Machine Learning para la detección de exoplanetas: revisión y nuevos enfoques

Trabajo de Fin de Máster

Máster en Ciencia de Datos

Enero 2023

Autor

Javier Gómez de Diego

Tutora

Laura Ruiz Dern

Trabajo de Fin de Máster

Machine Learning para la detección de exoplanetas: revisión y nuevos enfoques

Universitat Oberta de Catalunya

Enero 2023

Ad astra

Machine Learning para la detección de exoplanetas: revisión y nuevos enfoques

Javier Gómez de Diego

Astrophysics and Geoscience

Tutora de TFM

Laura Ruiz Dern

Profesor responsable de la asignatura

Jordi Casas Roma

22 de enero de 2023

Universitat Oberta de Catalunya

Índice

Lista de Figuras

Lista de Tablas

Lista de Ecuaciones

1. Introducción

La detección de exoplanetas es una tarea ardua que implica grandes cantidades de datos. Por lo tanto, la necesidad de crear procesos para automatizar su tratamiento y clasificación ha impulsado multitud de trabajos en torno a esta problemática. Este TFM parte como revisión y mejora del proyecto titulado *Machine Learning para el tratamiento de datos y la detección de exoplanetas mediante el método de tránsito* [1], realizado en 2020 como Trabajo de Fin de Grado.

Dicho proyecto trató la adecuación de los datos como principal problemática a resolver debido a varios factores:

- **Fuerte desbalanceamiento**. Los datos de entrenamiento constaban de 37 positivos (estrellas con exoplanetas confirmados) y 5.050 negativos (estrellas sin exoplanetas). Esto fue tratado con varias técnicas de *data augmentation* que elevaron el número de positivos hasta 2.849.
- **Alta dimensionalidad**. Cada uno de los 7.899 registros contaba con 3.197 dimensiones (mediciones de la cantidad de luz recibida en un momento dado). Esto haría que los modelos de clasificación no pudieran trabajar de forma eficaz y eficiente, por lo que se aplicó una reducción de dimensionalidad por *Principal Component Analysis* (PCA), resultando en un *dataset* de poco más de 600 dimensiones.

Tras todo el procesamiento de los datos, se construyó un modelo que logró clasificar 562 negativos y 2 positivos de forma correcta, por 3 falsos positivos y 3 falsos negativos.

En la Figura 1 se detalla el proceso seguido.

1.1. Contexto y justificación del Trabajo

Tras cursar el Máster en Ciencia de Datos, la motivación para continuar el proyecto se ha renovado tras haber profundizado más en el conocimiento en la materia. Es por ello por lo que en el presente TFM se pretende construir un procedimiento mejorado y más exhaustivo que el desarrollado en dicho TFG [1].

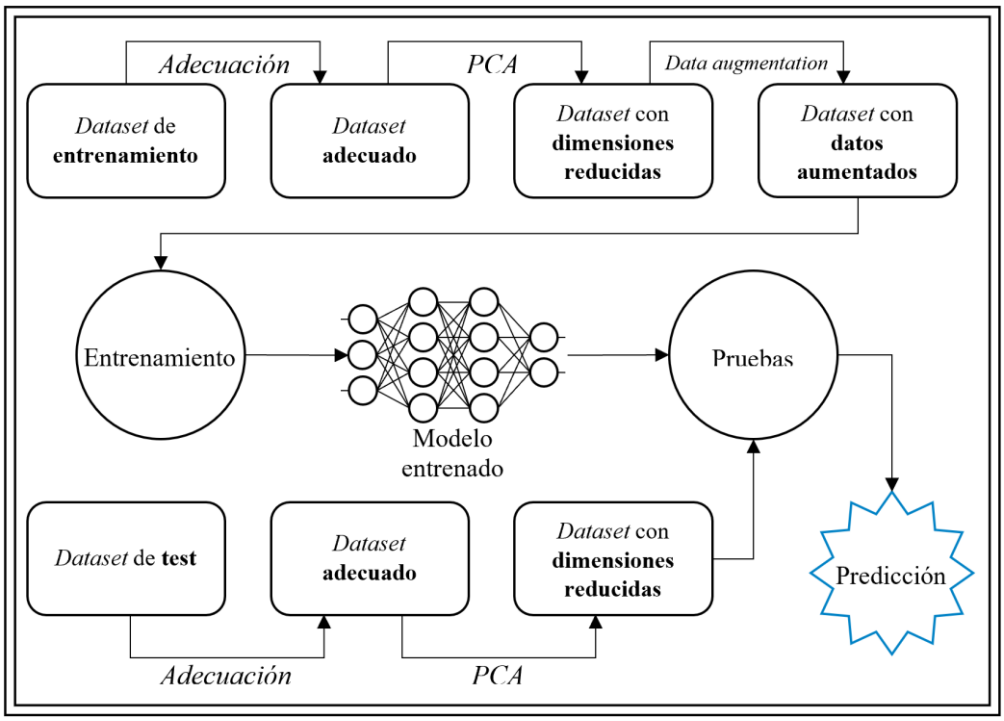

***Figura 1**. Diagrama de todo el proceso seguido en el TFG [1].*

1.2. Objetivos del Trabajo

General

El objetivo primordial del proyecto consiste en mejorar los resultados obtenidos en la clasificación de los datos en el TFG [1]. Para ello, se definen los siguientes objetivos específicos.

Específicos

- Revisión e identificación de puntos de mejora en el proceso de tratamiento de datos seguido en el TFG [1].
- Definición y desarrollo de nuevas técnicas para mejorar los puntos identificados anteriormente.
- Diseño e implementación de varias estrategias para la clasificación de los datos.
- Obtención y comparación de los resultados.

1.3. Estructura y metodología

Estructura

El trabajo seguirá el siguiente esquema:

1. **Revisión** de la metodología llevada a cabo en el TFG [1] e identificación de puntos de mejora.
2. Implementación de las **mejoras** anteriores.
3. Definición de las **estrategias** que se ejecutarán para la clasificación de los datos.
    A. **Distribuciones**. Cada línea del conjunto de datos es un vector cuyos valores representan el valor de la luminosidad de una estrella en un momento del tiempo. Esta estrategia consiste en generar un nuevo *dataset* en el que cada línea represente varias medidas estadísticas de la distribución de las mediciones de luz de las estrellas (es decir, cada fila es una distribución de los datos de las mediciones de luz de cada estrella, y cada columna es una de sus medidas: media, mediana, desviación…-).
    B. **Selección de dimensionalidad**. Establecer un mecanismo mediante el cual se identifican las curvas de luz resultantes de posibles tránsitos y sus posiciones. A partir de este punto, generar un nuevo dataset cuyas dimensiones representen los resultados obtenidos acerca de los posibles tránsitos y sus posiciones (para no perder la información de la frecuencia). Por último, definir y entrenar un modelo de clasificación con dicho *dataset*.
    C. **Mecanismo de atención**. Obtener un *vector atención* que pondere cada una de las dimensiones en base a la probabilidad de ser un tránsito y repetir las estrategias **A** y **B**.
4. **Implementación** de las estrategias y obtención de resultados.
5. **Comparación** de los resultados de las diferentes estrategias entre sí.
6. **Comparación** de los resultados de con los obtenidos en el TFG [1].

Metodología

Estrategia A

La estrategia de **Distribuciones** se planteará como continuación/mejora del TFG [1]. Se partirá de los mismos datos sin que sufran modificación alguna (salvo las posibles mejoras llevadas a cabo en el punto **2**).

Estrategia B

La estrategia de **Selección de dimensionalidad** da un nuevo enfoque a la metodología del TFG [1]: se parte de los mismos datos, pero el modelo elegido se entrenará con un nuevo *dataset* generado.

Estrategia C

La estrategia de **Mecanismo de atención** se implementará para, presumiblemente, potenciar las estrategias anteriores con la ponderación de los datos. Además, dada la naturaleza de esta estrategia, se podrá evaluar de forma sencilla su eficacia comparando sus resultados con los obtenidos inicialmente.

1.4. Planificación del trabajo

La planificación que se seguirá se muestra en el siguiente diagrama de Gantt.

Actividad	Octubre			Noviembre				Diciembre					Enero				
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Revisión de documentación y estado del arte																	
Revisión TFG e identificación de mejoras																	
Implementación de mejoras																	
Implementación estrategia A																	
Implementación estrategia B: B.1																	
Implementación estrategia B: B.2																	
Implementación estrategia C																	
Comparación de resultados																	
Redacción de la memoria																	
Preparación de la defensa																	
Defensa pública																	

PEC 2	Tiempo de trabajo
PEC 3	Tiempo de trabajo condicional
PEC 4	
PEC 5	
PEC 6	

2. Estado del arte

La automatización del proceso de la detección de exoplanetas mediante el método de tránsito apoyándose en Machine Learning es un problema aún no resuelto y que está en continua evolución. Nuevas técnicas que resultan más eficaces y eficientes van surgiendo, y nos acercan hacia soluciones cada vez más optimizadas.

Las últimas publicaciones en este campo muestran avances desde diferentes aproximaciones a lo largo de todo el proceso de tratamiento y clasificación de los datos.

Como ya se ha mencionado anteriormente, uno de los principales problemas encontrados que ya se trató en la realización del TFG [1] es el desbalanceamiento de los datos. Y es que todos los datos utilizados en la detección de exoplanetas presentan desbalanceamientos similares, por lo que solucionar este problema de la forma más efectiva posible resulta crucial. El artículo de *Cuéllar, S. et al.* [2] genera datos sintéticos que, combinados con datos reales, son utilizados para entrenar un modelo de predicción.

Datos sintéticos

En *Cuéllar, S et al.* [2] se introducen en el dataset curvas de luz sintéticas para mejorar el rendimiento del modelo. Éstas se generan utilizando el modelo cuadrático para las leyes de oscurecimiento de los extremos planteado en *Mandel, Kaisey et al.* [3]. Para un tránsito sobre un disco estelar con oscurecimiento cuadrático de los extremos, el flujo f es:

$$f(k,z) = 1 - \frac{(1-c)\lambda_e(k,z) + c\lambda_d(k,z) + u_2\eta_d(k,z)}{1 - u_1/3 - u_2/6}$$

Ecuación 1. *Modelo cuadrático para las leyes de oscurecimiento de los extremos [3].*

donde *k* es la relación del radio, z es la distancia proyectada, $c = u_1 + 2u_2$, u_1 y u_2 son los coeficientes cuadráticos de oscurecimiento de los extremos, y λ_e, λ_d y η_d son funciones de k y z definidas en [3]. Los parámetros para implementar el modelo a través de la librería *PyTransit* [4] son el período (T), la relación entre el radio del planeta y el radio de la estrella (r_p/r_s), la relación entre el semieje mayor orbital y el radio de la estrella (a/r_s) y la inclinación de la órbita (i).

Más adelante revisaremos otros métodos de *data augmentation* como SMOTE, empleado en *Singh, Amritanshu Kumar et al.* [5].

Con un dataset balanceado y debidamente procesado, el siguiente paso es el de generar un modelo para ser entrenado. La variedad de aproximaciones en este punto es muy elevada, lo que da lugar a multitud de trabajos con diversos métodos.

Detección de objetos bidimensionales

El trabajo de *Cui, Kaiming et al.* [6] explora una red neuronal de detección de objetos bidimensionales para identificar directamente las señales de los tránsitos, lo que se aproxima a la intuición visual humana. De esta forma, este trabajo se distancia de otros esquemas que emplean CNN unidimensionales y RNN.

En *Cui, Kaiming et al.* [6] se grafican las curvas de luz como una imagen bidimensional. Esto permite utilizar, directamente sobre dichas imágenes, algoritmos de *computer vision* ya desarrollados, lo que supone dos grandes ventajas:

- La frecuencia de muestreo para las curvas de luz es relativamente más permisiva ya que, en el caso de CNN unidimensionales y RNN, las dimensiones de entrada deben estar ya determinadas. Esto da más flexibilidad a la hora de utilizar este modelo con otros datasets.
- La salida del modelo combina tres escalas diferentes: grande, pequeña y mediana. De esta forma, se puede adaptar el output a las diferentes densidades de información generadas por distintos datasets sin modificar la estructura de la red.

En la figura 1 se muestra la arquitectura de la red escogida en *Cui, Kaiming et al.* [6]. Se trata de una red ampliamente conocida que se introdujo por primera vez en -*You Only Look Once (YOLOv3)- Redmon, Joseph et al.* [7], en el campo de *computer vision*. A ésta se le ha modificado ligeramente el canal de entrada y las dimensiones de salida.

