8418	0.0332	-0.0671	0.7268	0.7367
SVM_2	0.5067	0.0577	0.2313	0.8101	0.0577	-0.0721	0.7162	0.7251
SVM_3	0.5032	0.0538	0.2446	0.8136	0.0538	-0.0658	0.7294	0.7380
SVM_4	0.5493	0.0252	0.1945	0.8205	0.0252	-0.0767	0.6817	0.6929
SVM_5	0.5035	0.0571	0.2442	0.8378	0.0571	-0.0901	0.7321	0.7411

Tabla 4.16: Desempeño de los cinco mejores modelos SVM utilizando log-firma (`iisignature`) sobre datos simulados.

A partir de la matriz de confusión de prueba, se observa que el modelo logra un reconocimiento razonablemente equilibrado de las tres clases. En la clase AGN se obtienen 62 clasificaciones correctas de 101 observaciones, mientras que en Blazar se logran 24 aciertos de 30 casos. Por su parte, la clase QSO alcanza 171 clasificaciones correctas de 246 ejemplos. En conjunto, estos resultados muestran que el modelo mantiene una capacidad de separación adecuada entre las tres clases, con un comportamiento especialmente favorable en la identificación de Blazar y QSO.

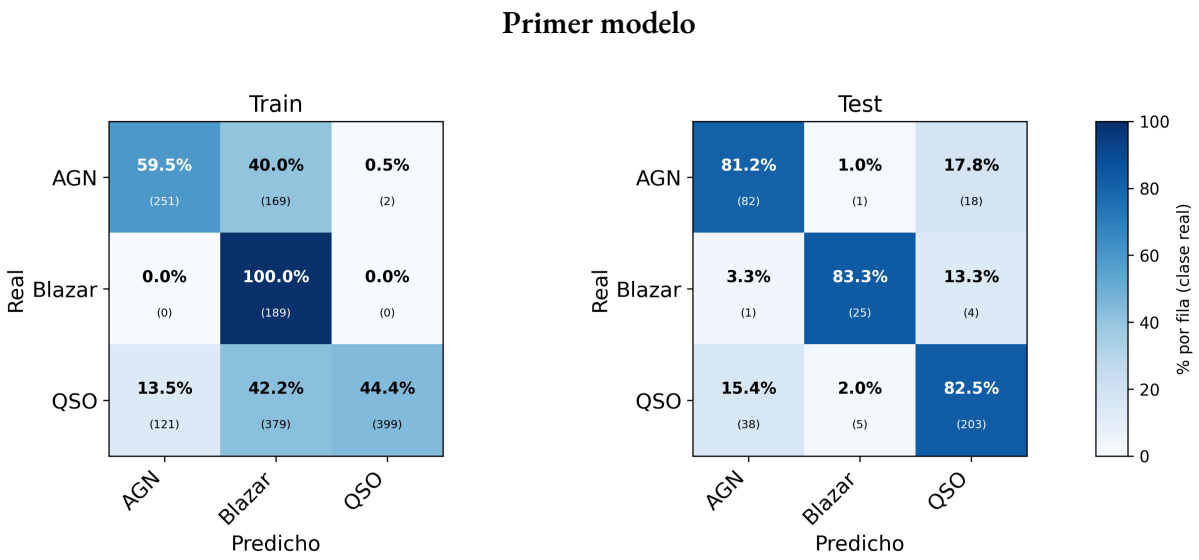


Figura 4.17: Matrices de confusión del entrenamiento y la prueba para el mejor modelo SVM utilizando log-firma (`iisignature`) sobre datos simulados.

```
Accuracy : 0.640
Precision: 0.650
Recall   : 0.640
F1-score : 0.650
```

```
Accuracy : 0.727
Precision: 0.770
Recall   : 0.727
F1-score : 0.737
```

Ahora se analiza la importancia de las variables agrupadas por nivel de la path signature, con el fin de identificar qué órdenes de la representación aportan más información al modelo SVM seleccionado con `iisignature`. En la figura, las barras representan la importancia promedio de las variables en cada nivel y los puntos muestran la mediana.

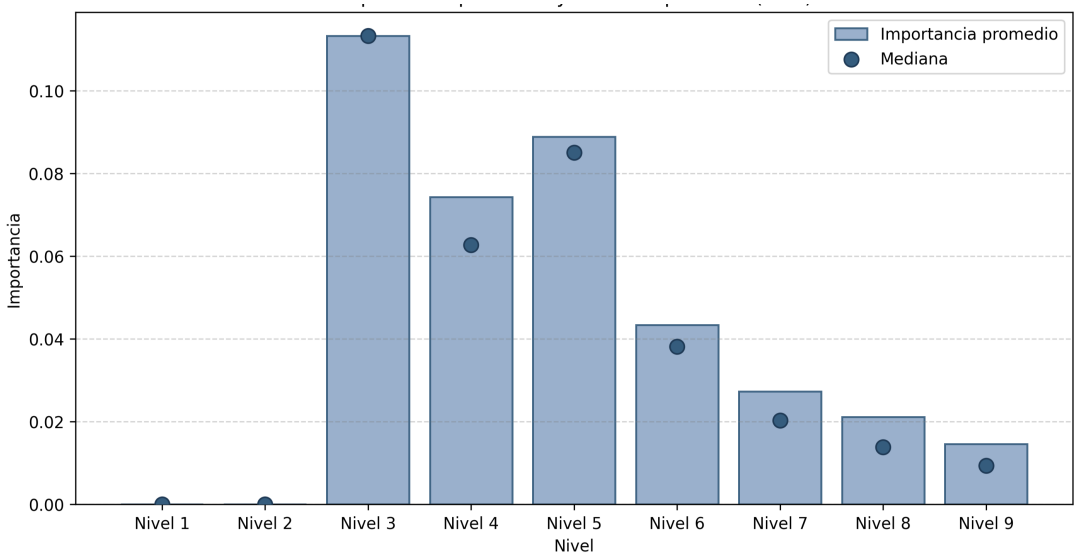


Figura 4.18: Importancia promedio y mediana por nivel para el modelo SVM seleccionado con log-firma `iisignature` sobre datos simulados.

Los resultados muestran que los niveles 1 y 2 no aportan al modelo final, mientras que la mayor importancia se concentra en el nivel 3. Luego, los niveles 4 y 5 también presentan una contribución importante, aunque menor que la observada en el nivel 3. A partir del nivel 6, la importancia disminuye de forma progresiva, y los niveles 8 y 9 muestran un aporte más reducido. En conjunto, esto sugiere que, para la representación basada en firma con `iisignature`, el modelo SVM obtiene la mayor parte de su capacidad discriminante a partir de niveles intermedios, especialmente desde el nivel 3 al 5, mientras que los términos de orden más alto tienen un papel menor. Este resultado sugiere que las variables más influyentes se concentran principalmente en niveles intermedios de la firma, por lo que agregar términos de orden superior no necesariamente implica una mejora importante en la contribución individual de las características.

Ahora se analiza el comportamiento del modelo SVM_5 al incorporar de manera acumulada los distintos niveles de la path signature obtenida con `iisignature`. Los resultados muestran que el desempeño en prueba mejora de forma general a medida que se agregan nuevos niveles de la representación. En particular, el AUC_{rep} aumenta desde 0.5135 en el nivel 1 hasta 0.7658 en el nivel 9, mientras que la accuracy y el F1-score ponderado también presentan una tendencia creciente, alcanzando en el nivel más alto valores de 0.7003 y 0.707, respectivamente. Esto indica que, en este caso, incorporar niveles superiores de la firma permite enriquecer la información disponible para el clasificador y mejorar su capacidad de discriminación entre las clases.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	AUC Train	Acc Test	F1 Test	AUC Test
1	2	0.3384	0.3707	0.5032	0.0796	0.0117	0.5135
2	6	0.4185	0.4352	0.5260	0.2255	0.2485	0.5330
3	14	0.5563	0.5182	0.5981	0.5809	0.5676	0.6035
4	30	0.5583	0.5301	0.6172	0.6021	0.5955	0.6581
5	62	0.5523	0.5358	0.6398	0.5703	0.5780	0.6803
6	126	0.5530	0.5483	0.6578	0.5836	0.5967	0.7171
7	254	0.5868	0.5815	0.6795	0.6207	0.6359	0.7395
8	510	0.6225	0.6136	0.7061	0.6870	0.6943	0.7616
9	1022	0.6404	0.6294	0.7183	0.7003	0.7070	0.7658

Tabla 4.17: Comparación del rendimiento del mejor modelo SVM con log-firma `iisignature` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos simulados.

Bajo este criterio, el mejor desempeño se alcanza en el nivel 9, con 1022 variables, por lo que este corresponde al nivel que maximiza el AUC en prueba. Sin embargo, también se observa que la mejora se vuelve más gradual en los niveles altos. Por ejemplo, entre los niveles 8 y 9 el AUC_{rep} aumenta solo de 0.7616 a 0.7658, lo que sugiere que, aunque los niveles superiores siguen aportando información útil, su contribución adicional es menor que la observada en los niveles intermedios. Estos resultados muestran que la representación se beneficia de la incorporación acumulada de niveles, aunque las mayores ganancias se producen antes de llegar al máximo nivel considerado.

Este resultado complementa el análisis de importancia por nivel presentado anteriormente. Aunque la mayor importancia promedio por variable se concentró de los niveles 3 al 5, el análisis acumulado muestra que seguir incorporando niveles hasta el nivel 9 continúa mejorando el rendimiento global del modelo. Esto sugiere que los niveles intermedios contienen las variables más influyentes de manera individual, mientras que los niveles superiores aportan información adicional que, aunque resulta más moderada, sigue ayudando al clasificador cuando se considera en conjunto.

Representación	Mejor modelo	AUC_{CV}	Acc_{train}	Acc_{test}	$F1_{w,test}$	Hiperparámetros							Tiempo total
						k best	C	γ	Class weight	Imp.	Scaler	SMOTE	
ESIG firma	SVM_5	0.5204	0.580	0.8462	0.8506	512	61.7502	0.05020	{1,0, 3,0, 0,7}	median	Standard	Sí	00:19:22
ESIG log-firma	SVM_1	0.5562	0.650	0.7480	0.7551	1023	10.7722	0.05826	balanced	median	Standard	Sí	00:03:55
IISIGNATURE firma	SVM_4	0.5082	0.560	0.8223	0.8256	1023	10.7722	0.05826	balanced	median	Standard	Sí	00:19:38
IISIGNATURE log-firma	SVM_1	0.5491	0.640	0.7268	0.7367	1023	10.7722	0.05826	balanced	median	Standard	Sí	00:04:01

Tabla 4.18: Comparación de los mejores modelos SVM obtenidos con `esig` e `iisignature`, usando firma y log-firma sobre datos simulados.

En conclusión, la representación basada en firma fue superior a la log-firma en ambas librerías también en el caso de SVM. Entre las cuatro combinaciones evaluadas, el mejor desempeño global se obtuvo con `esig` firma, que alcanzó el mayor AUC en el conjunto de prueba, junto con una accuracy de

0.8462 y un F1-score ponderado de 0.8506. `iisignature` firma presentó un comportamiento cercano, aunque levemente inferior, mientras que ambas variantes basadas en log-firma mostraron una disminución clara en desempeño. Desde el punto de vista computacional, las representaciones log-firma reducen de manera importante el tiempo de ejecución, ya que bajan de cerca de 20 minutos a alrededor de 4 minutos; sin embargo, esta ventaja se obtiene a costa de una pérdida apreciable en capacidad predictiva. En consecuencia, para este problema también se prioriza la firma estándar, y en particular la construida con `esig`, como la alternativa que ofrece el mejor equilibrio entre capacidad discriminativa y desempeño final del clasificador.

4.1.3. eXtreme gradient boosting

esig

El modelo XGBoost se construyó mediante gradient boosting con árboles de decisión, optimizando una pérdida multiclase y entregando probabilidades por clase. El ajuste se realizó mediante búsqueda aleatoria de hiperparámetros con validación cruzada estratificada repetida 5×2. En cada fold se aplicó early stopping, y el número final de árboles por configuración se fijó como la mediana del valor obtenido entre folds, con el fin de favorecer una selección más estable. La selección del mejor modelo se basó principalmente en el macro-F1 estimado con predicciones out-of-fold, complementado con estabilidad entre folds y métricas de balance entre clases. Se evaluaron distintas configuraciones de hiperparámetros mediante búsqueda aleatoria, considerando valores de número de estimadores entre 200 y 1500, profundidades máximas entre 3 y 12, tasas de aprendizaje entre 0.005 y 0.3, y distintas proporciones de observaciones y variables utilizadas en la construcción de cada árbol, variando entre 0.5 y 1.0. Además, se analizaron diferentes niveles de regularización mediante parámetros asociados al control de complejidad del árbol y penalizaciones L1 y L2, con el objetivo de reducir el sobreajuste y mejorar la capacidad de generalización del modelo. En primer lugar, se evaluó XGBoost utilizando como entrada las características derivadas de la signature. El proceso completo de compilación tuvo una duración total de 2 horas, 40 minutos y 12 segundos. Se resume el desempeño de las cinco mejores configuraciones obtenidas bajo esta representación.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
XGB_1	0.6021	0.0233	0.3639	0.6041	0.0240	0.3628	0.8302	0.8330
XGB_2	0.6049	0.0210	0.3662	0.6030	0.0186	0.3660	0.8223	0.8254
XGB_3	0.6030	0.0207	0.3674	0.6018	0.0206	0.3660	0.8117	0.8151
XGB_4	0.6015	0.0190	0.3793	0.6000	0.0191	0.3803	0.8568	0.8581
XGB_5	0.6045	0.0184	0.3206	0.6016	0.0196	0.3200	0.7692	0.7725

Tabla 4.19: Desempeño de los cinco mejores modelos XGBoost utilizando firma (`esig`) sobre datos simulados.

Los cinco modelos presentan un rendimiento muy similar en validación cruzada, con valores de AUC

Las penalizaciones L1 y L2 son técnicas de regularización utilizadas para reducir el sobreajuste; L1 favorece modelos más simples y L2 ayuda a estabilizar los coeficientes del modelo.

cercanos a 0.60 y desviaciones estándar bajas, lo que sugiere una estabilidad razonable entre particiones. Sin embargo, el desempeño en el conjunto de prueba permite distinguir mejor entre las configuraciones: el cuarto modelo alcanza mayor accuracy y el mayor F1-score ponderado, por lo que se selecciona como modelo final. Aunque este modelo exhibe un gap algo mayor, su mejor rendimiento sobre datos no vistos justifica su elección.

Cuarto modelo

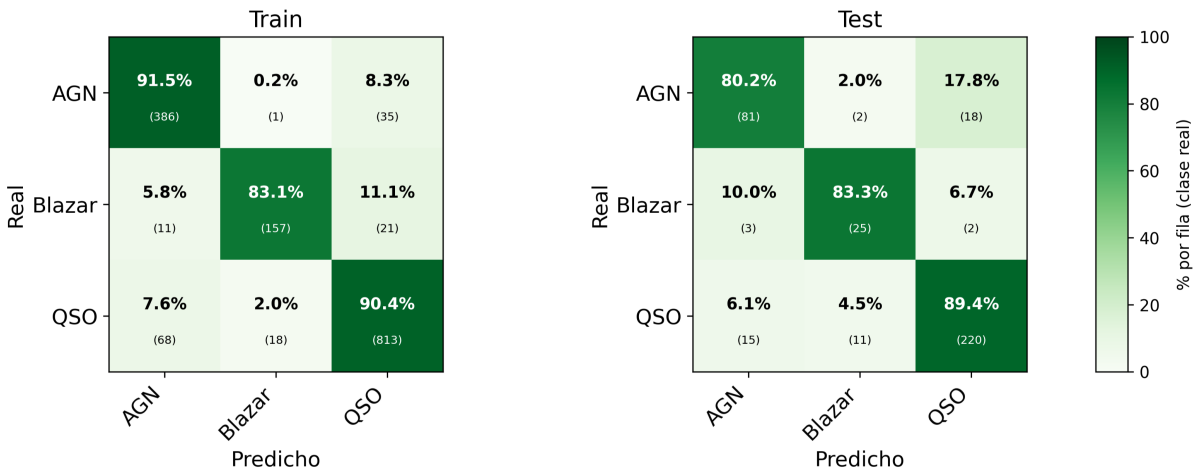


Figura 4.19: Matrices de confusión del entrenamiento y la prueba para el mejor modelo XGB utilizando firma (`esig`) sobre datos simulados.

```
Accuracy : 0.898
Precision: 0.885
Recall   : 0.883
F1-score : 0.883
```

```
Accuracy : 0.857
Precision: 0.896
Recall   : 0.857
F1-score : 0.859
```

Se analiza la importancia de las variables agrupadas por nivel de la path signature, las barras representan la importancia promedio de las variables en cada nivel y los puntos muestran la mediana. Los resultados muestran que la importancia no se concentra únicamente en los niveles más altos, sino que se distribuye de manera más amplia entre los niveles intermedios y superiores. En particular, el nivel 4 destaca como uno de los más relevantes, mientras que desde el nivel 5 hasta el nivel 9 las importancias promedio se mantienen en magnitudes similares y relativamente altas. En cambio, los niveles 1 y 2 presentan una contribución menor, y el nivel 3 marca una transición hacia una mayor relevancia. En conjunto, esto sugiere que XGBoost aprovecha información proveniente de distintos órdenes de la firma, combinando variables intermedias y altas para mejorar la clasificación.

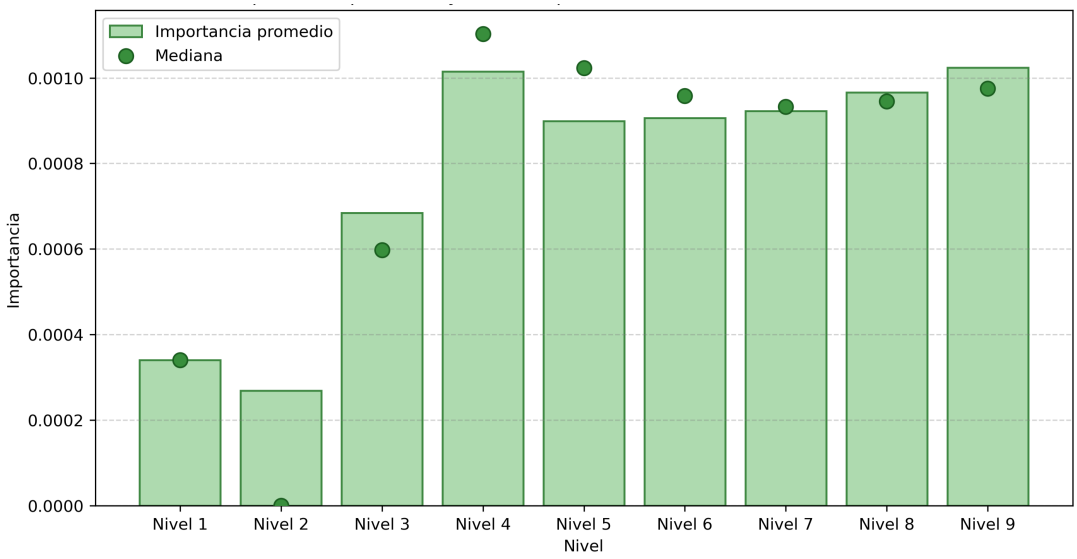


Figura 4.20: Importancia promedio y mediana por nivel para el modelo XGBoost seleccionado con firma `esig` sobre datos simulados.

A continuación, se evaluó si la representación completa de 1022 características correspondía a una elección adecuada.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	Acc Test	F1 Test	AUC Test
1	2	0.535	0.524	0.520	0.517	0.584
2	6	0.628	0.627	0.576	0.589	0.702
3	14	0.710	0.712	0.698	0.703	0.784
4	30	0.733	0.737	0.729	0.734	0.822
5	62	0.801	0.802	0.772	0.776	0.858
6	126	0.826	0.828	0.785	0.789	0.871
7	254	0.858	0.859	0.814	0.818	0.894
8	510	0.881	0.881	0.836	0.839	0.906
9	1022	0.901	0.901	0.857	0.858	0.920

Tabla 4.20: Comparación del rendimiento del mejor modelo XGB con firma `esig` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos simulados.

A medida que se incorporan niveles más altos de la signature, XGBoost mejora su desempeño en el conjunto de prueba: el AUC pasa de 0.584 en el nivel 1 a 0.920 en el nivel 9, acompañado también por aumentos en accuracy y F1-score. Bajo el criterio de maximizar el rendimiento en prueba, el mejor resultado se obtiene utilizando la representación completa.

A continuación, se aplicó el mismo procedimiento utilizando como entrada las características derivadas de la log-signature. El tiempo total de compilación fue de 11 minutos y 20 segundos, por lo que en este caso no se observó una ventaja computacional respecto de la signature estándar.

La siguiente tabla resume el desempeño de las cinco mejores configuraciones bajo esta representación.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
XGB_1	0.5689	0.0154	0.2902	0.5750	0.0224	0.2713	0.6897	0.6960
XGB_2	0.5627	0.0185	0.3760	0.5708	0.0232	0.3686	0.7745	0.7760
XGB_3	0.5711	0.0170	0.3409	0.5758	0.0216	0.3342	0.7507	0.7567
XGB_4	0.5598	0.0133	0.2509	0.5675	0.0199	0.2492	0.6817	0.6865
XGB_5	0.5604	0.0121	0.2550	0.5667	0.0190	0.2524	0.6790	0.6841

Tabla 4.21: Desempeño de los cinco mejores modelos XGBoost utilizando log-firma (`esig`) sobre datos simulados.

El mejor modelo con log-signature fue XGB_2, ya que alcanzó la mayor accuracy de prueba y el mayor F1-score ponderado entre las configuraciones evaluadas. Aun así, todos los resultados fueron inferiores a los obtenidos con signature, lo que confirma que la log-signature reduce el rendimiento predictivo del clasificador en este problema.

Segundo modelo

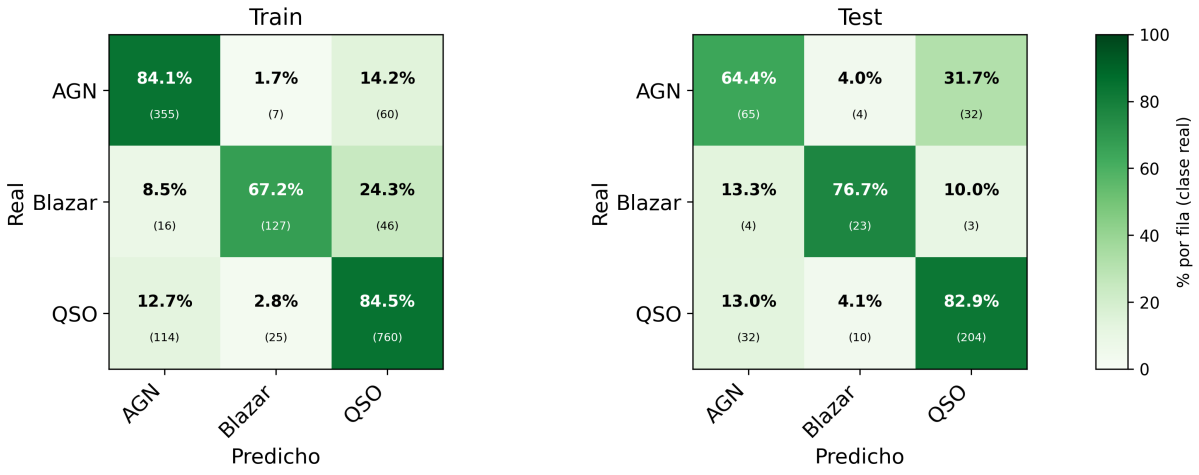


Figura 4.21: Matrices de confusión del entrenamiento y la prueba para el mejor modelo XGB utilizando log-firma (`esig`) sobre datos simulados.

```
Accuracy : 0.823
Precision: 0.803
Recall   : 0.786
F1-score : 0.791
```

```
Accuracy : 0.775
Precision: 0.706
Recall   : 0.747
F1-score : 0.776
```

En conjunto, los resultados muestran que XGBoost obtuvo un mejor desempeño con la firma estándar que con la log-firma. El mejor modelo global fue XGB_4 con signature, que superó al mejor modelo basado en log-signature en las métricas del conjunto de prueba. Esto indica que la representación completa de la trayectoria conserva de mejor forma la información útil para la clasificación, mientras que la log-signature pierde parte de esa capacidad discriminante.

