Universidad de Oviedo
Universidá d'Uviéu
University of Oviedo

Estudio con aprendizaje automático de la certificación de datos físicos del detector CMS del LHC (CERN)

Machine learning study of the physics data certification for the CMS detector at the LHC (CERN)

Ana Iglesias Rodríguez

Dirigido por Javier Fernández Menéndez
y Andrea Trapote Fernández
UNIVERSIDAD DE OVIEDO
Facultad de Ciencias
Grado en Física

Julio de 2021

Agradecimientos

En primer lugar, gracias a mi madre, a mi padre y a mi hermano por el apoyo durante la carrera y la paciencia en cada paso. En especial, gracias a mis padres por dejarme total libertad para escoger mi camino y no presionarme en momentos innecesarios. Aunque este año ha sido difícil, confiemos en que todo irá a mejor.

Gracias a David, porque tú siempre serás familia.

Por último, gracias a mis tutores Javier y Andrea. Gracias no sólo por estar siempre dispuestos a ayudarme con mis dudas, si no por descubrirme el mundo del Machine Learning. Habéis sido, sin quererlo, determinantes en mi futuro más próximo.

Resumen

El volumen de datos tomados en los experimentos en LHC (CERN) durante sus periodos activos es enorme. Tras tomarlos, todos estos datos son procesados con el objetivo de investigar diferentes aspectos de la física de partículas. En cambio, el primer filtro que deben pasar es un control de calidad, que determina si cada conjunto de datos es apto para la investigación posterior o tiene algún defecto.

En este trabajo, analizamos la implementación de técnicas de aprendizaje automático para este primer control. En concreto, buscamos el mejor modelo para datos tomados en CMS entre 2015 y 2018. El análisis se realiza a partir de las distribuciones estadísticas de diferentes magnitudes físicas registradas para muones en CMS durante este periodo.

La estructura del trabajo se divide en una parte teórica y una práctica. En el primer capítulo introducimos conceptos básicos asociados a la física de aceleradores de partículas. En el segundo, se presenta el proyecto CMS, así como el acelerador LHC. En el tercero, hablamos del grupo que se encarga actualmente de la certificación que buscamos replicar: el DQM y, los capítulos cuarto y quinto, están enfocados al aprendizaje automático. En concreto, en el cuarto se exponen todos los modelos y conceptos de forma teórica. Y, por último, en el quinto se expone el análisis con aprendizaje automático realizado sobre los datos de muones.

Índice general

Capítulo 1

Magnitudes y conceptos básicos

Este capítulo se presenta en forma de introducción para dar a conocer algunos conceptos importantes relacionados con el cuerpo del trabajo. Trataremos brevemente el Modelo Estándar (ME), así como las magnitudes asociadas a la física de aceleradores de partículas.

A no ser que se indique lo contrario, toda la información ha sido extraída del libro [3] *Particles and Nuclei* de B.Povh.

1.1. Modelo Estándar de la física de partículas

[1] Teoría desarrollada en torno a 1970 que permite explicar el conocimiento actual en física de partículas. En el ME, el Universo se puede explicar a partir de dos componentes básicos: las partículas elementales y las fuerzas o interacciones que las relacionan entre sí.

1.1.1. Partículas elementales

En la figura 1.1 podemos observar las partículas elementales, que se dividen en tres grandes grupos: los leptones, los quarks y los bosones. Tanto los leptones como los quarks tienen spin fraccionario e igual a 1/2, por lo

que son fermiones, mientras que los bosones tienen spin entero.

- **Quarks:** Hay seis distintos, agrupados por parejas en lo que conocemos como generaciones. La masa aumenta al subir de generación y los quarks más estables son los de la primera. En la primera fila de quarks en la figura 1.1 todos tienen carga $2/3e$, mientras que los de la segunda tienen carga $-1/3e$, donde e representa la carga fundamental. Otra propiedad conocida de los quarks es el color y solo forman combinaciones entre ellos que den un resultado de color neutro. Asociado a cada quark hay un antiquark. Estos tienen las mismas características que los quarks, a excepción de la carga eléctrica y el color, que son los opuestos. Como tienen carga eléctrica experimentan la interacción electromagnética y, por tener la propiedad de color, se ven afectados por la interacción fuerte. Además, también sufren la interacción electrodébil.

- **Leptones:** También se agrupan en tres generaciones. En cada una de ellas está un leptón con carga $-e$ y su neutrino correspondiente. El neutrino no tiene carga eléctrica y, a día de hoy, no se conoce el valor de su masa pero sí una cota superior que nos indica que son partículas sin masa o con una masa muy ligera. Los leptones no tienen carga de color pero sí tienen asociada una cantidad que se conoce como número leptónico y este es -1 para los que aparecen en la figura 1.1. Además, existen otras seis partículas análogas a los fermiones de la figura con carga opuesta y número leptónico $+1$. Estas son sus antipartículas. Por tener carga eléctrica experimentan interacción electrodébil, pero no se ven afectados por la fuerte pues no tienen carga de color. El leptón más conocido probablemente sea el electrón. En cambio, en este trabajo, tomará mayor relevancia el muón. Los muones [7] son leptones de la segunda generación, muy masivos en comparación al electrón ($105.6583745 \pm 0.0000024\ MeV/c^2$ frente a

$0.5109989461 \pm 0.0000000031\ MeV/c^2$) y con una vida media muy corta ($\tau = (2.1969811 \pm 0.0000022)e^{-06}s^{-1}$). Son leptones muy importantes en los estudios realizados en CMS, porque se les espera en la desintegración de potenciales partículas presentes en nueva física o física más allá del ME.

- **Bosones:** Los que aparecen en la imagen son las partículas portadoras de las interacciones fundamentales. Además de estos, está el bosón de Higgs, que fue descubierto recientemente y es el culpable de dotar de masa a las partículas a través de su interacción con el campo de Higgs. En este último caso el spin es nulo, mientras que para el resto es 1.

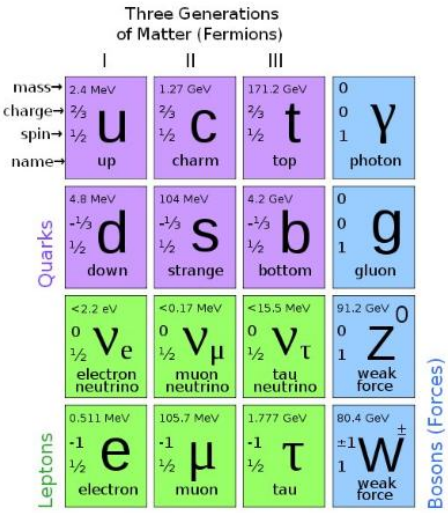


Figura 1.1: Partículas elementales del ME. Fuente: [4].

1.1.2. Interacciones fundamentales

Hay cuatro tipos: electromagnética, débil, fuerte y gravitatoria. Funcionan en distintos rangos y sus intensidades también varían. El ME es capaz de explicar las tres primeras, que se entienden a partir del intercambio entre partículas de los bosones portadores de la interacción. En cambio, nunca se

ha podido observar un hipotético "gravitón" que actúe como portador de la interacción gravitatoria. Aunque el ME no sea capaz de explicar la interacción gravitatoria, esta sólo domina a partir de los metros. En consecuencia, en física de partículas es una interacción despreciable.

- **Interacción electromagnética:** Es la interacción principal en los átomos y la responsable de su estructura, actúa sobre todo aquello que tenga carga eléctrica no nula y tiene un rango de actuación infinito. Su partícula mediadora son los fotones y se rige por la Ley de Coulomb. El potencial eléctrico cumple que

  $$V_e = \pm \frac{\alpha_{EM}}{R}$$

  donde el signo positivo corresponde al caso en el que las partículas tengan la misma carga (fuerza atractiva) y el negativo si las partículas tienen cargas opuestas (fuerza repulsiva), R es la distancia entre partículas y α_{EM} es la constante de estructura fina $\alpha_{EM} \approx \frac{1}{137}$.

- **Interacción fuerte**: Interacción responsable de la estructura de los núcleos atómicos y de la formación de hadrones. Tiene un rango de actuación de $\approx 1fm$ y su bosón portador es el gluón. Actúa sobre aquellas partículas que tengan carga de color, es decir, quarks y gluones. Su potencial viene dado por:

  $$V_S = -\frac{\alpha_S}{R} + KR$$

  donde $\alpha_S = 0.1179$ es una constante de acoplamiento, R la distancia entre las partículas y K otra constante.

- **Interacción débil**: Esta interacción no tiene asociada la formación de ninguna estructura en concreto pero sí la desintegración beta. Actúa a distancias menores del tamaño del protón, tiene como partículas portadoras al bosón Z y los bosones $W^{\pm}$ y es la responsable de la desintegración de los quarks o los leptones.

Aunque este modelo explique lo conocido hasta ahora en física de partículas, es una teoría incompleta que necesita las investigaciones del CERN. Un ejemplo de éxito relacionado con esta teoría fue el descubrimiento del bosón de Higgs que ratificó la existencia del campo de Higgs, que había sido teorizado anteriormente.

1.2. Equivalencia entre masa y energía.

En física de partículas es común utilizar unidades de energía para cantidades de masa o de momento lineal. Para entender esto tenemos que tener en cuenta las ecuaciones $E = mc^2$ y la de la energía relativista para fotones: $E = pc$. Además, debemos conocer el sistema de unidades empleado en el cual las constantes de velocidad de la luz en el vacío c y la constante de Planck reducida $\hbar$ ambas toman el valor unidad.

Por otro lado, las distancias se suelen medir en fermis (femtómetros, $1fm = 10^{-15}m$) y la energía en derivados de los electronvoltios, como GeV o TeV.