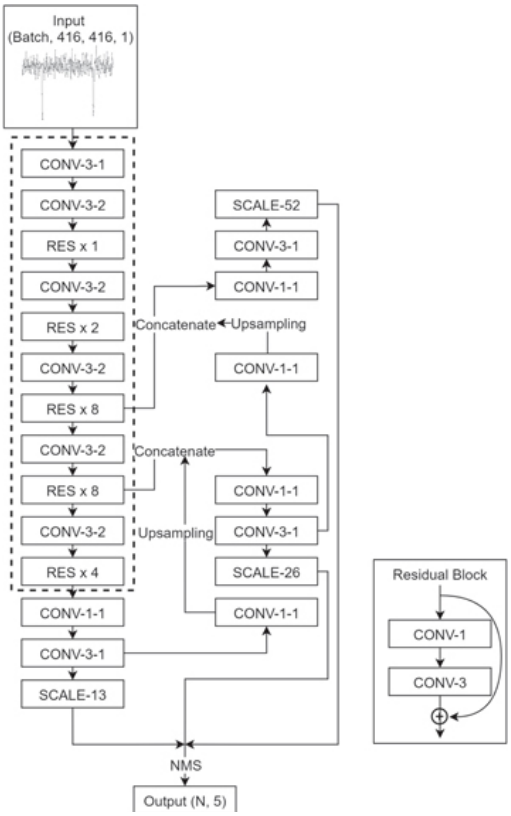

Figura 2. *Arquitectura de la red [6].*

La salida final combina los resultados de las tres escalas (grande, mediana y pequeña) mediante el algoritmo NSM (*Non Maximum Suppression*) (*Rosenfeld, A. et al.* [8]), que preserva de forma iterativa los mejores cuadros delimitadores en base a la confianza, eliminando el resto en base a un umbral.

También se modifica la función de pérdida original por la de la ecuación 2 para adaptarla a los delimitadores y los rangos de confianza específicos.

$$
\begin{aligned}
L = {} & L_{coord} + L_{obj} + L_{noobj}, \\
& = \lambda_{coord} \frac{1}{N} \sum_{i=0}^{S^2} 1_i^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\
& + \lambda_{coord} \frac{1}{N} \sum_{i=0}^{S^2} 1_i^{obj} \left[(w_i - \hat{w}_i)^2 + (h_i - \hat{h}_i)^2\right] \\
& - \frac{1}{N} \sum_{i=0}^{S^2} 1_i^{obj} \left[C_{\frac{S}{N}} \log(C_i) \right. \\
& \left. + \left(1 - C_{\frac{S}{N}}\right) \log(1 - C_i) \right] \\
& - \lambda_{noobj} \frac{1}{N'} \sum_{i=0}^{S^2} 1_i^{noobj} \log(1 - C_i).
\end{aligned}
$$

Ecuación 2. *Función de pérdida [6].*

La suma de los dos primeros términos es la pérdida del error cuadrático medio de las coordenadas. x_i , y_i , w_i , h_i son las coordenadas predichas de la celda i. S es el tamaño de la escala. 1_i^{obj} toma el valor 1 cuando hay un objeto en i, y 0 en caso contrario. 1_i^{noobj} toma el valor 1 cuando no hay un objeto en i, y 0 en caso contrario. N es el número de celdas para calcular el valor medio. El tercer y cuarto término muestran la *binary cross-entropy loss*. C_i es el valor de la confianza predicha para si habrá o no un objeto en i. λ_{coord} y λ_{noobj} son las constantes para incrementar la atención del modelo en ciertas partes. En este caso, se emplea λ_{coord} = λ_{noobj} = 10.

Una gran variedad de modelos, especialmente CNN, se han utilizado ampliamente para la detección de exoplanetas mediante el método de tránsito, produciendo distintos rendimientos y resultados. En el artículo *Singh, Amritanshu Kumar et al.* [5] se realiza una comparativa entre diferentes aproximaciones en cuanto a la elección de modelos de clasificación.

CNN vs SVM

Emplear distintos modelos con un mismo dataset permite comparar fielmente el rendimiento de los mismos. Sin embargo, y como hemos visto anteriormente, los datos de esta naturaleza suelen sufrir un fuerte desbalanceamiento que es necesario corregir. Es por ello por lo que, antes de entrenar los modelos, el artículo *Singh, Amritanshu Kumar et al.* [5] emplea la técnica SMOTE (*Synthetic Minority Oversampling Technique*) para contrarrestar la infrarrepresentación de las estrellas con tránsitos confirmados.

Esta técnica produce datos sintéticos de la clase minoritaria -*sin exoplaneta* en este caso- a partir de los vecinos más cercanos siguiendo la ecuación 3,

$$x_{new} = x_1 + \alpha(x_1 - x_2)$$

Ecuación 3. *Técnica SMOTE para la generación de datos sintéticos.*

donde x_{new} es la nueva muestra generada, x_1 es una muestra de la clase minoritaria, x_2 es su vecino más cercano y α es un número aleatorio entre 0 y 1.

Una vez balanceado el dataset, el trabajo de *Singh, Amritanshu Kumar et al.* [5] emplea dos modelos supervisados: CNN y SVM. Para los datos de test, se obtienen los resultados de la tabla 1 con CNN y los de la tabla 2 con SVM.

	precision	***recall***	***f1-score***	***support***
0	0.99	0.87	0.93	565
1	0.03	0.40	0.05	5

Tabla 1. *Resultados de test con CNN [5].*

Estos resultados indican que CNN consigue un mejor rendimiento que SVM. Esto se debe a que CNN se desenvuelve mejor que SVM con datasets que, originalmente, sufren un gran desbalanceamiento.

	precision	***recall***	***f1-score***	***support***
0	0.99	0.67	0.80	565
1	0.01	0.40	0.02	5

Tabla 2. *Resultados de test con SVM [5].*

Eficiencia computacional: series temporales

Otra aproximación interesante es la que se lleva a cabo en el artículo de *Malik, Abhishek et al.* [9]. Este trabajo emplea modelos de Machine Learning más clásicos que los usados en uno de los artículos considerados más punteros (*Shallue, Christopher J. et al.* [10]) al utilizar modelos de *Deep Learning*.

Sin embargo, esto conlleva un trabajo previo de extracción de características para utilizarlas junto al input del modelo. Para ello, en *Malik, Abhishek et al.* [9] se utiliza la extracción de características de series temporales basada en tests de hipótesis escalables de la librería de Python *tsfresh* [11] debido a que las curvas de luz son, en esencia, series temporales. Tras esto, se identifican en total 789 características que se preprocesan (eliminar características irrelevantes, imputar valores vacíos y escalar) y, por último, se introducen como input del modelo.

El modelo se trata de un GBT (*Gradient Boosted Tree*) que clasifica los datos en *candidato* y *no candidato* mediante el *framework lightGBM* [12]. Para establecer los hiperparámetros se emplea AUC (*Area Under the Curve* -ROC-) debido a que ofrece una integral sobre todos los posibles umbrales de decisión, por lo que es el método que menos desvirtuado se ve por el desbalanceamiento de los datos de test. En cuanto al rendimiento, la métrica que mejor se adapta es el *recall*, pero también se tiene en cuenta -con menor peso- la precisión. En la figura 2 se puede observar el proceso del trabajo.

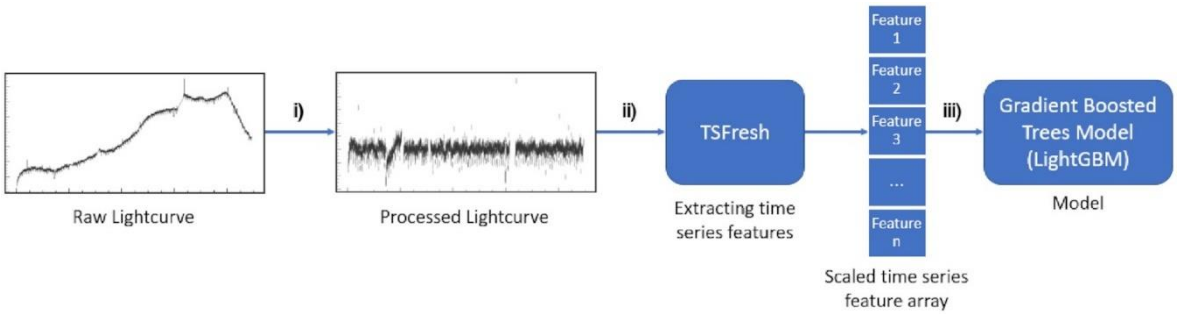

Figura 3. *Proceso de la metodología seguida [9].*

En la tabla 3 se muestran los resultados obtenidos sobre los datos de la misión TESS y se comparan con los obtenidos en *Yu, Liang et al.* [13].

Type	AUC	Recall	Precision
Yu, Liang et al. [13]	0.98	0.89	0.45
Malik, Abhishek et al. [9]	0.81	0.82	0.63

Tabla 3. *Comparación de resultados [9].*

De 49 muestras con tránsitos, el trabajo *Yu, Liang et al.* [13] fue capaz de encontrar 44, mientras que en *Malik, Abhishek et al.* [9] se identificaron 40. Sin embargo, los falsos positivos de este último fueron la mitad que en el primero, lo que mejora la precisión total del modelo.

Con otros datasets, como los provenientes de una de las campañas de Kepler o con datos sintéticos, el rendimiento del método empleado en *Malik, Abhishek et al.* [9] mejora, tal y como se observa en la figura 3.

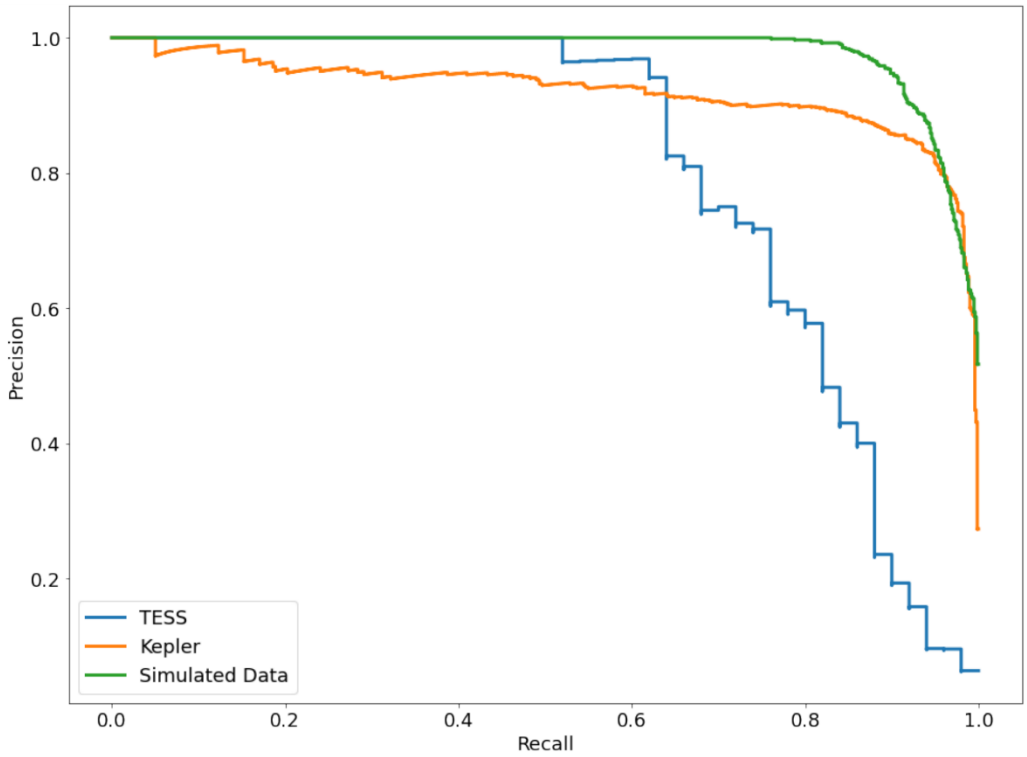

Figura 4. *Rendimiento -Precisión vs Recall- con diferentes datasets [9].*

Con resultados que no distan mucho de los obtenidos por técnicas más complejas de Deep Learning, *Malik, Abhishek et al.* [9] presenta las siguientes ventajas con respecto a éstas:

- Funciona con tan solo una visión global de la curva de luz, sin necesitar vistas plegadas o secundarias de las curvas de luz.
- El tiempo de entrenamiento es significativamente menor, requiriendo tan solo 5 minutos en un sistema con 2 CPUs, mientras que en modelos de Deep Learning esto requiere de unas 5 horas -además de la optimización de los hiperparámetros-.
- Esta estructura se adapta fácilmente a nuevos datos provenientes de distintas fuentes, ya que no requiere una re-optimización de los hiperparámetros.
- Las características de los datos pueden ser identificadas automáticamente, lo que facilita la comprensión de los datos y los procesos físicos subyacentes. Por otro lado,

los modelos de Deep Learning suelen ser cajas negras, lo que dificulta la interpretación de los resultados.
- El entrenamiento no requiere de hardware dedicado -GPUs-.

Por otro lado, las desventajas de la metodología desarrollada en *Malik, Abhishek et al.* [9] con respecto a las más complejas son:

- Los resultados no logran ser tan excepcionales como los obtenidos por modelos de Deep Learning debidamente entrenados.
- Se requiere como input unas características de los datos que han debido ser extraídas previamente, mientras que los modelos de Deep Learning pueden utilizar directamente las vistas globales y plegadas como input.

Métodos supervisados y no supervisados

Hasta ahora, los trabajos revisados empleaban modelos de aprendizaje supervisado para realizar la tarea de clasificación de los datos. El artículo *Jin, Yucheng et al.* [14] realiza un trabajo de comparación entre modelos supervisados -árboles de decisión, random forest, naïve Bayes y redes neuronales- y el uso posterior de modelos no supervisados -k-means- para encontrar planetas potencialmente habitables.

Lo que destaca de este trabajo es que utiliza, además de un dataset de las curvas de luz, un dataset con diferentes variables de exoplanetas confirmados -radio, período orbital, masa, densidad, temperatura, etc.-, por lo que éste va más allá de la mera detección de exoplanetas: también trata de clasificar los exoplanetas confirmados en función de su potencial como planetas habitables.