Ahora se analiza la importancia de las variables agrupadas por nivel de la log-signature, con el fin de identificar qué órdenes de esta representación aportan más información al modelo XGB_2 construido con `esig` sobre datos simulados. Para ello, se utilizaron las importancias internas entregadas por XGBoost y luego se agruparon las variables según el nivel correspondiente. En la figura, las barras representan la importancia promedio de las variables en cada nivel y los puntos muestran la mediana.

Importancia promedio y mediana por nivel

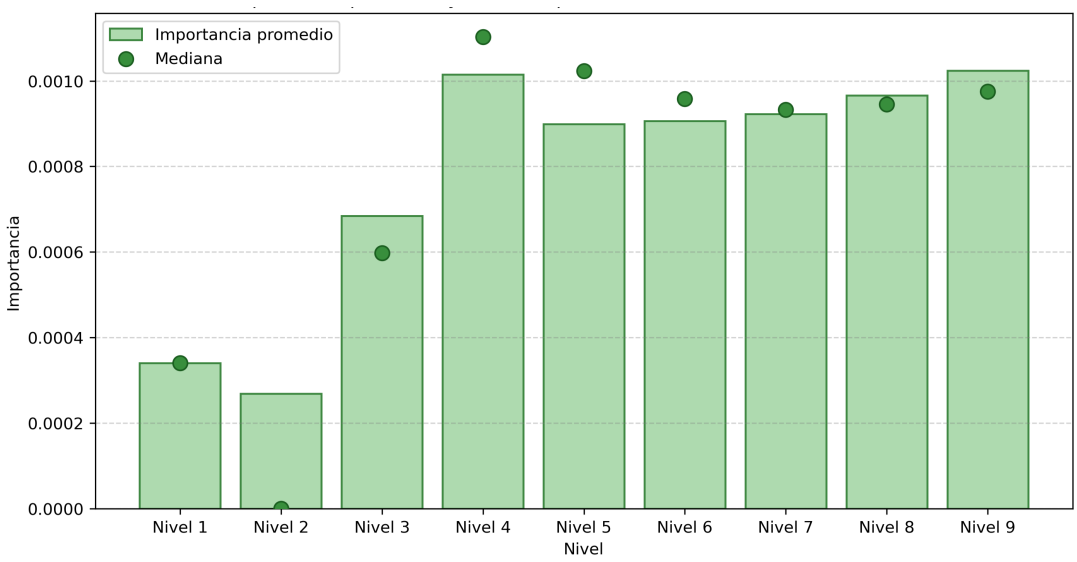


Figura 4.22: Importancia promedio y mediana por nivel para el modelo XGBoost seleccionado con log-firma `esig` sobre datos simulados.

Los resultados muestran que la importancia se distribuye de manera relativamente homogénea entre los distintos niveles de la log-signature. Sin embargo, los niveles 1 y 5 presentan las mayores importancias promedio, mientras que el resto de los niveles se mantiene en magnitudes similares. A diferencia de lo observado con la firma estándar, aquí no se aprecia un predominio tan marcado de los niveles altos, sino una contribución más repartida entre varios órdenes de la representación. Esto sugiere que el modelo extrae información útil desde distintos niveles de la log-signature para realizar la clasificación.

Ahora se analiza el comportamiento del segundo modelo al incorporar de manera acumulada los distintos niveles de la log-signature sobre los datos simulados. Los resultados muestran una mejora progresiva en las métricas de prueba desde el nivel 1 hasta el nivel 5. En particular, el AUC_{rep} aumenta

desde 0.5335 en el nivel 1 hasta 0.5884 en el nivel 5, acompañado también por un aumento en accuracy y F1-score ponderado. Sin embargo, a partir del nivel 6 el rendimiento en prueba deja de mejorar de manera consistente: el AUC_{rep} se mantiene cercano a 0.58 e incluso disminuye en los niveles más altos, mientras que las métricas de entrenamiento continúan aumentando.

Bajo el criterio de seleccionar el nivel más simple que maximiza el rendimiento en prueba, el nivel 5 corresponde a la mejor alternativa, ya que alcanza el mayor AUC_{rep} utilizando 14 variables. En conjunto, estos resultados sugieren que, para esta representación, la información más útil para la clasificación se concentra en los primeros niveles de la log-signature, mientras que incorporar términos de orden superior no mejora el desempeño final y puede introducir complejidad innecesaria.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	AUC Train	Acc Test	F1 Test	AUC Test
1	2	0.5020	0.4974	0.6156	0.4271	0.4385	0.5335
2	3	0.5285	0.5315	0.6588	0.4164	0.4333	0.5393
3	5	0.5755	0.5816	0.7218	0.4403	0.4542	0.5599
4	8	0.5987	0.6054	0.7350	0.4483	0.4619	0.5768
5	14	0.6139	0.6213	0.7592	0.4509	0.4660	0.5884
6	23	0.6285	0.6350	0.7726	0.4695	0.4829	0.5871
7	41	0.6503	0.6562	0.7960	0.4668	0.4803	0.5761
8	71	0.6834	0.6870	0.8207	0.4721	0.4809	0.5763
9	127	0.7060	0.7087	0.8376	0.4775	0.4845	0.5691

Tabla 4.22: Comparación del rendimiento del mejor modelo XGB con log-firma `esig` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos simulados.

Aunque la importancia se distribuye de manera relativamente homogénea entre varios niveles, los niveles 1 y 5 destacan como especialmente relevantes. Esto sugiere que el modelo aprovecha información útil desde distintos órdenes de la log-signature, pero que el mejor equilibrio entre información y complejidad se alcanza antes de incorporar la representación completa.

iisignature

Ahora se repitió el mismo experimento, pero generando las variables de entrada con `iisignature` en vez de la anterior. Para que la comparación fuera justa, se mantuvo exactamente lo mismo en el resto del proceso: la misma separación train/test, el mismo manejo del desbalance con pesos por clase y la misma validación cruzada repetida (5×2) para buscar hiperparámetros.

Cuarto modelo

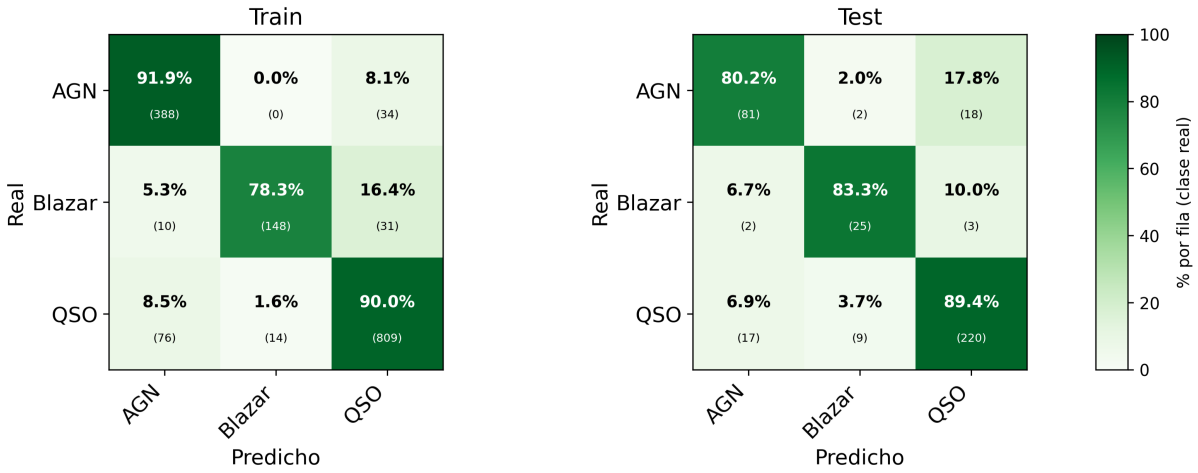


Figura 4.23: Matrices de confusión del entrenamiento y la prueba para el mejor modelo XGB utilizando firma (`iisignature`) sobre datos simulados.

```
Accuracy : 0.898
Precision: 0.885
Recall   : 0.883
F1-score : 0.883
```

```
Accuracy : 0.865
Precision: 0.797
Recall   : 0.843
F1-score : 0.867
```

La búsqueda con 30 configuraciones demoró 2 horas, 32 minutos y 22 segundos.

Al evaluar en el conjunto de prueba, el modelo que destacó fue el cuarto que logró los mejores resultados globales en test, macro-F1 y F1 ponderado, además, clasificó bien la clase minoritaria.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
XGB_1	0.5947	0.0212	0.2633	0.6017	0.0203	0.2675	0.7374	0.7425
XGB_2	0.5965	0.0240	0.2610	0.6030	0.0227	0.2649	0.7188	0.7254
XGB_3	0.5948	0.0191	0.2454	0.5973	0.0214	0.2558	0.7135	0.7227
XGB_4	0.5948	0.0234	0.3858	0.6009	0.0214	0.3814	0.8647	0.8656
XGB_5	0.5932	0.0230	0.2497	0.5989	0.0239	0.2555	0.7162	0.7241

Tabla 4.23: Desempeño de los cinco mejores modelos XGBoost utilizando firma (`iisignature`) sobre datos simulados.

Ahora se analiza la importancia de las variables agrupadas por nivel de la path signature, con el fin de identificar qué órdenes de esta representación aportan más información al modelo XGB_4 sobre datos simulados. Para ello, se utilizaron las importancias internas entregadas por XGBoost y luego se agruparon las variables según el nivel correspondiente. En la figura, las barras representan la importancia promedio de las variables en cada nivel y los puntos muestran la mediana.

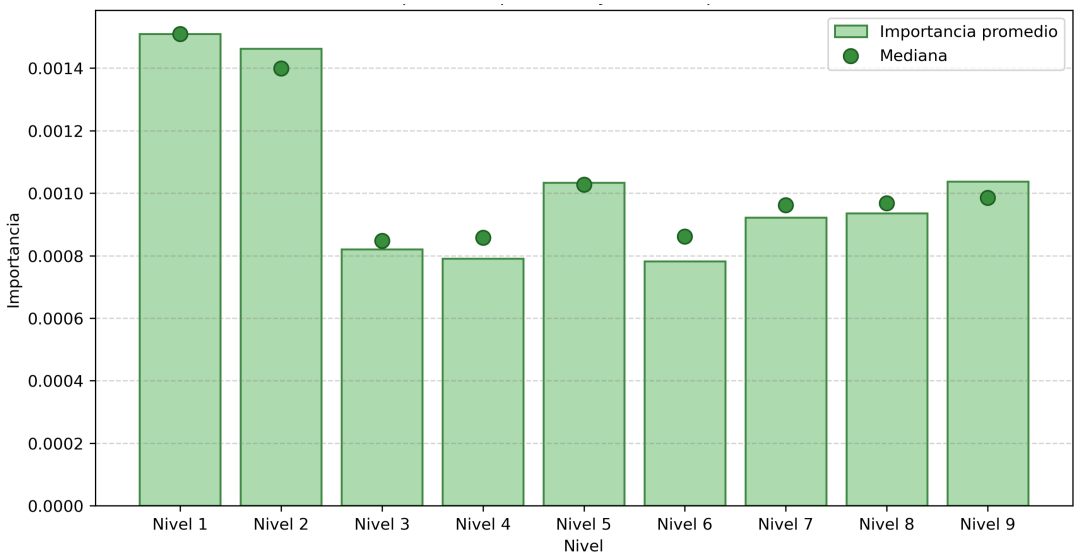


Figura 4.24: Importancia promedio y mediana por nivel para el modelo XGBoost seleccionado con firma `iisignature` sobre datos simulados.

La importancia se distribuye de manera relativamente amplia entre los distintos niveles de la firma. Los niveles 1 y 2 presentan las mayores importancias promedio, mientras que los niveles 5 y 9 también muestran contribuciones relevantes. En cambio, los niveles 3, 4 y 6 al 8 mantienen valores menores. En conjunto, esto sugiere que el modelo no depende exclusivamente de los niveles más altos, sino que combina información de niveles bajos, intermedios y altos para realizar la clasificación.

Se evalúa el comportamiento del modelo al incorporar de manera acumulada los distintos niveles de la path signature. Los resultados muestran una mejora progresiva en las métricas de prueba a medida que se agregan niveles de la representación.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	AUC Train	Acc Test	F1 Test	AUC Test
1	2	0.5219	0.5158	0.6534	0.4271	0.4267	0.5204
2	6	0.6192	0.6178	0.7576	0.4377	0.4474	0.5393
3	14	0.7007	0.7042	0.8387	0.4668	0.4791	0.5797
4	30	0.7245	0.7280	0.8645	0.4881	0.5006	0.5982
5	62	0.7901	0.7913	0.9175	0.5093	0.5161	0.5970
6	126	0.8311	0.8314	0.9370	0.5358	0.5403	0.6110
7	254	0.8536	0.8543	0.9511	0.5305	0.5381	0.6278
8	510	0.8828	0.8827	0.9692	0.5305	0.5326	0.6254
9	1022	0.8980	0.8980	0.9756	0.5411	0.5453	0.6341

Tabla 4.24: Comparación del rendimiento del mejor modelo XGB con firma `iisignature` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos simulados.

Bajo el criterio de seleccionar el nivel más simple que maximiza el rendimiento en prueba, el mejor resultado se obtiene en el nivel 9, con 1022 variables. Estos resultados sugieren que la representación completa de la firma estándar resulta más informativa que sus versiones truncadas para este modelo, ya que el desempeño mejora de manera sostenida hasta incorporar todos los niveles.

Se aplica el mismo procedimiento, pero esta vez utilizando como entrada las características de la log-signature. Se mantiene la misma partición de prueba y entrenamiento, el esquema del desbalance y la validación cruzada estratificada. El proceso de compilación tuvo una duración total de 9 minutos y 34 segundos.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
XGB_1	0.5689	0.0154	0.2902	0.5750	0.0224	0.2713	0.6897	0.6960
XGB_2	0.5627	0.0185	0.3760	0.5708	0.0232	0.3686	0.7745	0.7760
XGB_3	0.5711	0.0170	0.3409	0.5758	0.0216	0.3342	0.7507	0.7567
XGB_4	0.5598	0.0133	0.2509	0.5675	0.0199	0.2492	0.6817	0.6865
XGB_5	0.5604	0.0121	0.2550	0.5667	0.0190	0.2524	0.6790	0.6841

Tabla 4.25: Desempeño de los cinco mejores modelos XGBoost utilizando log-firma (`iisignature`) sobre datos simulados.

En la representación basada en log-signature, el mejor desempeño sobre el conjunto de prueba correspondió a XGB_4. Este modelo alcanzó una accuracy de 0.682, un F1-score de 0.687 y un macro-F1 de 0.677, por lo que se selecciona como el modelo final para esta representación.

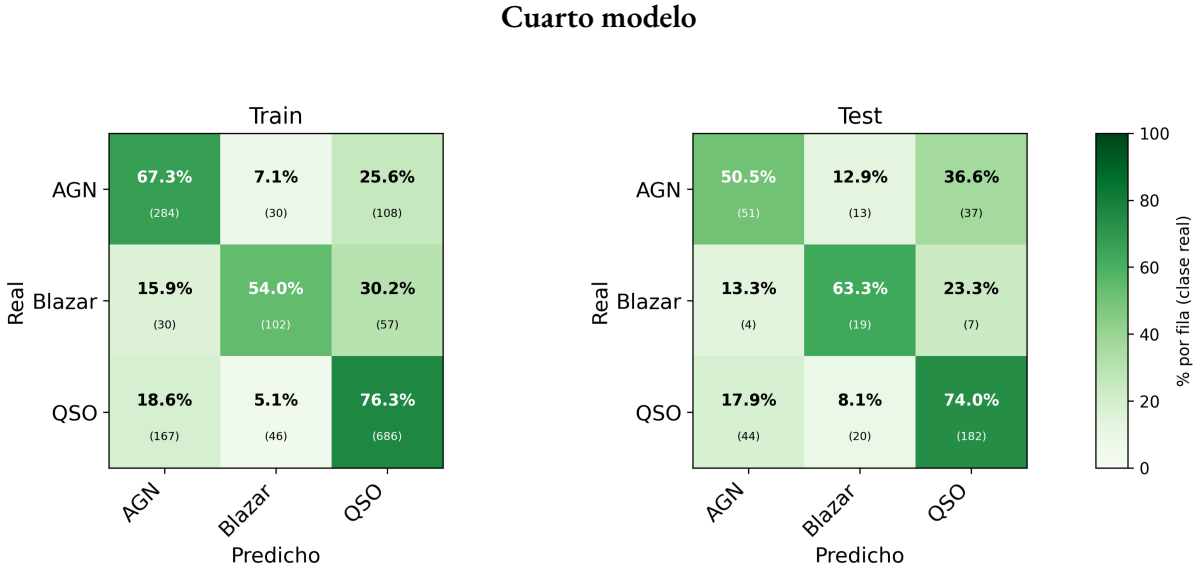


Figura 4.25: Matrices de confusión del entrenamiento y la prueba para el mejor modelo XGB utilizando log-firma (`iisignature`) sobre datos simulados.

```
Accuracy : 0.795
Precision: 0.762
Recall   : 0.763
F1-score : 0.760
```

```
Accuracy : 0.682
Precision: 0.650
Recall   : 0.730
F1-score : 0.687
```

Se estudia la importancia de las variables agrupadas por nivel de la log-signature, con el fin de identificar qué órdenes de esta representación aportan más información al cuarto modelo construido con `iisignature` sobre datos simulados. Para ello, se utilizaron las importancias internas entregadas por XGBoost y luego se agruparon las variables según el nivel correspondiente. En la figura, las barras representan la importancia promedio de las variables en cada nivel y los puntos muestran la mediana.

Los resultados muestran que la importancia se distribuye de manera relativamente homogénea entre los distintos niveles de la log-signature. Sin embargo, el nivel 1 presenta la mayor importancia promedio, seguido por el nivel 4. A partir del nivel 5, las importancias se mantienen en valores similares, sin diferencias muy marcadas entre los niveles superiores. Esto sugiere que, en esta representación, el modelo extrae información útil desde distintos órdenes de la log-signature, aunque con un mayor peso de los primeros niveles.

Importancia promedio y mediana por nivel

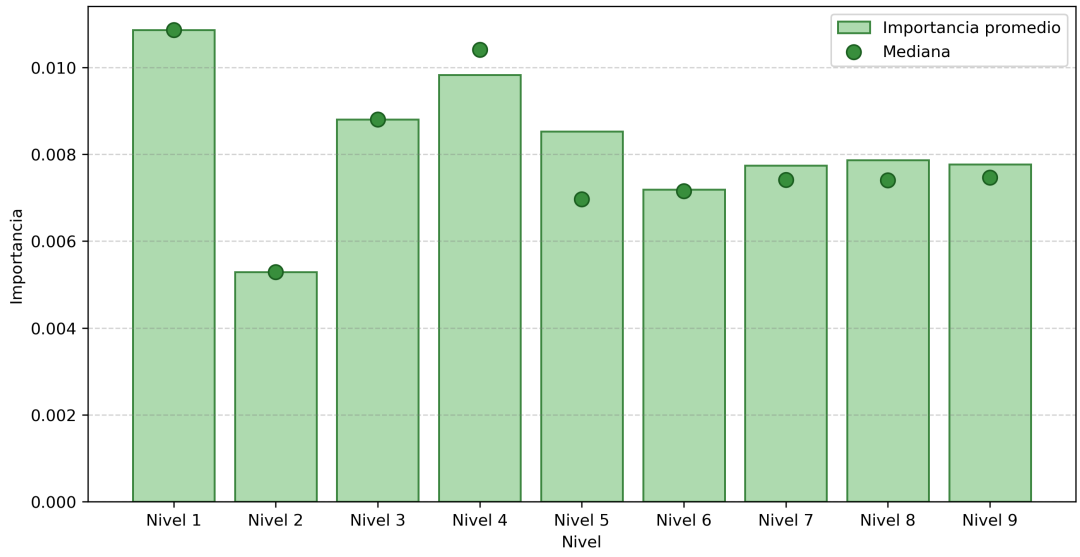


Figura 4.26: Importancia promedio y mediana por nivel para el modelo XGBoost seleccionado con log-firma `iisignature` sobre datos simulados.

Ahora se evalúa el comportamiento del modelo XGB_4 al incorporar de manera acumulada los distintos niveles de la log-signature. Los resultados muestran una mejora progresiva en el AUC_{rep} desde el nivel 1 hasta el nivel 8, donde se alcanza el mejor valor de 0.5977. A partir de ese punto, el rendimiento deja de mejorar, ya que en el nivel 9 el AUC_{rep} desciende levemente a 0.5971. Algo similar ocurre

con las demás métricas de prueba, que no muestran una ganancia clara al incorporar la representación completa.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	AUC Train	Acc Test	F1 Test	AUC Test
1	2	0.5329	0.5201	0.6421	0.4264	0.4151	0.5251
2	3	0.5446	0.5402	0.6775	0.4031	0.3991	0.5335
3	5	0.6318	0.6307	0.7674	0.4341	0.4321	0.5516
4	8	0.6851	0.6868	0.8231	0.4612	0.4679	0.5676
5	14	0.7267	0.7285	0.8617	0.4419	0.4520	0.5725
6	23	0.7490	0.7516	0.8846	0.4651	0.4693	0.5858
7	41	0.8072	0.8079	0.9074	0.4961	0.4949	0.5898
8	71	0.8198	0.8214	0.9242	0.4651	0.4631	0.5977
9	127	0.8459	0.8469	0.9337	0.4651	0.4664	0.5971

Tabla 4.26: Comparación del rendimiento del mejor modelo XGB con log-firma `iisignature` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos simulados.

Bajo el criterio de seleccionar el nivel más simple que maximiza el rendimiento en prueba, el nivel 8 corresponde a la mejor alternativa, ya que logra el mayor $\mathrm{AUC}_{\mathrm{rep}}$ utilizando 71 variables. En conjunto, estos resultados sugieren que la información más útil de la log-signature se concentra en los primeros ocho niveles, y que incorporar el nivel 9 no mejora el desempeño final del modelo.

Representación	Mejor modelo	AUC_{CV}	Acc_{train}	Acc_{test}	$\mathrm{F1}_{w,test}$	Hiperparámetros							Tiempo total
						n	η	depth	min_child	subsample	colsample	reg	
ESIG firma	XGB_4	0.6015	0.8980	0.8568	0.8581	103	0.02322	5	11.92	0.8937	0.9800	λ=3,70, $\alpha\approx$0	02:04:09
ESIG log-firma	XGB_2	0.5627	0.8225	0.7745	0.7760	71	0.02322	5	11.92	0.8937	0.9800	λ=3,70, $\alpha\approx$0	00:11:20
IISIGNATURE firma	XGB_4	0.4587	0.8907	0.8647	0.8658	94	0.02322	5	11.92	0.8937	0.9800	λ=3,70, $\alpha\approx$0	02:32:22
IISIGNATURE log-firma	XGB_4	0.4170	0.7954	0.7401	0.7476	52	0.04212	5	16.57	0.8871	0.7896	λ=22,89, $\alpha\approx$0	00:09:34

Tabla 4.27: Comparación de los mejores modelos XGBoost obtenidos con `esig` e `iisignature`, usando firma y log-firma sobre datos simulados.