1.3. Sección eficaz

Es una de las magnitudes más importantes en la descripción de experimentos de dispersión. Se denota por σ y representa la referencia a la probabilidad de que el choque entre dos partículas ocurra. Analíticamente su expresión es (1.1), donde Γ es la tasa de reacciones por partícula incidente y Φ es el número de partículas incidentes por unidad de área y tiempo.

$$\Gamma = \Phi\sigma \tag{1.1}$$

Tiene unidades de área y su valor no depende del montaje experimental. En física de partículas se utiliza el barn como unidad: $1barn = 1b = 10^{-28}m^2$. Un valor típico para un haz de $10GeV$ de protones es $\sigma_{pp}(10GeV) \approx 40mb$.

1.4. Luminosidad

La luminosidad instantánea, denotada por L, proporciona junto a la sección eficaz σ, el número de choques por unidad de tiempo, denotado por N, a partir de : $N = \mathrm{L}\sigma$. En el LHC, la luminosidad [11] viene dada por:

$$\mathrm{L} = \frac{\gamma f k_B N_p^2}{4\pi\epsilon_n \beta^*} F$$

donde γ es el factor de Lorentz, f es la frecuencia de revolución, k_B el número de paquetes de protones, N_p el número de protones por paquetes, ϵ_n es la emitancia transversa normalizada (diseñado para que tome el valor $3.74\mu m$), β^* es la función de betatrón en el IP, y F el factor de reducción asociado al ángulo de cruce. Para conocer más parámetros asociados al LHC, consultar [11].

La energía nominal para cada haz de protones es de 7 TeV. Cada paquete de protones tiene $1.15e^{11}$ protones, que se enfocan en un área de $16\mu m \times 16\mu m$ en el punto donde colisionan. La probabilidad de que un protón del paquete que viene de la izquierda choque con uno de los que vienen de la derecha depende de su tamaño y de la sección eficaz del paquete (σ^2 con $\sigma = 16\mu m$). Teniendo en cuenta esto y, que solo aproximadamente la mitad de las interacciones son útiles, se tiene que cada cruce produce 20 colisiones útiles. Por último, como hay 11245 cruces por segundo, hay 600 millones de colisiones por segundo. Para ver estos cálculos con más detalle, consultar [12].

Derivado de la luminosidad instantánea tenemos la luminosidad integrada $\int \mathrm{L}dt$, que nos da el número de interacciones que se observan durante un periodo de tiempo concreto.

Capítulo 2

El colisionador LHC (CERN)

Para entender qué tan importante o útil puede ser la automatización del procesamiento de datos en el CERN, primero tenemos que conocer qué son estos datos, cómo y dónde se toman y cuál es el volumen de datos que se producen durante los periodos de funcionamiento (Runs) de los aceleradores. Nuestro análisis se realizará con datos obtenidos en el experimento Compact Muon Solenoid (CMS) situado en el acelerador LHC. En este capítulo se presenta una introducción básica sobre el acelerador para entender el LHC.

2.1. Estructura del LHC

[2] La estructura experimental del CERN consiste en una sucesión de aceleradores en los que las partículas se aceleran de forma progresiva. En cada paso de esta cadena, los haces de partículas van adquiriendo más y más energía. El último acelerador se conoce como Large Hadron Collider (LHC) y en él las partículas pueden ser aceleradas hasta $7TeV$, pudiendo producir choques de $14TeV$ en centro de masas.

El LHC es un acelerador circular de 27 km de circunferencia, instalado en el túnel subterráneo preexistente que había sido construido para el LEP, un acelerador previo al LHC. Los hadrones, en concreto protones o iones pesados, viajan a lo largo del acelerador en dos tubos: en uno de ellos circulan en sentido horario y en el otro, en antihorario. Estos tubos se juntan en

cuatro puntos del acelerador, donde se sitúan diferentes detectores y en torno a los cuales se desarrollan los principales experimentos del CERN.

En el LHC no sólo se aceleran protones, si no que también se realizan experimentos con átomos de plomo (Pb) o átomos pesados. En cambio, como los protones son los hadrones más relevantes en sus experimentos, vamos a ver cuál es el camino de los mismos hasta llegar a cualquiera de los detectores situados en el LHC. En la imagen 2.1 podemos ver un esquema de los aceleradores y, resaltado en morado, el camino que vamos a describir a continuación. Una explicación similar para los átomos pesados se puede encontrar en la referencia [2] de la bibliografía.

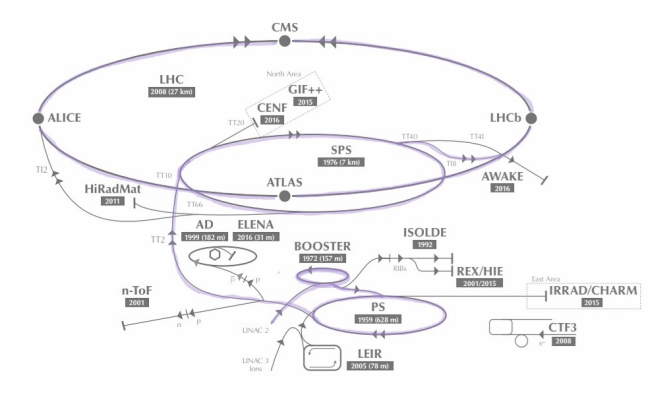


Figura 2.1: Esquema del sistema de aceleradores del CERN, en el que se remarca el camino que sigue un protón para ser acelerado. Fuente: [2] con modificaciones propias.

La fuente de protones son átomos de hidrógeno a los que se les arranca el electrón. Estos son inyectados al Proton Synchrotron Booster (PSB) desde el acelerador lineal Linac2 con una energía de 50 MeV. Una vez aquí, se aceleran hasta 1.4 GeV para enviarlos al Proton Synchotron (PS), en el que aumentan su energía hasta 25 GeV. El último paso antes del LHC es el Super Proton Synchotron (SPS) donde alcanzarán los 450 GeV. Al LHC son transferidos en dos corrientes diferentes que circulan en sentidos opuestos y

en donde se aceleran a 7 TeV. Una vez aquí, las corrientes de protones se pueden pasar horas girando en el acelerador antes de colisionar en alguno de los cuatro puntos, que también se observan en la figura 2.1. Es importante destacar que los protones vienen empaquetados en lo que conocemos como bunches o paquetes de protones.

Cada vez que entran protones desde el SPS al LHC conocemos ese conjunto de bunches como un fill. Estos se mantienen en el LHC para que colisionen durante un periodo de tiempo y, después, se reciclan inyectando un nuevo fill.

2.2. Periodos de funcionamiento

El proyecto del LHC surge en la época de 1980 en el CERN, pero su construcción no fue aprobada por el consejo del CERN hasta diciembre de 1994. Los proyectos ATLAS, ALICE, CMS y LHCb son presentados y aceptados entre 1996 y 1998. A partir de aquí y desde el final de la construcción se han ido intercalando periodos de funcionamiento con otros en los que el acelerador está parado, que se aprovechan para realizar mejoras sobre el mismo. Cuando el acelerador está en funcionamiento llamamos a estos periodos Runs y, cuando no lo está, decimos que estamos en un Long Shutdown.

En concreto, desde el comienzo del proyecto ha habido dos Runs completos y dos Long Shutdowns. A principios del año que viene (2022) se prevé que se ponga en marcha el Run 3, que durará hasta 2024. Durante este Run se afinarán y comprobarán las técnicas de aprendizaje automático para certificación de datos. Tras este periodo, el Long Shutdown 3 se prevé que se alargue entre 2025 y mediados del 2027, y durante el mismo se espera una mejora sustancial del LHC para convertirlo en su sucesor: el High Luminosity LHC. En resumen, las fechas relevantes respecto a los Runs en el LHC son las siguientes:

- 2009- Febrero 2013: Run 1 con $\sqrt{(s)} = 7 - 8TeV$.
- 2013-2014: Long Shutdown 1
- Principios 2015- Diciembre 2018: Run 2 con $\sqrt{(s)} = 13TeV$.
- 2019-2021: Long Shutdown 2
- 2022-2024: Run 3 con $\sqrt{s} = 13 - 14TeV$.
- 2025-mediados 2027: Long Shutdown 3. En este periodo se espera una mejora para convertir al LHC en su sucesor: el High Luminosity LHC.

En la figura 2.2 podemos ver la evolución de la luminosidad integrada desde 2010 hasta 2018. Se observa claramente una mejora en la luminosidad integrada total con el paso del tiempo. Además de ello, se indica cuál era la energía en centro de masa disponible ($\sqrt{s}$) para cada periodo. Podemos ver como, tras el LS1, las condiciones del LHC mejoraron sustancialmente.

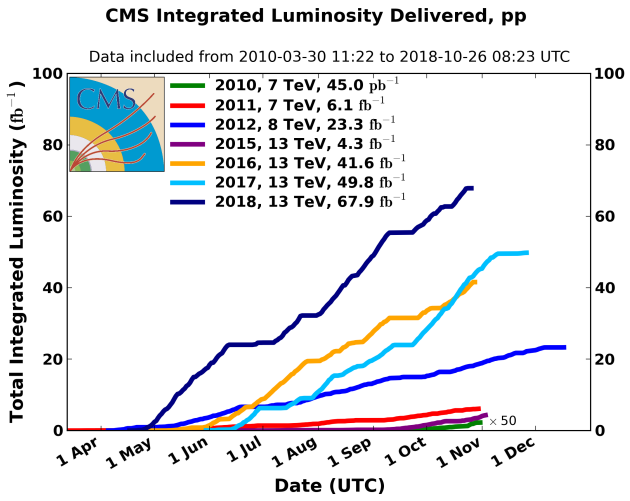


Figura 2.2: Luminosidad integrada total desde 2010 hasta 2018. Fuente: [5].