Primero, se lleva a cabo un preprocesamiento de los datos -eliminación de columnas mayoritariamente vacías e imputación de valores vacíos en columnas con menor proporción de estos valores-, un análisis exploratorio -correlación entre variables para reducir colinealidad y redundancia, análisis univariado y bivariado- y una selección de variables en base a las correlaciones.

Tras esto, se lleva a cabo la implementación de cada modelo.

Árbol de decisión

Se emplea *DecisionTreeClassifier* de la librería *Scikit-learn* con un número mínimo de muestras requeridas para dividir un nodo de 2 a 100 y una profundidad máxima de 1 a 20, eligiendo el valor exacto en función de la maximización del *accuracy*. Esto resulta en un número mínimo de muestras de 53 y una profundidad máxima de 7.

El *accuracy* obtenido con este modelo es de 99.06% sobre los datos de test.

En la tabla 4 se pueden observar las variables más relevantes halladas por el modelo.

Variable	Importance	Meaning	Description
koi_fpflag_ss	0.30512	Stellar Eclipse	1 means some phenomena caused by an eclipsing binary observed
koi_fpflag_co	0.27971	Centroid Offset	1 means the source of the signal is from a nearby star
koi_fpflag_nt	0.27158	Non-Transit-Like	1 means the light curve is not consistent with that of a transiting planet
koi_period	0.05115	Orbital Period	Measured in days and is taken in log scale
koi_count	0.03464	Number of Planets	Number of exoplanet candidates identified in a solar system
koi_fpflag_ec	0.03245	Ephemeris Match Indicates Contamination	1 means the candidate shares the same period and epoch as another object
koi_time0	0.01978	Transit Epoch	Measured in Barycentric Julian Day (BJD)

Tabla 4. *Variables más relevantes establecidas por el árbol de decisión [14].*

Random forest

Se emplea *RandomFprestRegressor* de la librería *Scikit-learn*. Siguiendo la misma estrategia que con el árbol de decisión, se establece un rango de 20 a 1000 árboles con un *step* de 20, logrando una mayor *accuracy* con 40 árboles.

El *accuracy* obtenido con este modelo es de 92.11% sobre los datos de test.

Naïve Bayes

Se emplea ComplementNB de la librería *Scikit-learn*. Antes de realizar el entrenamiento del modelo, se estandarizan las variables de entrada y se codifica la clase como un vector *one-hot*, estableciendo cada uno como la proporción de cada clase en el espacio muestral.

El *accuracy* obtenido con este modelo es de 88.50% sobre los datos de test.

Perceptrón multicapa

Se emplea *MLP Regressor* de la librería *Scikit-learn*. Tras el proceso de optimización, se establece *tanh* como función de activación, *Adam* como optimizador, un tamaño de la capa de 25 y un *learning rate* de 0.003.

El *accuracy* obtenido con este modelo es de 99.79% sobre los datos de test.

K-means

Se emplea *KMeans* de la librería *Scikit-learn*. El objetivo es encontrar qué exoplanetas están en el mismo grupo que la Tierra como planetas potencialmente habitables, que son los que se muestran en la figura 5.

Figura 5. *Cluster de los planetas potencialmente habitables; resultado de K-means [14].*

ThetaRay: detección de anomalías

La potencia del Machine Learning está haciendo que su uso esté cada vez más extendido en diversos campos, automatizando tareas de todo tipo; esto hace que surjan modelos muy avanzados en ciertos campos que, al ser empleados para realizar otras tareas para las que originalmente no fueron desarrollados, produzcan rendimientos excelentes con tan solo un ligero reentrenamiento. Este es el caso llevado a cabo en el artículo *Ofman, Leon et al.* [15].

Este trabajo emplea un set de métodos tanto semisupervisados como no supervisados llamado *ThetaRay* (desarrollado por *ThetaRay, Inc.* [16]), cuya aplicación original es la detección de anomalías en instituciones financieras para detectar crímenes financieros y brechas de seguridad cibernéticas y de IoT. Dado que el tránsito de un planeta por delante de su estrella es esencialmente una anomalía, este set de modelos podría permitir su detección. En este caso, el objetivo es automatizar el proceso de confirmación de candidatos a exoplanetas.

Los modelos que emplea *ThetaRay* son análisis harmónicos, geometría difusa y procesamiento estocástico, descomposición de matrices de bajo rango, álgebra lineal aleatoria, teoría de medidas geométricas, aprendizaje múltiple, redes neuronales de aprendizaje profundo y representación compacta de diccionarios.

Para adaptar el modelo a la detección de tránsitos, se emplean datos etiquetados de Kepler para el entrenamiento y la validación; y, posteriormente, se aplica a los datos de TESS, logrando mapear 3 nuevos tránsitos de entre casi 11.000 casos.

Los modelos de *ThetaRay* utilizados para lograr estos resultados son:

- **Algoritmo NY**. Basado en la metodología de mapas difusos para la reducción de dimensiones no lineal. Éste crea un espacio de dimensiones reducidas que describe geométricamente los datos clasificados como *normales*. Cuando se introduce un nuevo dato, se le aplica la misma transformación geométrica y, si éste cae dentro del espacio dimensional reducido descrito, se cataloga como *normal*; en caso contrario, se clasifica como *anomalía*. En la figura 6 se muestra el diagrama de flujo del algoritmo.

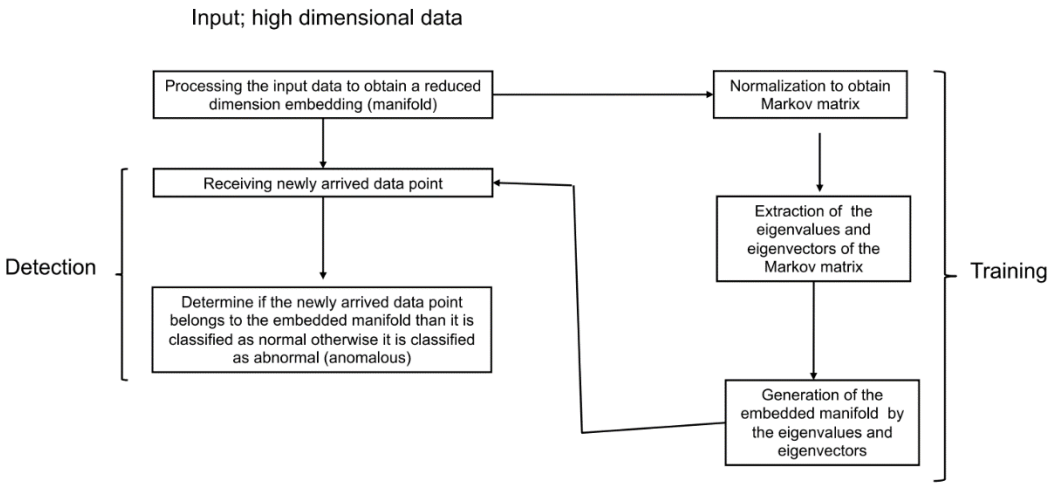

Figura 6. *Diagrama de flujo del algoritmo NY empleado en ThetaRay [15].*

- **Algoritmo LU**. Está basado en la descomposición matricial de bajo rango *Shabat, Gil et al.* [17]. A grandes rasgos, construye un diccionario a partir de los datos de entrenamiento; cuando se introduce un nuevo dato que no se puede definir bien en éste, se cataloga como *anomalía*.
- **Algoritmo DK**. Emplea iteraciones sucesivas de los algoritmos NY y LU, construyendo primero un diccionario y, posteriormente, reduciendo geométricamente su espacio dimensional.
- **Algoritmo AE**. Basado en un autoencoder variacional que, una vez entrenado, es capaz de reducir y reconstruir la representación de datos original. Cuando se introduce un nuevo dato que resulta ser una anomalía, la representación obtenida del autoencoder no encajará con la introducida como input. Para ello se establece un criterio para medir la similitud entre el input y el output.

Los resultados de aplicar un modelo de forma interdisciplinar, *ThetaRay* en este caso, muestran la eficiencia de estos modelos para la automatización del proceso de búsqueda de candidatos a exoplanetas.

2.1. Conclusión

Aunque el objetivo del presente trabajo sea el mismo que en la mayoría de los anteriormente citados, la metodología que se utilizará difiere de todos ellos. La idea de entrenar un modelo, no con los datos de las distintas mediciones sino con cómo éstos se distribuyen en cada registro, proporciona un nuevo enfoque para abordar la búsqueda de una solución a la problemática planteada.

3. Desarrollo e implementación

Datos

Los datos a tratar fueron obtenidos en la Campaña 3 de la misión *K2* de la NASA en 2016. Estos datos se publican abiertamente por la agencia espacial norteamericana en formato *.fits* a través del portal *Mikulski Archive* del *Space Telescope Science Institute* [18], y han recibido un primer tratamiento para un mejor estudio.

Primeramente, la NASA lleva a cabo un proceso para eliminar el ruido en los datos producidos por el propio telescopio. A continuación, por cada archivo correspondiente a cada estrella, se ha traspasado cada una de las distintas mediciones de luz a una columna distinta llamada ***FLUX.n***, siendo *n* el número de la medición. Por último, se han etiquetado en la columna ***LABEL*** con un 2 aquellas estrellas -filas- cuyas mediciones han confirmado la existencia de algún exoplaneta en su órbita, y con un 1 aquellas que no han sido confirmadas. Al momento de realizar este proceso, los datos dejaban entrever que apenas existían estrellas con exoplanetas, por lo que se le han añadido algunos datos de estrellas otras campañas que sí han sido confirmadas con exoplanetas. Por último, se han dividido los datos en dos *datasets*, uno para el entrenamiento del modelo y el otro para test.

Entrenamiento

En la tabla 5 se muestran los datos de entrenamiento.

	LABEL	FLUX.1	FLUX.2	FLUX.3	…	FLUX.3196	FLUX.3197
0	2	93.85	83.81	20.10	…	5.08	-39.54
1	2	-38.88	-33.83	-58.54	…	16.00	19.93
2	2	532.64	535.92	513.73	…	-70.02	-96.67
…	…	…	…	…	…	…	…
5085	1	3.82	2.09	-3.29	…	-6.41	-2.55
5086	1	323.28	306.36	293.16	…	-14.09	27.82
5087 filas x 3198 columnas							

Tabla 5. *Muestra de los datos de entrenamiento*

- **5087 filas**, representando cada una de las estrellas, de las cuales:
  - **37** tienen al menos un exoplaneta en su órbita -***LABEL*** = ***2***-.
  - **5050** no tienen ningún exoplaneta en su órbita -***LABEL*** = ***1***-.
- **3198 columnas**, conformadas por:
  - **1** columna -***LABEL***- para etiquetar las estrellas.
  - **3197** columnas -***FLUX.1*** – ***FLUX.3197***-, que son las distintas mediciones del brillo de las estrellas, cada una tomada cada 36 minutos aproximadamente o 0,00046 Hz -se realizan 3.197 observaciones en 80 días: $\frac{80}{3197} \approx 0.025 \rightarrow 0.025 \times 24 \times 60 = 36\ minutos \rightarrow \frac{1}{36 \times 60} \approx 0.00046\ Hz$ -, pues la unidad de

tiempo utilizada para realizar las mediciones es el Tiempo Dinámico Baricéntrico (TDB). Por su parte, la unidad de medida de estas columnas es fotones por segundo ($e^{-}s^{-1}$), que mide la radiancia espectral.

Test

En la tabla 6 se muestran los datos de test.

	LABEL	FLUX.1	FLUX.2	FLUX.3	…	FLUX.3196	FLUX.3197
0	2	93.85	83.81	20.10	…	5.08	-39.54
1	2	-38.88	-33.83	-58.54	…	16.00	19.93
2	2	532.64	535.92	513.73	…	-70.02	-96.67
…	…	…	…	…	…	…	…
5085	1	3.82	2.09	-3.29	…	-6.41	-2.55
5086	1	323.28	306.36	293.16	…	-14.09	27.82
5087 filas x 3198 columnas							

Tabla 6. *Muestra de los datos de test*

- **570 filas**, representando cada una de las estrellas, de las cuales:
  - **5** tienen al menos un exoplaneta en su órbita -***LABEL*** = ***2***-.
  - **565** no tienen ningún exoplaneta en su órbita -***LABEL*** = ***1***-.
- **3198 columnas**, conformadas por:
  - **1** columna -***LABEL***- para etiquetar las estrellas.
  - **3197** columnas -***FLUX.1*** – ***FLUX.3197***-.

3.1. Revisión y mejoras

Preprocesamiento

En el TFG [1] se llevó a cabo el siguiente proceso:

1. **Abstracción de tendencia**. Consiste en aplicar un suavizado gaussiano a los datos, resultando en lo que se puede interpretar como la tendencia general del brillo de la estrella. Esta tendencia no es relevante para la identificación de exoplanetas, pues un tránsito ocurre en un tiempo y magnitud mucho menores. Por lo tanto, eliminamos dichas tendencias de los datos. En la primera imagen de la figura 7 se muestra el flujo en bruto de las mediciones de luz de una estrella; en la segunda, la tendencia obtenida tras el procesamiento; y en la tercera, el resultado de abstraer dicha tendencia al flujo original.
2. **Normalización** de los datos con la media centrada en 0.
3. Eliminación de **datos atípicos** superiores.