XGBoost muestra un patrón claro: cuando se usa la firma el modelo tiende a lograr un mejor equilibrio entre desempeño y capacidad de generalización que con la log-firma. Esto sugiere que, en este escenario, la firma conserva más información útil para separar las clases, permitiendo que el boosting aproveche mejor interacciones no lineales y combinaciones de variables; en cambio, al pasar a log-firma parte de esa estructura se resume y se pierde señal discriminante, lo que se refleja en una disminución del rendimiento en el conjunto de prueba. Además, las diferencias entre bibliotecas indican que no solo importa el modelo, sino también cómo se construye la representación: `esig` e `iisignature` pueden generar características con propiedades numéricas distintas, y eso impacta la estabilidad del entrenamiento y el resultado final. En consecuencia, para los datos simulados, la evidencia apunta a preferir la firma como representación base para XGBoost, ya que entrega resultados más consistentes al evaluar clases desbalanceadas.

4.2. Datos reales

El análisis ahora se enfoca en los datos observados reales, el objetivo es determinar si la path signature extraída directamente de las curvas de luz observadas puede ser utilizada para la clasificación de los objetos astronómicos.

El primer problema al comparar las curvas de luz reales es que han sido observadas en ventanas de tiempo (dominios) diferentes, pero para que la path signature pueda compararlas, estas deben compartir un punto de inicio y un punto final. Sino, se oculta el comportamiento real del objeto. Para solucionar esto, se aplica un paso de normalización temporal. Primero, se identifican el tiempo mínimo absoluto (T_{min}) y el tiempo máximo absoluto (T_{max}) de todo el conjunto de datos. Luego, para cada curva de luz individual se calcula su propia magnitud media ($\bar{m}_{curva}$).

```
> range(newdata$mjd)
[1] 58242.22 60316.16
```

Finalmente, a cada curva se le añaden dos puntos: un punto inicial en $(T_{min}, \bar{m}_{curva})$ y un punto final en $(T_{max}, \bar{m}_{curva})$, así se fuerza a todas las curvas a existir en el mismo dominio temporal $[T_{min}, T_{max}]$.

Se muestran observaciones de la primera y última curva de luz:

```
                id        tiempo  banda   magnitud     error
0       ZTF17aaadhjr  58242.220000      1  16.351614  0.010000
1       ZTF17aaadhjr  58480.464942      1  17.250767  0.028059
2       ZTF17aaadhjr  58493.245961      1  17.280317  0.019254
3       ZTF17aaadhjr  58522.181250      1  17.630146  0.025279
4       ZTF17aaadhjr  58534.234630      1  17.698772  0.068816
...              ...           ...    ...        ...       ...
805282  ZTF18acpdfyw  60289.395961      1  17.868067  0.054846
805283  ZTF18acpdfyw  60291.440961      1  17.865416  0.054394
805284  ZTF18acpdfyw  60296.498183      1  17.853710  0.064757
805285  ZTF18acpdfyw  60315.418009      1  17.881319  0.045689
805286  ZTF18acpdfyw  60316.160000      1  17.929876  0.010000
```

En complemento, también se ven las primeras 3 curvas de luz junto a su primera y última observación, para dar cuenta de que efectivamente todas poseen el mismo dominio:

```
                  id    tiempo  banda   magnitud  error
0       ZTF17aaadhjr  58242.22      1  16.351614   0.01
402477  ZTF17aaadhjr  60316.16      1  16.351614   0.01

203     ZTF17aaahyqr  58242.22      1  19.363590   0.01
402592  ZTF17aaahyqr  60316.16      1  19.363590   0.01

318     ZTF17aaaivav  58242.22      1  19.827703   0.01
402831  ZTF17aaaivav  60316.16      1  19.827703   0.01
```

A continuación, el vector numérico generado por la path signature de nivel 9:

```
     0            1         2             3            4            5  \
0  1.0  1834.949201  3.140854  1.683519e+06  3207.725793  2555.581746
1  1.0  1768.926620  3.288600  1.564551e+06  1843.093348  3974.198736
2  1.0  1827.860058  0.564786  1.670536e+06   303.222622   729.127149
3  1.0  1807.210162  0.474224  1.633004e+06   449.560182   407.462250
4  1.0  1747.222002  0.636582  1.526392e+06   579.782017   532.468060

          6             7             8             9  ...       1014  \
0  4.932482  1.029724e+09  2.149502e+06  1.587011e+06  ...  59.481266
1  5.407445  9.225251e+08  9.716102e+05  1.317077e+06  ...  69.806622
2  0.159492  1.017835e+09  1.740532e+05  2.061422e+05  ...   0.000063
3  0.112444  9.837273e+08  3.270725e+05  1.583047e+05  ...   0.000020
4  0.202618  8.889821e+08  3.776766e+05  2.576547e+05  ...   0.000180

           1015           1016          1017       1018           1019  \
0  1.129599e+07   35597.994060  11664.901974  57.911424   13618.513250
1  1.100275e+08  113900.053222  82278.056430  92.757837  131915.130288
2  3.222270e+03       0.566531      0.423036   0.000079       0.617128
3  1.025089e+02       0.053540      0.014609   0.000014       0.013341
4  6.957743e+02       0.449795      0.169255   0.000160       0.165046

         1020        1021          1022           id
0   38.777977   17.005190  8.197222e-02  ZTF17aaadhjr
1  125.698175  123.396339  1.239757e-01  ZTF17aaahyqr
2    0.000102    0.000094  1.611359e-08  ZTF17aaajmlg
3    0.000007    0.000002  3.342633e-09  ZTF17aabirwa
4    0.000105    0.000045  4.730708e-08  ZTF17aabtkto
```

El conjunto de datos está compuesto por 1071 objetos tipo QSO, 500 AGN y 228 Blazars. Esto indica que las clases no están balanceadas.

```
Clases únicas: ['Blazar' 'AGN' 'QSO']

Cantidad de ejemplos por clase:
 tipo
QSO       1071
AGN        500
Blazar     228
Name: count, dtype: int64
```

4.2.1. Random forest

esig

Se presentan los resultados obtenidos al aplicar Random Forest sobre el conjunto de datos reales, utilizando características construidas mediante la librería `esig`. A diferencia del análisis previo sobre datos simulados, el interés aquí se centra en evaluar el comportamiento del modelo frente a observaciones reales, donde la variabilidad intrínseca de las curvas y la superposición entre clases planean un escenario de clasificación más exigente. La búsqueda de hiperparámetros se realizó mediante Randomized Search sobre 200 configuraciones, y el proceso completo tuvo una duración total de 23 horas, 37 minutos y 58 segundos.

A partir de esta búsqueda se seleccionaron las cinco configuraciones con mejor AUC promedio en validación cruzada. Los resultados muestran que las diferencias entre ellas son pequeñas, con valores de AUC_{CV} entre 0.7523 y 0.7545. Dentro de este conjunto, el primer modelo obtuvo el mayor AUC promedio en validación cruzada, acompañado de un AUC promedio de 0.7584 en validación repetida y de un gap menor que el observado en otras configuraciones de desempeño similar. Por esta razón, y manteniendo como criterio principal la capacidad discriminativa estimada por validación cruzada, RF_1 puede considerarse el modelo seleccionado en esta etapa.

No obstante, al evaluar el desempeño sobre el conjunto de prueba, el modelo RF_2 alcanzó los mejores resultados, con AUC_{test}=0.8908, accuracy de 0.7939 y F1-score ponderado de 0.7923, superando al primero. Esto indica que, si se quiere mantener coherencia con el cirterio de selección establecido, conviene conservar RF_1 como el modelo final y destacar a RF_2 como la configuración que logró el mejor rendimiento en evaluación final.

Desde el punto de vista de la clasificación por clase, las matrices de confusión muestran que la categoría QSO es la mejor identificada por el modelo, mientras que AGN y Blazar presentan mayores niveles de confusión entre sí y QSO. En el caso del primer random forest, las métricas por clase en test alcanzan F1 de 0.63 para AGN, 0.62 para Blazar y 0.86 para QSO.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
RF_1	0.7545	0.0248	0.1655	0.7584	0.0230	0.1614	0.7799	0.7788
RF_2	0.7537	0.0250	0.1800	0.7570	0.0228	0.1766	0.7939	0.7923
RF_3	0.7530	0.0238	0.1529	0.7571	0.0231	0.1484	0.7772	0.7779
RF_4	0.7528	0.0257	0.1605	0.7566	0.0238	0.1563	0.7688	0.7675
RF_5	0.7523	0.0256	0.1341	0.7558	0.0239	0.1302	0.7549	0.7541

Tabla 4.28: Desempeño de los cinco mejores modelos RF utilizando firma (`esig`) sobre datos reales.

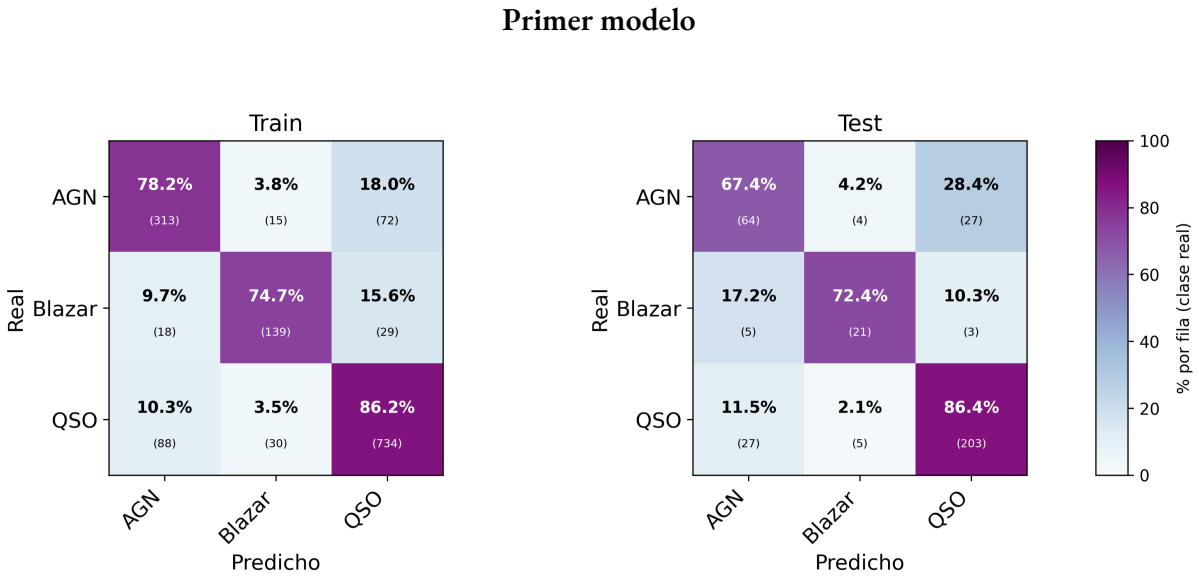


Figura 4.27: Matrices de confusión del entrenamiento y la prueba para el mejor modelo RF utilizando firma (`esig`) sobre datos reales.

```
Accuracy : 0.803
Precision: 0.802
Recall   : 0.803
F1-score : 0.802
```

```
Accuracy : 0.780
Precision: 0.778
Recall   : 0.780
F1-score : 0.779
```

Con la representación basada en log-firma, las cinco mejores configuraciones presentan desempeños muy cercanos en validación cruzada, con valores de AUC_{CV} entre 0.7203 y 0.7219. El mayor AUC promedio en validación cruzada correspondió a RF_1, con un valor de 0.7219, por lo que esta configuración puede considerarse la mejor bajo el criterio principal de selección definido previamente. No obstante, al evaluar el comportamiento en el conjunto de prueba, el mejor AUC fue obtenido por el tercer modelo (0.8354), mientras que el segundo modelo alcanzó el mayor F1-score ponderado (0.7387), ambos por encima del desempeño del primero. Esto sugiere que, aunque RF_1 fue el modelo más sólido según validación cruzada, RF_2 y RF_3 mostraron una mejor capacidad de generalización empírica en datos no vistos. En términos de clasificación por clase, los mejores resultados volvieron a concentrarse en QSO, mientras que AGN y Blazar presentaron mayores dificultades de separación. En

conjunto, estos resultados, a pesar de ser competitivos, presentan un desempeño inferior al observado previamente con la firma estándar.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{val}	SD_{val}	Gap_{val}	Acc_{test}	$F1_{w,test}$
RF_1	0.7219	0.0330	0.1610	0.7202	0.0210	0.1624	0.7242	0.7260
RF_2	0.7210	0.0347	0.1668	0.7177	0.0210	0.1697	0.7354	0.7387
RF_3	0.7207	0.0337	0.1769	0.7195	0.0209	0.1781	0.7354	0.7357
RF_4	0.7207	0.0348	0.1470	0.7170	0.0217	0.1504	0.6992	0.7053
RF_5	0.7203	0.0335	0.1426	0.7175	0.0208	0.1449	0.7047	0.7098

Tabla 4.29: Desempeño de los cinco mejores modelos RF utilizando log-firma (`esig`) sobre datos reales.

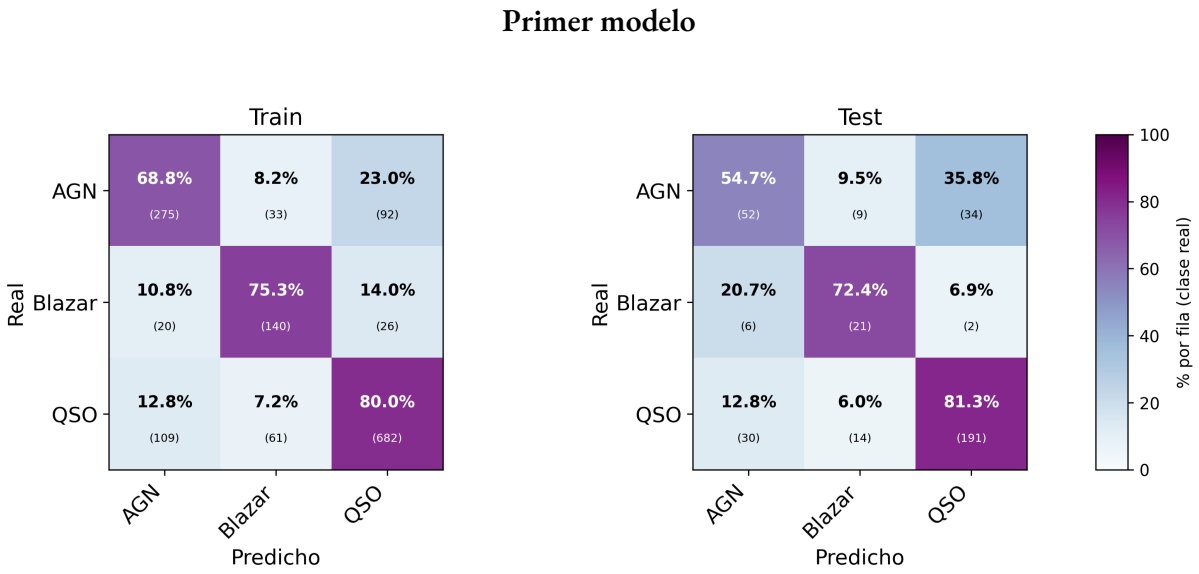


Figura 4.28: Matrices de confusión del entrenamiento y la prueba para el mejor modelo RF utilizando log-firma (`esig`) sobre datos reales.

```
Accuracy : 0.755
Precision: 0.760
Recall   : 0.755
F1-score : 0.756
```

```
Accuracy : 0.724
Precision: 0.731
Recall   : 0.724
F1-score : 0.726
```

El tiempo total de cómputo fue de 1 hora, 45 minutos y 30 segundos.

IISIGNATURE

La búsqueda de hiperparámetros se realizó mediante Randomized Search sobre 200 configuraciones, con un tiempo total de ejecución de 13 horas, 28 minutos y 17 segundos. A partir de este proceso, se seleccionaron las cinco configuraciones con mejor AUC promedio en validación cruzada, cuyos resultados muestran un comportamiento muy parejo. Dentro de este conjunto, el primer modelo obtuvo

el mayor AUC en validación cruzada (0.7763), acompañado además del mayor AUC en el conjunto de prueba (0.8756), por lo que resulta la alternativa más óptima bajo el criterio principal de selección definido previamente. No obstante, RF_4 presentó la mejor accuracy de prueba (0.7825) y el mayor F1-score ponderado (0.7830), posicionándose como el competidor más cercano en términos de desempeño final. Esto indica que la firma permite obtener modelos robustos y competitivos sobre datos reales, aunque persiste una mayor dificultad para separar las clases AGN y Blazar en comparación con QSO.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{val}	SD_{val}	Gap_{val}	AUC_{test}	Acc_{test}	$F1_{w,test}$
RF_1	0.7763	0.0148	0.1566	0.7779	0.0186	0.1550	0.8756	0.7745	0.7769
RF_2	0.7757	0.0158	0.1402	0.7759	0.0187	0.1395	0.8584	0.7639	0.7652
RF_3	0.7754	0.0163	0.1471	0.7767	0.0183	0.1457	0.8636	0.7719	0.7729
RF_4	0.7751	0.0177	0.1613	0.7758	0.0186	0.1604	0.8750	0.7825	0.7830
RF_5	0.7738	0.0150	0.1363	0.7764	0.0184	0.1333	0.8507	0.7480	0.7512

Tabla 4.30: Desempeño de los cinco mejores modelos RF utilizando firma (`iisignature`) sobre datos reales.

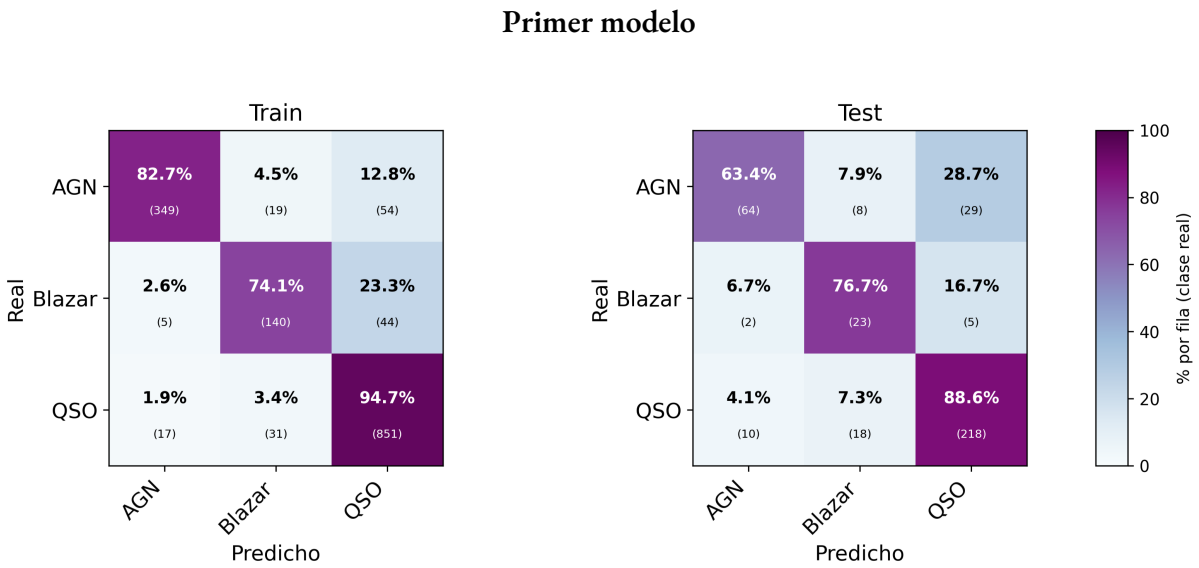


Figura 4.29: Matrices de confusión del entrenamiento y la prueba para el mejor modelo RF utilizando firma (`iisignature`) sobre datos reales.

```
Accuracy : 0.824
Precision: 0.829
Recall   : 0.824
F1-score : 0.825
```

```
Accuracy : 0.775
Precision: 0.780
Recall   : 0.775
F1-score : 0.777
```

Las cinco mejores configuraciones tuvieron resultados muy similares en validación cruzada. Aunque RF_1 obtuvo el mayor AUC promedio y fue la mejor según el criterio de selección, RF_3 mostró el mejor

desempeño en el conjunto de prueba, con mayor capacidad de generalización. En la clasificación por clase, QSO volvió a ser la categoría mejor identificada, mientras que AGN y Blazar siguieron siendo más difíciles de separar. En conjunto, esta representación fue competitiva, pero mantuvo limitaciones en las clases minoritarias.

Se replica el estudio, pero ahora considerando la log-firma.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{val}	SD_{val}	Gap_{val}	AUC_{test}	Acc_{test}	$F1_{w,test}$
RF_1	0.7322	0.0270	0.1584	0.7328	0.0221	0.1572	0.8207	0.7082	0.7173
RF_2	0.7298	0.0286	0.1291	0.7304	0.0221	0.1275	0.7933	0.6711	0.6818
RF_3	0.7298	0.0276	0.1761	0.7317	0.0221	0.1731	0.8366	0.7321	0.7408
RF_4	0.7294	0.0282	0.1548	0.7307	0.0220	0.1523	0.8162	0.7029	0.7137
RF_5	0.7290	0.0282	0.1427	0.7306	0.0220	0.1404	0.8046	0.6870	0.6997

Tabla 4.31: Desempeño de los cinco mejores modelos RF utilizando log-firma (`iisignature`) sobre datos reales.

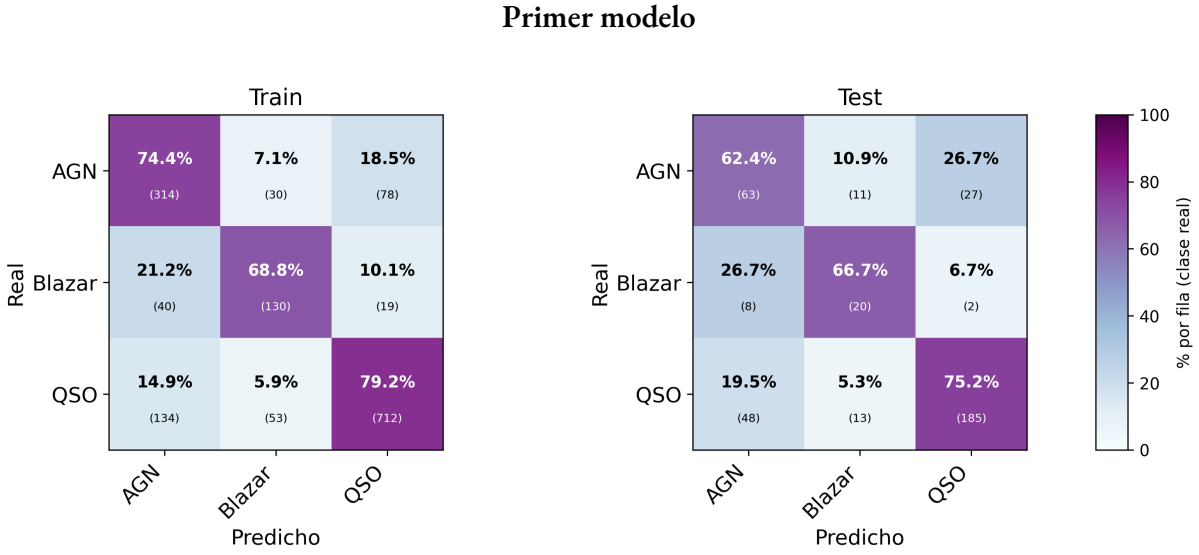


Figura 4.30: Matrices de confusión del entrenamiento y la prueba para el mejor modelo RF utilizando log-firma (`iisignature`) sobre datos reales.

```
Accuracy : 0.765
Precision: 0.774
Recall   : 0.765
F1-score : 0.768
```

```
Accuracy : 0.708
Precision: 0.733
Recall   : 0.708
F1-score : 0.717
```

Se resumen los mejores modelos obtenidos con las librerías `esig` e `iisignature`, considerando tanto la representación basada en firma como en log-firma. Esta comparación permite evaluar simultáneamente el rendimiento predictivo, la configuración óptima de hiperparámetros y el costo computacional asociado a cada alternativa, con el propósito de identificar la combinación más conveniente para el problema de clasificación estudiado.