En relación al Run 3, tenemos que la luminosidad instantánea será de $2 \cdot 10^{34} cm^{-2} s^{-1}$, lo que implica en promedio unos $70 fb^{-1}$ al año frente a los $68 fb^{-1}$ que, como máximo, se alcanzaron en 2018. Este último dato

lo podemos ver en la figura 2.2. Por otro lado, el High Luminosity LHC (HL-LHC) supone una mejora de un orden de magnitud en el número de colisiones por segundo, alcanzando una luminosidad integrada de más de $250 fb^{-1}$ por año durante los Runs 4 y 5, en comparación a los $70 fb^{-1}$ que se esperan alcanzar durante el Run 3. Un esquema de las fechas de cada Run, su energía en centro de masas y su luminosidad integrada se puede ver en la figura 2.3, donde también se indica la evolución del HL-LHC y las expectativas futuras.

Figura 2.3: Evolución del LHC y del proyecto HL-LHC. Fuente: [6].

2.3. Experimentos en el LHC

En el LHC hay cuatro experimentos principales en torno a los puntos de colisión: ATLAS, ALICE, CMS y LHCb. Además, hay otros más pequeños instalados cerca de los anteriores. Un resumen breve de los objetivos de los mismos es el siguiente.

- ALICE (A Large Ion Collider Experiment): Detector especializado en medir y analizar colisiones de iones de plomo. Estudia propiedades del

plasma de quarks y gluones.

- ATLAS (A Toroidal LHC ApparatuS): Detector de propósito general construido para abarcar un amplio espectro de investigaciones en la física de partículas, como pueden ser medidas de precisión sobre el bosón de Higgs o búsqueda de física más allá del Modelo Estándar
- CMS (Compact Muon Solenoid): Otro detector de propósito general similar a ATLAS. Es el experimento del que proceden nuestros datos y, por ello, se explicará con mayor detalle en la siguiente sección.
- LHCb (Large Hadron Collider beauty experiment): Experimento especializado en el estudio de la asimetría entre la materia y la antimateria en colisiones de partículas que contengan el quark b (B-particles).

Alrededor de estos cuatro proyectos principales estarían:

- LHCf (Large Hadron Collider forward): Busca contrastar modelos relacionados con los rayos cósmicos de alta energía. Para ello, analiza las partículas que se generan muy cerca de la dirección de los haces de partículas.
- MOEDAL (Monopole and Exotic Detector At the LHC): Experimento basado en la búsqueda de partículas hipotéticas super ionizantes, como monopolos magnéticos.
- TOTEM (TOTal Elastic and difracctive cross section Measurement): Mide el tamaño efectivo (sección eficaz) de los protones en el LHC.

2.4. El proyecto CMS: detector y objetivos

El experimento CMS consiste en un detector de propósito general útil para explorar nueva física a altas energías. Se construyó con el objetivo de estudiar diferentes mecanismos relacionados con el Modelo Estándar.

[9] En él participan hasta 5000 físicos de partículas, ingenieros, técnicos, estudiantes y personal asociado de 200 instituciones de hasta 50 países (datos de 2019), haciendo de CMS una de las colaboraciones científicas más grandes de la historia.

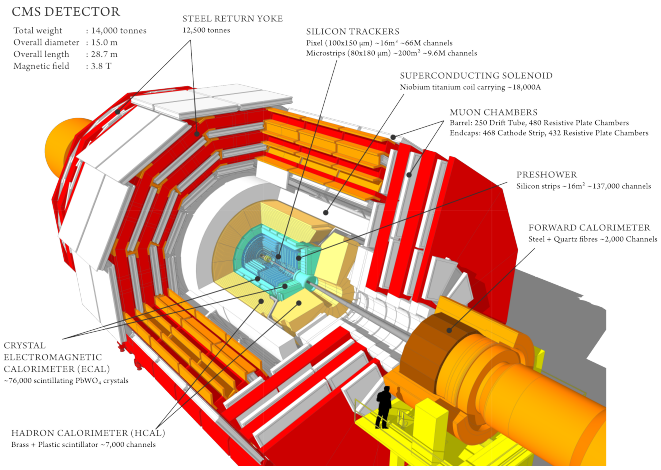


Figura 2.4: Esquema del detector de CMS. Fuente: [13].

[11] El detector mide 22 metros de largo, 15 metros de diámetro y pesa 12.500 toneladas. En la figura 2.4 podemos ver un esquema del mismo, donde se aprecia la simetría cilíndrica del detector. Una de las piezas más importantes, y la que le da nombre, es el solenoide superconductor. Este es capaz de generar un campo magnético de $4T$, que es necesario para poder modificar la trayectoria de las partículas que se producen en las colisiones. Poder de curvar la trayectoria de las partículas cargadas es importante para poder medir con precisión el momento de las mismas. Este imán mide $13m$ de largo y $5.9m$ de diámetro. Dentro del mismo están los detectores de trazas y los calorímetros electromagnético y hadrónico. Rodeándolo están las cámaras de muones. Hablaremos a continuación brevemente de estos cuatro subdetectores de CMS, así como de magnitudes interesantes relativas a las medidas tomadas en este detector.

2.4.1. Magnitudes relativas a las medidas

- **Granularidad de los datos**: Los datos con los que vamos a trabajar pueden estudiarse con dos granularidades diferentes. El primer caso es estudiar los datos en lumisections (LS) y, el segundo, en runs. Cada run se divide en periodos más pequeños, de aproximadamente 23 segundos, conocidos como LS, en los que se consigue que la luminosidad sea aproximadamente constante.

  Es importante notar que, los runs de los que hablamos a partir de aquí no corresponden al Run 1 ó 2 que mencionamos antes. En este caso, son los periodos de tiempo sobre los que tenemos comentarios por parte de los expertos, como veremos en el capítulo 5. De este modo y, en relación a lo que vimos en secciones anteriores, cada Run se compone por un conjunto de fills. En cada fill se realizan una serie de colisiones y, se conoce como run, a un intervalo dentro del fill con condiciones del detector CMS constantes en el que se toman datos. A su vez, estos runs se dividen en LS.

- **Ejes de coordenadas**: [11] El sistema de coordenadas toma como referencia el punto de colisión, situado dentro del detector. El eje vertical será nuestro eje Y, el eje Z corresponde a la dirección en la que viaja el haz de partículas y el eje X el perpendicular a los dos anteriores. En la figura 2.5 podemos ver lo anterior, con una vista frontal y una lateral de un esquema del detector. Asociados a las medidas tenemos también dos ángulos: el azimutal, ϕ , medido en el plano XY desde el eje X y el polar, θ , medido desde el eje Z.

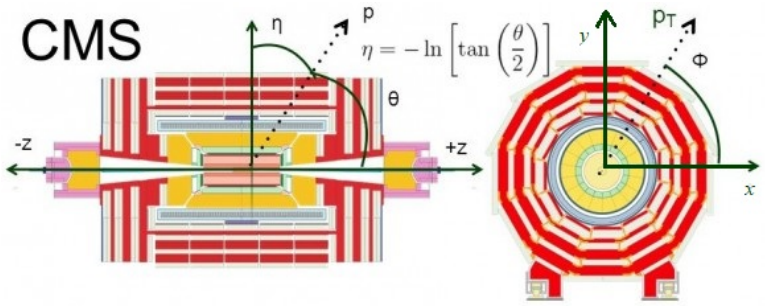


Figura 2.5: Esquema del sistema de referencia empleado en las medidas de datos. Fuente: [8].

- **Pseudorapidez**: Medida que nos indica qué tan distante está la partícula respecto al plano transverso. La expresión analítica de la misma es

$$\eta = -\ln\left[\tan\left(\frac{\theta}{2}\right)\right]$$

En la figura 2.6 podemos observar un esquema de un sector del detector en el que se representan las direcciones asociadas a distintos valores de η . Las etiquetas presentes en la figura corresponden a los distintos subdetectores, de los que hablaremos en la siguiente sección.

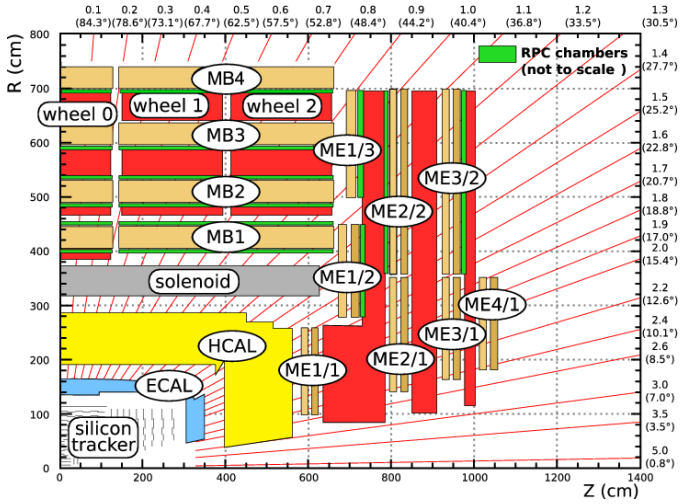


Figura 2.6: Vista RZ de un sector del detector CMS. Fuente: [21].

- **Cuadrimomento**: El cuadrimomento de una partícula es un vector en el que se detalla su energía según:

$$p = \left(\frac{E}{c}, p_x, p_y, p_z\right)$$

Considerando el momento en el plano transverso p_T dado por $p_T = \sqrt{p_x^2 + p_y^2}$, el cuadrimomento anterior se puede expresar también como:

$$p = \left(\frac{E}{c}, p_T, \theta, \phi\right)$$

Análogamente al momento transverso p_T, también podemos considerar la energía en el plano transverso E_T.

2.4.2. Subdetectores de CMS

El detector del experimento CMS tiene forma cilíndrica y de forma habitual se emplean las palabras Barrel y Endcap para referirse a la parte asociada al cuerpo del detector (zona barril) y a los tapones, respectivamente. En esta sección hablaremos de los cuatro subdetectores más importantes de CMS y de sus partes.