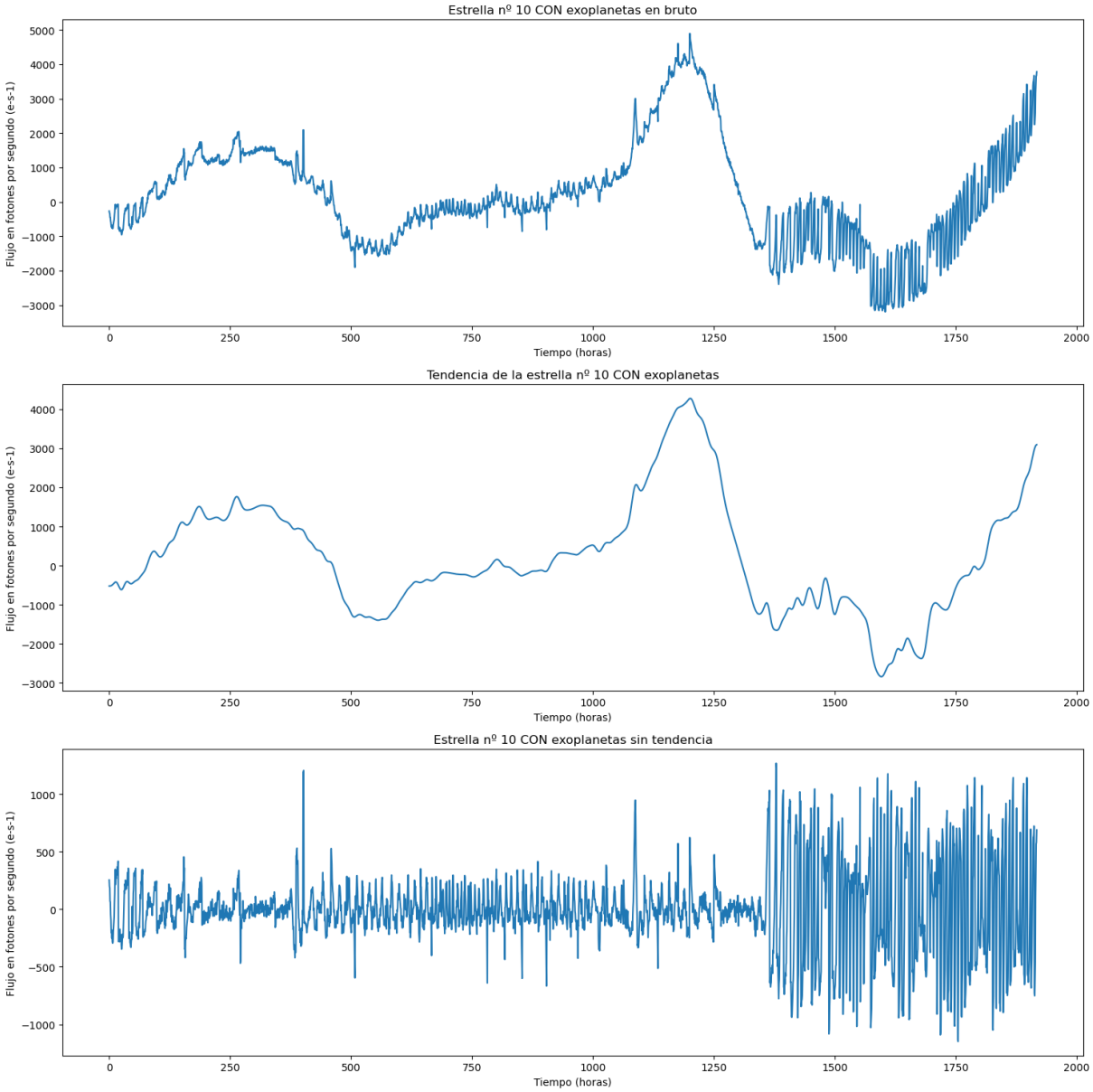

Figura 7. *Flujo en bruto (arriba), tendencia (medio) y flujo sin tendencia (abajo).*

Este proceso presenta el problema de editar los datos (tratamiento de *outliers*) después de normalizarlos, lo que desvirtúa el objetivo de la normalización al quedar cada registro del conjunto de datos en rangos muy dispares. Para corregirlo, en este trabajo se ha realizado:

1. **Abstracción de tendencia**. Mismo mecanismo que el seguido en el TFG [1].
2. **Tratamiento de outliers superiores**. Además de llevarlo a cabo antes de la normalización, se ha modificado el proceso en sí: se considera outlier aquel dato que supere en 1.5 veces la media de las 100 mediciones más cercanas, y es imputado con el valor de dicho límite. Esto parámetros se pueden adaptar específicamente para cada flujo de datos dependiendo de sus características a través de un modelo entrenado para ello, pero esto no entra dentro del alcance del presente proyecto.
3. **Normalización**. Todos los datos quedan comprendidos dentro del rango **[0,1].**

En la figura 8 se muestra un flujo totalmente preprocesado siguiendo los pasos anteriores.

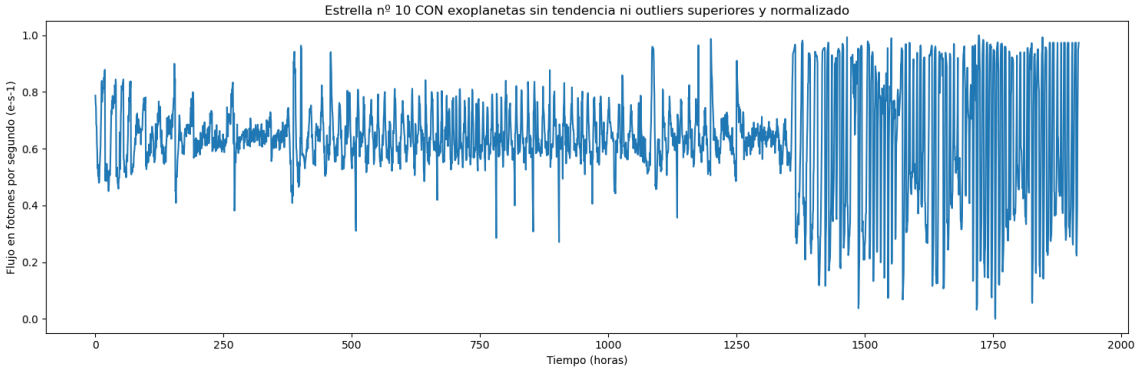

***Figura 8**. Flujo sin tendencia, sin outliers superiores y normalizado [0,1].*

Además, se han modificado las etiquetas para que los positivos queden identificados como 1 y los negativos como 0.

Data Augmentation

A diferencia del TFG [1], este trabajo no requiere de reducción de dimensionalidad de los datos. Esto es debido a que, en las estrategias que se desarrollarán más adelante, los datos se utilizarán para generar nuevos *datasets* con un número de dimensiones notablemente más reducidas. Por lo tanto, el siguiente paso realizado es el de balancear el *dataset* mediante la generación de datos positivos (con exoplanetas) a partir de los disponibles.

En el TFG [1] se realizó lo siguiente:

1. **Aumentar y disminuir** los 37 datos positivos por los factores 0.02, 0.04, 0.06, 0.08 y 0.1, lo que resulta en $10 \times 37 + 37 = 407$ datos positivos totales.
2. **Aplicación de ruido** aleatorio con 10 varianzas distintas a cada uno de los anteriores 407 registros, resultando en $407 \times 6 + 407 =$ **2849 datos positivos totales**.

Este proceso resultó en unos datos con una variabilidad suficiente como para evitar el sobreentrenamiento en el posterior modelo de clasificación.

Nuevas consideraciones

Un factor relevante pasado por alto en el TFG [1] es que, debido a las técnicas de *data augmenta*tion aplicadas (aumento/disminución y rudio), el rango de los datos normalizados se ve afectado.

Para corregir estas desviaciones, se ha tenido en cuenta que modificar las mediciones que superan el rango inferior podría perturbar la forma y medición de curvas de luz reales, alterando así la naturaleza de las detecciones.

Por lo tanto, todos los registros se han aumentado para que el valor de su medición mínima sea 0 para, posteriormente, realizar un aplanamiento superior de forma que todas las mediciones por encima de 1 sean imputadas con dicho valor.

Tras todo este procesamiento de los datos, las dimensiones de los *datasets* son las siguientes:

- **Entrenamiento**: 2849 positivos y 5050 negativos (7899 filas x 3197 columnas).
- **Test**: 5 positivos y 565 negativos (570 filas x 3197 columnas).

3.2. Modelos de clasificación

A continuación, se desarrollan las diferentes estrategias implementadas para la clasificación de los datos.

3.2.1. Estrategia A - Distribuciones

Datos

Esta estrategia utiliza los mismos datos que los del TFG [1] (pero con el proceso de tratamiento mejorado) para generar un nuevo dataset que recoja las medidas de las distribuciones de las medidas de cada registro. Éste consta de las siguientes dimensiones:

- **Media** del conjunto de mediciones por cada registro.
- **Varianza** del conjunto de mediciones por cada registro.
- **Desviación estándar** del conjunto de mediciones por cada registro.
- **Q0**: quintil 0 (0%: valor mínimo).
- **Q1**: quintil 1 (10%).
- **Q2**: quintil 2 (20%).
- **Q3**: quintil 3 (30%).
- **Q4**: quintil 4 (40%).
- **Q5**: quintil 5 (50%: valor mediano).
- **Q6**: quintil 6 (60%).
- **Q7**: quintil 7 (70%).
- **Q8**: quintil 8 (80%).
- **Q9**: quintil 9 (90%).
- **Q10**: quintil 10 (100%: valor máximo).

De esta manera, los datos quedan de la siguiente forma:

- **Entrenamiento**: 7899 filas x **14** columnas.
- **Test**: 570 filas x **14** columnas.

En la figura 9 se muestran los datos de la primera estrella del conjunto de datos de entrenamiento junto a una representación gráfica de su distribución y quintiles definidos.

LABEL	Media	Varianza	Desviacion estandar	q0	q1	q2	q3	q4	q5	q6	q7	q8	q9	q10
1	0.72645	0.005152	0.071778	0.0	0.638416	0.676148	0.6978	0.713316	0.725316	0.739893	0.756983	0.7797	0.811828	0.998687

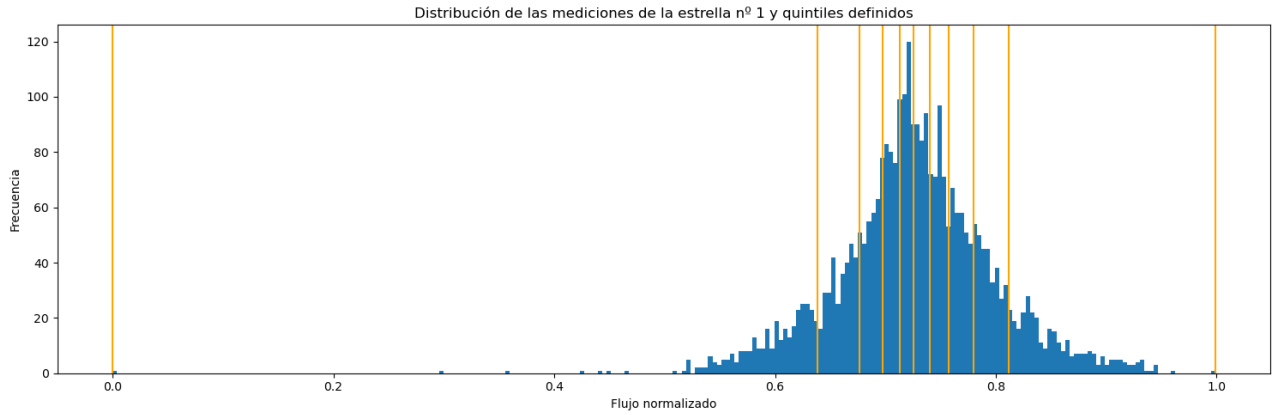

***Figura 9**. Datos de la primera línea del nuevo dataset (arriba) y distribución de la misma con los quintiles (abajo).*

Modelos

Con este conjunto de datos se han entrenado varios modelos con distintas configuraciones de hiperparámetros. Las tres arquitecturas de redes son:

- **Modelo 5 capas ocultas**:
  - Capa de entrada de **14** neuronas.
  - Capa densa de **128** neuronas con activación ***relu***.
  - Capa densa de **64** neuronas con activación ***relu***.
  - Capa densa de **32** neuronas con activación ***relu***.
  - Capa densa de **10** neuronas con activación ***relu***.
  - Capa densa de **8** neuronas con activación ***relu***.
  - Capa de salida de **1** neurona y activación ***sigmoid***.
- **Modelo 4 capas ocultas**:
  - Capa de entrada de **14** neuronas.
  - Capa densa de **64** neuronas con activación ***relu***.
  - Capa densa de **32** neuronas con activación ***relu***.
  - Capa densa de **10** neuronas con activación ***relu***.
  - Capa densa de **8** neuronas con activación ***relu***.
  - Capa de salida de **1** neurona y activación ***sigmoid***.
- **Modelo 3 capas ocultas**:
  - Capa de entrada de **14** neuronas.
  - Capa densa de **32** neuronas con activación ***relu***.
  - Capa densa de **10** neuronas con activación ***relu***.
  - Capa densa de **8** neuronas con activación ***relu***.
  - Capa de salida de **1** neurona y activación ***sigmoid***.