Representación	Mejor modelo	AUC_{CV}	Acc_{train}	Acc_{test}	$F1_{w,test}$	Hiperparámetros							Tiempo total
						Crit.	Depth	Max feat.	Max samp.	Min leaf	Min split	Trees	
ESIG firma	RF_1	0.7545	0.803	0.7799	0.7788	gini	None	0.5	0.6	14	24	2273	13:37:58
ESIG log-firma	RF_1	0.7219	0.755	0.7242	0.7260	gini	None	0.5	0.6	14	24	2273	01:45:30
IISIGNATURE firma	RF_1	0.7763	0.824	0.7745	0.7769	entropy	15	0.8	0.8	13	41	3141	13:28:17
IISIGNATURE log-firma	RF_1	0.7322	0.765	0.7082	0.7173	gini	None	0.5	0.6	14	24	2273	01:51:57

Tabla 4.32: Comparación de los mejores modelos RF obtenidos con `esig` e `iisignature`, usando firma y log-firma sobre datos reales.

En términos generales, la firma estándar supera a la log-firma en ambas librerías, lo que confirma que esta representación conserva mejor la información discriminante de los datos reales. Bajo el criterio principal de selección basado en AUC en validación cruzada, el mejor resultado corresponde a `iisignature` con firma estándar. Sin embargo, `esig` con firma estándar presenta un desempeño ligeramente superior en accuracy y F1-score sobre el conjunto de prueba. En consecuencia, ambas configuraciones destacan por sobre las restantes, aunque `iisignature` con firma estándar puede considerarse la alternativa más sólida según el criterio de selección adoptado.

Para interpretar el mejor modelo seleccionado, se analizó la importancia de las características agrupadas por nivel de la firma, considerando tanto la importancia promedio como la mediana. Los resultados muestran una tendencia general al aumento de la relevancia hacia los niveles superiores, aunque no de forma estrictamente monótona. En particular, el nivel 9 destaca claramente al presentar la mayor importancia promedio y también la mediana más alta, seguido por los niveles 8 y 7. En contraste, el nivel 1 aporta una contribución prácticamente marginal, mientras que los niveles intermedios muestran una participación más moderada. Además, en varios niveles altos el promedio supera a la mediana, lo que sugiere que dentro de esos grupos existen algunas variables especialmente influyentes que elevan la importancia media, aun cuando la mayoría de las variables mantiene valores más contenidos. En conjunto, estos resultados indican que los términos de orden superior concentran la mayor parte de la información discriminante del modelo y cumplen un papel central en la clasificación de AGN, Blazar y QSO.

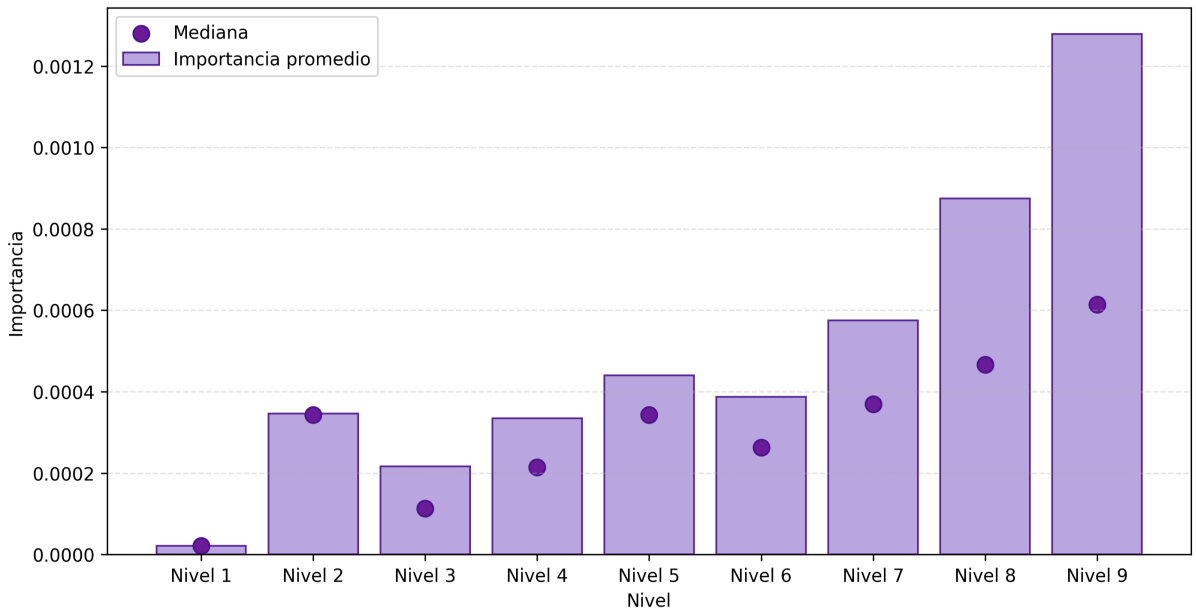


Figura 4.31: Importancia promedio y mediana por nivel para el modelo RF seleccionado con firma `iisignature` sobre datos reales.

Para terminar, se analizó el efecto de utilizar subconjuntos acumulativos de características hasta cada nivel de firma. Los resultados muestran una mejora general del rendimiento a medida que se incorporan términos de mayor orden. En particular, el AUC de prueba aumenta desde 0.512 en el nivel 1 hasta 0.747 en el nivel 9, donde además se obtienen la mayor accuracy de prueba (0.611) y el mayor F1-score (0.616). En consecuencia, bajo un criterio de selección basado en AUC sobre el conjunto de prueba, el nivel más simple que maximiza el desempeño corresponde al nivel 9, lo que respalda el uso de la representación de 1022 características.

Nivel de firma	Nº de características	Acc_{train}	$F1_{train}$	Acc_{test}	$F1_{test}$	AUC_{test}
1	2	0.587	0.477	0.590	0.470	0.512
2	6	0.536	0.551	0.402	0.415	0.533
3	14	0.662	0.670	0.545	0.547	0.654
4	30	0.706	0.713	0.566	0.573	0.662
5	62	0.742	0.747	0.577	0.586	0.683
6	126	0.775	0.778	0.593	0.602	0.704
7	254	0.795	0.798	0.595	0.604	0.723
8	510	0.820	0.822	0.601	0.606	0.737
9	1022	0.837	0.839	0.611	0.616	0.747

Tabla 4.33: Comparación del rendimiento del mejor modelo RF con firma `iisignature` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos reales.

Los resultados obtenidos con Random Forest sobre datos reales muestran que las representaciones basadas en firma estándar superan consistentemente a sus contrapartes basadas en log-firma, tanto con `esig` como con `iisignature`. Esto indica que, en este escenario, la firma completa conserva de mejor manera la información discriminante necesaria para la clasificación de AGN, Blazar y QSO. En particular, las representaciones con log-firma reducen de forma importante el tiempo de cómputo, pero esta ventaja computacional se obtiene a costa de una pérdida apreciable en desempeño predictivo.

Al comparar ambas librerías, se observa que los mejores resultados globales se concentran en las configuraciones construidas con firma estándar. Bajo el criterio principal de selección basado en AUC en validación cruzada, el modelo más sólido corresponde a `iisignature` con firma estándar. Sin embargo, `esig` con firma estándar presenta un desempeño muy similar e incluso ligeramente superior en algunas métricas de prueba, lo que muestra que ambas implementaciones entregan resultados competitivos cuando se utiliza la representación completa. En consecuencia, la diferencia principal no se encuentra tanto entre librerías, sino entre el uso de firma y log-firma.

El análisis de importancia de características mostró que los niveles superiores de la firma concentran la mayor parte de la información relevante para el modelo, especialmente los niveles más altos. De manera coherente con ello, la evaluación mediante subconjuntos acumulativos de características indicó que el mejor rendimiento se alcanza al utilizar la representación completa, lo que respalda el uso del nivel 9 en el análisis final. En conjunto, estos resultados permiten concluir que, para Random Forest sobre datos reales, la estrategia más adecuada consiste en utilizar la firma estándar completa, ya que es la que ofrece el mejor equilibrio entre capacidad discriminativa, estabilidad y desempeño global.

4.2.2. Support vector machine

esig

Se evaluó la representación basada en path signature calculada con `esig` sobre los datos reales, manteniendo la misma estrategia de validación cruzada y el mismo esquema general de ajuste utilizados en los análisis anteriores. A partir de la búsqueda realizada, se seleccionaron las cinco configuraciones con mejor desempeño. En este caso, el mayor AUC promedio en validación cruzada correspondió a SVM_3, con un valor de 0.7102. Sin embargo, al comparar el comportamiento final sobre el conjunto de prueba, el mejor desempeño correspondió a SVM_2, que alcanzó un AUC_{rep} de 0.8564, una accuracy de 0.8078 y un F1-score ponderado de 0.8064. Por esta razón, SVM_2 fue seleccionado como modelo final para la representación basada en firma con `esig` en datos reales, privilegiando su mejor capacidad de generalización en datos no vistos. El tiempo total de cómputo fue de 23 minutos y 25 segundos.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
SVM_1	0.7002	0.0152	0.2193	0.8484	0.0152	0.0711	0.7632	0.7620
SVM_2	0.6857	0.0124	0.2546	0.8564	0.0124	0.0839	0.8078	0.8064
SVM_3	0.7102	0.0309	0.0871	0.7484	0.0309	0.0490	0.7019	0.6870
SVM_4	0.6568	0.0108	0.2687	0.8409	0.0108	0.0846	0.6797	0.6981
SVM_5	0.6922	0.0229	0.0712	0.7227	0.0229	0.0407	0.6825	0.6612

Tabla 4.34: Desempeño de los cinco mejores modelos SVM utilizando firma (`esig`) sobre datos reales.

A partir de la matriz de confusión de prueba, se observa que el modelo logra un reconocimiento equilibrado de las tres clases. En la clase AGN se obtienen 61 clasificaciones correctas de 95 observaciones, mientras que en Blazar se logran 23 aciertos de 29 casos. Por su parte, la clase QSO alcanza 206 clasificaciones correctas de 235 ejemplos. Esto se traduce en recalls aproximados de 0.64 para AGN, 0.79 para Blazar y 0.88 para QSO. En conjunto, estos resultados muestran que SVM_2 presenta un comportamiento más homogéneo entre clases que el resto de las configuraciones evaluadas, con una capacidad especialmente favorable para reconocer objetos QSO y un buen desempeño en la clase Blazar.

Además, los valores de Gap_{CV} y Gap_{rep} fueron positivos en los cinco modelos evaluados, lo que indica que el desempeño en entrenamiento fue superior al observado tanto en validación cruzada como en prueba. Esto sugiere la presencia de cierto grado de sobreajuste, aunque en el caso de SVM_2 el rendimiento final sobre el conjunto de prueba siguió siendo el mejor entre las configuraciones comparadas.

Segundo modelo

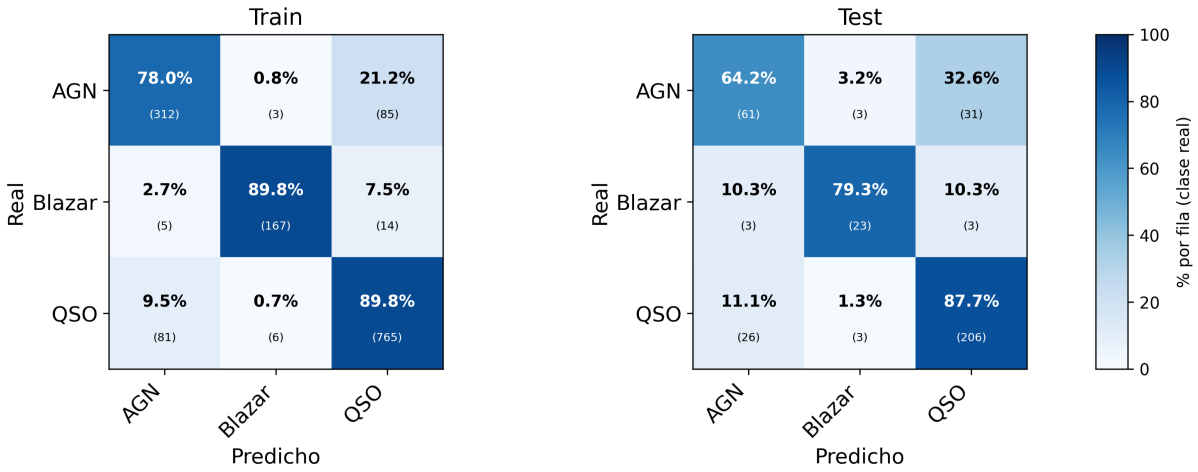


Figura 4.32: Matrices de confusión del entrenamiento y la prueba para el mejor modelo SVM utilizando firma (`esig`) sobre datos reales.

```
Accuracy : 0.865
Precision: 0.865
Recall   : 0.865
F1-score : 0.865
```

```
Accuracy : 0.808
Precision: 0.810
Recall   : 0.808
F1-score : 0.806
```

Si bien el uso de SMOTE puede hacer que las métricas de entrenamiento no se vean tan altas al evaluarse sobre la distribución original de los datos, este efecto no tiene por qué presentarse siempre con la misma intensidad. En este caso, el mejor comportamiento en entrenamiento sugiere que la representación sobre los datos reales conserva una estructura de clases más clara que la observada en los datos simulados. Así, aunque el modelo fue entrenado sobre una muestra balanceada artificialmente, la frontera de decisión aprendida sigue siendo consistente con la distribución original del conjunto real, lo que permite obtener métricas de entrenamiento más favorables.

A continuación, se evaluó la representación basada en log-firma sobre los datos reales, manteniendo la misma partición de los datos, la estrategia de validación cruzada y el mismo esquema general de ajuste utilizados en los análisis anteriores. A partir de la búsqueda realizada, se seleccionaron las cinco configuraciones con mejor desempeño. En este caso, el mayor valor de la métrica robusta en validación cruzada fue 0.3703, con una configuración que utilizó kbest $=$ 127, $C = 212{,}3514$, ponderación de clases $\{1{,}0,\ 3{,}0,\ 0{,}7\}$ y $\gamma = 0{,}02729$. Sin embargo, al comparar el comportamiento final sobre el conjunto de prueba, el mejor desempeño correspondió a SVM_3, que alcanzó un AUC_{rep} de 0.7809, una accuracy de 0.6462 y un F1-score ponderado de 0.6646. Por esta razón, SVM_3 fue seleccionado como modelo final para la representación basada en log-firma con `esig` en datos reales, privilegiando el rendimiento observado en datos no vistos. El tiempo de cómputo total fue de 3 minutos y 29 segundos.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
SVM_1	0.5903	0.0205	0.2131	0.7368	0.0205	0.0665	0.5933	0.6230
SVM_2	0.5902	0.0100	0.2700	0.7667	0.0100	0.0935	0.6657	0.6825
SVM_3	0.5650	0.0157	0.3056	0.7809	0.0157	0.0897	0.6462	0.6646
SVM_4	0.6608	0.0258	0.0563	0.7010	0.0258	0.0160	0.6546	0.6175
SVM_5	0.6566	0.0236	0.0312	0.6876	0.0236	0.0001	0.6435	0.6020

Tabla 4.35: Desempeño de los cinco mejores modelos SVM utilizando log-firma (`esig`) sobre datos reales.

Aunque SVM_4 y SVM_5 presentan mayores valores de AUC promedio en validación cruzada, su desempeño final sobre el conjunto de prueba es inferior al de SVM_3. En particular, SVM_3 logra el mayor AUC_{rep}, lo que indica una mejor capacidad de generalización en datos no vistos.

Además, todos los modelos muestran valores positivos de Gap_{CV} y Gap_{rep}, lo que indica que el desempeño en entrenamiento es superior al observado tanto en validación cruzada como en prueba. Este comportamiento evidencia la presencia de sobreajuste en todas las configuraciones evaluadas, siendo más pronunciado en los modelos con mayor gap. El tiempo total del proceso fue de 00:03:29, lo que confirma que la representación basada en log-firma reduce de manera importante el costo computacional en comparación con la firma estándar.

Tercer modelo

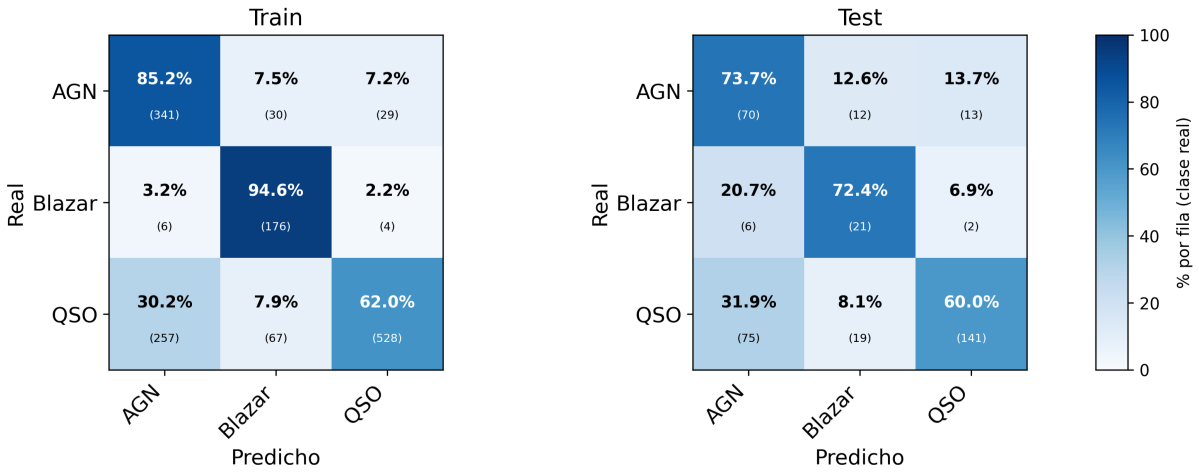


Figura 4.33: Matrices de confusión del entrenamiento y la prueba para el mejor modelo SVM utilizando log-firma (`esig`) sobre datos reales.

```
Accuracy : 0.727
Precision: 0.726
Recall   : 0.727
F1-score : 0.731
```

```
Accuracy : 0.646
Precision: 0.751
Recall   : 0.646
F1-score : 0.665
```

A partir de la matriz de confusión de prueba, se observa que el modelo presenta un desempeño diferenciado entre clases. En la clase AGN se obtienen 70 clasificaciones correctas de 95 observaciones, mientras que en Blazar se logran 21 aciertos de 29 casos. Por su parte, la clase QSO alcanza 141 clasificaciones correctas de 235 ejemplos. Esto se traduce en recalls aproximados de 0.74 para AGN, 0.72 para Blazar y 0.60 para QSO.

En conjunto, estos resultados indican que el modelo logra una buena identificación de las clases AGN y Blazar, mientras que la principal limitación se presenta en la clase QSO, donde se observa una mayor confusión hacia las otras categorías. A pesar de esto, SVM_3 mantiene el mejor equilibrio global entre las métricas de desempeño en el conjunto de prueba dentro de las configuraciones evaluadas.

Estos resultados muestran que la representación basada en log-firma permite construir un clasificador con desempeño aceptable sobre datos reales, aunque inferior al obtenido previamente con la firma estándar. Aun así, el modelo seleccionado logra una separación razonable entre las clases y destaca por requerir un tiempo de ejecución mucho menor, lo que convierte a esta representación en una alternativa computacionalmente atractiva cuando se busca reducir el costo del ajuste.

IISIGNATURE

Se evaluó la representación basada en log-firma utilizando `esig` sobre los datos reales, manteniendo la misma partición de los datos, la estrategia de validación cruzada y el mismo esquema general de ajuste utilizados en los análisis anteriores. A partir de la búsqueda realizada, se seleccionaron las cinco configuraciones con mejor desempeño. En este caso, el mayor valor de la métrica robusta en validación cruzada fue 0.5443, con una configuración que utilizó kbest $= 1022$, $C = 784{,}8082$, ponderación de clases `balanced` y $\gamma = 8{,}22 \times 10^{-5}$.

Sin embargo, al comparar el comportamiento final sobre el conjunto de prueba, el mejor desempeño correspondió a SVM_3, que alcanzó un AUC_{rep} de 0.8469, una accuracy de 0.7613 y un F1-score ponderado de 0.7604. Por esta razón, SVM_3 fue seleccionado como modelo final para la representación basada en log-firma con `esig` en datos reales, privilegiando el rendimiento observado en datos no vistos. El tiempo de cómputo total fue de 23 minutos y 53 segundos.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
SVM_1	0.7320	0.0056	0.0691	0.7609	0.0056	0.0402	0.7135	0.6999
SVM_2	0.7069	0.0146	0.2129	0.8313	0.0146	0.0885	0.7215	0.7198
SVM_3	0.6988	0.0197	0.2454	0.8469	0.0197	0.0973	0.7613	0.7604
SVM_4	0.7270	0.0145	0.0480	0.7370	0.0145	0.0381	0.7003	0.6810
SVM_5	0.6850	0.0187	0.2397	0.8254	0.0187	0.0993	0.7056	0.7195

Tabla 4.36: Desempeño de los cinco mejores modelos SVM utilizando log-firma (`esig`) sobre datos reales.

Aunque SVM_1 y SVM_4 presentan valores de AUC promedio en validación cruzada comparables o incluso superiores, su desempeño final sobre el conjunto de prueba es inferior al de SVM_3. En particular, SVM_3 logra el mayor AUC_{rep}, lo que indica una mejor capacidad de generalización en datos no vistos.

Además, todos los modelos presentan valores positivos de Gap_{CV} y Gap_{rep}, lo que indica que el desempeño en entrenamiento es superior al observado tanto en validación cruzada como en prueba. Esto evidencia la presencia de sobreajuste en todas las configuraciones evaluadas, siendo más pronunciado en modelos como SVM_3 y SVM_5, que presentan mayores valores de gap. A pesar de esto, SVM_3 mantiene el mejor equilibrio entre capacidad de ajuste y desempeño en generalización.

El tiempo total del proceso fue de 00:23:53, lo que, si bien es considerable, sigue siendo inferior al requerido por algunas configuraciones basadas en firma estándar.

Tercer modelo

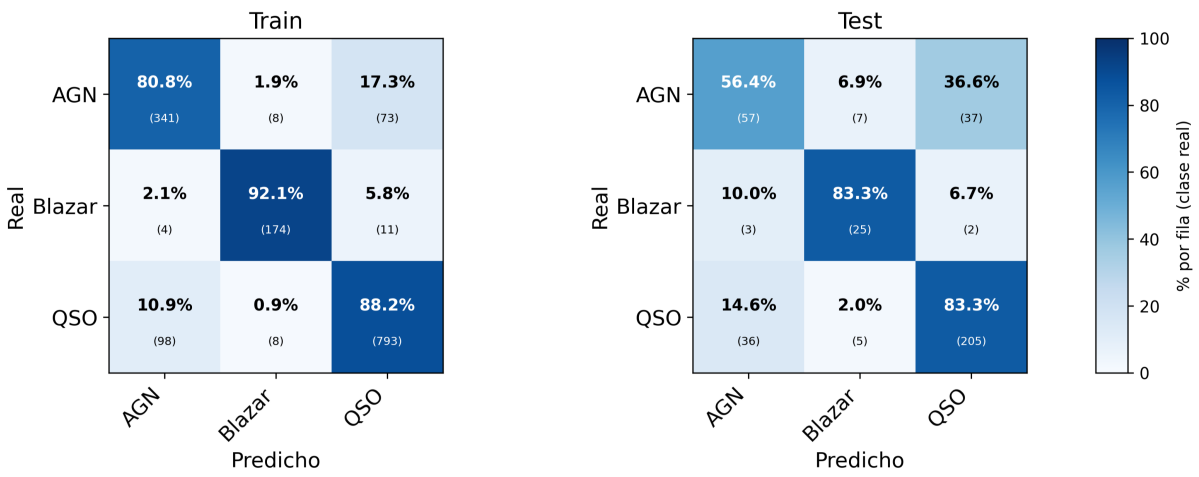


Figura 4.34: Matrices de confusión del entrenamiento y la prueba para el mejor modelo SVM utilizando firma (`iisignature`) sobre datos reales.

```
Accuracy : 0.866
Precision: 0.867
Recall   : 0.866
F1-score : 0.867
```

```
Accuracy : 0.761
Precision: 0.762
Recall   : 0.761
F1-score : 0.760
```

A partir de la matriz de confusión de prueba, se observa que el modelo logra un reconocimiento sólido de las tres clases. En la clase AGN se obtienen 57 clasificaciones correctas de 101 observaciones, mientras que en Blazar se logran 25 aciertos de 30 casos. Por su parte, la clase QSO alcanza 205 clasificaciones correctas de 246 ejemplos. Esto se traduce en recalls aproximados de 0.56 para AGN, 0.83 para Blazar y 0.83 para QSO.