Tracker o Detector de trazas

[11] [14] 75 millones de sensores de silicio (material resistente a la radiación) electrónicos componen esta primera capa del detector. El Tracker es la componente encargada de medir las trayectorias de las partículas cargadas que lo atraviesan, siendo capaz de alcanzar una precisión de $10\mu m$. Tiene un diámetro de $2.6m$ y una longitud de $5.8m$. A partir de estas trayectorias, es posible calcular el momento de las partículas.

El Tracker se divide en tres regiones en función del tamaño de los detectores. La primera es la zona más cercana al punto de colisión ($r \approx 10\ cm$) y es a la que llega el mayor flujo de partículas. En ella los detectores son muy pequeños, $100 \times 150\mu m^2$, para asegurar una buena resolución. La segunda región es la intermedia $10cm < r < 55cm$ y, en ella, los detectores son microcélulas de silicio de un tamaño aproximado de $10cm \times 80\mu m$. Por último, la región más externa ($r > 55\ cm$) es similar a la anterior, excepto que el tamaño de sus células es mayor: de $25\ cm \times 180\mu m$.

Por otro lado, la zona barril también se divide en dos niveles: **Tracker Inner Barrel (TIB)**, formada por 4 capas de detectores situados en la región $\mid z \mid < 65\ cm$ y **Tracker Outer Barrel (TOB)**, formada por 6 capas de detectores de silicio y cubriendo la zona $\mid z \mid < 110\ cm$. Es importante evitar zonas sin detector, por lo que la zona externa (TOB) es ligeramente más larga que la interna (TIB). Con el mismo objetivo de evitar puntos muertos hay tres discos en la zona de transición entre la zona cilíndrica y

los tapones.

Análogamente, en la zona de los tapones se consideran también dos regiones diferenciadas: el **Tracker End Cap (TEC)** formado por 9 discos y cubriendo 120 $cm <| z |< 280$ cm y los **Tracker Inner Disks (TID)** que son los tres discos que rellenan la zona entre TIB y TEC.

En la figura 2.7 se puede observar un esquema de los componentes del Tracker en una región del mismo.

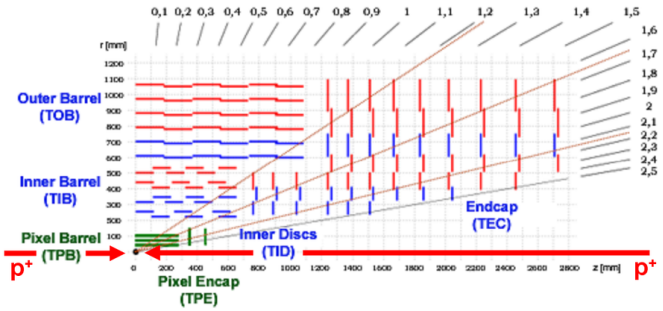


Figura 2.7: Esquema de un sector del Tracker en el que se detallan sus diferentes componentes. Fuente: [22].

Calorímetro electromagnético (ECAL)

[11][15] ECAL es un calorímetro formado por cristales de tugnstenato de plomo ($PbWO_4$), divididos en una parte central, denominada Barrel Section (EB), y dos tapones o Endcaps (EE).

La región cilíndrica (EB, de **ECAL Barrel**) tiene un radio interno de $1.29m$ y está constituída por 36 módulos idénticos. Cubre señales en un rango de $0 <| \eta |< 1.479$ en pseudorapidez. Por otro lado, los **tapones (EE)** están a $3.14m$ del punto de colisión y cubren un rango en pseudorapidez de $1.479 <| \eta |< 3.0$. La zona EE está precedida por un sistema de **Preshower (ES)** formado por tiras de silicio que generan fotones cuando un electrón o positrón les atraviesa. En la figura 2.8 podemos ver un esquema del ECAL.

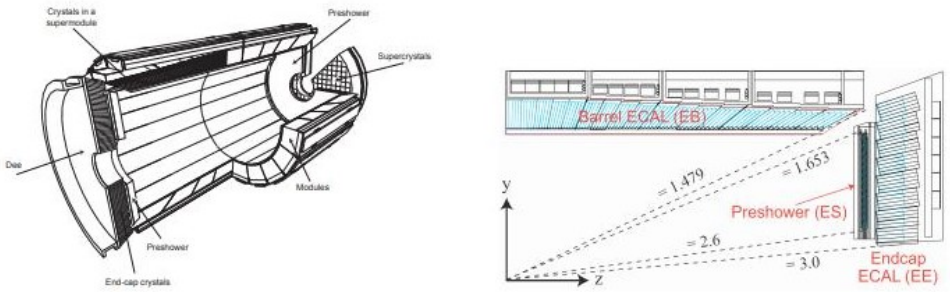


Figura 2.8: Esquema del ECAL de CMS. A la izquierda: Disposición de los EE, EB, y ES. A la derecha: Pseudorapidez en un sector del ECAL. Fuente: [19].

Para detectar las partículas, los cristales de $PbWO_4$ están conectados a fotodiodos de avalancha de silicio (APDs) en la zona central y a fototriodos de vacío (VPTs) en los tapones. Cuando un electrón o un fotón pasan por el calorímetro electromagnético producen una señal luminosa, proporcional a la energía de la partícula, que es registrada por el detector.

Calorímetro de hadrones (HCAL)

[11] [16] Los hadrones, formados por quarks y gluones, son capaces de atravesar el ECAL y se detienen en la capa externa del HCAL. El calorímetro hadrónico mide la energía de estas partículas y proporciona, de forma indirecta, la presencia de partículas sin carga que no interactúan, como los neutrinos.

El HCAL es un calorímetro de muestras: detecta la posición, la energía y el tiempo en el que llega la partícula. Al igual que el ECAL, el HCAL se divide en dos secciones cilíndricas: una dentro del diámetro del solenoide (barrel (HB)) y otra externa (outer(HO)) y dos tapones (Endcap (HE)) y fordward sections (HF)), como podemos observar en la imagen 2.9.

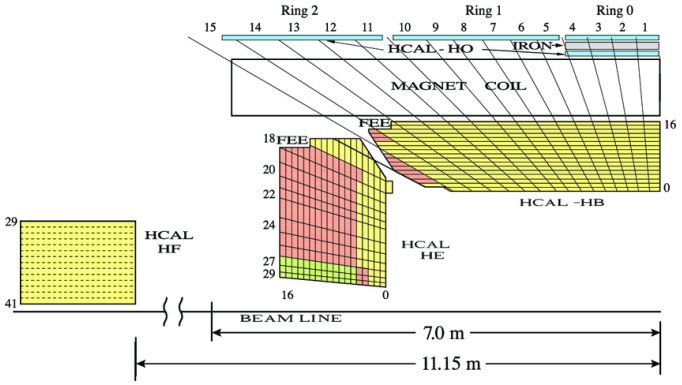


Figura 2.9: Esquema del HCAL donde se aprecian de forma diferenciada sus partes: HB, HO, HF y HE. Fuente: [20].

El **Hadron Barrel (HB)** está formado por 15 tiras de latón que absorben las partículas y dos placas externas de acero inoxidable. Tras salir del ECAL, las partículas chocan contra unas placas centelleadoras de unos $9mm$ de grosor. Cubre un rango en pseudorapidez de $\mid \eta \mid < 1.4$.

El **Hadron Outer (HO)** se encuentra en la parte externa del solenoide, en la región $\mid \eta \mid < 1.26$. Está dividido en 5 secciones, de $2.5m$ cada una, a lo largo de η numeradas del -2 al 2. La capa 0 tiene dos capas de material centelleador separadas por una capa absorbente de $18cm$ de hierro. Proporciona la energía de los hadrones o quarks que son capaces de escapar las primeras capas del HCAL, lo que mejora sustancialmente la resolución de la energía transversa perdida (E_T^{Miss}).

El **Hadron Endcap (HE)** cubre la región $1.3 < \eta < 3.0$. [20] Está formado por discos de latón, intercalados con material centelleador que cubren 20° en ϕ, divididos en 5 sectores.

Por último, el **Hadron Fordward (HF)** está fabricado en fibra de cuarzo y hierro. Se sitúa a $11.2m$ del punto de colisión y tiene un espesor de $1.65m$. [20] Está separado en dos capas diferentes. En la primera, formada por las fibras más largas de cuarzo, se mide toda la radiación. Las capas se separan entre sí con acero. Esta disposición permite la separación de casca-

das generadas por electrones y fotones de aquellas generadas por hadrones. Los fotodetectores usados en el HF son fotomultiplicadores de ocho etapas (PMT).

Cámaras de muones

[11][18] Una de las misiones principales de CMS es detectar correctamente la presencia de los muones, pues se les espera en la desintegración de potenciales nuevas partículas. Al ser partículas cargadas, son detectadas en primer lugar en el detector de trazas. Tras salir del mismo, atraviesan el resto del detector y son frenados en lo que se conoce como las cámaras de muones: la capa más externa del detector.

Por lo tanto, los muones se manifiestan en dos subdetectores: en el detector del trazas y en la cámara de muones. De hecho, en función del momento transverso del muón, la medida de los mismos tiene mejor o peor precisión en un componente u otro del detector. Como consecuencia de esto aparecen dos términos nuevos: los muones del Tracker y los muones globales.

- **Muones del Tracker**: Cuando el momento transverso de un muón es bajo, algunos se dispersan antes de llegar a las primeras cámaras de muones. Por ello, la mejor resolución es la que nos proporciona la trayectoria del Tracker en función de la curvatura de la misma. Los muones con una señal en el Tracker de $p_T > 0.5GeV$ y $p > 2.5GeV$ y que corresponden, al menos, con una señal en alguna de las cámaras de muones los conocemos como **muones del Tracker**.

- **Muones globales**: En este caso la señal en las cámaras es clara y es posible reconstruir la trayectoria del muón desde el Tracker.