Los diferentes hiperparámetros utilizados son:

- ***Learning rate***: 0.01, 0.001 y 0.0001.
- ***Batch size***: 128 y 256.
- ***Epochs*** máximas: 1500.

Se han definido los siguientes callbacks:

- ***EarlyStopping***. Configurado para detener el entrenamiento cuando el ***accuracy*** con los datos de entrenamiento deje de crecer durante **200** *epochs*.
- ***ModelCheckpoint***. Guarda los pesos de la red en el momento en el que se obtiene el máximo ***F1 Score*** con los datos de validación.

Resultados

En la tabla 7 se muestran los resultados de cada red con cada combinación de los hiperparámetros descritos.

Modelo	Learning rate	Batch size	Accuracy	Precision	Recall	F1 Score
5 capas	**0.01**	**128**	99.3%	66.7%	40.0%	50.0%
5 capas	**0.01**	**256**	99.1%	50.0%	40.0%	44.4%
5 capas	**0.001**	**128**	98.2%	22.2%	40.0%	28.6%
5 capas	**0.001**	**256**	98.4%	25.0%	40.0%	30.8%
5 capas	**0.0001**	**128**	97.5%	15.4%	40.0%	22.2%
5 capas	**0.0001**	**256**	97.7%	16.7%	40.0%	23.5%
4 capas	**0.01**	**128**	99.1%	50.0%	40.0%	44.4%
4 capas	**0.01**	**256**	99.3%	66.7%	40.0%	50.0%
4 capas	**0.001**	**128**	99.1%	50.0%	40.0%	44.4%
4 capas	**0.001**	**256**	98.6%	28.6%	40.0%	33.3%
4 capas	**0.0001**	**128**	98.2%	22.2%	40.0%	28.6%
4 capas	**0.0001**	**256**	92.5%	6.8%	60.0%	12.2%
3 capas	**0.01**	**128**	99.3%	66.7%	40.0%	50.0%
3 capas	**0.01**	**256**	98.8%	33.3%	40.0%	36.4%
3 capas	**0.001**	**128**	99.1%	50.0%	40.0%	44.4%
3 capas	**0.001**	**256**	98.6%	33.3%	60.0%	42.9%
3 capas	**0.0001**	**128**	96.3%	10.0%	40.0%	16.0%
3 capas	**0.0001**	**256**	98.4%	25.0%	40.0%	30.8%

Tabla 7. *Resultados de todos los modelos de la Estrategia A.*

Mejor modelo

El criterio para seleccionar el mejor modelo de los anteriores es el ***F1 Score***, debido a que combina las medidas de *recall* y *precision* en un solo valor, lo que refleja mejor el rendimiento de los modelos.

En este caso, hay 3 modelos con los mejores resultados. Uno es el de **5 capas** ocultas, *learning rate* de **0.01** y *batch size* de **128**; el otro es de **4 capas** ocultas, *learning rate* de **0.01** y *batch size* de **256**; y el último es de **3 capas** ocultas, *learning rate* de **0.01** y *batch size* de **128**. En la figura 10 se muestra la matriz de confusión con las predicciones de los mejores modelos sobre los datos de test.

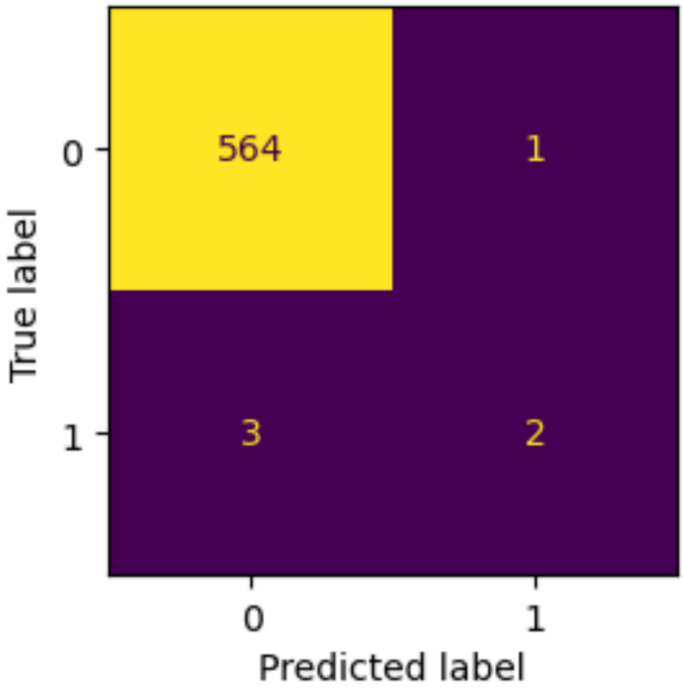

***Figura 10**. Matriz de confusión de los mejores modelos de la Estrategia A.*

Conclusión

Pese a que la clasificación del mejor modelo del TFG [1] logró ser muy buena dadas las limitaciones, la clasificación obtenida con estos modelos de la Estrategia A la ha mejorado, puesto que ha conseguido dos falsos positivos menos, resultando en tan solo **1 falso positivo** y **3 falsos negativos**.

Además, estos excelentes resultados se han conseguido con tan solo 14 dimensiones, por las más de 600 de las empleadas en el TFG [1]. Esto demuestra la eficacia de la Estrategia A al condensar toda la información de los datos en forma de distribuciones.

3.2.2. Estrategia B – Selección de dimensionalidad

Datos

Esta estrategia parte de los mismos datos que los del TFG [1] (pero con el proceso de tratamiento mejorado) y realiza lo siguiente:

1. **Detección de posibles tránsitos**. Por cada línea, se trata de identificar tránsitos de forma similar a cómo se detectaron los *outliers* superiores en el preprocesamiento (un tránsito no deja de ser una anomalía, pero por debajo de los valores medios en este caso). El algoritmo utilizado considera una medición como tránsito cuando ésta es **0.82** (82%) del valor medio de las **32** mediciones más cercanas.
2. **Almacenar tránsitos**. Tras identificar un tránsito, se almacena su valor junto a los de las **8** mediciones más cercanas.
3. **Almacenar posiciones**. Además de los valores anteriores, también se almacenan las posiciones de los mismos.

El resultado de este proceso son 3 vectores:

- **Vector tránsitos**. Contiene los valores de las mediciones detectadas como tránsitos, incluyendo los valores de las 8 mediciones más cercanas.
- **Vector posiciones**. Cada valor equivale a la posición que ocupa el valor correspondiente del vector tránsitos en los datos originales.
- **Vector posiciones únicas**. Contiene la posición del valor mínimo de cada tránsito (sin incluir mediciones más cercanas). Su longitud equivale al número de tránsitos detectados en cada línea.

En la primera imagen de la figura 11 se muestran todas las mediciones de la primera estrella de los datos originales (preprocesados); en la segunda, los posibles tránsitos identificados (en verde, el valor mínimo de cada tránsito; en naranja, los vecinos más cercanos de los anteriores); y en las dos últimas se muestran las mismas mediciones de las dos primeras centradas en el rango [150,250].

El vector de posiciones únicas contiene la información de la posición de los tránsitos con respecto a los datos originales. Ésta no es interesante debido a la aleatoriedad de la aparición de un tránsito en los datos, lo que supondría un problema con las diferentes magnitudes de las posiciones (el primer tránsito de una estrella podría encontrarse en la posición 11, mientras que en otra podría darse en la posición 250).

Para solucionarlo, se calcula un nuevo vector:

- **Vector frecuencias**. Cada valor de este vector es la diferencia entre las posiciones de los tránsitos del vector posiciones únicas, por lo que contiene las frecuencias con las que aparecen los tránsitos en los datos.

A partir de estos vectores, se genera un nuevo dataset que representa las distribuciones de los valores de las mediciones de los posibles tránsitos y sus frecuencias de aparición en los datos. Las 21 dimensiones de este nuevo conjunto de datos son:

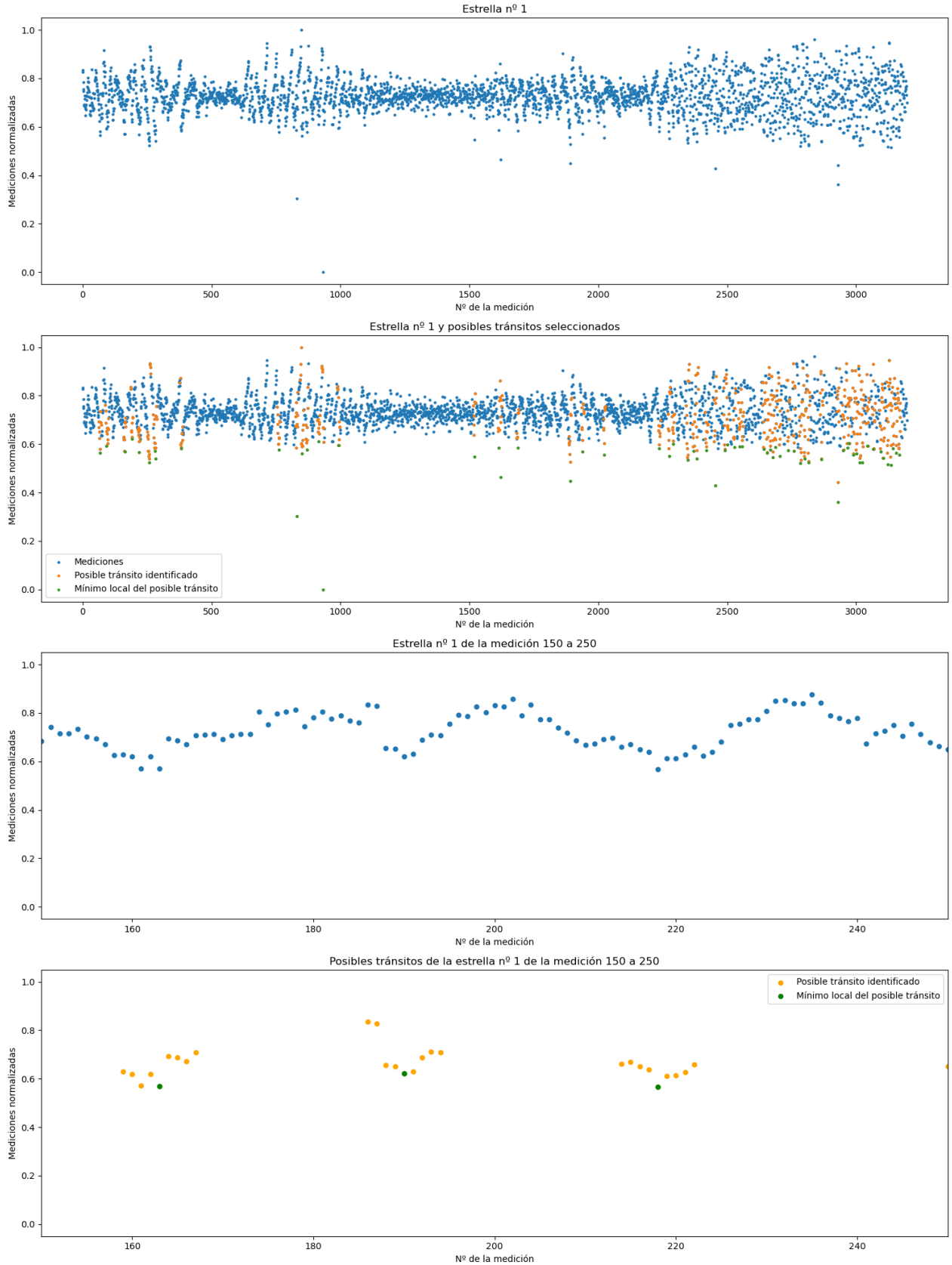

***Figura 11**. Detección de tránsitos*

- **Tránsitos**: número de los posibles tránsitos identificados.
- **Frec. media**: Media de la frecuencia de aparición de los posibles tránsitos.
- **Frec. varianza**: Varianza de la frecuencia de aparición de los posibles tránsitos.
- **Frec. desviación estándar**: Desviación estándar de la frecuencia de aparición de los posibles tránsitos.
- **Frec. mediana**: Mediana de la frecuencia de aparición de los posibles tránsitos.
- **Frec. Q0**: quintil 0 (0%: frecuencia mínima).
- **Frec. Q1**: quintil 1 (20%).
- **Frec. Q2**: quintil 2 (40%).
- **Frec. Q3**: quintil 3 (60%).
- **Frec. Q4**: quintil 4 (80%).
- **Frec. Q5**: quintil 5 (100%: frecuencia máxima).
- **Media** de las mediciones de los posibles tránsitos y sus vecinos más cercanos.
- **Varianza** de las mediciones de los posibles tránsitos y sus vecinos más cercanos.
- **Desviación estándar**: de las mediciones de los posibles tránsitos y sus vecinos más cercanos.
- **Mediana** de las mediciones de los posibles tránsitos y sus vecinos más cercanos.
- **Q0**: quintil 0 (0%: valor mínimo).
- **Q1**: quintil 1 (20%).
- **Q2**: quintil 2 (40%).
- **Q3**: quintil 3 (60%).
- **Q4**: quintil 4 (80%).
- **Q5**: quintil 5 (100%: valor máximo).

De esta manera, los datos quedan de la siguiente forma:

- **Entrenamiento**: 7899 filas x **21** columnas.
- **Test**: 570 filas x **21** columnas.

En la primera imagen de la figura 12 se muestra la distribución de los valores de las mediciones de los posibles tránsitos de la primera estrella del dataset y los quintiles definidos; en la segunda, la distribución de las frecuencias de aparición de los mismos y sus quintiles.