En conjunto, estos resultados muestran que el modelo presenta un desempeño equilibrado entre clases, destacando especialmente en la identificación de Blazar y QSO, mientras que la principal dificultad se mantiene en la clase AGN, donde se observa una mayor confusión hacia las otras categorías.

A continuación, se evaluó la representación basada en log-firma utilizando `iisignature` sobre los datos reales, manteniendo la misma partición de los datos, la estrategia de validación cruzada y el mismo esquema general de ajuste utilizados en los análisis anteriores. A partir de la búsqueda realizada, se seleccionaron las cinco configuraciones con mejor desempeño. En este caso, el mejor valor de la métrica robusta en validación cruzada fue 0.4398, con una configuración que utilizó kbest $=$ 127, $C = 212{,}3514$, ponderación de clases $\{1{,}0,\ 3{,}0,\ 0{,}7\}$ y $\gamma = 0{,}02729$.
Sin embargo, al comparar el comportamiento final sobre el conjunto de prueba, el mejor desempeño correspondió a SVM_3, que alcanzó un AUC_{rep} de 0.7707, una accuracy de 0.6790 y un F1-score ponderado de 0.6921. Por esta razón, SVM_3 fue seleccionado como modelo final para la representación basada en log-firma con `iisignature` en datos reales, privilegiando el rendimiento observado en datos no vistos. El tiempo de cómputo total fue de 3 minutos y 35 segundos.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
SVM_1	0.6165	0.0073	0.2048	0.7161	0.0073	0.1053	0.5915	0.6068
SVM_2	0.6072	0.0030	0.2615	0.7528	0.0030	0.1160	0.6737	0.6829
SVM_3	0.6027	0.0039	0.2808	0.7707	0.0039	0.1128	0.6790	0.6921
SVM_4	0.6304	0.0279	0.1085	0.6841	0.0279	0.0548	0.3767	0.4267
SVM_5	0.6120	0.0110	0.1733	0.6965	0.0110	0.0888	0.3979	0.3794

Tabla 4.37: Desempeño de los cinco mejores modelos SVM utilizando log-firma (`iisignature`) sobre datos reales.

Aunque SVM_4 presenta el mayor AUC promedio en validación cruzada, su desempeño en el conjunto de prueba es considerablemente inferior al del resto de las configuraciones. En contraste, SVM_3 logra el mayor AUC_{rep}, junto con las mejores métricas globales de clasificación, lo que indica una mejor capacidad de generalización en datos no vistos.

Además, todos los modelos presentan valores positivos de Gap_{CV} y Gap_{rep}, lo que indica que el desempeño en entrenamiento es superior al observado en validación cruzada y en prueba. Este comportamiento evidencia la presencia de sobreajuste en todas las configuraciones evaluadas, siendo más pronunciado en modelos como SVM_3, que presenta uno de los mayores gaps. A pesar de ello, su rendimiento final en prueba sigue siendo el mejor entre las alternativas consideradas.

El tiempo total del proceso fue de 00:03:35, lo que confirma que la representación basada en log-firma permite reducir significativamente el costo computacional en comparación con la firma estándar.

Tercer modelo

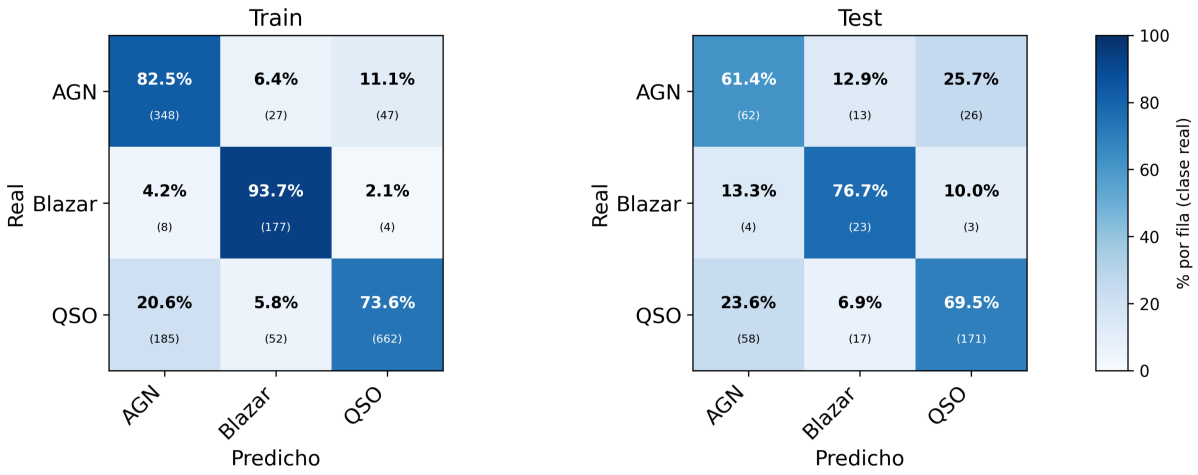


Figura 4.35: Matrices de confusión del entrenamiento y la prueba para el mejor modelo SVM utilizando log-firma (`iisignature`) sobre datos reales.

```
Accuracy : 0.786
Precision: 0.791
Recall   : 0.786
F1-score : 0.791
```

```
Accuracy : 0.679
Precision: 0.731
Recall   : 0.679
F1-score : 0.692
```

A partir de la matriz de confusión de prueba, se observa que el modelo logra un reconocimiento razonable de las tres clases, aunque con diferencias entre ellas. En la clase AGN se obtienen 62 clasificaciones correctas de 101 observaciones, mientras que en Blazar se logran 23 aciertos de 30 casos. Por su parte, la clase QSO alcanza 171 clasificaciones correctas de 246 ejemplos.

Esto se traduce en recalls aproximados de 0.61 para AGN, 0.77 para Blazar y 0.70 para QSO. En conjunto, estos resultados muestran que el modelo presenta un mejor desempeño en la identificación de la clase Blazar, mantiene un comportamiento intermedio en QSO y presenta mayores dificultades en la clase AGN, donde se observa una mayor dispersión de errores hacia las otras categorías.

En conjunto, estos resultados muestran que la log-firma con `iisignature` permite obtener un modelo con desempeño aceptable sobre los datos reales, aunque más bajo que el observado en las mejores configuraciones basadas en firma estándar. Aun así, el modelo logra distinguir de forma razonable entre las clases y tiene la ventaja de requerir poco tiempo de ejecución, por lo que puede ser una opción útil cuando se busca reducir el costo computacional.

Con el fin de resumir de manera conjunta los resultados obtenidos en datos reales, se presenta el desempeño de los mejores modelos SVM construidos con `esig` e `iisignature`, considerando tanto la

firma como la log-firma. En ella se comparan las métricas principales de validación cruzada y prueba, junto con los hiperparámetros seleccionados y el tiempo total de ejecución de cada configuración.

Representación	Mejor modelo	AUC_{CV}	Acc_{train}	Acc_{test}	$F1_{w,test}$	Hiperparámetros							Tiempo total
						k best	C	γ	Class weight	Imp.	Scaler	SMOTE	
ESIG firma	SVM_2	0.6857	0.865	0.8078	0.8064	1022	10.7722	0.05826	balanced	median	Standard	Sí	00:23:25
ESIG log-firma	SVM_3	0.5650	0.727	0.6462	0.6646	127	212.3514	0.02729	{1,0, 3,0, 0,7}	median	Standard	Sí	00:03:29
IISIGNATURE firma	SVM_3	0.6988	0.866	0.7613	0.7604	1022	784.8082	$8{,}22 \times 10^{-5}$	balanced	median	Standard	Sí	00:23:53
IISIGNATURE log-firma	SVM_3	0.6027	0.786	0.6790	0.6921	127	212.3514	0.02729	{1,0, 3,0, 0,7}	median	Standard	Sí	00:03:35

Tabla 4.38: Comparación de los mejores modelos SVM obtenidos con `esig` e `iisignature`, usando firma y log-firma sobre datos reales.

En términos generales, la representación basada en firma mostró un desempeño superior a la log-firma en ambas librerías. Entre las cuatro combinaciones evaluadas, el mejor rendimiento global se obtuvo con `esig` firma, que alcanzó el mayor desempeño en el conjunto de prueba, con una accuracy de 0.8078 y un F1-score ponderado de 0.8064, evidenciando una mejor capacidad de generalización.

La representación basada en `iisignature` firma también presentó un desempeño competitivo, aunque levemente inferior. En contraste, las configuraciones basadas en log-firma mostraron una disminución consistente en las métricas de clasificación, lo que sugiere que, si bien esta representación reduce significativamente el costo computacional, lo hace a costa de una pérdida relevante de información discriminante necesaria para separar adecuadamente las clases astronómicas. Estos resultados indican que la firma estándar constituye la mejor representación para este problema, especialmente cuando se prioriza el rendimiento predictivo por sobre la eficiencia computacional.

Ahora se analiza la importancia de las variables agrupadas por nivel de la path signature, con el fin de identificar qué órdenes de esta representación aportan más información al modelo SVM seleccionado con `esig` en datos reales. Para ello, se calculó la importancia de las variables mediante permutation importance sobre el pipeline final y luego se agruparon según el nivel correspondiente. En la figura, las barras representan la importancia promedio de las variables en cada nivel y los puntos muestran la mediana.

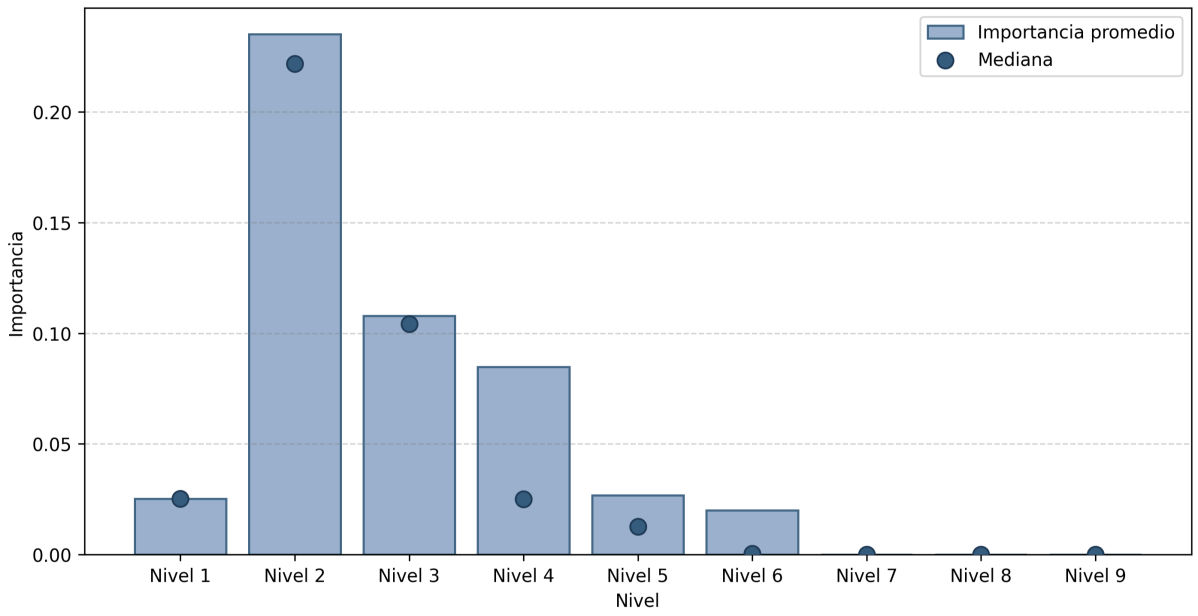


Figura 4.36: Importancia promedio y mediana por nivel para el modelo SVM seleccionado con firma `esig` sobre datos reales.

Los resultados muestran que la mayor importancia se concentra claramente en el nivel 2, seguido por los niveles 3 y 4. El nivel 1 presenta un aporte menor, mientras que desde el nivel 5 en adelante la contribución disminuye de forma importante. En particular, los niveles 7, 8 y 9 prácticamente no aportan al modelo final. En conjunto, esto sugiere que, para la representación basada en firma con `esig` en datos reales, el modelo SVM obtiene la mayor parte de su capacidad discriminante a partir de niveles bajos e intermedios, especialmente entre los niveles 2 y 4, mientras que los términos de orden más alto tienen un papel mucho más reducido.

Este resultado sugiere que las variables más influyentes se concentran principalmente en los primeros niveles de la firma, por lo que las mayores ganancias del modelo deberían esperarse antes de incorporar los niveles más altos de la representación.

Ahora se analiza el comportamiento del modelo SVM_2 al incorporar de manera acumulada los distintos niveles de la path signature construida con `esig` sobre los datos reales. Los resultados muestran una mejora progresiva en todas las métricas de prueba a medida que se agregan niveles de la representación. En particular, el AUC_{rep} aumenta desde 0.5100 en el nivel 1 hasta 0.8564 en el nivel 9, mientras que la accuracy y el F1-score ponderado también crecen de forma sostenida, alcanzando en el nivel más alto valores de 0.8078 y 0.8064, respectivamente. Esto sugiere que, en este caso, la incorporación de niveles superiores de la firma permite capturar información adicional que mejora la capacidad de discriminación del clasificador.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	AUC Train	Acc Test	F1 Test	AUC Test
1	2	0.5932	0.4829	0.5294	0.6462	0.5332	0.5100
2	6	0.5723	0.5196	0.6138	0.6212	0.5668	0.5811
3	14	0.6551	0.6404	0.7603	0.6713	0.6588	0.7396
4	30	0.6961	0.6876	0.7959	0.6741	0.6655	0.7571
5	62	0.7239	0.7199	0.8286	0.6825	0.6800	0.7815
6	126	0.7740	0.7719	0.8648	0.7437	0.7405	0.8092
7	254	0.8088	0.8079	0.8980	0.7549	0.7523	0.8320
8	510	0.8394	0.8394	0.9242	0.7744	0.7735	0.8441
9	1022	0.8651	0.8651	0.9403	0.8078	0.8064	0.8564

Tabla 4.39: Comparación del rendimiento del mejor modelo SVM con firma `esig` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos reales.

Bajo este criterio, el mejor desempeño se alcanza en el nivel 9, con 1022 variables, por lo que este corresponde al nivel más simple que maximiza el rendimiento en prueba. A diferencia de otros casos en los que el desempeño se estabiliza o disminuye al agregar niveles altos, aquí la mejora se mantiene hasta el final. En conjunto, estos resultados indican que la representación completa de la firma estándar resulta más informativa que sus versiones truncadas para clasificar los datos reales.

Este resultado complementa el análisis de importancia por nivel presentado anteriormente. Aunque las variables más influyentes en promedio se concentran en los niveles bajos e intermedios, el análisis acumulado muestra que los niveles superiores siguen aportando información útil cuando se incorporan en conjunto. En otras palabras, los primeros niveles contienen las variables más fuertes de manera individual, pero la inclusión de niveles más altos permite mejorar gradualmente el rendimiento global del modelo hasta alcanzar su mejor resultado en el nivel 9.

En conclusión, los resultados obtenidos con SVM sobre datos reales muestran que la firma estándar fue superior a la log-firma en ambas librerías. Entre las cuatro configuraciones evaluadas, el mejor desempeño global se obtuvo con `esig` firma, cuyo modelo final alcanzó un AUC_{rep} de 0.8564, junto con una accuracy de 0.8078 y un F1-score ponderado de 0.8064, por lo que se selecciona como la mejor alternativa para este problema. La representación basada en firma con `iisignature` también mostró un desempeño competitivo, aunque algo menor, mientras que ambas variantes basadas en log-firma presentaron una disminución clara en capacidad predictiva. Sin embargo, la log-firma tuvo la ventaja de reducir de manera importante el tiempo de ejecución, pasando de alrededor de 23 minutos a poco más de 3 minutos. En conjunto, estos resultados indican que, para los datos reales, la firma estándar conserva mejor la información discriminante necesaria para la clasificación, y en particular la construida con `esig` ofrece el mejor equilibrio entre desempeño y robustez.

4.2.3. eXtreme gradient boosting

esig

Se evaluó la representación basada en path signature sobre los datos reales, manteniendo una estrategia de validación cruzada repetida 5×2 y priorizando inicialmente el desempeño OOF (out-of-fold) para seleccionar los modelos más estables. A partir de esta búsqueda, se identificaron cinco configuraciones destacadas, cuyas métricas de entrenamiento y prueba fueron evaluadas posteriormente sobre el conjunto de datos de prueba. El proceso de compilación tuvo una duración total de 2 horas, 5 minutos y 16 segundos.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
XGB_1	0.7597	0.0174	0.1789	0.7631	0.0135	0.1738	0.6685	0.6747
XGB_2	0.7567	0.0210	0.1731	0.7611	0.0161	0.1767	0.6992	0.7026
XGB_3	0.7575	0.0214	0.1793	0.7611	0.0160	0.1772	0.7047	0.7066
XGB_4	0.7580	0.0192	0.2036	0.7602	0.0140	0.2025	0.6880	0.6914
XGB_5	0.7615	0.0213	0.1698	0.7638	0.0159	0.1636	0.6825	0.6879

Tabla 4.40: Desempeño de los cinco mejores modelos XGBoost utilizando firma (`esig`) sobre datos reales.

Los cinco modelos presentan resultados muy similares entre sí, con valores de AUC y métricas de clasificación cercanos, lo que sugiere una estabilidad razonable entre configuraciones. Sin embargo, las diferencias observadas en el conjunto de prueba permiten distinguir mejor su capacidad de generalización en datos no vistos.

Al comparar el comportamiento final sobre el conjunto de prueba, el mejor equilibrio entre accuracy, F1-score ponderado y macro-F1 corresponde a XGB_3, que alcanza una accuracy de 0.7047, un F1-score ponderado de 0.7066 y un macro-F1 de 0.6518. Por esta razón, XGB_3 se selecciona como la configuración final, ya que presenta el mejor rendimiento global en datos no vistos.

Si bien los valores de AUC son muy similares entre los modelos, las diferencias en métricas de clasificación permiten distinguir mejor su desempeño real. En particular, XGB_3 supera levemente a XGB_2 y XGB_4 en accuracy y F1-score, mientras que XGB_1 y XGB_5 presentan resultados inferiores.

Además, todos los modelos exhiben valores positivos de Gap, lo que indica que el desempeño en entrenamiento es superior al observado en prueba. Esto evidencia la presencia de sobreajuste en todas las configuraciones, aunque con distinta intensidad. En particular, XGB_4 presenta el mayor gap, lo que sugiere una menor capacidad de generalización.

Tercer modelo

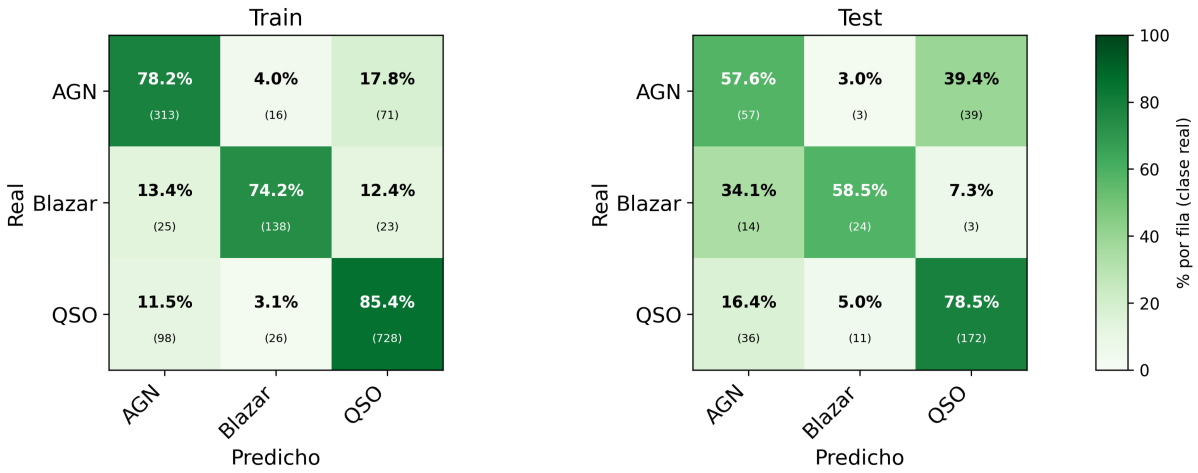


Figura 4.37: Matrices de confusión del entrenamiento y la prueba para el mejor modelo XGB utilizando firma (`esig`) sobre datos reales.

```
Accuracy : 0.820
Precision: 0.820
Recall   : 0.793
F1-score : 0.791
```

```
Accuracy : 0.705
Precision: 0.710
Recall   : 0.705
F1-score : 0.706
```

A partir de la matriz de confusión de prueba, se observa que el modelo logra una separación razonable entre las clases, aunque con errores más notorios que los observados en los mejores modelos SVM. En la clase AGN se obtienen 57 clasificaciones correctas de 99 observaciones, mientras que en Blazar se logran 24 aciertos de 41 casos. Por su parte, la clase QSO alcanza 172 clasificaciones correctas de 219 ejemplos. Los resultados muestran que el modelo presenta un mejor desempeño en la identificación de la clase QSO, mientras que mantiene un comportamiento más limitado en AGN y Blazar, donde se observa una mayor confusión entre clases.

Posteriormente, se evaluó la representación basada en log-firma sobre los datos reales, manteniendo la misma estrategia de validación cruzada repetida 5×2 y el mismo esquema general de búsqueda utilizado en el análisis anterior. A partir de este proceso, se seleccionaron cinco configuraciones destacadas según su desempeño out-of-fold. Sin embargo, para elegir el modelo final se consideró principalmente su comportamiento sobre el conjunto de prueba. El tiempo de cómputo fue de 12 minutos y 5 segundos.

Con el fin de resumir estos resultados, se presenta el desempeño de los cinco mejores modelos XGBoost obtenidos con log-firma en datos reales.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
XGB_1	0.7248	0.0273	0.1508	0.7271	0.0211	0.1586	0.6407	0.6471
XGB_2	0.7250	0.0294	0.1251	0.7254	0.0224	0.1332	0.6100	0.6183
XGB_3	0.7277	0.0278	0.1674	0.7277	0.0227	0.1638	0.6212	0.6285
XGB_4	0.7222	0.0238	0.2383	0.7221	0.0201	0.2380	0.6295	0.6331
XGB_5	0.7242	0.0285	0.1150	0.7250	0.0215	0.1211	0.5989	0.6086

Tabla 4.41: Desempeño de los cinco mejores modelos XGBoost utilizando log-firma (`esig`) sobre datos reales.