En resumen, la diferencia entre los dos tipos de muones es la intensidad de la señal que dejan en las cámaras de muones: alta para los muones globales y baja para los del Tracker. En nuestro caso, emplearemos datos asociados

a muones globales.

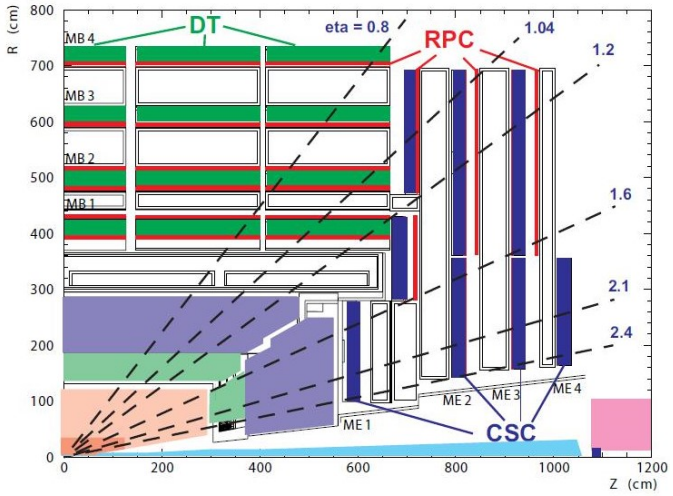


Figura 2.10: Esquema de un sector del detector en el que se detalla la posición de las RPCs, las CSCs y las DTs. Fuente: [11].

Debido a las diferentes condiciones presentes en el detector hay tres tipos de cámaras de muones, las cuales podemos observar en la figura 2.10. En la región cilíndrica ($|\eta| < 1.2$) hay un campo magnético residual bajo y una baja tasa de muones. Por lo tanto, los detectores que aquí se colocan son **Drift Tubes (DT)**. En la zona de los tapones ($1.2 < |\eta| < 2.4$), en cambio, hay un alto campo magnético residual así como una alta tasa de muones. Los detectores de esta zona se conocen como **Cathode Strip Chambers (CSC)**. Por último, en ambas regiones hay otro tipo de detectores que proporcionan una respuesta rápida con buena resolución temporal de la presencia de muones que se conocen como **Resistive Plate Chambers (RPC)**. A partir de los datos de las RPCs se puede decidir si los datos se registran o no. Las DTs y las CSCs funcionan de forma independiente de las RPCs, proporcionando fuentes de información complementarias.

En total hay 1400 cámaras de muones que se dividen en 250 DTs, 540 CSCs y 610 RPCs. En su conjunto proporcionan una cobertura total para

la toma de datos asociados a muones en $\mid \eta \mid < 2.4$.

2.4.3. Detección de las diferentes partículas en CMS

La figura 2.11 representa los subdetectores que acabamos de describir y esquematiza hasta dónde son capaces de llegar cada tipo de partícula detectada.

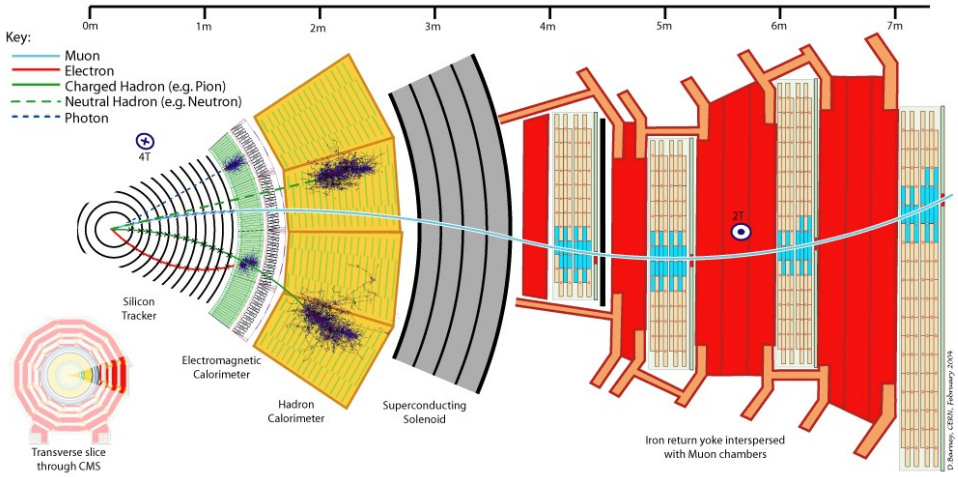


Figura 2.11: Detalle del alcance máximo de diferentes tipos de partículas en el detector CMS. Fuente: [10].

Para empezar, los **muones** se diferencian del resto por ser las únicas partículas detectadas capaces de llegar a las cámaras de muones. Por ello, su detección se basa en señales en estas cámaras acompañadas de una señal en el Tracker. Como hemos visto en la sección anterior, hay dos tipos de muones. En nuestro caso, emplearemos datos asociados a muones globales y, por ello, en la figura 2.12 se puede ver un ejemplo de la trayectoria de un muón global.

Los **hadrones** se detectan a partir de deposiciones en el calorímetro hadrónico. Para diferenciar aquellos hadrones neutros de los cargados, tenemos que fijarnos en si hay algún rastro en el detector de trazas, pues sólo las partículas cargadas interaccionan con las placas de silicio del Tracker.

Los **electrones** dejarán una deposición en el calorímetro electromagnético así como una señal en el detector de trazas, por ser partículas cargadas. En el caso de tener una deposición en el calorímetro electromagnético sin una traza asociada, la señal corresponde a un **fotón**.

Por último, hay otro tipo de partículas que no se detectan directamente a partir de deposiciones en los subdetectores. Los **neutrinos** están asociados a una falta en la energía transversa que se calcula a partir de la conservación del momento lineal en el plano transverso. En el momento de la colisión se considera que $p_T(t=0) = 0$ pues $p_z >> p_x, p_y$ así que la suma vectorial de los momentos tras la colisión ha de ser nula. En caso de no serlo, aquella energía necesaria para que lo sea es la energía transversa faltante.

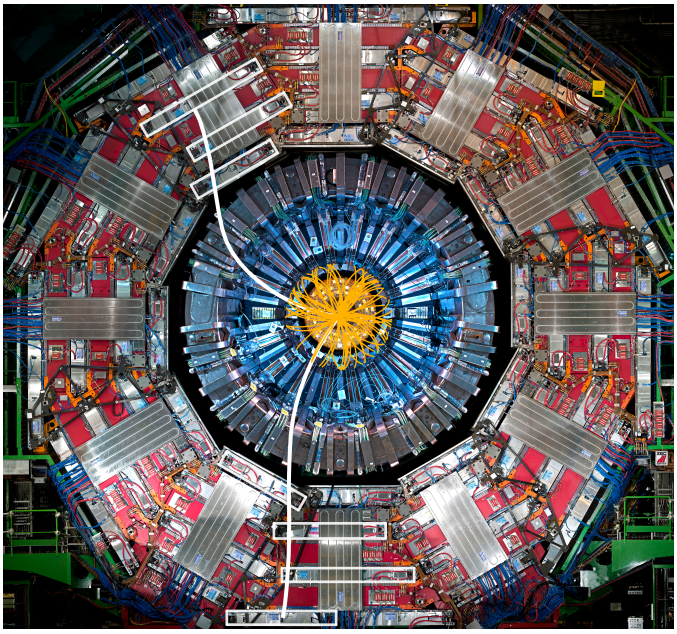


Figura 2.12: Detalle, en blanco, de la reconstrucción de la trayectoria asociada a un muón global en el detector de CMS. En amarillo aparecen las trazas en el Tracker. Fuente: [17].

Capítulo 3

Monitorización y certificación de datos en CMS

Durante los periodos activos del LHC se producen gran cantidad de datos por segundo. Por ejemplo, en experimentos como ATLAS o CMS se puede producir hasta 1GB por segundo. Para, entre otras cosas, clasificar, revisar y presentar a otros expertos de CMS estos datos está el **DQM** o **Data Quality Monitoring group**.

3.1. Data Quality Monitoring group (DQM)

[23] El DQM tiene tres propósitos principales:

- Proporcionar una monitorización del detector y avisar en el caso de que alguna parte del mismo tenga algún problema. Esta tarea se realiza en tiempo real y puede aceptar una pequeña tasa de error.
- Certificar los datos recogidos teniendo en cuenta en qué fracción de los mismos había algún problema en el detector. En este caso, los resultados no tienen que ser inmediatos si no que se puede permitir un pequeño retraso en los mismos, pero se busca una tasa de error mínima.

- Depuración de los programas (debugging) que proporcionan a los expertos información detallada.

3.1.1. La estructura del DQM

[24] En la figura 3.1 podemos observar un esquema de la estructura del DQM, donde los componentes de software se resaltan en gris. El DQM actúa como parte del software de CMS (CMSSW) en gran parte de su estructura. Esencialmente podríamos diferenciar dos partes: la online y la offline, y destacar el DQMGUI como elemento central del DQM.

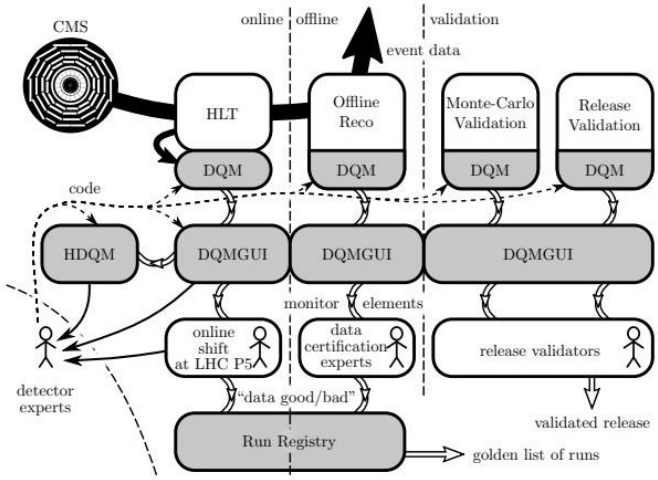


Figura 3.1: Esquema de la estructura del DQM. Fuente: [24].