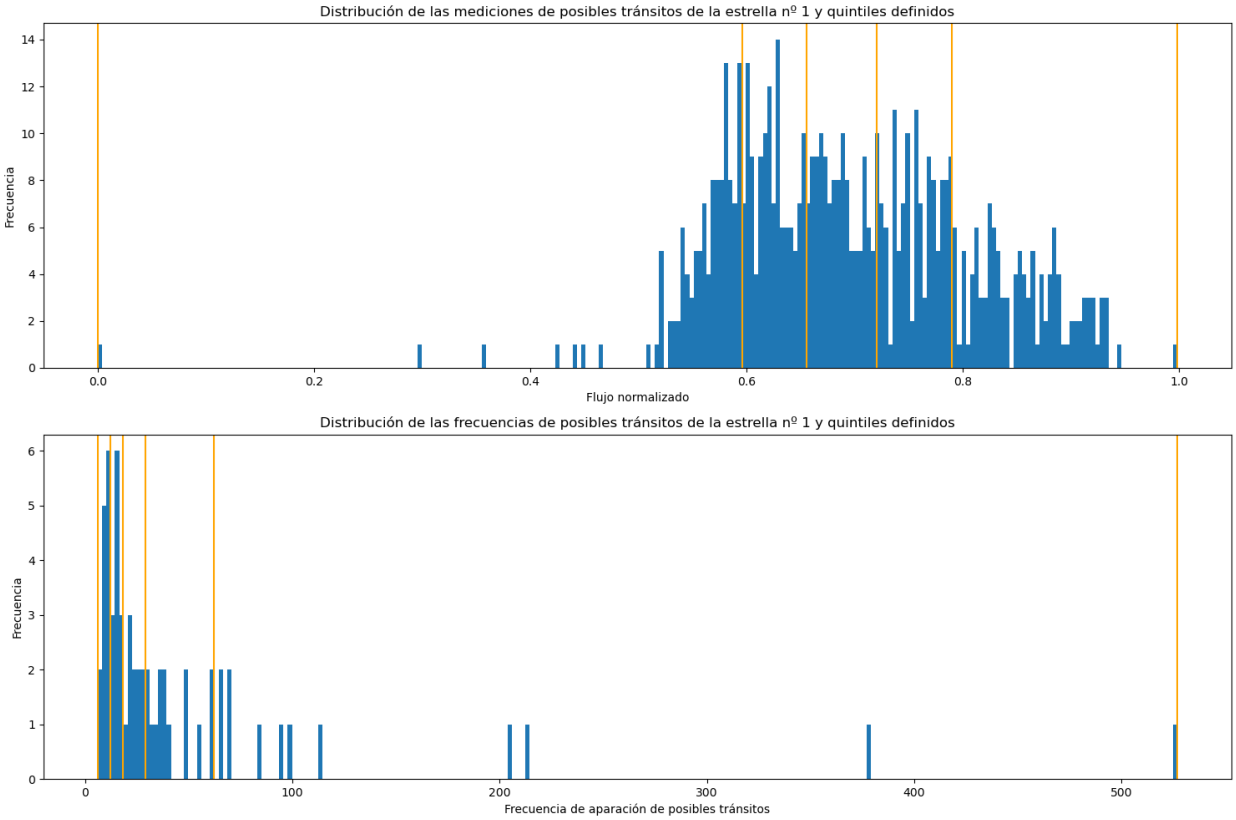

***Figura 12**. Distribución de los posibles tránsitos y sus frecuencias de aparición.*

Modelos

Con este conjunto de datos se han entrenado varios modelos con distintas configuraciones de hiperparámetros. Las tres arquitecturas de redes son:

- **Modelo 6 capas ocultas**:
  - Capa de entrada de **21** neuronas.
  - Capa densa de **256** neuronas con activación ***relu***.
  - Capa densa de **128** neuronas con activación ***relu***.
  - Capa densa de **64** neuronas con activación ***relu***.
  - Capa densa de **32** neuronas con activación ***relu***.
  - Capa densa de **10** neuronas con activación ***relu***.
  - Capa densa de **8** neuronas con activación ***relu***.
  - Capa de salida de **1** neurona y activación ***sigmoid***.
- **Modelo 5 capas ocultas**:
  - Capa de entrada de **14** neuronas.
  - Capa densa de **256** neuronas con activación ***relu***.
  - Capa densa de **64** neuronas con activación ***relu***.
  - Capa densa de **32** neuronas con activación ***relu***.
  - Capa densa de **10** neuronas con activación ***relu***.
  - Capa densa de **8** neuronas con activación ***relu***.
  - Capa de salida de **1** neurona y activación ***sigmoid***.

- **Modelo 4 capas ocultas**:
  - Capa de entrada de **14** neuronas.
  - Capa densa de **128** neuronas con activación ***relu***.
  - Capa densa de **32** neuronas con activación ***relu***.
  - Capa densa de **10** neuronas con activación ***relu***.
  - Capa densa de **8** neuronas con activación ***relu***.
  - Capa de salida de **1** neurona y activación ***sigmoid***.

Los diferentes hiperparámetros utilizados son:

- ***Learning rate***: 0.01, 0.001 y 0.0001.
- ***Batch size***: 64 y 128.
- ***Epochs*** máximas: 2500

Se han definido los siguientes callbacks:

- ***EarlyStopping***. Configurado para detener el entrenamiento cuando el ***accuracy*** con los datos de entrenamiento deje de crecer durante **200** *epochs*.
- ***ModelCheckpoint***. Guarda los pesos de la red en el momento en el que se obtiene el máximo ***F1 Score*** con los datos de validación.

Resultados

En la tabla 8 se muestran los resultados de cada red con cada combinación de los hiperparámetros descritos.

Modelo	Learning rate	Batch size	Accuracy	Precision	Recall	F1 Score
6 capas	**0.01**	**64**	99.1%	0.0%	0.0%	0.0%
6 capas	**0.01**	**128**	99.1%	0.0%	0.0%	0.0%
6 capas	**0.001**	**64**	93.7%	8.1%	60.0%	14.3%
6 capas	**0.001**	**128**	99.1%	0.0%	0.0%	0.0%
6 capas	**0.0001**	**64**	98.2%	22.2%	40.0%	28.6%
6 capas	**0.0001**	**128**	94.4%	6.5%	40.0%	11.1%
5 capas	**0.01**	**64**	93.9%	3.1%	20.0%	5.4%
5 capas	**0.01**	**128**	92.8%	5.0%	40.0%	8.9%
5 capas	**0.001**	**64**	97.0%	16.7%	60.0%	26.1%
5 capas	**0.001**	**128**	96.1%	13.0%	60.0%	21.4%
5 capas	**0.0001**	**64**	94.0%	10.8%	80.0%	19.0%
5 capas	**0.0001**	**128**	96.7%	15.0%	60.0%	24.0%
4 capas	**0.01**	**64**	98.8%	25.0%	20.0%	22.2%
4 capas	**0.01**	**128**	94.4%	6.5%	40.0%	11.1 %
4 capas	**0.001**	**64**	95.8%	8.7%	40.0%	14.3%
4 capas	**0.001**	**128**	96.1%	9.5%	40.0%	15.4%
4 capas	**0.0001**	**64**	94.2%	11.1%	80.0%	19.5%
4 capas	**0.0001**	**128**	98.6%	28.6%	40.0%	33.3%

Tabla 8. *Resultados de todos los modelos de la Estrategia B.*

Mejor modelo

Siguiendo el mismo criterio que en la Estrategia A, el mejor modelo es el de **4 capas** ocultas entrenado con un *learning rate* de **0.0001** y un *batch size* de **128**, logrando un *accuracy* del **98.6%**, una *precision* de **28.6%**, un *recall* de **40%** y un *F1 Score* de 33.3%. En la figura 13 se muestra la matriz de confusión con las predicciones del mejor modelo sobre los datos de test.

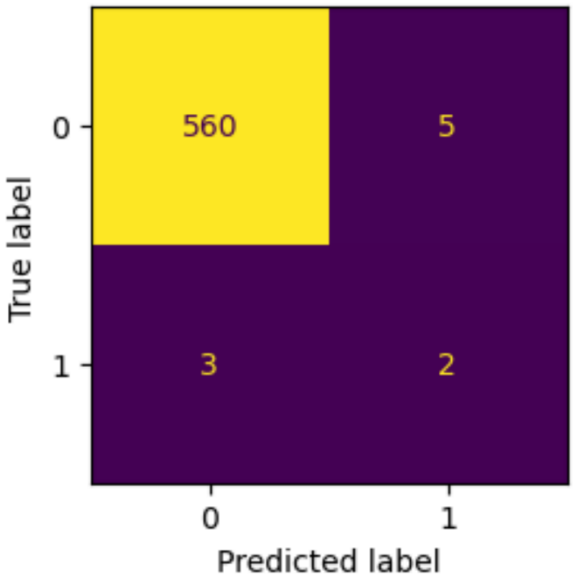

Figura 13. *Matriz de confusión del mejor modelo de la Estrategia B.*

Conclusión

Esta estrategia ha cometido **5 falsos positivos**, más que en el TFG [1] y en la Estrategia A; y los mismos **3 falsos negativos**, por lo que el rendimiento de la Estrategia B es algo inferior con respecto a los anteriores.

3.2.3. Estrategia C – Mecanismo de atención

Esta estrategia consiste en obtener, por cada línea del conjunto de datos, un **vector atención** que represente las ponderaciones de cada medición en función de sus características relativas. De esta forma, al ponderar los datos originales preprocesados, las mediciones más propensas de formar parte de un tránsito se verán más exageradas, lo que podría mejorar el rendimiento de los modelos de las estrategias **A** y **B**.

Datos

Cada componente del vector atención se calcula según la ecuación 4.

Ecuación 4. *Vector atención*

$$valor_{atención} = \begin{cases} \dfrac{valor_{medición} - mínimo}{threshold - mínimo}, & valor_{medición} \leq threshold \\ 1, & valor_{medición} > threshold \end{cases}$$

donde

$$f \in F_n, \qquad mínimo \leq f$$

y

$$threshold = \bar{F_n} \cdot (1 - \sigma_{F_n})$$

donde

$$F_n = \{n\ vecinos\ más\ cercanos\ de\ valor_{medición}\}$$

El vector atención se puede entender como la normalización entre el rango [0,1] de los valores de las mediciones que se encuentran por debajo de $threshold$.

En la primera imagen de la figura 14 se muestra la estrella 8 del conjunto de entrenamiento sin aplicar el mecanismo de atención; en la segunda, la misma estrella tras la ponderación por el vector atención; la tercera se centra en un rango específico de longitud n, junto a la $\bar{F_n}$ (naranja) y $threshold$ (rojo); y la cuarta es idéntica a la tercera tras la ponderación con el vector atención. Se puede apreciar cómo el mecanismo de atención resalta los posibles tránsitos.

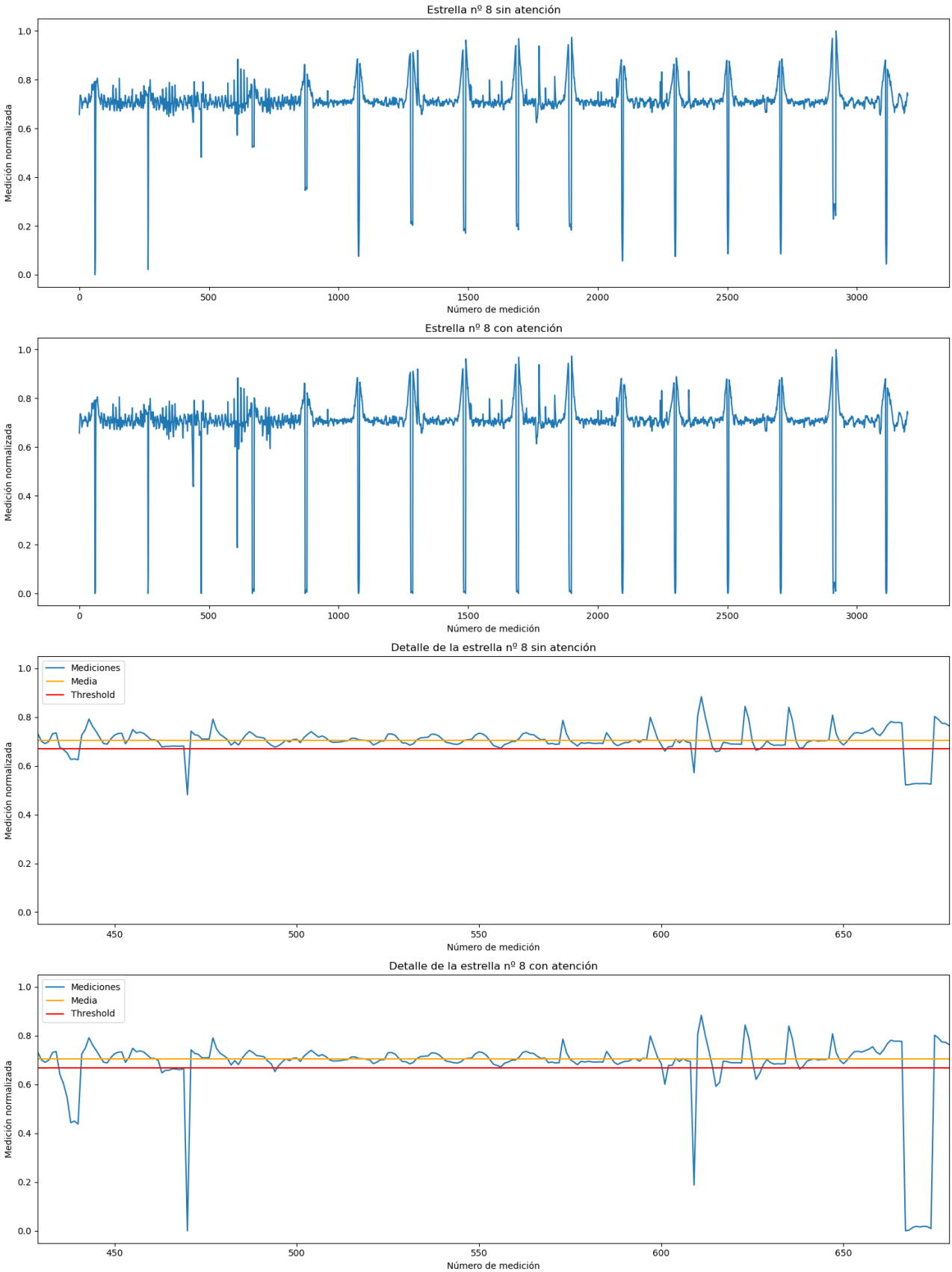

Figura 14. *Aplicación del mecanismo de atención.*

3.2.4. Estrategia C_A – Distribuciones con mecanismo de atención

Con los nuevos datos resultantes de aplicar el mecanismo de atención, se vuelve a construir el dataset de distribuciones de la **Estrategia A** y a entrenar las mismas configuraciones de modelos e hiperparámetros.