En términos generales, los cinco modelos presentan desempeños similares tanto en validación cruzada como en prueba, lo que sugiere una estabilidad razonable entre las configuraciones evaluadas. No obstante, las diferencias en las métricas de clasificación permiten distinguir su capacidad de generalización.

Al comparar el comportamiento final sobre el conjunto de prueba, el mejor equilibrio entre accuracy y F1-score ponderado corresponde a XGB_1, que alcanza una accuracy de 0.6407 y un F1-score ponderado de 0.6471. Por esta razón, XGB_1 se selecciona como el modelo final para la representación basada en log-firma con `esig` en datos reales.

Además, todos los modelos presentan valores positivos de Gap_{CV} y Gap_{rep}, lo que indica que el desempeño en entrenamiento es superior al observado en prueba. Esto evidencia la presencia de sobreajuste en todas las configuraciones evaluadas, siendo más pronunciado en modelos como XGB_4, que presenta la mayor diferencia entre entrenamiento y prueba.

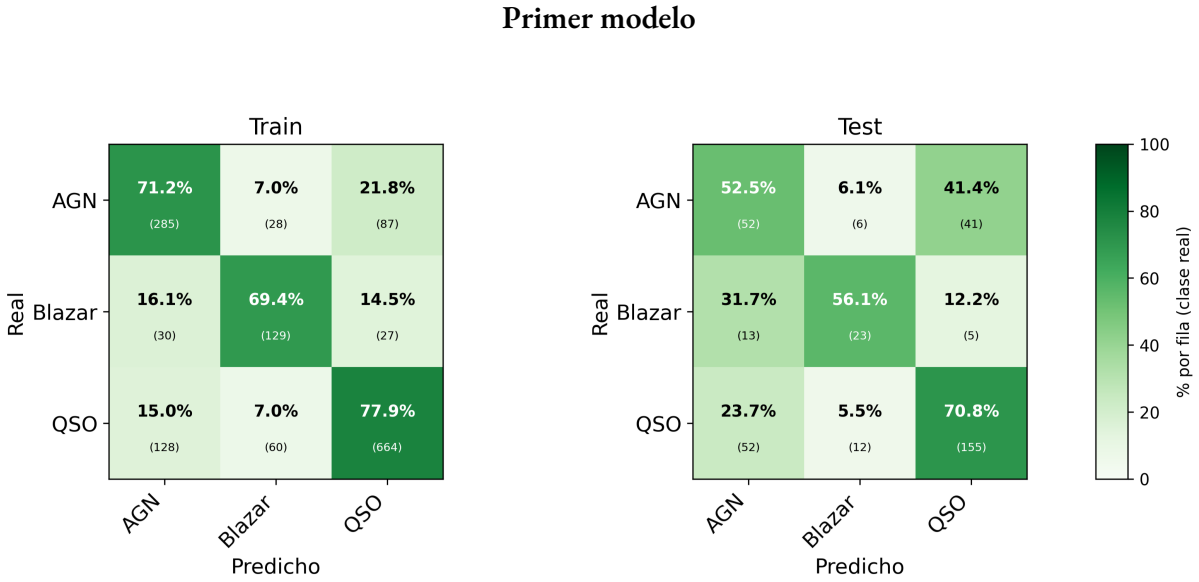


Figura 4.38: Matrices de confusión del entrenamiento y la prueba para el mejor modelo XGB utilizando log-firma (`esig`) sobre datos reales.

```
Accuracy : 0.750
Precision: 0.760
Recall   : 0.750
F1-score : 0.750
```

```
Accuracy : 0.641
Precision: 0.660
Recall   : 0.641
F1-score : 0.647
```

A partir de la matriz de confusión de prueba, se observa que el modelo logra una separación aceptable entre las clases, con un mejor desempeño en la clase QSO y mayores dificultades en AGN. En la clase AGN se obtienen 52 clasificaciones correctas de 99 observaciones, mientras que en Blazar se logran 23 aciertos de 41 casos. Por su parte, la clase QSO alcanza 155 clasificaciones correctas de 219 ejemplos. Esto se traduce en recalls aproximados de 0.53 para AGN, 0.56 para Blazar y 0.71 para QSO. En conjunto, estos resultados muestran que el modelo reconoce con mayor precisión la clase mayoritaria (QSO), mientras que presenta una mayor dispersión de errores en las clases minoritarias.

En conjunto, estos resultados indican que XGBoost con log-firma `esig` permite obtener un desempeño aceptable sobre los datos reales, aunque inferior al observado previamente con la firma estándar. Aun así, el modelo logra distinguir de manera razonable entre las clases y conserva la ventaja de trabajar con una representación más compacta y un tiempo total de ejecución relativamente bajo.

iisignature

Se entrenaron distintas configuraciones de XGBoost mediante validación cruzada estratificada repetida 5×2, priorizando inicialmente el desempeño out-of-fold y la estabilidad entre folds. La búsqueda total tomó 2 horas, 27 minutos y 42 segundos, y luego se evaluaron en detalle los cinco modelos mejor posicionados.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$\text{F1}_{\text{w,test}}$
XGB_1	0.7800	0.0129	0.1341	0.7816	0.0237	0.1309	0.5995	0.6100
XGB_2	0.7825	0.0127	0.1405	0.7831	0.0270	0.1362	0.6048	0.6138
XGB_3	0.7846	0.0194	0.1435	0.7862	0.0282	0.1362	0.6021	0.6112
XGB_4	0.7881	0.0131	0.1990	0.7862	0.0243	0.1993	0.6048	0.6119
XGB_5	0.7850	0.0134	0.1517	0.7840	0.0258	0.1522	0.5995	0.6090

Tabla 4.42: Desempeño de los cinco mejores modelos XGBoost utilizando firma (`iisignature`) sobre datos reales.

A primera vista, las diferencias entre los cinco modelos no son grandes, ya que todos se ubican en rangos muy próximos tanto en AUC como en las métricas finales de prueba. Sin embargo, al comparar directamente el rendimiento sobre el conjunto no visto, XGB_2 resulta más conveniente: comparte el mejor valor de accuracy en test y, además, alcanza el mayor F1-score ponderado. Por esta razón, se adopta como modelo final para esta representación. En cambio, XGB_4 logra valores de AUC algo más altos, pero su brecha entre entrenamiento y prueba es considerablemente mayor, lo que sugiere un ajuste menos equilibrado.

Segundo modelo

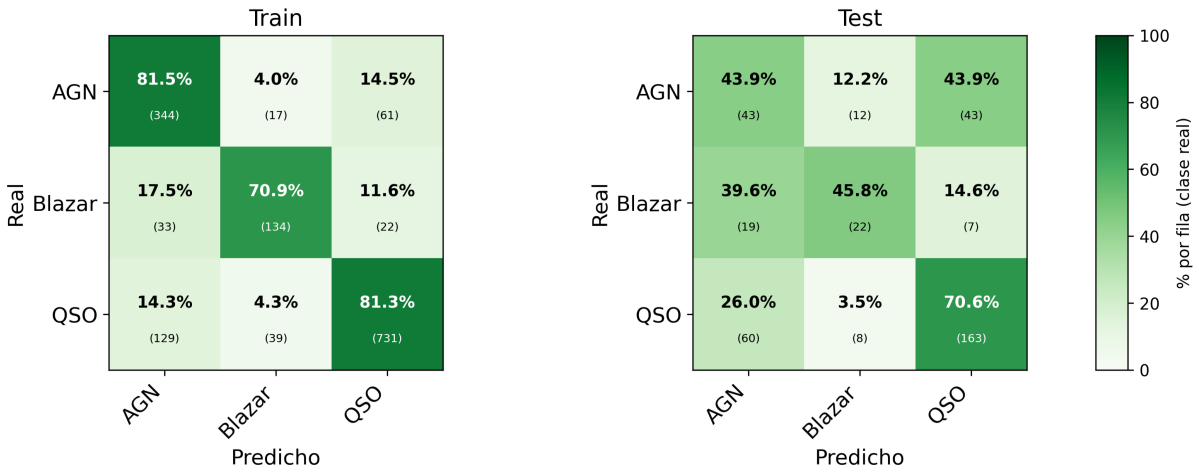


Figura 4.39: Matrices de confusión del entrenamiento y la prueba para el mejor modelo XGB utilizando firma (`iisignature`) sobre datos reales.

```
Accuracy : 0.801
Precision: 0.810
Recall   : 0.801
F1-score : 0.800
```

```
Accuracy : 0.605
Precision: 0.626
Recall   : 0.605
F1-score : 0.614
```

A partir de la matriz de confusión de prueba, se observa que el modelo presenta dificultades relevantes en las clases AGN y Blazar, mientras que mantiene un mejor desempeño en la clase QSO. En la clase AGN se obtienen 43 clasificaciones correctas de 98 observaciones, mientras que en Blazar se logran 22 aciertos de 48 casos. Por su parte, la clase QSO alcanza 163 clasificaciones correctas de 231 ejemplos. Esto se traduce en recalls aproximados de 0.44 para AGN, 0.46 para Blazar y 0.71 para QSO. En conjunto, estos resultados muestran que el modelo reconoce con mayor precisión la clase mayoritaria, mientras que presenta una mayor dispersión de errores en las clases minoritarias.

En el caso de la log-firma utilizando `iisignature`, el rendimiento general fue considerablemente menor en comparación con la firma estándar. Aunque las diferencias entre los cinco modelos fueron pequeñas, el mejor comportamiento sobre el conjunto de prueba correspondió a XGB_1, que alcanzó una accuracy de 0.4922 y un F1-score ponderado de 0.4907. Por esta razón, XGB_1 se selecciona como el modelo final para esta representación. En comparación con la firma estándar, se observa una caída importante en la capacidad de generalización, lo que sugiere que la log-firma conserva menos información discriminante para la clasificación en datos reales.

Modelo	AUC_{CV}	SD_{CV}	Gap_{CV}	AUC_{rep}	SD_{rep}	Gap_{rep}	Acc_{test}	$F1_{w,test}$
XGB_1	0.5581	0.0191	0.3705	0.5542	0.0164	0.3672	0.4922	0.4907
XGB_2	0.5525	0.0112	0.4145	0.5570	0.0109	0.4111	0.4690	0.4690
XGB_3	0.5450	0.0268	0.3136	0.5457	0.0207	0.3144	0.4845	0.4799
XGB_4	0.5579	0.0109	0.3888	0.5569	0.0109	0.3860	0.4496	0.4484
XGB_5	0.5589	0.0125	0.3788	0.5590	0.0128	0.3733	0.4690	0.4673

Tabla 4.43: Desempeño de los cinco mejores modelos XGBoost utilizando log-firma (`iisignature`) sobre datos reales.

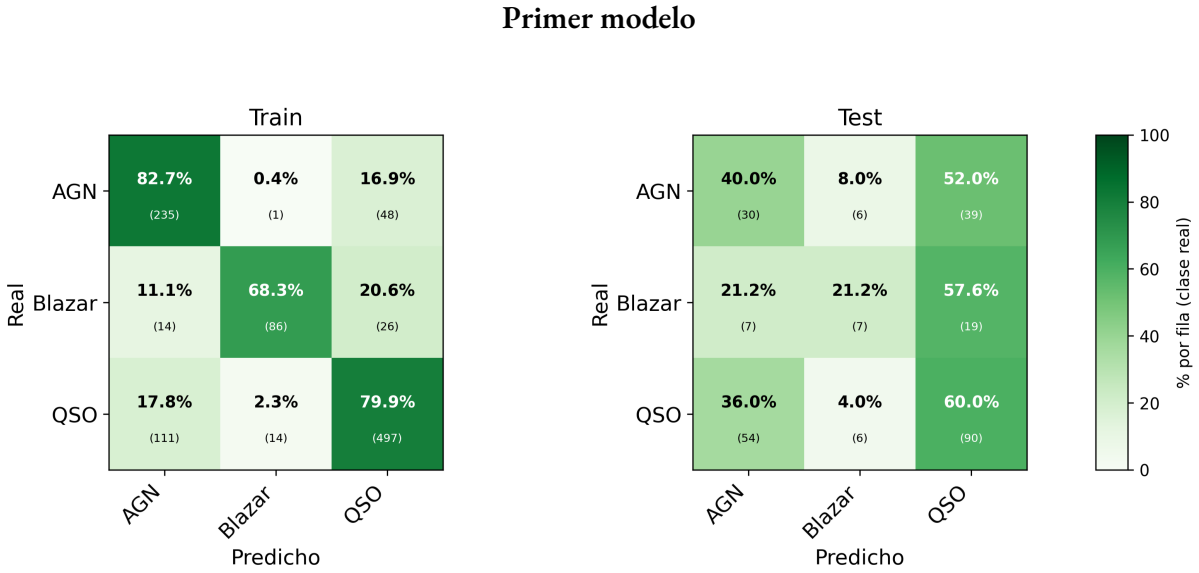


Figura 4.40: Matrices de confusión del entrenamiento y la prueba para el mejor modelo XGB utilizando log-firma (`iisignature`) sobre datos reales.

```
Accuracy : 0.793
Precision: 0.810
Recall   : 0.793
F1-score : 0.800
```

```
Accuracy : 0.492
Precision: 0.500
Recall   : 0.492
F1-score : 0.491
```

A partir de la matriz de confusión de prueba, se observa que el modelo presenta dificultades importantes en las tres clases, especialmente en AGN y Blazar. En la clase AGN se obtienen 30 clasificaciones correctas de 75 observaciones, mientras que en Blazar se logran 7 aciertos de 33 casos. Por su parte, la clase QSO alcanza 90 clasificaciones correctas de 150 ejemplos. Esto se traduce en recalls aproximados de 0.40 para AGN, 0.21 para Blazar y 0.60 para QSO. En conjunto, estos resultados muestran que el modelo logra capturar parcialmente la estructura de la clase mayoritaria, pero presenta una baja capacidad de discriminación en las clases minoritarias, lo que explica el bajo desempeño global observado.

Ahora se resumen los mejores modelos XGBoost obtenidos en datos reales, considerando las cuatro combinaciones entre librería y representación.

Representación	Mejor modelo	AUC_{CV}	AUC_{rep}	Acc_{test}	$F1_{w,test}$	Hiperparámetros							Tiempo total
						Trees	LR	Depth	Leaves	Min child	Gamma	Subsample	
ESIG firma	XGB_3	0.7575	0.7611	0.7047	0.7066	97	0.0445	4	85	39.8736	0.00010	0.6920	02:05:16
ESIG log-firma	XGB_1	0.7248	0.7271	0.6407	0.6471	131	0.0470	4	107	47.9510	0.7977	0.7649	00:12:05
IISIGNATURE firma	XGB_2	0.7825	0.7831	0.6048	0.6138	80	0.0232	5	128	11.9212	0.00384	0.8937	02:28:32
IISIGNATURE log-firma	XGB_1	0.5581	0.5542	0.4922	0.4907	33	0.0567	4	77	12.7809	0.00048	0.9941	00:06:50

Tabla 4.44: Comparación de los mejores modelos XGBoost obtenidos con `esig` e `iisignature`, usando firma y log-firma sobre datos reales.

A partir de esta comparación, la alternativa más conveniente en datos reales fue la firma calculada con `esig`. Aunque la configuración basada en `iisignature` con firma alcanzó los AUC promedio más altos, el modelo construido con `esig` entregó el mejor rendimiento final sobre el conjunto de prueba, tanto en accuracy como en F1-score ponderado. En consecuencia, se privilegia esta combinación dentro del estudio con XGBoost.

Para complementar la interpretación del modelo final, se analizó la importancia de las características agrupadas por nivel de la firma, donde las barras representan la importancia promedio y los puntos la mediana dentro de cada nivel. Este resumen permite distinguir no solo qué niveles aportan más, sino también si ese aporte proviene de muchas variables o de unas pocas especialmente influyentes.

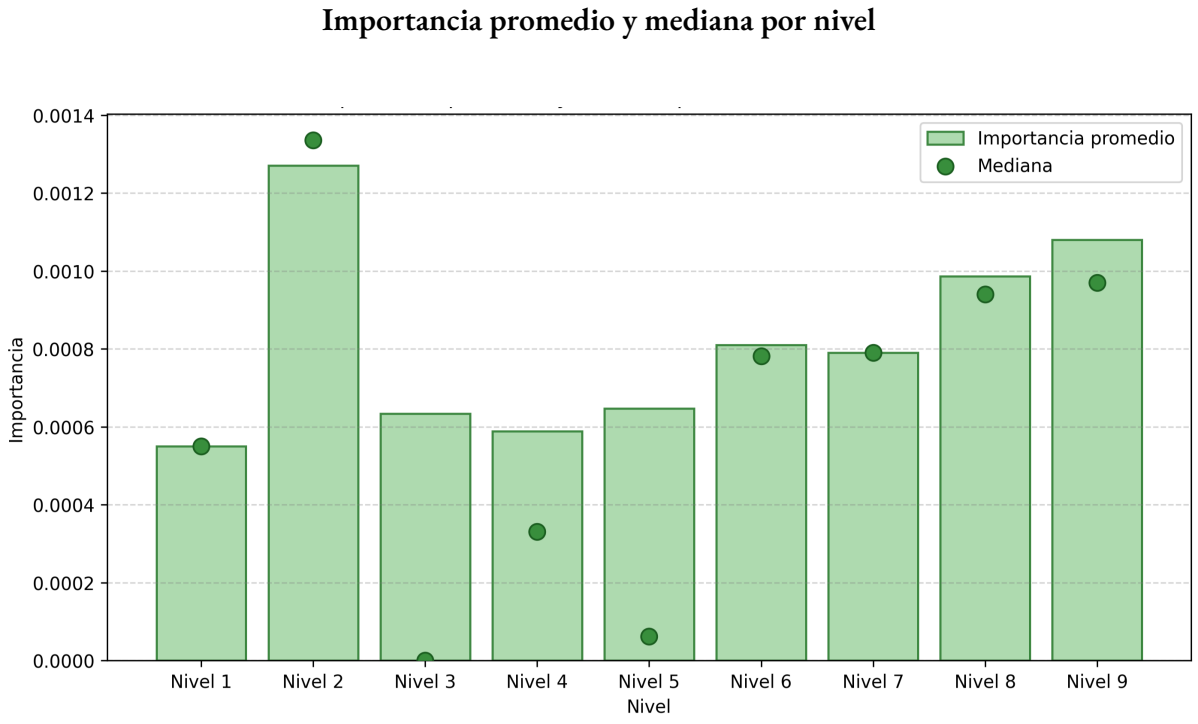


Figura 4.41: Importancia promedio y mediana por nivel para el modelo XGBoost seleccionado con firma `esig` sobre datos reales.

El gráfico muestra que el nivel 2 concentra la mayor importancia promedio y también la mediana más alta, lo que indica que en ese nivel hay variables particularmente informativas para el modelo. Aun así, los niveles superiores no desaparecen del análisis: los niveles 6 al 9 conservan importancias relevantes y

relativamente estables. En cambio, en algunos niveles intermedios, como 3 y 5, la distancia entre promedio y mediana sugiere que el peso del nivel depende de pocas variables destacadas. Esto sugiere que XGBoost aprovecha información distribuida en varios órdenes de la firma.

Finalmente, se evaluó si la representación completa de 1022 características era realmente necesaria o si podía lograrse un rendimiento similar usando solo una parte de la firma. Para ello, se comparó el modelo utilizando subconjuntos acumulativos hasta cada nivel.

Nivel de Firma	Nº de Características	Acc Train	F1 Train	AUC Train	Acc Test	F1 Test	AUC Test
1	2	0.5848	0.4723	0.5239	0.5850	0.4714	0.4862
2	6	0.5076	0.5178	0.6646	0.4596	0.4685	0.5607
3	14	0.6551	0.6621	0.8089	0.5432	0.5554	0.6860
4	30	0.6905	0.6957	0.8444	0.5850	0.5942	0.7136
5	62	0.7309	0.7346	0.8760	0.6351	0.6427	0.7417
6	126	0.7608	0.7633	0.8961	0.6490	0.6576	0.7754
7	254	0.7872	0.7893	0.9121	0.6741	0.6799	0.7952
8	510	0.7990	0.8008	0.9263	0.6769	0.6822	0.8026
9	1022	0.8206	0.8218	0.9377	0.6852	0.6889	0.8131

Tabla 4.45: Comparación del rendimiento del mejor modelo XGB con firma `esig` al utilizar subconjuntos acumulativos de características hasta cada nivel, sobre datos reales.

La evolución por niveles muestra una tendencia bastante clara: desde el nivel 3 en adelante, el rendimiento en prueba mejora de forma sostenida a medida que se incorporan características de mayor orden. El mejor resultado se obtiene en el nivel 9, que coincide con la representación completa. En otras palabras, aunque algunos niveles bajos contienen variables muy influyentes, la mejor capacidad de clasificación aparece cuando el modelo dispone de toda la estructura de la firma.

Para concluir, el estudio con XGBoost sobre datos reales deja tres ideas principales. Primero, la firma estándar funcionó mejor que la log-firma en ambas librerías, lo que indica que conservar la representación completa de la trayectoria resulta más útil para distinguir entre AGN, Blazar y QSO. Segundo, `esig` ofreció el mejor balance entre discriminación y desempeño final en prueba, por encima de las variantes construidas con `iisignature`. Y tercero, dentro del mejor caso, la representación completa volvió a ser la más efectiva: aunque los niveles bajos aportan señales importantes, el modelo mejora de manera consistente cuando incorpora también información de orden superior. Por lo tanto, para XGBoost en datos reales, la configuración final preferida corresponde a la firma estándar calculada con `esig`, utilizando todos los niveles disponibles.

5 Conclusión

La técnica de path signature representa un avance relevante en la clasificación astronómica, al mostrar que la geometría de las curvas de luz contiene información suficiente para identificar objetos en el espacio. A diferencia de clasificadores que incorporan variables externas como coordenadas, colores u otras características derivadas, este estudio obtiene resultados competitivos utilizando únicamente la firma del camino. Esto muestra que la forma en que varía la luz puede ser una base informativa y efectiva para clasificar objetos astronómicos, incluso en casos de variabilidad compleja, tanto en datos simulados como en datos reales.

Algoritmo	Representación	Mejor modelo	AUC_{CV}	AUC_{rep}	Acc_{test}	$\text{F1}_{w,test}$	Tiempo de cómputo
RF	ESIG firma	RF_1	0.7545	0.7584	0.7799	0.7788	13:37:58
RF	ESIG log-firma	RF_1	0.7219	0.7202	0.7242	0.7260	01:45:30
RF	IISIGNATURE firma	RF_1	0.7763	0.7779	0.7745	0.7769	13:28:17
RF	IISIGNATURE log-firma	RF_1	0.7322	0.7328	0.7082	0.7173	01:51:57
SVM	ESIG firma	SVM_2	0.6857	0.8564	0.8078	0.8064	00:23:25
SVM	ESIG log-firma	SVM_3	0.5650	0.7809	0.6462	0.6646	00:03:29
SVM	IISIGNATURE firma	SVM_3	0.6988	0.7707	0.6790	0.6921	00:23:53
SVM	IISIGNATURE log-firma	SVM_3	0.6027	0.7707	0.6790	0.6921	00:03:35
XGB	ESIG firma	XGB_3	0.7575	0.7611	0.7047	0.7066	02:05:16
XGB	ESIG log-firma	XGB_1	0.7248	0.7271	0.6407	0.6471	00:12:05
XGB	IISIGNATURE firma	XGB_2	0.7825	0.7831	0.6048	0.6138	02:28:32
XGB	IISIGNATURE log-firma	XGB_1	0.5581	0.5542	0.4922	0.4907	00:06:50

Tabla 5.1: Resumen de los mejores modelos obtenidos sobre datos reales para cada combinación de algoritmo, librería y representación. La fila verde destaca el mayor AUC_{CV} y la fila azul destaca la mayor accuracy en prueba.