El sector online recibe datos y genera histogramas con una frecuencia de 10-15Hz. Constituye una parte esencial en el preprocesamiento de los datos. Por otro lado, en la sección offline se realizan numerosas tareas que podríamos dividir en dos etapas comunes:

1. Se generan los elementos de monitorización (monitor elements) y se les agrega la información de la base de datos de CMS.

2. Se extraen los histogramas y se suman todos los correspondientes al

mismo run, para obtener así una información estadística completa del mismo.

Los histogramas finales se comparan con otros almacenados en el HDQM (historic DQM) o con histogramas de referencia. En el HDQM se pueden ver tendencias globales de elementos de monitorización sobre muchos runs en un año de datos, así como los diferentes observables monitorizados por el DQM.

El DQMGUI es el principal elemento del DQM. Es una interfaz de usuario gráfica (Graphical User Interface) que proporciona acceso a todos los histogramas de CMS. Es capaz de garantizar una visualización de todas las necesidades del DQM para todos los subsistemas, tanto para datos tomados en el momento como para aquellos almacenados en el HDQM.

3.1.2. Problemas afrontados por el DQM hasta el Run 2

El DQM ha proporcionado un buen servicio durante el Run 2. En cambio, en algunas ocasiones fue necesaria la intervención manual en los códigos del CMSSW para resolver algunos problemas. De cara al Run 3, uno de los objetivos es depurar estos códigos y tener localizados los cambios realizados.

Otro de los principales problemas que ha surgido con el paso del tiempo es cómo almacenar la cantidad de datos tomada. Para hacernos a la idea, el sistema offline DQMGUI recoge datos de $3.8 \cdot 10^{10}$ eventos procesados. Todos ellos están almacenados en una sola base de datos de $4.1TB$. En el caso del online DQMGUI, se almacena una información más reducida, siendo la base de datos de $650GB$. Obviamente, con el paso del tiempo y en periodos de funcionamiento la cantidad de datos a almacenar crece de manera considerable, haciendo de la memoria necesaria un problema a abordar.

3.1.3. Expectativas futuras

El sistema de DQM funciona bien. En cambio, se busca una mejora de su actuación de cara a nuevos periodos de toma de datos. En concreto, los cambios planteados están enfocados a mejorar pequeños detalles en los que se ha necesitado intervenir de forma manual durante los Runs 1 y 2. Por ejemplo, uno de los proyectos a largo plazo es la depuración del código empleado en el CMSSW.

Otro de los proyectos activos es la mejora de la certificación de datos en torno al registro de runs. Esta se plantea a partir de un nuevo diseño del registro, así como la introducción de nuevas herramientas de certificación, como técnicas de aprendizaje automático.

Además, se busca ampliar y mejorar las herramientas ahora disponibles y, por supuesto, garantizar que la infraestructura de DQM sea capaz de soportar el volumen de datos a tratar.

3.2. Certificación actual de datos en CMS.

Ahora mismo, la certificación de datos en CMS se basa en la opinión de expertos. Para clasificar los datos como buenos o malos, estos especialistas comparan visualmente las distribuciones conseguidas con unas predefinidas. Tras estas decisiones, los datos quedan clasificados en dos clases: buenos y malos. Los buenos son aquellos que podrían ser útiles para realizar análisis físicos. Por otro lado, los malos son aquellos que han sido tomados cuando alguna parte del detector presentaba un problema y no sirven para posteriores análisis.

Como veremos en el capítulo 5, tendremos una tercera clase. Los datos de la misma serán clasificados como no etiquetados y simplemente no han sido revisados por los expertos por las condiciones en las que han sido tomados. Por ejemplo, son runs no etiquetados aquellos en los que no se registran

colisiones o no tienen interés físico.

Aunque este modelo de certificación humano ha sido eficiente hasta el momento, implementar técnicas informáticas podría mejorar este rendimiento. La certificación humana está condicionada a la fatiga, falta de concentración e incluso una incorrecta visualización de los datos. En cambio, un análisis realizado por un ordenador no tiene asociados estos problemas. Implementando modelos de aprendizaje automático no solo podría mejorarse la certificación, si no que podría ahorrarse un gasto humano considerable, que podría ser empleado en otras tareas. Además, con las mejoras en luminosidad que se esperan para el Run 3 y posteriores, este gasto irá incrementando cada vez más: cuantos más datos tengamos, más expertos necesitamos. Es por ello que, en este trabajo, trataremos de incluir estas novedosas técnicas con el objetivo de automatizar la certificación y ahorrar tiempo y trabajo a las personas encargadas actualmente de la tarea.

Capítulo 4

Introducción al aprendizaje automático (Machine Learning)

[25][26] El aprendizaje automático (o Machine Learning, ML) es la ciencia de programar a los ordenadores para que puedan aprender a partir de los datos. Está basada en la inferencia estadística y su objetivo puede ser predictivo o descriptivo. En el primer caso, se buscan aproximaciones que proporcionen cómo se van a comportar ciertos parámetros de los datos en un futuro próximo y, en el segundo, lo que se espera obtener es conocimiento acerca de los datos como pueden ser patrones o irregularidades.

Las opciones de aplicación cubren un amplio rango de posibilidades y ámbitos diferentes, desde proporcionar un filtro para el spam en nuestro correo hasta la detección de tumores en escáneres o las sugerencias de productos cuando realizamos una compra.

Para clasificar los algoritmos disponibles podemos fijarnos en si son supervisados por algún experto (supervised, unsupervised, semisupervised learning...), si pueden o no aprender de forma instantánea de una base de datos que aumenta constantemente (online vs batch learning) o si se basan en datos pasados para aprender o emplean el reconocimiento de patrones en datos de entrenamiento y generan un modelo (aprendizaje basado en la evidencia o en un modelo). Estos criterios de clasificación no son únicos e

incluso se pueden combinar entre ellos para obtener el algoritmo más apropiado para cada tarea.

En nuestro caso, usaremos algoritmos supervisados a los que proporcionaremos un conjunto de datos etiquetado separado en diferentes grupos: uno para el entrenamiento del modelo y otro para su evaluación.

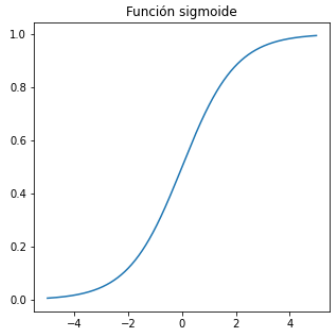
4.1. Técnicas supervisadas de clasificación

En esta sección vamos a presentar cómo funcionan cada uno de los modelos que evaluaremos en el siguiente capítulo. Todos ellos están pensados para problemas de clasificación binaria, como es nuestro caso. Decimos que una clasificación es binaria cuando tan solo tenemos dos clases posibles como resultado de la clasificación (0 ó 1 para nuestros datos). En el caso de nuestro problema tendremos las clases de datos buenos y de datos malos y los etiquetaremos como 0 y 1, respectivamente.

4.1.1. Regresión logística balanceada con pesos

La regresión logística estima la probabilidad de que un caso pertenezca a una clase concreta. Cuando esta probabilidad es mayor del 50 % el modelo predice que esta es la clase correcta y, la descarta, en otro caso.

El modelo calcula una suma ponderada de los datos de entrada (junto a un término de sesgo) y calcula la logística de este valor. La función logística es una sigmoide dada por la ecuación (4.1) y cuya forma podemos ver en la figura 4.1. Cuando el modelo tiene una estimación de probabilidad $\hat{p} = h_\theta(x) = \sigma(x^T\theta)$ donde x son los datos de entrada y θ los pesos asociados, la predicción de clasificación $\hat{y}$ es simplemente la recogida en la expresión (4.2).



$$\sigma(t) = \frac{1}{1 + \exp(-t)} \tag{4.1}$$

Figura 4.1: Representación gráfica de la función sigmoide.

$$\hat{y} = \begin{cases} 0 \quad si \quad \hat{p} < 0.5 \\ 1 \quad si \quad \hat{p} \geq 0.5 \end{cases} \tag{4.2}$$

El conjunto de entrenamiento nos sirve para calcular los pesos recogidos en θ, buscando altas probabilidades para casos favorables y bajas para casos negativos. Para conseguir estos valores se emplea una ecuación de coste, que se minimiza globalmente con algún algoritmo complementario como el Gradient Descent (GD).

En ML tenemos diferentes tipos de penalizaciones para la función de coste, lo que nos genera diferentes modelos matemáticos. Por ejemplo, el modelo Lasso [34] emplea una del tipo (4.3) y es más apropiado para seleccionar los parámetros más importantes de los datos y descartar el resto. En cambio, la del modelo Ridge viene dada por la expresión (4.4) y es más apropiada para evitar el overfitting.

$$\sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{n} x_{ij}\theta_j \right)^2 + \lambda \sum_{j=1}^{n} \mid \theta_j \mid \tag{4.3}$$

$$\sum_{i=1}^{n} \left(y_i - \sum_{j=1}^{n} x_{ij}\theta_j \right)^2 + \lambda \sum_{j=1}^{n} \theta_j^2 \tag{4.4}$$

4.1.2. K-nearest neighborgs (KNN)

[26] El estimador KNN (K-nearest neighbors) es no paramétrico y adapta el método en función de las densidades locales. Calcula una densidad en función de las distancias de un punto x a los datos de la muestra x^t. En concreto, sean $d_1(x) < d_2(x) < ... < d_N(x)$ las distancias desde x a los puntos de la muestra y $k << N$ con N tamaño de la muestra, la densidad estimada por el método KNN viene dada por:

$$\hat{p} = \frac{k}{2Nd_k(x)} \tag{4.5}$$

Es un estimador naive donde $h = 2d_k(x)$ y en el que, en lugar de ver cuántas muestras caen en nuestro bin, adaptamos la anchura del bin para que contenga las muestras deseadas. No es continuo y claramente no es una función de densidad de probabilidad pues la integral no vale 1, si no ∞.