Resultados

En la tabla 9 se muestran los resultados de cada red con cada una de las combinaciones de los hiperparámetros utilizadas en la Estrategia A.

Modelo	Learning rate	Batch size	Accuracy	Precision	Recall	F1 Score
5 capas	**0.01**	**128**	92.6%	4.9%	40.0%	8.7%
5 capas	**0.01**	**256**	90.5%	5.5%	60.0%	10.0%
5 capas	**0.001**	**128**	91.2%	5.9%	60.0%	10.7%
5 capas	**0.001**	**256**	93.5%	7.9%	60.0%	14.0%
5 capas	**0.0001**	**128**	89.6%	5.0%	60.0%	9.2%
5 capas	**0.0001**	**256**	94.2%	6.2%	40.0%	10.8%
4 capas	**0.01**	**128**	91.2%	5.9%	60.0%	10.7%
4 capas	**0.01**	**256**	97.2%	13.3%	60.0%	20.0%
4 capas	**0.001**	**128**	90.5%	5.5%	60.0%	10.0%
4 capas	**0.001**	**256**	93.2%	7.5%	60.0%	13.3%
4 capas	**0.0001**	**128**	90.5%	5.5%	60.0%	10.0%
4 capas	**0.0001**	**256**	94.4%	9.1%	60.0%	15.8%
3 capas	**0.01**	**128**	92.5%	8.7%	80.0%	15.7%
3 capas	**0.01**	**256**	94.0%	8.6%	60.0%	15.0%
3 capas	**0.001**	**128**	93.2%	7.5%	60.0%	13.3%
3 capas	**0.001**	**256**	91.9%	6.4%	60.0%	11.5%
3 capas	**0.0001**	**128**	92.1%	6.5%	60.0%	11.8%
3 capas	**0.0001**	**256**	91.9%	6.4%	60.0%	11.5%

Tabla 9. *Resultados de todos los modelos de la Estrategia C_A.*

Mejor modelo

Siguiendo el mismo criterio, el mejor modelo es el de **4 capas** ocultas entrenado con un *learning rate* de **0.01** y un *batch size* de **256**., logrando un *accuracy* de **97.2%**, una *precision* de **13.3%**, un *recall* de **60%** y un *F1 Score* de **20%**. En la figura 15 se muestra la matriz de confusión con las predicciones del mejor modelo sobre los datos de test.

Conclusión

Tras aplicar el mecanismo de atención, la Estrategia C_A ha cometido los mismo **3 falsos negativos**, pero el número de falsos positivos ha aumentado con respecto a la Estrategia A. Por ello, para esta aproximación no ha resultado beneficioso ponderar los datos por el vector atención obtenido previamente.

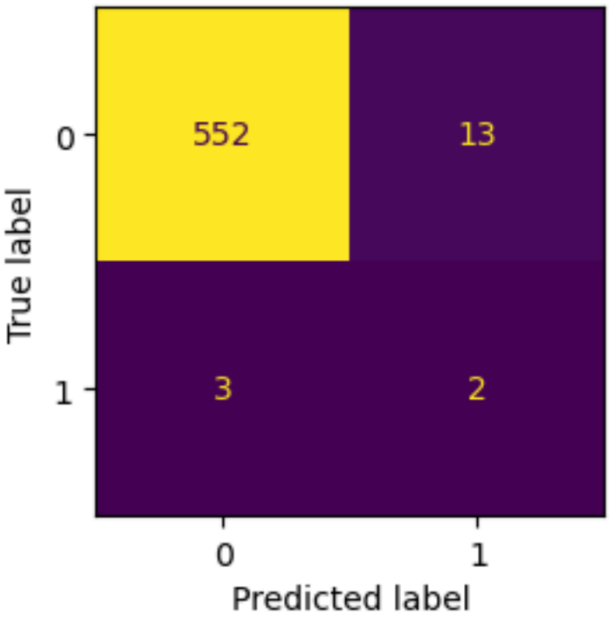

***Figura 15**. Matriz de confusión del mejor modelo de la Estrategia C_A.*

3.2.5. Estrategia C_B – Selección de dimensionalidad con mecanismo de atención

Con los nuevos datos resultantes de aplicar el mecanismo de atención, se vuelve a detectar los posibles tránsitos y a construir el dataset de distribuciones de mediciones y frecuencias de la **Estrategia B** y a entrenar las mismas configuraciones de modelos e hiperparámetros.

En la figura 16 se muestra, de manera análoga a la figura 11, la detección de tránsitos potenciales en una estrella tras la aplicación del mecanismo de atención.

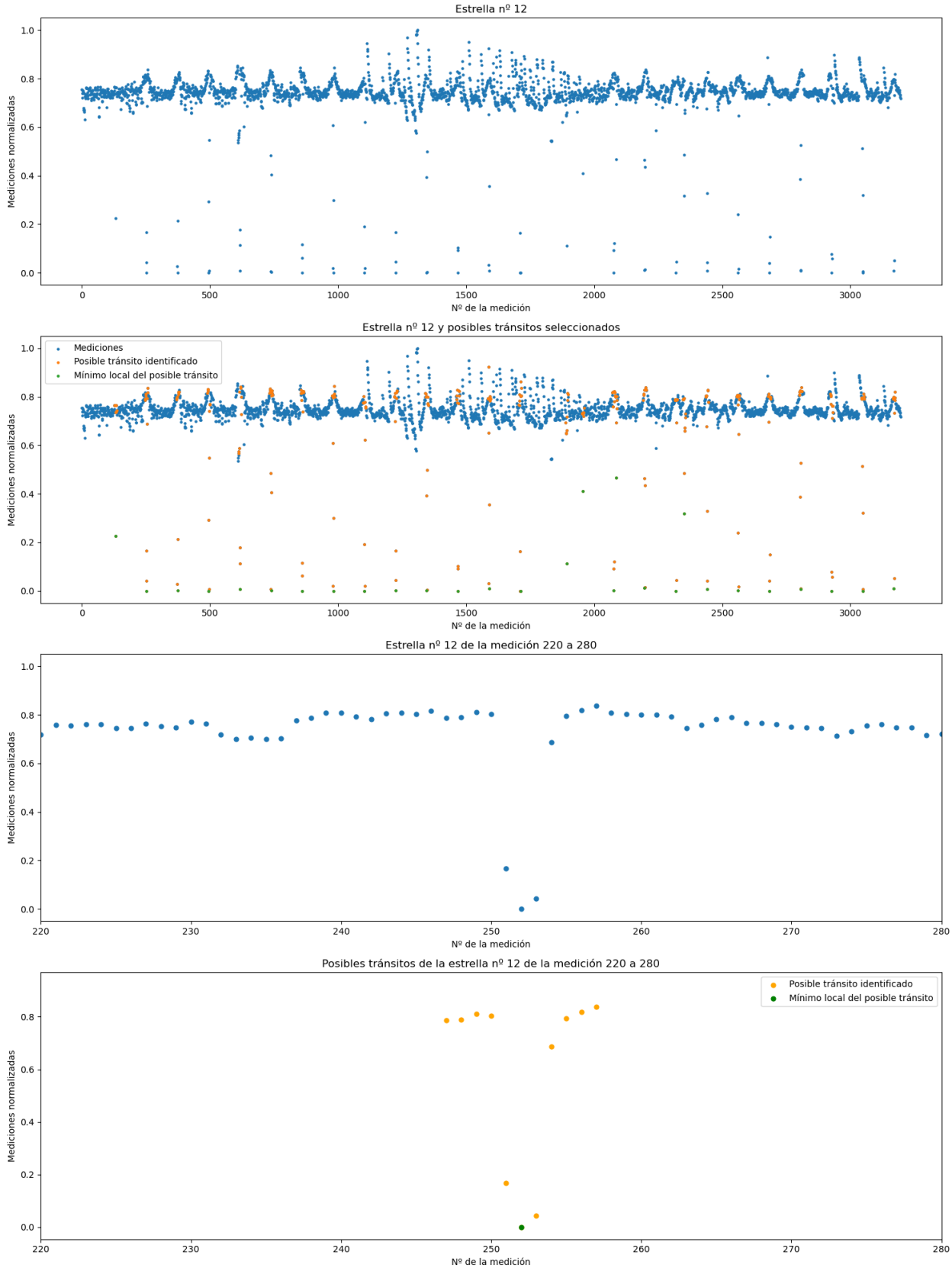

Figura 16. *Detección de tránsitos tras el mecanismo de atención*

Resultados

En la tabla 10 se muestran los resultados de cada red con cada una de las combinaciones de los hiperparámetros utilizadas en la **Estrategia B**.

Modelo	Learning rate	Batch size	Accuracy	Precision	Recall	F1 Score
6 capas	**0.001**	**64**	99.1%	0.0%	0.0%	0.0%
6 capas	**0.001**	**128**	99.1%	50.0%	20.0%	28.6%
6 capas	**0.0001**	**64**	96.0%	12.5%	60.0%	20.7%
6 capas	**0.0001**	**128**	97.9%	23.1%	60.0%	33.3%
6 capas	**0.00001**	**64**	95.8%	12.0%	60.0%	20.0%
6 capas	**0.00001**	**128**	96.8%	15.8%	60%	25.0%
5 capas	**0.001**	**64**	99.1%	0.0%	0.0%	0.0%
5 capas	**0.001**	**128**	99.1%	0.0%	0.0%	0.0%
5 capas	**0.0001**	**64**	94.4%	9.1%	60.0%	15.8%
5 capas	**0.0001**	**128**	95.4%	13.8%	80.0%	23.5%
5 capas	**0.00001**	**64**	98.2%	27.3%	60.0%	37.5%
5 capas	**0.00001**	**128**	96.8%	15.8%	60.0%	25.0%
4 capas	**0.001**	**64**	94.6%	9.4%	60.0%	16.2%
4 capas	**0.001**	**128**	97.2%	17.6%	60.0%	27.3%
4 capas	**0.0001**	**64**	93.2%	7.5%	60.0%	13.3%
4 capas	**0.0001**	**128**	98.8%	37.5%	60.0%	46.2%
4 capas	**0.00001**	**64**	98.1%	25.0%	60.0%	35.3%
4 capas	**0.00001**	**128**	99.5%	75.0%	60.0%	66.7%

Tabla 10. *Resultados de todos los modelos de la Estrategia* C_B.

Mejor modelo

Siguiendo el mismo criterio, el mejor modelo es el de **4 capas** ocultas entrenado con un *learning rate* de **0.00001** y un *batch size* de **128**; obteniendo un *accuracy* de **99.5%**, una *precision* del **75%**, un *recall* del **60%** y un *F1 Score* de **66.7%**. En la figura 17 se muestra la matriz de confusión con las predicciones del mejor modelo sobre los datos de test.

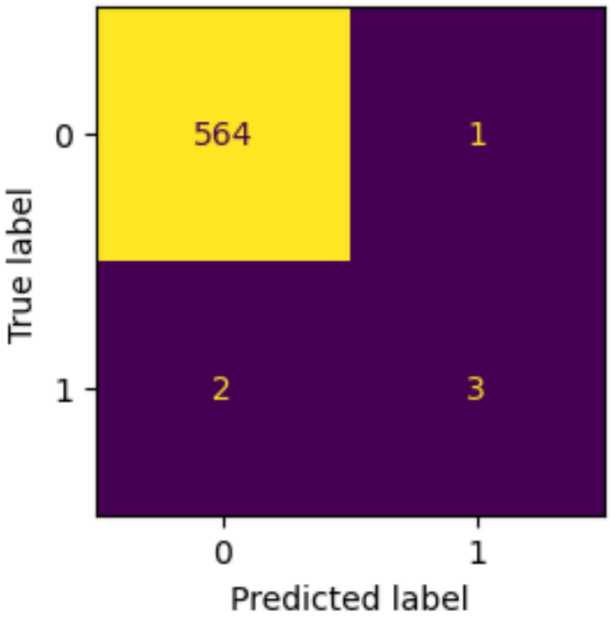

Figura 17. *Matriz de confusión del mejor modelo de la Estrategia* C_B

Conclusión

Esta estrategia ha logrado cometer tan solo **2 falsos negativos** y **1 único falso positivo**, prediciendo correctamente **3 positivos**. Estos resultados mejoran espectacularmente los obtenidos a lo largo de todo el presente trabajo y el TFG [1].