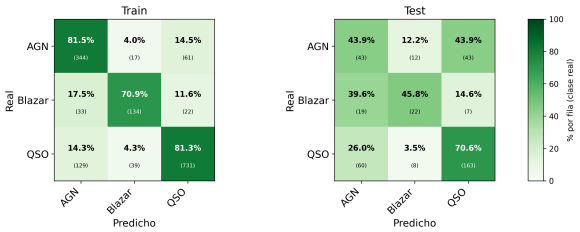
Por un lado, el mayor valor de AUC en validación cruzada fue alcanzado por XGBoost con firma estándar obtenida con `iisignature` (XGB_2), con un valor de 0.7825, lo que indica la mejor capacidad

discriminativa promedio entre las configuraciones evaluadas. Por otro lado, el mejor rendimiento final sobre el conjunto de prueba correspondió a SVM con firma estándar obtenida con `esig` (SVM_2), con una accuracy de 0.8078 y un F1-score ponderado de 0.8064. A pesar de estos buenos resultados, en los gráficos se observa cierta sobredispersión en las predicciones, especialmente en las clases AGN y Blazar, lo que se traduce en mayor confusión entre ellas y refleja la dificultad de separarlas completamente. Esto es esperable, ya que el modelo utiliza únicamente información derivada de la path signature, sin incorporar características adicionales que podrían ayudar a mejorar la separación entre clases.

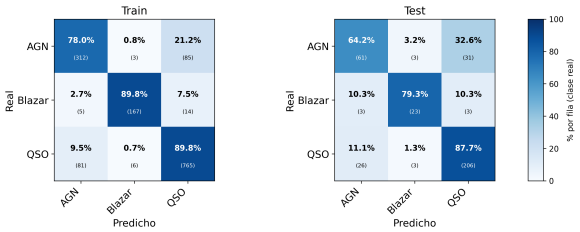
Mientras XGBoost destaca por su capacidad de separar las clases de manera global según la métrica AUC, SVM muestra un mejor comportamiento en la clasificación efectiva de ejemplos no vistos. En consecuencia, ambos modelos resultan relevantes para interpretar los resultados finales del estudio: XGBoost como el modelo con mayor capacidad discriminativa promedio, y SVM como el modelo con mejor desempeño práctico en el escenario real.

No obstante, el modelo Random Forest con firma estándar utilizando `esig` también muestra un buen desempeño, con un AUC_{CV} de 0.7545, una accuracy en prueba de 0.7799 y un F1-score ponderado de 0.7788. Si bien no alcanza los valores más altos en ninguna métrica en particular, sus resultados son consistentes tanto en validación como en test. En comparación con los mejores modelos respecto a AUC_{CV} y accuracy, la diferencia es de aproximadamente 3 % a 4 %, lo que sigue siendo un buen rendimiento. Además, a diferencia de SVM, que presenta cierto subajuste, y de XGBoost, que muestra indicios de sobreajuste, Random Forest mantiene un comportamiento más estable entre entrenamiento y validación. Por esto, Random Forest con firma estándar se puede considerar como el modelo más equilibrado del estudio.

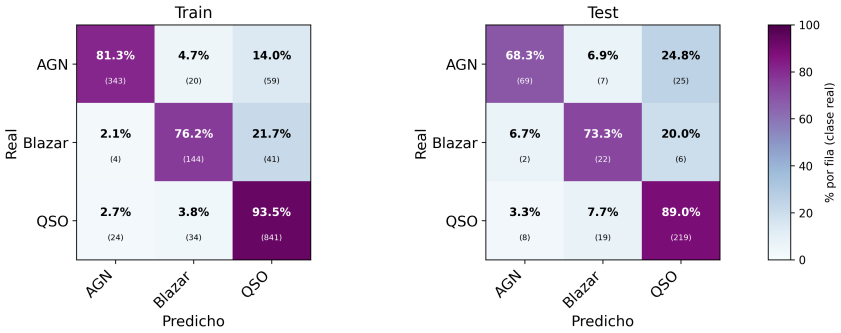
En general, la clase QSO es la mejor identificada, mientras que AGN y Blazar presentan mayor confusión entre sí. Esto es consistente con su naturaleza astronómica, ya que ambos corresponden a núcleos activos con propiedades observacionales similares, cuya principal diferencia radica en la orientación del sistema respecto al observador, o sea, en el ángulo desde el cual se observa la emisión del objeto. Esta similitud dificulta su separación cuando se utilizan únicamente características derivadas de curvas de luz.



(a) Matriz de confusión del modelo XGB_2, correspondiente a la configuración con mayor AUC en validación cruzada sobre datos reales.



(b) Matriz de confusión del modelo SVM_2, correspondiente a la configuración con mejor accuracy sobre el conjunto de prueba en datos reales.



(c) Matriz de confusión del modelo RF_1, correspondiente a la configuración más equilibrada entre desempeño y generalización.

Figura 5.1: Comparación entre las matrices de confusión de los modelos más relevantes sobre datos reales: XGBoost con mayor AUC en validación cruzada, SVM con mejor desempeño en el conjunto de prueba y Random Forest como modelo más equilibrado.

Al comparar estos resultados con el clasificador reportado por ALeRCE, es importante tener en cuenta que ambos enfoques no utilizan la misma información de entrada. La matriz de confusión publicada por Sánchez-Sáez et al. (2021) [19] muestra un buen desempeño, aunque mantiene cierta confusión entre las clases AGN, Blazar y QSO, lo que refleja la dificultad de distinguir objetos con patrones observacionales similares. En este trabajo, los modelos seleccionados también enfrentan esa misma dificultad; sin embargo, los resultados obtenidos muestran que la representación basada en path signature permite alcanzar una separación competitiva entre estas tres categorías utilizando únicamente la información contenida en la curva de luz.

En términos cuantitativos, se observa que dicho enfoque logra clasificar correctamente cerca de un 87 % de los objetos QSO, mientras que en esta tesis se alcanza aproximadamente un 78 %. Si bien existe una diferencia cercana a 9 puntos porcentuales, esta resulta esperable considerando que en el trabajo de referencia se incorporan características adicionales, tales como información espectral y parámetros derivados del dominio del tiempo, mientras que en esta tesis se utilizan exclusivamente representaciones basadas en la curva de luz mediante path signature. En este contexto, el desempeño alcanzado es consistente y demuestra que, aun con un conjunto de variables más limitado, el modelo es capaz de capturar una proporción importante de la información discriminante, lo que refuerza el potencial de esta metodología como herramienta de clasificación.

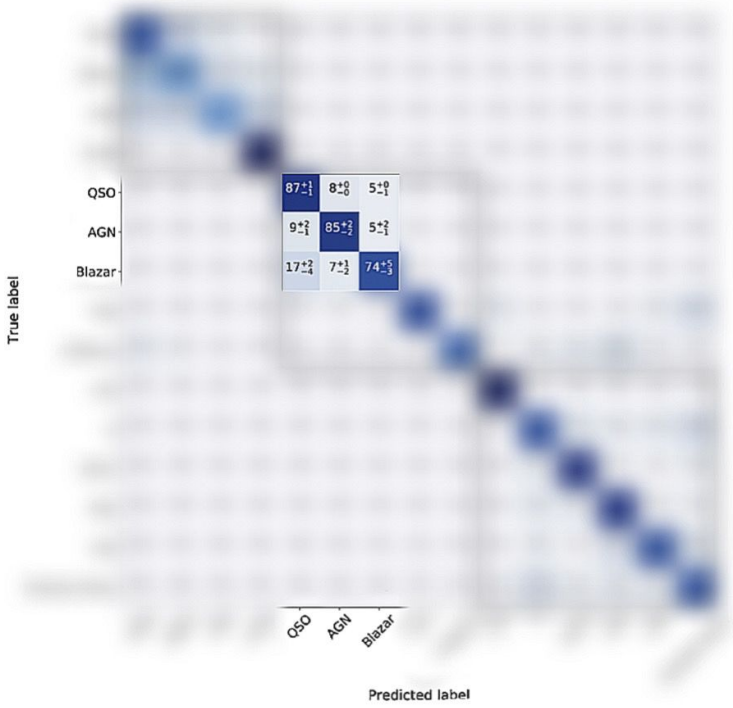


Figura 5.2: Matriz de confusión publicada por Sánchez-Sáez et al. (2021) para el clasificador del broker ALeRCE [19].

En particular, el modelo SVM con firma estándar presentó una clasificación más equilibrada entre clases y un mejor desempeño final en el conjunto de prueba, mientras que el modelo XGBoost con firma estándar mostró la mejor capacidad discriminativa promedio según validación cruzada. En conjunto, ambos resultados apoyan una misma conclusión: la principal fortaleza del enfoque propuesto no depende exclusivamente de un algoritmo específico, sino del uso de la firma estándar como representación de las curvas de luz. Esto refuerza la idea de que la geometría de la trayectoria observada contiene información útil para distinguir objetos extragalácticos complejos.

Trabajar primero con datos simulados fue una etapa metodológicamente útil, ya que permitió probar los modelos en un escenario más controlado antes de pasar a los datos reales, que son más complejos y ruidosos. Esta fase sirvió para observar cómo se comportaban las distintas representaciones y algoritmos, identificar cuáles mostraban mejores resultados y entender mejor sus limitaciones. Además, permitió llegar al análisis con datos reales con una base más clara sobre qué configuraciones valía la pena estudiar con mayor detalle. En ese sentido, la simulación no fue un paso previo que ayudó a orientar y respaldar el análisis final.

Algoritmo	Representación	Mejor modelo	AUC_{CV}	Acc_{test}	$F1_{w,test}$	Tiempo de cómputo
RF	ESIG firma	RF_1	0.6047	0.8223	0.8265	14:11:36
RF	ESIG log-firma	RF_1	0.5795	0.7798	0.7775	02:18:30
RF	IISIGNATURE firma	RF_1	0.6020	0.8090	0.8114	14:50:44
RF	IISIGNATURE log-firma	RF_1	0.5555	0.7558	0.7502	01:48:23
SVM	ESIG firma	SVM_5	0.5204	0.8462	0.8506	00:19:22
SVM	ESIG log-firma	SVM_1	0.5562	0.7480	0.7551	00:03:55
SVM	IISIGNATURE firma	SVM_4	0.5082	0.8223	0.8256	00:19:38
SVM	IISIGNATURE log-firma	SVM_1	0.5491	0.7268	0.7367	00:04:01
XGB	ESIG firma	XGB_4	0.6015	0.8568	0.8581	02:04:09
XGB	ESIG log-firma	XGB_2	0.5627	0.7745	0.7760	00:11:20
XGB	IISIGNATURE firma	XGB_4	0.4587	0.8647	0.8658	02:32:22
XGB	IISIGNATURE log-firma	XGB_4	0.4170	0.7401	0.7476	00:09:34

Tabla 5.2: Resumen de los mejores modelos obtenidos sobre datos simulados para cada combinación de algoritmo, librería y representación. La fila morada destaca el mayor AUC_{CV} y la fila verde destaca la mayor accuracy en prueba.

En los datos simulados, los modelos en general tuvieron buenos resultados, aunque no todos destacaron en lo mismo. Random Forest con firma estándar usando `esig` obtuvo el mayor AUC en validación cruzada (0.6047), superando levemente a su equivalente con `iisignature` (0.6020). Por otro lado, XGBoost con firma estándar usando `iisignature` alcanzó la mayor accuracy en prueba (0.8647) y el mayor F1-score ponderado (0.8658), superando a SVM en aproximadamente 1.8 puntos porcentuales en accuracy. SVM también mostró un desempeño alto, especialmente con firma estándar usando `esig`, con una accuracy de 0.8462 y un F1-score ponderado de 0.8506, quedando a menos de 2 puntos porcentuales del mejor resultado. En conjunto, estos resultados muestran que en los datos simulados la firma estándar funcionó mejor que la log-firma en los tres algoritmos, y también que el mejor modelo depende de la métrica que se considere: Random Forest destacó en AUC, XGBoost en accuracy y F1-score, y SVM presentó un rendimiento competitivo y cercano a los mejores valores observados.

Al comparar estos resultados con los obtenidos en datos reales, se observa que el desempeño disminuye en todos los casos, lo que era esperable porque las curvas reales son más complejas, ruidosas y presentan mayor solapamiento entre clases. Por ejemplo, en Random Forest con firma estándar (`esig`), la accuracy pasa de 0.8223 a 0.7799, lo que corresponde a una caída de aproximadamente un 5 %. En SVM con firma estándar (`esig`), la accuracy disminuye desde 0.8462 a 0.8078 (alrededor de un 4.5 %), mientras que en XGBoost con firma estándar (`iisignature`), la caída es más pronunciada, desde 0.8647 a 0.6048 (cerca de un 30 %). Aun así, se mantuvo una tendencia clara: la firma estándar siguió siendo mejor que la log-firma, con diferencias entre un 3 % y 7 % en accuracy dependiendo del modelo, mientras que RF mostró el comportamiento más sólido en el escenario real. Además, los modelos que destacan en datos simulados coinciden con los que presentan mejor desempeño en datos reales, lo que sugiere una consistencia en los resultados y refuerza la validez del enfoque utilizado. Por ende, los da-

tos simulados sirvieron como una referencia útil para orientar el análisis.

Otro resultado consistente a lo largo del estudio fue la superioridad de la firma por sobre la log-firma. Aunque la log-firma redujo el costo computacional en varias configuraciones, sus resultados fueron sistemáticamente inferiores en comparación con la firma estándar, tanto en datos simulados como en datos reales. Esto indica que la firma completa truncada al nivel 9 conserva de mejor manera la información necesaria para separar clases astronómicas con comportamiento similar. En consecuencia, el uso de la firma estándar se consolidó como la representación más adecuada dentro de las alternativas evaluadas.

Asimismo, las comparaciones entre librerías mostraron que tanto `esig` como `iisignature` permiten construir modelos competitivos, aunque con diferencias en rendimiento según el algoritmo utilizado. En términos generales, `esig` destacó especialmente en el modelo SVM con mejor accuracy final, mientras que `iisignature` alcanzó el mayor AUC con XGBoost en los datos reales. Esto sugiere que las diferencias entre implementaciones pueden influir en el resultado, pero que el factor más determinante sigue siendo la elección de la representación basada en firma.

El tiempo de cómputo también fue un criterio importante en la comparación entre modelos. Random Forest presentó los mayores costos computacionales, mientras que SVM alcanzó el mejor desempeño sobre datos reales con tiempos bastante menores, lo que refuerza su conveniencia práctica. XGBoost, por su parte, fue una alternativa intermedia, con buena capacidad discriminativa, aunque sin lograr el mejor rendimiento final.

A pesar de los resultados obtenidos, este trabajo presenta algunas limitaciones propias del enfoque utilizado. En primer lugar, se trabajó únicamente con características derivadas de path signature, sin incorporar otras variables relevantes como información espectral o parámetros físicos del objeto, lo que limita la comparación con modelos más completos. En segundo lugar, el análisis se enfoca solo en tres clases (AGN, Blazar y QSO), por lo que los resultados no necesariamente se generalizan a otros problemas de clasificación astronómica. Finalmente, el uso de firmas de nivel alto implica una gran cantidad de variables (más de 1000 en nivel 9) y, por lo tanto, una alta carga computacional. Esto se refleja en tiempos de entrenamiento elevados, especialmente en Random Forest, lo que puede dificultar su uso en conjuntos de datos más grandes o en aplicaciones en tiempo real.

En síntesis, los resultados de esta investigación permiten concluir que las características derivadas de path signature constituyen una buena representación para la clasificación de AGN, Blazar y QSO. Si el criterio principal es la capacidad discriminativa promedio estimada mediante validación cruzada, el modelo más destacado corresponde a XGBoost con firma estándar obtenida con `iisignature`. En cambio, si el criterio prioritario es el desempeño final sobre datos no vistos, el modelo más conveniente es SVM con firma estándar obtenida con `esig`. No obstante, el modelo basado en Random Forest

con firma estándar obtenida con `esig` presenta el comportamiento más equilibrado, ya que mantiene un buen desempeño tanto en validación cruzada como en el conjunto de prueba, sin diferencias marcadas entre ambas métricas. Por esta razón, se considera como la alternativa más robusta dentro de los modelos evaluados. En este sentido, la contribución principal de este trabajo no radica únicamente en identificar un algoritmo ganador, sino en demostrar que la representación basada en firma estándar permite mejorar la separación entre clases y constituye una alternativa prometedora para futuros problemas de clasificación astronómica.

5.1. Trabajos futuros

- Extender el estudio a más clases astronómicas, no solo AGN, Blazar y QSO. Esto permitiría evaluar si la representación basada en path signature mantiene su capacidad discriminativa en escenarios de clasificación más exigentes, donde exista una mayor diversidad de patrones de variabilidad. Además, incorporar un conjunto más amplio de objetos haría posible analizar con mayor profundidad la escalabilidad del enfoque y su utilidad en problemas astronómicos más cercanos a contextos reales de operación.

- Probar otros modelos de aprendizaje automático que puedan capturar dependencias temporales más complejas, por ejemplo, redes neuronales. Si bien en este trabajo se evaluaron modelos clásicos como Random Forest, SVM y XGBoost, resulta de interés estudiar modelos capaces de modelar relaciones no lineales de mayor complejidad y estructuras temporales más profundas. Esto permitiría analizar si la información contenida en la path signature puede seguir siendo aprovechada por modelos más flexibles, o si incluso conviene comparar su desempeño con enfoques que trabajen directamente sobre las curvas de luz sin necesidad de una representación previa.

- Analizar hasta qué punto aproximarse a la firma infinita mediante niveles de truncamiento más altos mejora el desempeño, considerando el equilibrio entre una mejor representación de la curva de luz, el costo computacional y el riesgo de sobreajuste. En este estudio, la firma truncada de nivel 9 mostró ser una representación efectiva; sin embargo, queda abierta la pregunta sobre si niveles superiores podrían capturar información adicional útil para distinguir clases con patrones muy similares. Al mismo tiempo, avanzar hacia truncamientos más altos implica un crecimiento importante en la dimensión del espacio de características, por lo que también sería necesario estudiar con mayor detalle el compromiso entre riqueza descriptiva, estabilidad estadística y factibilidad computacional.

- Incorporar información multibanda, con el fin de evaluar si el uso conjunto de distintas bandas fotométricas mejora la capacidad de clasificación. En este trabajo se utilizó una sola banda con el objetivo de mantener una representación más simple y homogénea de las curvas de luz; sin embargo, muchas clases astronómicas presentan diferencias relevantes cuando se observan en distintas longitudes de onda. Por ello, integrar información multibanda podría optimizar la

representación del objeto y aportar información complementaria que ayude a mejorar la separación entre clases con comportamientos similares en una sola banda.

- Evaluar estrategias más específicas para enfrentar el desbalance de clases, especialmente en objetos menos representados, como Blazar. Aunque en este estudio se incorporaron mecanismos para abordar este problema, el desbalance siguió siendo un factor importante en la dificultad de clasificación. En este sentido, sería relevante analizar con mayor detalle técnicas de remuestreo, reponderación, entre otros, con el fin de mejorar el reconocimiento de las clases minoritarias sin deteriorar el desempeño global del modelo.

- Explorar reducciones de dimensionalidad o métodos de selección de variables más avanzados, que permitan disminuir el costo computacional sin perder capacidad predictiva. La representación basada en path signature genera vectores de características de alta dimensión, lo que puede afectar tanto el tiempo de cómputo como la estabilidad de algunos modelos. Por ello, sería útil estudiar estrategias que permitan conservar la información más relevante de la firma, reduciendo al mismo tiempo la complejidad del problema y facilitando su aplicación en conjuntos de datos más grandes.

Anexo

5.2. Hiperparámetros

Random forest

Hiperparámetro	Definición	Impacto en el modelo
`n_estimators`	Número de árboles que componen el bosque.	Un mayor número de árboles suele aumentar la estabilidad del modelo, aunque incrementa el tiempo de entrenamiento y predicción.
`criterion`	Criterio usado para medir la calidad de una división, como `gini` o `entropy`.	Determina cómo se construyen los nodos del árbol. Puede afectar levemente la calidad de la separación entre clases.
`max_depth`	Profundidad máxima permitida para cada árbol.	Árboles más profundos capturan patrones más complejos, pero también aumentan el riesgo de sobreajuste.
`max_features`	Proporción o número de variables consideradas al buscar la mejor división.	Valores menores aumentan la diversidad entre árboles; valores mayores pueden mejorar la precisión, pero también la correlación entre ellos.
`max_samples`	Proporción de observaciones usadas para entrenar cada árbol cuando se aplica muestreo.	Controla la variabilidad entre árboles. Puede reducir sobreajuste y afectar la robustez del ensamble.
`min_samples_leaf`	Número mínimo de observaciones requeridas en una hoja terminal.	Valores mayores suavizan el modelo y reducen el sobreajuste, aunque pueden perder detalle en la clasificación.
`min_samples_split`	Número mínimo de observaciones necesarias para dividir un nodo.	Restringe divisiones muy específicas. Ayuda a controlar la complejidad del árbol.
`class_weight`	Peso asignado a cada clase durante el entrenamiento.	Permite compensar el desbalance de clases, dando mayor importancia a las categorías minoritarias.

Tabla 5.3: Principales hiperparámetros de Random Forest, su definición e impacto.

Hiperparámetro	Valores evaluados
Número de árboles	600–5000 (valores enteros generados aleatoriamente)
Profundidad máxima	{None, 4, 5, 6, ..., 20}
Número de variables por división	{`sqrt`, `log2`, 0.20, 0.30, 0.40, 0.50, 0.60, 0.70, 0.80}
Muestras mínimas para dividir un nodo	20–200 (valores enteros generados aleatoriamente)
Muestras mínimas por hoja	5–50 (valores enteros generados aleatoriamente)
Proporción de muestras utilizadas por árbol	{0.4, 0.6, 0.8}
Criterio de división	{`gini`, `entropy`}
Ponderación de clases	AGN = 2.0, Blazar = 3.0, QSO = 1.0
Bootstrap	`True`

Tabla 5.4: Espacio de hiperparámetros evaluado para el modelo Random Forest.

Support vector machine

Hiperparámetro	Definición	Impacto en el modelo
`C`	Parámetro de regularización. Controla cuánto penaliza el modelo los errores de clasificación.	Valores altos buscan clasificar mejor los datos de entrenamiento, pero pueden aumentar el sobreajuste. Valores bajos generan un modelo más regularizado.
`gamma`	Parámetro del kernel RBF que controla el alcance de influencia de cada observación.	Valores altos generan fronteras más complejas y locales; valores bajos producen fronteras más suaves y globales.
`kernel`	Función utilizada para proyectar los datos a un espacio donde puedan separarse mejor.	Determina la forma de la frontera de decisión. En este trabajo, el kernel RBF permitió capturar relaciones no lineales.
`kbest`	Número de variables seleccionadas antes de entrenar el modelo.	Reduce dimensionalidad y ruido. Una selección adecuada puede mejorar el rendimiento y disminuir el costo computacional.
`class_weight`	Peso asignado a cada clase durante el entrenamiento.	Ayuda a enfrentar el desbalance de clases, forzando al modelo a prestar mayor atención a las clases menos representadas.
`scaler`	Método de escalamiento aplicado a las variables, como `StandardScaler`.	Es fundamental en SVM, ya que el modelo es sensible a la escala de las variables.
`SMOTE`	Técnica de sobremuestreo sintético aplicada solo al conjunto de entrenamiento.	Mejora la representación de las clases minoritarias y puede aumentar la capacidad de clasificación en escenarios desbalanceados.

Tabla 5.5: Principales hiperparámetros y componentes de SVM, su definición e impacto.