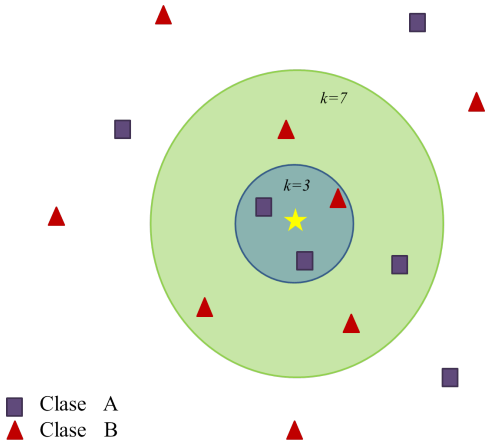


Figura 4.2: Representación gráfica del funcionamiento del modelo KNN. Fuente: Elaboración propia.

Su funcionamiento es muy sencillo. Para cada dato, se comprueban las clases de los k vecinos más cercanos y se asigna la clase mayoritaria entre estos. Es decir, si tenemos 3 vecinos de cierta clase A y 8 de una clase B, el resultado proporcionado por el modelo es que el dato pertenece a la clase B.

Para evitar empates en el número de muestras de cada clase, se suele tomar un número impar de vecinos. Debemos notar que el modelo es sensible al valor de k, como podemos ver en la imagen 4.2. En este ejemplo, si $k = 3$ el modelo se decidiría por la clase A y, si $k = 7$, por la clase B.

4.1.3. Árboles de decisión

Los árboles de decisión son algoritmos muy versátiles en el aprendizaje automático, que sirven tanto para tareas de clasificación como para regresión. Es uno de los modelos más sencillos de interpretar por los humanos, pues podemos crear diagramas sencillos que convierten nuestra clasificación en algo visual. Son, además, la parte fundamental de los Random Forest, uno de los algoritmos más prometedores que emplearemos.

Un ejemplo de árbol de decisión se presenta en la figura 4.3. En ella, podemos ver todos los componentes: cada nodo corresponde a un test o un atributo, cada rama a un valor de dicho test, cada hoja (nodo final) corresponde a una clase final y cada camino es el conjunto de ramas que nos lleva a la decisión final.

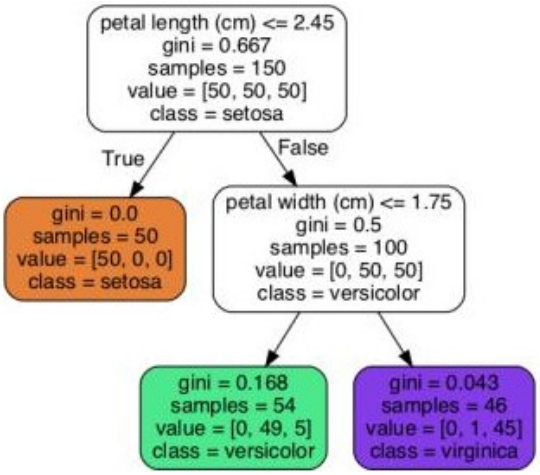


Figura 4.3: Ejemplo de árbol de decisión para datos asociados al tamaño de las hojas de diferentes plantas de la misma especie. Fuente: [25].

Su construcción es sencilla: en cada paso se evalúan ciertos atributos de los datos en razón a su poder de separación y se escoge el que nos proporcione mejores resultados. No hay un sólo algoritmo para realizar esta división y sus principales diferencias radican en qué criterio emplean para separar los nodos. Por ejemplo, en ML se emplean $ID3$, $C4.5$ o $C5.0$ y en estadística el modelo CART (Classification and Regression Tree). Para más información sobre los algoritmos anteriores consultar: [28], [29] y [30].

En nuestro caso, como vamos a emplear el algoritmo del paquete de Scikit-learn [27] de Python, emplearemos el modelo CART con el índice GINI. En este modelo las decisiones tomadas son siempre binarias, así que cada división solo puede tener dos hijos como máximo. El índice GINI es una medida de la pureza del nodo: un nodo es puro ($GINI = 0$) si el conjunto de entrenamiento solo posee elementos de una clase. En concreto, la fórmula del índice es la recogida en la expresión (4.6), donde $p_{i,k}$ es el ratio de la clase k entre los datos de entrenamiento del nodo $i-$ésimo.

$$G_i = 1 - \sum_{k=1}^{n} p_{i,k}^2 \tag{4.6}$$

Alternativamente al índice GINI está la entropía, pero los resultados suelen ser bastante similares y el tiempo de procesamiento es mayor con la entropía. Por ello, vamos a emplear el índice GINI.

Los árboles de decisión tienen como problema principal que es muy probable que haya overfitting si les dejamos total libertad al entrenamiento. En consecuencia, vamos a tener que regularizarlo, es decir, restringir hiperparámetros como la profundidad máxima o el mínimo de muestras que debe tener cada nodo para que se produzca la separación.

Por otro lado, un árbol nos proporciona información pero esta puede llegar a ser insuficiente. En cambio, si juntamos varios podemos obtener una información más precisa. Esto se conoce como *ensemble learning*, cuya idea radica en generar múltiples modelos básicos en el conjunto de entrenamiento

y combinarlos para obtener un modelo más fuerte que mejore la estabilidad y la actuación de los modelos por separado. Las opciones más usadas son bagging y boosting.

Bagging

Método de agrupación que responde a los siguientes pasos:

1. Supongamos que la muestra de entrenamiento tiene N observaciones y se seleccionan M submuestras de manera aleatoria con reemplazamiento (bootstrapping).

2. Con cada una de las submuestras anteriores, generamos un árbol de decisión que dejamos crecer totalmente. Estos árboles estarán poco influenciados y habrá mucha variedad entre ellos.

3. Una predicción para nuevos datos se construirá en función de los M árboles anteriores. Por ejemplo, se podrá usar la media o la clase mayoritaria.

Boosting

Cada ejecución del modelo determinará sobre qué muestra trabajará la siguiente ejecución. El algoritmo asigna pesos a cada modelo resultante, dependiendo del desempeño individual. En consecuencia, una predicción sobre nuevos datos se realiza a partir de una combinación balanceada con pesos de los modelos individuales.

Random Forest (RF)

Los Random Forest son una combinación de árboles de decisión entrenados mediante bagging. Para cada muestra bootstrap empleada en uno de los árboles habrá datos del conjunto de entrenamiento que no sean empleados. En consecuencia, habrá datos del conjunto que no formarán parte del

proceso de creación de ninguno de los árboles del modelo. Estas muestras no usadas se denominan Out of The Bag y pueden ser útiles para estimar el error del test final.

Boosted Decision Trees (BDT)

Análogo al Random Forest pero construyendo el modelo final a partir de boosting en lugar de bagging.

4.2. Posibles problemas asociados al ML

Cuando aplicamos algoritmos de aprendizaje automático estamos haciendo inferencia sobre una muestra, por lo tanto, los problemas vendrán o bien de un mal algoritmo o de una mala muestra. Veamos algunos ejemplos de ambos casos:

1. Poca cantidad de datos: Teniendo en cuenta que el algoritmo se basa en la experiencia, necesitamos muchos ejemplos para que los algoritmos funcionen correctamente. Además, se ha comprobado que modelos bastante simples pueden alcanzar la misma precisión que modelos muy complejos cuando tienen la suficiente cantidad de datos.

2. Conjunto de entrenamiento no representativo: O bien porque la proporción de datos es muy pequeña o porque no representa bien la muestra (por ejemplo, si solo proporcionamos datos de las colas en una distribución normal el modelo nunca va a aproximarnos los datos a la distribución real). En consecuencia, es crucial usar casos representativos de aquellos que quieres generalizar para obtener una buena aproximación.

3. Overfitting: Cuando el modelo generado sólo se adapta bien a la muestra proporcionada, el error del modelo tiende a dispararse. Ocurre

cuando tenemos modelos muy complicados en relación a la cantidad y la calidad de nuestros datos. Por ello, una forma de reducir el riesgo de overfitting es simplificar el modelo, lo que se conoce como regularización. La cantidad de regularización que aplicamos se puede controlar a partir de los hiperparámetros, que son los parámetros del propio algoritmo.

4. Underfitting: Es lo contrario al overfitting y surge como consecuencia de tener un modelo tan simple que no describa ni la propia muestra.

4.3. Herramientas y medidas de evaluación de modelos

Una vez generado el modelo con los datos de entrenamiento (*training set*) tenemos que evaluarlo con la porción de datos destinada a esto (*test set*) para saber qué tan bueno es. Tendremos una estimación del error en la muestra y del error en datos que no pertenecen a la muestra (error de generalización) y, la diferencia entre ambos, nos dirá si existe o no overfitting.

Cross-validation

Una forma de escoger qué modelo es el mejor es comparar sus errores de generalización para distintos valores de sus hiperparámetros. En cambio, puede ser que esto nos devuelva el mejor modelo particularizado a estos datos y no generalizado. Para abordar esta situación se extrae de la muestra de entrenamiento una pequeña parte, que llamaremos *conjunto de validación* y será esta la que evalúe cuál es el mejor modelo. Normalmente esta es una buena opción pero puede producir problemas cuando la muestra tomada es muy pequeña o muy grande. Por ello, lo que normalmente se realiza son pruebas de *cross validation*, que es generar muchos conjuntos de validación y analizar la precisión media.