La combinación de la selección de posibles tránsitos con las distribuciones de sus mediciones y frecuencias, junto con la aplicación del mecanismo de atención ha logrado los mejores resultados. Esto se debe a que esta Estrategia C_B recoge toda la información posible: posibles tránsitos, sus mediciones y sus frecuencias, además de estar ponderados por el vector atención calculado.

El código con toda la implementación en formato html *se puede descargar aquí.*

3.3. Comparación de resultados

En la tabla 11 se muestran los resultados obtenidos en el TFG [1] y en las diferentes estrategias de este trabajo.

Mejor modelo	Verdaderos		Falsos		F1 Score	Dimensiones
	Negativos	Positivos	Negativos	Positivos		
TFG [1]	**562**	**2**	**3**	**3**	40.0%	603
Estrategia A	**564**	**2**	**3**	**1**	50.0%	14
Estrategia B	**560**	**2**	**3**	**5**	33.3%	21
Estrategia C_A	**552**	**2**	**3**	**13**	20.0%	14
Estrategia C_B	**564**	**3**	**2**	**1**	66.7%	21

Tabla 11. *Resultados*

Tras haber condensado todas las dimensiones originales en distribuciones y haber usado las medidas de las mismas para entrenar los modelos, la **Estrategia A** ha mejorado los resultados del TFG [1] al haber clasificado correctamente 2 negativos más y, por tanto, haber cometido 2 errores menos en cuanto a los falsos positivos.

El nuevo enfoque aplicado en la **Estrategia B**, mediante el cual se realizaba una selección de las dimensiones que podrían ser tránsitos y, posteriormente, se comprimían en distribuciones tanto de sus mediciones como de sus frecuencias de aparición, no ha conseguido mejorar ni a la Estrategia A ni al TFG [1].

La también novedosa **Estrategia C** consistía en obtener un vector atención que, al multiplicarlo por los datos originales, los posibles tránsitos quedaban más marcados y eran más diferenciables. Con estos nuevos datos se repetían las dos estrategias anteriores, resultando en las estrategias C_A y C_B.

La **Estrategia C_A** empeoró considerablemente los resultados al cometer 13 falsos positivos, 12 más que empleando los datos originales. Esto demuestra que, en este caso, la ponderación de los datos por el vector atención no ha sido beneficiosa.

Sin embargo, la **Estrategia C_B** obtuvo 3 verdaderos positivos, más que ninguna otra estrategia, y tan solo 1 falso positivo, igualando el mejor registro obtenido en la Estrategia A. Estos resultados suponen la mejor clasificación de todo el proyecto y el TFG [1], obteniendo un *accuracy* de **99.5%**, una *precision* del **75%**, un *recall* del **60%** y un *F1 Score* de **66.7%**, cometiendo únicamente 3 errores en total. En este caso, el mecanismo de atención sí resultó determinante para mejorar la clasificación.

4. Conclusión

Pese a la dificultad técnica de los datos trabajados y los buenos resultados obtenidos en el TFG [1], aún existía un margen de mejora tanto en el procesamiento como en la aproximación para tratar la reducción de dimensionalidad y la clasificación.

El procesamiento de los datos se ha mejorado siguiendo un proceso más lógico y teniendo en cuenta mayores consideraciones que podían influir, directamente, en los modelos de clasificación posteriores. El resultado de este apartado ha sido un *dataset* más consistente y limpio.

Uno de los puntos que más se prestaba a ser mejorado era el de la reducción de la dimensionalidad. En el TFG [1] se empleó el *Principal Analysis Component* (PCA) para pasar de 3197 dimensiones a 603. No obstante, dada la naturaleza de los datos, esta técnica no resulta ser la más acertada debido a la fuerte relación de todas las dimensiones entre sí (cada dimensión es la misma medida en distintos momentos de tiempo). Por lo tanto, tratar de comprimir toda la información esencial de las dimensiones originales en un espacio de dimensiones reducido en base a combinaciones lineales de las mismas, en unos datos cuyas dimensiones son más bien abstracciones temporales de una misma medida que varía, hace que los datos resultantes pierdan gran parte de su información intrínseca.

Gracias a la aproximación empleada en las diferentes estrategias, toda la abstracta información esencial de los datos originales para clasificarlos en función de si existen exoplanetas se logra mantener al mismo tiempo que se reduce la dimensionalidad (y de forma más agresiva con 14 dimensiones) al interpretar los datos como distribuciones de una única medida unidimensional.

El resultado de esta compresión de la información en la **Estrategia A** como mejora del proceso seguido en el TFG [1] se ha visto ratificada con unos mejores resultados; además de haber sido obtenidos con una mayor eficiencia al contar con un número de dimensiones menor.

La **Estrategia B** ha consistido en ir un paso más lejos. En lugar de comprimir todos los datos originales, se ha realizado una selección de datos algorítmica en función de sus características relativas que las hacen más o menos propensas a ser tránsitos (que son indicadores que ayudan a discernir entre la existencia y no existencia de exoplanetas orbitando una estrella). Tras este proceso, se han comprimido los datos de la misma manera que en la Estrategia A, añadiendo además datos sobre la frecuencia de aparición de estas dimensiones seleccionadas para no perder la información temporal de los datos originales. Pese a lograr unos resultados decentes, no han sido mejores que los anteriores, lo que lleva a pensar que la selección algorítmica de las dimensiones tenga margen de mejora si se realizase de forma inteligente.

La **Estrategia C** ha incorporado una nueva aproximación: el mecanismo de atención. Tras obtener un vector atención de forma algorítmica, que se puede interpretar como información añadida acerca de qué características son más relevantes, se han repetido las dos

estrategias anteriores. Gracias a este nuevo enfoque, se ha logrado la mejor clasificación de todo el proyecto al ejecutar la Estrategia B con el mecanismo de atención.

Esto se puede explicar gracias a que el mecanismo de atención ha facilitado la selección de dimensionalidad realizada algorítmicamente, lo que ha resultado en un *dataset* de distribuciones de las mediciones y frecuencias con mayor información acerca de los posibles tránsitos, lo que ha ayudado a los modelos a identificarlos con más facilidad.

Para poner en perspectiva los resultados obtenidos, el trabajo de *Singh, Amritanshu Kumar et al.* [5],visto previamente en *Estado del arte*, utiliza los mismos datos de test de la Campaña 3 de la misión *K2* empleados en el presente proyecto. Tal y como se muestra en las tablas 1 y 2, tras aplicar SMOTE para *data augmentation* y utilizar CNN y SVM para la clasificación, los mejores resultados constan de 489 verdaderos negativos, 2 verdaderos positivos, 3 falsos negativos y 76 falsos positivos, lo que supone un F1 Score de 5%. Estos resultados plasman la dificultad del tratamiento y clasificación de los datos empleados.

Teniendo en cuenta lo anterior, la referencia utilizada como punto de partida de este trabajo se trata de un resultado bueno -F1 Score de 40% obtenido en el TFG [1]-, por lo que todo lo que supere un F1 Score de 40% o 50% en la clasificación de los datos de test es muy positivo. Esto se ha conseguido tanto con la Estrategia A como con la **Estrategia C_B** con un 40% y un **66.7%**, respectivamente; por lo tanto, los resultados de esta última estrategia se puede considerar como excelentes.

5. Trabajos futuros

Las nuevas aproximaciones han logrado buenos resultados, lo que abre una nueva vía para investigar en esta línea. Los puntos de mejora más claros que tiene este proyecto se focalizan en los procesos algorítmicos utilizados en distintas ocasiones:

- **Detección de datos atípicos superiores**. Algoritmo para detectar e imputar outliers que superan cierto límite en el rango superior de los datos.
- **Selección de dimensionalidad**. Algoritmo para detectar posibles tránsitos en función de ciertas características relativas.
- **Vector atención**. Algoritmo para obtener unas ponderaciones en función de ciertas características relativas.

Por lo tanto, la mejora de los resultados obtenidos en este trabajo pasa por explorar la utilización de modelos como RNN u otro tipo de red tanto para la detección de tránsitos en los datos como para la ponderación de los mismos, en lugar de emplear una aproximación algorítmica.

Otro punto a tratar podría ser la implementación de las diferentes estrategias en *datasets* de otras campañas para estudiar el comportamiento de los modelos con otros datos y observar su generalización. Para ello habría que reconsiderar el procesamiento de los datos y adaptarlo a sus características, dado el desbalanceamiento o dimensionalidad de los mismos.

Por último, dado que la aproximación utilizada para desarrollar las estrategias es muy novedosa al tratarse, esencialmente, de interpretar los datos como distribuciones, se abre una nueva vía para explorar diferentes modelos sobre estos nuevos datos reinterpretados, como podrían ser los modelos y algoritmos de detección de anomalías *ThetaRay* [16] vistos en *Estado del arte*, ente otros.

6. Bibliografía

[1] J. Gómez de Diego, *Machine Learning para el tratamiento de datos y la detección de exoplanetas mediante el método de tránsito,* Universidad Politécnica de Madrid, E.T.S.I. de Sistemas Informáticos, 2020.

[2] S. Cuéllar et al., *Deep learning exoplanets detection by combining real and synthetic data,* Sathishkumar V E, Hanyang University, KOREA, REPUBLIC OF, 2022.

[3] K. Mandel and E. Agol, *Analytic Light Curves for Planetary Transit Searches,* 2002.

[4] H. Parviainen, *PyTransit: fast and easy exoplanet transit modelling in PYTHON,* 3 ed., vol. 450, Monthly Notices of the Royal Astronomical Society, 2015, pp. 3233-3238.

[5] A. K. Singh and V. A. Kumbhare, *Detection of Exoplanets using Machine Learning,* International Journal of Research Publication and Reviews (IJRPR), 2022.

[6] K. Cui, J. Liu, F. Feng and J. Liu, *Identify Light-curve Signals with Deep Learning Based Object Detection Algorithm. I. Transit Detection,* The Astronomical Journal, 2021.

[7] J. Redmon and A. Farhadi, *Yolov3: An incremental improvement,* Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[8] A. Rosenfeld and M. Thurston, *Edge and Curve Detection for Visual Scene Analysis,* IEEE Transactions on Computers, 1971.

[9] A. Malik, B. P. Moster and C. Obermeier, *Exoplanet detection using machine learning,* 4 ed., vol. 513, Monthly Notices of the Royal Astronomical Society, 2022, p. 5505–5516.

[10] C. J. Shallue and A. Vanderburg, *Identifying Exoplanets with Deep Learning: A Five-planet Resonant Chain around Kepler-80 and an Eighth Planet around Kepler-90,* The American Astronomical Society, 2018.

[11] M. Chrsit, N. Braun, J. Neuffer and A. W. Kempa-Liehr, *Time Series FeatuRe Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package),* vol. 307, Neurocomputing, 2018, pp. 72-77.

[12] K. G. et. al. *Advances in Neural Information Processing Systems 30,* Curran Associates, 2017.

[13] L. Yu, A. Vanderburg, C. Huang, C. J. Shallue, I. J. M. Crossfield, B. S. Gaudi, T. Daylan, A. Dattilo, D. J. Armstrong, G. R. Ricker, R. K. Vanderspek, D. W. Latham, S. Seager, J. Dittmann and J. P. Doty, *Identifying Exoplanets with Deep Learning. III. Automated Triage and Vetting of TESS Candidates,* 1 ed., vol. 158, The Astronomical Journal, 2019.

[14] Y. Jin, L. Yang and C.-E. Chiang, *Identifying Exoplanets with Machine Learning Methods: a Preliminary Study,* International Journal on Cybernetics & Informatics, 2022.

[15] L. Ofman, A. Averbuch, A. Shliselberg, I. Benaun, D. Segev and A. Rissman, *Automated identification of transiting exoplanet candidates in NASA Transiting*

Exoplanets Survey Satellite (TESS) data with machine learning methods, vol. 91, New Astronomy, 2022.

[16] "ThetaRay, Inc.," [Online]. Available: https://www.thetaray.com/.

[17] G. Shabat, Y. Shmueli, Y. Aizenbu and A. Averbuch, *Randomized LU decomposition,* vol. 44, Applied and Computational Harmonic Analysis, 2018, pp. 246-272.

[18] "Mikulski Archive," Space Telescope Science Institute, NASA, [Online]. Available: https://archive.stsci.edu/.

[19] S. T. S. Institute, "Mikulski Archive," NASA, [Online]. Available: https://archive.stsci.edu/. [Accessed Junio 2020].

Código:

https://drive.google.com/file/d/1pCkNOSwsVKROEjhA0i-bUSr_lRdsEboT/view?usp=share_link

Trabajo de Fin de Máster

Machine Learning para la detección de exoplanetas: revisión y nuevos enfoques

Autor:
Javier Gómez de Diego

Tutora:
Laura Ruiz Dern

Universitat Oberta de Catalunya
Máster en Ciencia de Datos
Enero 2023