Hiperparámetro	Valores evaluados
Número de variables seleccionadas	{128, 256, 384, 512, 768, 1023}
Parámetro de regularización	0.5–2000 (valores generados aleatoriamente en escala logarítmica)
Parámetro asociado al alcance de influencia de cada observación	10^{-6}–0.1 (valores generados aleatoriamente en escala logarítmica)
Pesos de clase	{balanced}, {2.0, 4.0, 1.0}, {2.5, 5.0, 1.0}, {1.5, 3.5, 0.8}, {3.0, 6.0, 1.0}, {1.0, 3.0, 0.7}
Estrategia de imputación	Mediana
Estandarización	Activada
SMOTE	Activado dentro de cada fold de validación cruzada
Kernel	RBF

Tabla 5.6: Espacio de hiperparámetros evaluado para los modelos Support Vector Machine.

Hiperparámetro	Definición	Impacto en el modelo
`n_estimators`	Número de árboles que se agregan secuencialmente al modelo.	Más árboles pueden mejorar el ajuste, pero también aumentar el tiempo de cómputo y el riesgo de sobreajuste.
`learning_rate` (η)	Tasa de aprendizaje con que cada árbol corrige a los anteriores.	Valores pequeños suelen mejorar la generalización, aunque requieren más árboles para alcanzar buen desempeño.
`max_depth`	Profundidad máxima de cada árbol.	Controla la complejidad del modelo. Profundidades mayores capturan patrones más complejos, pero aumentan el riesgo de sobreajuste.
`min_child_weight`	Peso mínimo requerido en un nodo hijo para permitir una nueva división.	Valores altos hacen el modelo más conservador y reducen divisiones poco informativas.
`subsample`	Proporción de observaciones utilizadas para construir cada árbol.	Introduce aleatoriedad y puede mejorar la generalización al reducir sobreajuste.
`colsample_bytree`	Proporción de variables usadas al construir cada árbol.	Reduce dependencia entre árboles y puede mejorar robustez cuando existen muchas variables.
`reg_lambda`	Regularización L2 aplicada a los pesos del modelo.	Ayuda a controlar el sobreajuste, favoreciendo soluciones más estables.
`reg_alpha`	Regularización L1 aplicada a los pesos del modelo.	Puede inducir soluciones más simples y disminuir la influencia de variables poco relevantes.
`objective`	Función objetivo optimizada durante el entrenamiento, por ejemplo `multi:softprob`.	Define el tipo de problema que resuelve el modelo y la forma de sus predicciones.
`eval_metric`	Métrica usada para monitorear el entrenamiento.	Permite evaluar el progreso del modelo y apoyar decisiones como *early stopping*.

Tabla 5.7: Principales hiperparámetros de XGBoost, su definición e impacto.

Hiperparámetro	Valores evaluados
Tasa de aprendizaje	0.007–0.06 (valores continuos generados aleatoriamente)
Profundidad máxima	2–5 (valores enteros generados aleatoriamente)
Peso mínimo por nodo hijo	10–150 (valores continuos generados aleatoriamente)
Proporción de observaciones utilizadas por árbol	0.65–1.00 (valores continuos generados aleatoriamente)
Proporción de variables utilizadas por árbol	0.65–1.00 (valores continuos generados aleatoriamente)
Proporción de variables utilizadas por nodo	0.65–1.00 (valores continuos generados aleatoriamente)
Reducción mínima de pérdida para dividir un nodo	0.0001–8.0 (valores continuos generados aleatoriamente)
Regularización L1	10^{-10}–1.0 (valores continuos generados aleatoriamente)
Regularización L2	1.0–200.0 (valores continuos generados aleatoriamente)
Política de crecimiento	depthwise, lossguide
Número máximo de hojas	16–128 (valores enteros generados aleatoriamente)
Número máximo de estimadores	20000 con detención temprana de 250 iteraciones

Tabla 5.8: Espacio de hiperparámetros evaluado para los modelos XGBoost.

5.3. Conceptos relevantes

Label Encoding

El label encoding es una técnica de preprocesamiento utilizada para transformar variables categóricas en representaciones numéricas, asignando a cada categoría un valor entero único. Este procedimiento resulta necesario en muchos algoritmos de aprendizaje automático que requieren entradas numéricas para su funcionamiento [31].

Formalmente, sea una variable categórica X que puede tomar valores en un conjunto finito de categorías $\mathcal{C} = \{c_1, c_2, \ldots, c_K\}$. El label encoding define una función de transformación:

$$\varphi : \mathcal{C} \to \{0, 1, \ldots, K-1\},$$

tal que a cada categoría c_k se le asigna un valor entero distinto.
De este modo, una observación $x \in \mathcal{C}$ se transforma en:

$$\varphi(x) = k - 1, \quad \text{si } x = c_k.$$

Si bien esta técnica es simple y eficiente, introduce un orden implícito entre las categorías que no necesariamente existe en los datos originales. Por esta razón, su uso es adecuado principalmente para variables objetivo o para variables categóricas cuando el modelo empleado no es sensible al orden numérico, como en el caso de árboles de decisión o métodos basados en ensambles.

AUC multiclase OVR ponderado

El AUC multiclase basado en la estrategia One-vs-Rest (OVR) corresponde a una extensión del área bajo la curva ROC para problemas de clasificación con más de dos clases. En este enfoque, el problema multiclase se descompone en K problemas binarios, donde cada clase k se considera como clase positiva frente al resto de las clases, que actúan como clase negativa [32].

Para cada clase k, se calcula el AUC binario, denotado como AUC_k, a partir de la curva ROC correspondiente. Este valor mide la capacidad del modelo para distinguir correctamente la clase k frente a las demás. Formalmente, puede interpretarse como la probabilidad de que el modelo asigne una puntuación mayor a una observación positiva que a una negativa:

$$\text{AUC}_k = \mathbb{P}\big(s_k(x^+) > s_k(x^-)\big),$$

donde $s_k(x)$ corresponde al puntaje o probabilidad estimada para la clase k, x^+ representa una observación perteneciente a la clase k y x^- una observación perteneciente a otra clase.

Una vez obtenidos los valores de AUC_k para cada clase, estos se combinan mediante un promedio ponderado, donde el peso de cada clase depende de su proporción en el conjunto de datos. Sea n_k el número de observaciones de la clase k y N el total de observaciones, el AUC multiclase OVR ponderado se define como:

$$\text{AUC}_{\text{OVR}}^{\text{weighted}} = \sum_{k=1}^{K} \frac{n_k}{N} \cdot \text{AUC}_k.$$

Este permite obtener una medida global del desempeño del modelo, considerando tanto su capacidad discriminativa como la distribución de las clases en el conjunto de datos, siendo particularmente adecuado en escenarios con desbalance de clases [33].

One-vs-Rest

La estrategia One-vs-Rest es un enfoque utilizado para extender clasificadores binarios a problemas de clasificación multiclase. La idea central consiste en descomponer un problema con K clases en K problemas binarios independientes. Para cada clase k, se construye un clasificador que distingue entre las observaciones pertenecientes a dicha clase (clase positiva) y aquellas que pertenecen a cualquiera de las otras clases (clase negativa).

Formalmente, para cada clase $k \in \{1, \ldots, K\}$, se define un clasificador $f_k(x)$ tal que:

$$f_k(x) = \begin{cases} 1, & \text{si } x \text{ pertenece a la clase } k, \\ 0, & \text{en caso contrario.} \end{cases}$$

Cada clasificador produce una puntuación o probabilidad $s_k(x)$, que indica el grado de pertenencia

de la observación x a la clase k. La predicción final del modelo se obtiene seleccionando la clase con mayor puntuación:

$$\hat{y}(x) = \arg \max_{k \in \{1,\ldots,K\}} s_k(x).$$

Permite reutilizar modelos diseñados para clasificación binaria en contextos multiclase, manteniendo una implementación simple y eficiente. Además, es especialmente útil en combinación con métricas como el AUC multiclase, donde cada clase puede evaluarse de forma independiente frente al resto [34].

Kernel RBF

El kernel RBF (Radial Basis Function), también conocido como kernel gaussiano, es una función ampliamente utilizada en máquinas de soporte vectorial cuando la separación entre clases no es lineal. Su principal ventaja es que permite modelar fronteras de decisión más flexibles que las obtenidas con un kernel lineal. Matemáticamente, se expresa como:

$$K(x_i, x_j) = \exp\left(-\gamma \|x_i - x_j\|^2\right),$$

donde x_i y x_j representan dos observaciones y γ controla el alcance de influencia de cada ejemplo de entrenamiento. Intuitivamente, valores bajos de γ implican una influencia más amplia, mientras que valores altos generan una influencia más local. En conjunto con el parámetro C, que regula la penalización de errores de clasificación, el kernel RBF permite capturar patrones no lineales complejos en los datos [35].

SMOTE

SMOTE (Synthetic Minority Over-sampling Technique) es una técnica de sobre-muestreo sintético utilizada para mitigar el desbalance entre clases en problemas de clasificación. A diferencia del sobre-muestreo aleatorio, que replica observaciones existentes, SMOTE genera nuevos ejemplos sintéticos a partir de interpolaciones entre instancias de la clase minoritaria y sus vecinos más cercanos. De este modo, se busca entregar una representación más amplia de la clase menos frecuente y favorecer una frontera de decisión más adecuada durante el entrenamiento del modelo [36, 37].

EARLY STOPPING

El early stopping es una estrategia de regularización utilizada para prevenir el sobreajuste durante el entrenamiento de modelos predictivos. Su idea central consiste en monitorear el desempeño del modelo sobre un conjunto de validación y detener el proceso de entrenamiento cuando la métrica de interés deja de mejorar durante un número predefinido de iteraciones. De este modo, se busca conservar la versión del modelo con mejor capacidad de generalización y evitar que continúe ajustándose en exceso a los datos de entrenamiento [38].

En métodos iterativos, como el gradient boosting, esta técnica permite además seleccionar de manera automática un número adecuado de iteraciones, equilibrando desempeño predictivo y costo computacional. En particular, en XGBoost el parámetro `early_stopping_rounds` activa este mecanismo y exige que la métrica de validación mejore al menos una vez dentro de un número dado de rondas para continuar el entrenamiento [39].

OUT-OF-FOLD

Las predicciones out-of-fold corresponden a las estimaciones obtenidas para observaciones que no fueron utilizadas durante el entrenamiento del modelo en una iteración dada de validación cruzada. En este esquema, cada partición dejada fuera actúa como conjunto de prueba, mientras que el modelo se ajusta sobre los folds restantes. De este modo, cada observación es predicha por un modelo que no la ha visto previamente, lo que permite obtener una estimación más realista del desempeño predictivo. Este tipo de predicciones es especialmente útil en procedimientos donde el modelo final se entrena a partir de predicciones cruzadas generadas por los modelos base [40, 41].

5.4. Resultados de la implementación

En esta sección se presentan resultados complementarios que no fueron incorporados en el cuerpo principal. En particular, se incluyen matrices de confusión y métricas detalladas de las configuraciones no seleccionadas como modelo final, tanto para las características construidas con `esig` como con `iisignature`, bajo las representaciones firma y log-firma de los datos simulados y de los datos reales.

5.4.1. RStudio

Realiza el preprocesamiento de la base de datos, incorporando la media de magnitud al inicio y al final de cada curva de luz real, con el fin de homogeneizar su estructura temporal. Posteriormente, a partir de estas curvas corregidas, se genera curvas de luz sintéticas. Además, permite estimar el parámetro φ asociado a cada serie y generar su representación gráfica, para realizar un análisis comparativo entre las curvas observadas y las simuladas.

`Preprocesamiento de los datos`

5.4.2. Python

Por motivos de extensión, en este anexo no se incorpora el contenido completo del notebook, sino únicamente una descripción general de su estructura y un fragmento representativo del procedimiento implementado. El código completo utilizado en este trabajo se encuentra disponible en los siguientes enlaces:

Path signature

Limpiar la base, filtrar solo la banda 1, cargar las curvas simuladas, calcular la path signature con profundidad 9 para `esig` e `iisignature` y concatenar las características del vector numérico con las clases de los objetos.

`Path Signature`

Modelos de clasificación

Incluye la implementación de los modelos Random Forest, Support Vector Machine y eXtreme Gradient Boosting para curvas simuladas y curvas reales, considerando las representaciones signature y log-signature, tanto con `esig` como con `iisignature`. En particular, los archivos se organizan en las siguientes secciones: curvas simuladas con ESIG, curvas simuladas con IISIGNATURE, curvas reales con ESIG, curvas reales con IISIGNATURE, además de bloques complementarios para gráficos de importancia de variables y análisis de profundidad.

`Random Forest` `Support Vector Machine` `eXtreme Gradient Boosting`

Agradecimientos

Quiero comenzar agradeciendo a toda mi familia, que ha sido mi apoyo más constante y mi refugio en cada etapa de este camino. Gracias por estar siempre a mi lado, por recordarme una y otra vez que soy capaz de lograr lo que me proponga, incluso cuando yo misma lo olvido. Gracias por acompañarme durante toda mi vida y por tenerme paciencia en mis días más difíciles, especialmente cuando el estrés, el cansancio o la frustración parecen ganarme. Destaco enormemente a mis padres y a mi hermana, por cada risa compartida, por cada enojo, por cada abrazo, por cada sueño cumplido y por todos los que aún nos quedan por vivir. Gracias por ser mi lugar seguro, mi fuerza y el mejor hogar que podría haber tenido jamás. Lulú, Kleer, Albito, Hachi y los que nos cuidan desde el cielo, los amo.

Mami, gracias por confiar siempre en mí, por corregirme con amor y por no abandonarme en ningún momento. Te amo profundamente. Todo lo que soy es gracias a ti, y sé que en esta vida no me alcanzará el tiempo para retribuirte todo lo que me has dado, pero me esforzaré siempre intentándolo.

Gordi, como no somos tan de piel, guardo en mi corazón con un amor inmenso cada palabra de aliento que me has dado. Tal vez por eso cada abrazo tuyo, cada gesto de amor y cada muestra de cariño tienen para mí un valor tan grande. Gracias por quererme, por amarme y por estar en mi vida. Gracias por ser el mejor papá que pude tener.

Hermani, mi persona favorita. Fuiste mi primer amor y la razón por la que mi corazón se hizo más grande desde que llegaste a llenar de alegría la casa. Gracias por tus consejos, por tantas risas cómplices y por la luz que llevas contigo. No dejes nunca de ser ese solcito que ilumina mi vida.

También quiero agradecer al amor de mi eternidad, Javier, por hacerme tan feliz desde que llegaste a mi vida. Eres una persona tan linda, cielo, y agradezco profundamente todo el amor con el que me llenas cada día. Gracias por hacerme sentir la mujer más afortunada del mundo. Sigamos siempre compartiendo nuestros logros caminando juntos de la manito.

Gracias a mis amigas y amigos, por ser un pilar fundamental en mi desarrollo personal y por acompañarme en distintas etapas de mi vida. Niss y Tami, gracias por tantos años de amistad, por estar siempre ahí y por haberse convertido en mis hermanas de distinta madre. Geral, mi floja favorita, gracias por hacerme tan feliz y por regalarme tantos momentos lindos siempre que estamos juntitas. Grupo Playa, gracias por llegar a mi vida de una forma tan inesperada; jamás pensé encariñarme tanto, pero terminaron siendo mis confidentes más preciados durante la carrera; Juan Verdadero, gracias por obligarme a no rendirme, fuiste una de las primeras personas que creyó en mí y te estaré agradecida toda mi vida

por ello. Camisan, gracias por ser mi mejor amiga, por confiarme tantos secretos, por cuidar mis penas como si fueran tuyas, por hacerme reír y por escucharme siempre. Tu amistad ha sido un regalo muy precioso en mi vida. Pau, Dani, Cony, Néstor, Jarita, Xami, Sofi, Javi, Denisse Quilaleo, Nathy y podría seguir nombrando a muchas personas más... gracias por acompañarme a lo largo de la carrera. Seamos o no cercanos en la actualidad, valoro profundamente el cariño que me entregaron en su momento y también el que algunos de ustedes me siguen entregando hasta hoy. Qué linda se vuelve la vida cuando una tiene la suerte de compartirla con amigos.

Para terminar, quiero agradecer a mi comisión por acompañarme en este arduo proceso, por orientarme cuando lo necesité y por el tiempo y dedicación que me entregaron.

¡Gracias a todos por acompañarme en este camino y ser parte de este proceso que hoy culmina en mi formación como Ingeniera Estadística! ♥

Bibliografía

[1] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proceedings of the 13th International Conference on Machine Learning*, 1996.

[2] L. Eyer and N. Mowlavi, "Updated version of the variability tree presented in eyer & mowlavi (2008), separated by pulsators and eruptive variables," 2018, figura en ResearchGate. [Online]. Available: https://www.researchgate.net/figure/Updated-version-of-the-variability-tree-presented-in-Eyer-Mowlavi-2008-separated_fig1_329541119

[3] ——, "Variable stars across the observational hr diagram," *Journal of Physics: Conference Series*, vol. 118, 2008. [Online]. Available: http://stacks.iop.org/1742-6596/118/i=1/a=012010

[4] F. Förster, "Alerce: Towards real-time alert classification in the lsst era," octubre 2020, presentación, ZTF Celebration, Caltech. [Online]. Available: https://sites.astro.caltech.edu/ztf/csac/Presentations/ALeRCE%20ZTF%20Celebration%2020201023.pdf

[5] M. Catelan and H. A. Smith, *Pulsating stars*. Wiley-VCH, 2015.

[6] P. Moskalik, "Multi-mode oscillations in classical cepheids and rr lyrae-type stars," in *Precision asteroseismology*, J. A. Guzik, W. J. Chaplin, G. Handler, and A. Pigulski, Eds. Cambridge University Press, 2014, vol. 301.

[7] ESA/Hubble, "Active galactic nucleus," n.d. [Online]. Available: https://esahubble.org/wordbank/active-galactic-nucleus/

[8] NASA, "Quasars," n.d. [Online]. Available: https://science.nasa.gov/universe/galaxies/quasars/

[9] NASA Fermi Science Support Center, "Blazars," n.d. [Online]. Available: https://fermi.gsfc.nasa.gov/science/eteu/blazars/

[10] Sociedad Española de Astronomía, "Supernova [glosario]," 2025, recuperado el 8 de julio de 2025. [Online]. Available: https://www.sea-astronomia.es/glosario/supernova

[11] A. Udalski, "Ogle cepheids and rr lyrae stars in the milky way," *EPJ Web of Conferences*, vol. 152, 2017.

[12] European Space Agency, *The HIPPARCOS and TYCHO catalogues: Astrometric and photometric star catalogues derived from the ESA HIPPARCOS Space Astrometry Mission*, 1997, eSA Special Publication No. 1200.

[13] M. Catelan, I. Dékány, M. Hempel, and D. Minniti, "Stellar variability in the vvv survey: An update," Jun. 2014.

[14] J. Debosscher, L. M. Sarro, C. Aerts, J. Cuypers, B. Vandenbussche, R. Garrido, and E. Solano, "Automated supervised classification of variable stars: I. methodology," *Astronomy & Astrophysics*, vol. 475, 2008.

[15] J. W. Richards, D. L. Starr, N. R. Butler, J. S. Bloom, J. M. Brewer, A. Crellin-Quick, J. Higgins, R. Kennedy, and M. Rischard, "On machine-learned classification of variable stars with sparse and noisy time-series data," *The Astrophysical Journal*, 2011, preprint: arXiv:1101.1959.

[16] J. J. W. Richards, D. L. Starr, H. Brink, A. A. Miller, J. S. Bloom, N. R. Butler, J. B. James, J. P. Long, and J. Rice, "Active learning to overcome sample selection bias: Application to photometric variable star classification," *The Astrophysical Journal*, vol. 744, 2012.

[17] F. Elorrieta, S. Eyheramendy, A. Jordán, I. Dékány, M. Catelan, R. Angeloni, J. Alonso-García, R. Contreras-Ramos, F. Gran, G. Hajdu, N. Espinoza, R. K. Saito, and D. Minniti, "A machine learned classifier for rr lyrae in the vvv survey," *Astronomy & Astrophysics*, vol. 592, 2016.

[18] I. Nun, P. Protopapas, B. Sim, and W. Chen, "Ensemble learning method for outlier detection and its application to astronomical light curves," *The Astronomical Journal*, vol. 152, 2016.

[19] P. Sánchez-Sáez, I. Reyes, C. Valenzuela, F. Förster, S. Eyheramendy, F. Elorrieta, F. E. Bauer, G. Cabrera-Vives, P. A. Estévez, M. Catelan, G. Pignata, P. Huijse, D. De Cicco, P. Arévalo, R. Carrasco-Davis, J. Abril, R. Kurtev, J. Borissova, J. Arredondo, E. Castillo-Navarrete, D. Rodriguez, D. Ruz-Mieres, A. Moya, L. Sabatini-Gacitúa, C. Sepúlveda-Cobo, and E. Camacho-Iñiguez, "Alert classification for the alerce broker system: The light curve classifier," *The Astronomical Journal*, vol. 161, 2021.

[20] S. Eyheramendy, F. Elorrieta, and W. Palma, "An irregular discrete time series model to identify residuals with autocorrelation in astronomical light curves," *Monthly Notices of the Royal Astronomical Society*, 2018.

[21] C. Ojeda, W. Palma, S. Eyheramendy, and F. Elorrieta, "Extending time-series models for irregular observational gaps with a moving average structure for astronomical sequences," *RAS Techniques and Instruments*, 2023.

[22] I. Chevyrev and A. Kormilitzin, "A primer on the signature method in machine learning," *Foundations of Machine Learning*, 2016.

[23] P. K. Friz and N. B. Victoir, *Multidimensional Stochastic Processes as Rough Paths: Theory and Applications*. Cambridge University Press, 2010.

[24] T. J. Lyons, *Differential Equations Driven by Rough Paths*. Springer, 2007.

[25] L. Breiman, "Random forests," *Machine Learning*, vol. 45, 2001.

[26] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[27] B. Schölkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002.

[28] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016, pp. 785–794. [Online]. Available: https://doi.org/10.1145/2939672.2939785

[29] DataSig (datasig-ac-uk). (2025) esig: signature-based python package. [Online]. Available: https://github.com/datasig-ac-uk/esig

[30] J. F. Reizenstein and B. Graham, "Algorithm 1004: The iisignature library: Efficient calculation of iterated-integral signatures and log signatures," *ACM Transactions on Mathematical Software*, vol. 46, no. 2, 2020.

[31] scikit-learn developers. (2026) Labelencoder — scikit-learn documentation. Accessed: 2026-04-05. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.LabelEncoder.html

[32] ——. (2026) roc_auc_score — scikit-learn documentation. Accessed: 2026-04-05. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_auc_score.html

[33] ——. (2026) Multiclass receiver operating characteristic (roc). Accessed: 2026-04-05. [Online]. Available: https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html

[34] ——. (2026) Onevsrestclassifier — scikit-learn documentation. Accessed: 2026-04-05. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.multiclass.OneVsRestClassifier.html

[35] ——. (2026) Rbf svm parameters — scikit-learn example. Accessed: 2026-04-04. [Online]. Available: https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html

[36] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, pp. 321–357, 2002.

[37] imbalanced-learn developers. (2025) Smote. Accessed: 2026-04-04. [Online]. Available: https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html

[38] scikit-learn developers. (2026) Early stopping in gradient boosting. Accessed: 2026-04-04. [Online]. Available: https://scikit-learn.org/stable/auto_examples/ensemble/plot_gradient_boosting_early_stopping.html

[39] XGBoost developers. (2020) Python api reference: early_stopping_rounds. Accessed: 2026-04-04. [Online]. Available: https://xgboost.readthedocs.io/en/release_1.2.0/python/python_api.html

[40] scikit-learn developers. (2026) cross_val_predict — scikit-learn documentation. Accessed: 2026-04-04. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_predict.html

[41] ——. (2026) Stackingclassifier — scikit-learn documentation. Accessed: 2026-04-04. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.StackingClassifier.html