4.3.1. Medidas de evaluación

En ocasiones el porcentaje de acierto, calculado como datos bien clasificados entre el total, no es una buena magnitud para diferenciar un buen modelo de uno malo. Si, por ejemplo, tenemos un modelo que predice si la imagen de un número manuscrito corresponde a un 3 o no y este nos devuelve siempre que no, acertaría un 90 % de las veces pues tenemos 10 posibilidades (del 0 al 9). En cambio, no es un buen modelo. Por ello en ocasiones se prefieren otras medidas de evaluación y, en especial, si tratamos con conjuntos de datos asimétricos como será nuestro caso.

Matriz de confusión y métricas asociadas

La matriz de confusión (figura 4.4) es una forma gráfica de identificar falsos positivos (Error de tipo I) y falsos negativos (Error de tipo II). O, lo que es lo mismo, la cantidad de veces que nuestro modelo clasifica la clase A como B y viceversa. Cada columna de la matriz corresponde con una clase real, así como cada una de las filas está asignada a una clase predicha. El objetivo es minimizar lo máximo posible los errores de tipo I y II, obteniendo una matriz lo más diagonal posible.

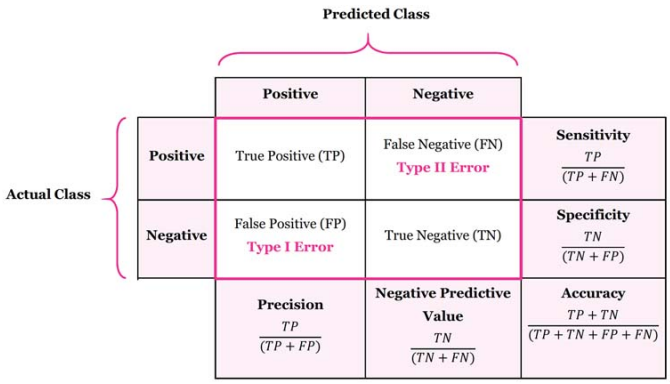


Figura 4.4: Esquema de la matriz de confusión con los diferentes estimadores de precisión. Fuente: [32].

En la figura 4.4 podemos ver también diferentes estimadores que nos darán una idea de qué tan bueno es nuestro modelo:

1. Precisión: Es el ratio entre la cantidad de datos positivos clasificados correctamente y el total de los predichos como positivos. Nos proporciona una idea de qué tan cerca está nuestro modelo del valor verdadero.
$$\frac{TP}{TP + FP} \tag{4.7}$$
2. Especifidad (Specifity): Es la proporción de casos negativos que fueron correctamente clasificados.
$$\frac{TN}{TN + FP} \tag{4.8}$$
3. Sensibilidad (Recall): Análogo de la especifidad para los casos positivos.
$$\frac{TP}{TP + FN} \tag{4.9}$$
4. Exactitud (Accuracy): Proporción de predicciones correctas.
$$\frac{TP + TN}{TP + TN + FP + FN} \tag{4.10}$$

Función F_1

Los valores de precisión y recall usualmente se estudian juntos para medir la efectividad del modelo y el objetivo es que ambos sean lo más altas posibles. En cambio, no se pueden incrementar ambas a la vez de manera arbitraria, si no que cuando incrementas una la otra se reduce. A esto se le conoce como precision/recall trade-off. Por ello se busca un equilibrio en el que ambas puedan tomar el máximo valor posible. Se emplea entonces la función F_1, definida a partir de la media armónica entre la precisión y el recall. Esto es,

$$F_1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \tag{4.11}$$

Esta función favorece el caso en el que ambas métricas son similares y penaliza cuando uno (o ambos) es muy bajo.

Curva ROC y AUC

Una curva ROC (de Receiver Operating Characteristic o Característica Operativa del Receptor) es otra herramienta útil para clasificaciones binarias. Consiste en representar la tasa de positivos bien clasificados (o recall) frente la tasa de falsos positivos o, lo que es lo mismo, $1 - specifity$.

Conjuntamente con la curva, se suele representar una línea discontinua que representa a un modelo puramente aleatorio y corresponde con la bisectriz del cuadrante. El objetivo es conseguir una curva que se aleje lo máximo posible de esta línea, en la mitad superior del cuadrante.

Para analizar la eficiencia de nuestros modelos, debemos comparar el área bajo la curva ROC (AUC) para los conjuntos de evaluación y entrenamiento. Cuanto más cerca estén de 1, mejor poder predictivo tiene nuestro modelo. En cambio, cuando haya una diferencia mayor del 2 % consideraremos que el overfitting a la muestra de entrenamiento no será despreciable. Buscamos entonces, valores altos pero similares.

4.4. Paquetes de Python necesarios

En esta sección vamos a introducir los paquetes o bibliotecas de Python necesarios para el análisis del siguiente capítulo. En concreto, hemos usado pandas, scikit-learn, NumPy, matplotlib y seaborn.

- **NumPy**: Es la biblioteca de referencia para la programación científica en Python. Permite trabajar de forma sencilla con arrays y recoge una gran colección de funciones matemáticas complejas. Es la base de otros paquetes como Pandas o Scikit-learn. Su documentación puede consultarse en: `https://numpy.org/doc/stable/`.

- **Matplotlib**: Junto con NumPy, es una de las bibliotecas más usadas de Python. En este caso, está centrada en la representación gráfica y en la visualización de datos. Para este trabajo, se ha usado en todas las representaciones de histogramas y gráficas. Su documentación puede consultarse en: `https://matplotlib.org/stable/contents.html`.

- **Pandas**: Es un paquete potente y fácil de usar para el análisis y la manipulación de datos en Python. Para revisar la documentación, consultar: `https://pandas.pydata.org/docs/`. El uso principal que le hemos dado es para crear la estructura de nuestros datos, analizarlos y modificarlos cuando era necesario.

- **Scikit-learn**: Este paquete proporciona herramientas fáciles y eficientes para el análisis predictivo de datos. Está codificado sobre las librerías NumPy, SciPy y Matplotlib y contiene los algoritmos que han sido necesarios en nuestro análisis. En algunos casos, las funciones que hemos usado están en submódulos suyos como metrics, tree o model_selection. Para acceder a la documentación, consultar: `https://scikit-learn.org/stable/`.

- **Seaborn**: Esta es una biblioteca avanzada de representación gráfica, basada en Matplotlib. Solo la hemos empleado en los mapas de color de las matrices de confusión. La documentación asociada se puede consultar en: `https://seaborn.pydata.org/tutorial.html`.

Capítulo 5

Aplicación de las técnicas de Machine Learning a la certificación de muones de CMS

En este capítulo vamos a comparar la eficiencia de los modelos supervisados presentados en el capítulo anterior para clasificar conjuntos de datos tomados en CMS entre 2015 y 2018. Los datos corresponden a medidas de observables físicos de muones globales.

Nuestro objetivo principal es determinar si distintos grupos de datos son aptos para estudiar nueva física o tienen algún problema. Los del primer tipo los clasificaremos como buenos y les asignaremos la etiqueta '1' y, los del segundo, como malos con etiqueta '0' .

5.1. Descripción de los datos

Originalmente, nuestro conjunto de datos tiene tres etiquetas posibles: buenos (GOOD), malos (BAD) y sin etiquetar (NO LABEL), correspondiendo esta clasificación a la opinión de expertos que han revisado los diferentes parámetros. Los datos que no están etiquetados provienen de runs que han sido tomados en condiciones en las que no son útiles, como por ejemplo en

runs de calibración, de prueba o demasiado cortos. Una forma de evaluar si nuestro modelo es bueno será a partir de la clasificación de estos datos sin etiqueta, pues deberían ser catalogados como malos.

Además de estas tres etiquetas, los datos se diferencian entre ellos a través del número de run. Las decisiones se tomarán en función de cinco variables físicas que tienen a su vez asociadas otras cinco magnitudes. En concreto, las variables que vamos a emplear son: la pseudorapidez η, el momento lineal p, el momento transverso p_T, el ángulo ϕ y un valor asociado a una distribución χ^2 [1] sobre sus grados de libertad $\frac{\chi^2_n}{n}$. Esta última distribución es consecuencia de cómo se reconstruye a los muones globales. Como hemos mencionado en el capítulo 2, un muón global está reconstruído a partir de señales en el Tracker y en las cámaras de muones. La reconstrucción se realiza a partir de un ajuste Kalman-Filter (KF) [31] y, en consecuencia, aparece este valor de $\frac{\chi^2_n}{n}$. Para cada una de las variables, tenemos las magnitudes:

1. Número de registros (entries): Es la cantidad de muones registrados en el run asociado. Es importante recordar que realizamos el análisis sobre runs, que son los periodos de tiempo que han sido etiquetados por los expertos y no sobre LS.

2. Media: Promedio, por run, de la distribución de la variable asociada (momento medio, η media...)

3. El valor cuadrático medio o raíz de la media cuadrática (RMS) que se calcula a partir de la expresión (5.1). Es una magnitud útil cuando el

[1] Una distribución χ^2 es una distribución estadística correspondiente a una distribución gamma de parámetros $(\frac{n}{2}, \frac{1}{2})$ donde n es la cantidad de grados de libertad, i.e, $\chi^2_n = \gamma(\frac{n}{2}, \frac{1}{2})$.

signo de la variable no es relevante y nos interesa más el valor absoluto de la misma.

$$RMS = \sqrt{\frac{1}{N}\sum_{k=1}^{N} x_k^2} \tag{5.1}$$

4. Asimetría estadística (skewness) [33]: Magnitud que nos permite determinar qué tan simétrica es la distribución de nuestros datos respecto a la media. Puede tomar valores tanto positivos (es asimétrica hacia la derecha y su cola hacia la derecha es más larga), negativos (asimétrica hacia la izquierda) o ser nula (totalmente simétrica).

   Para obtener una idea más visual de esta magnitud, se añade la figura 5.1. Normalmente, para valores de skewness (en valor absoluto) entre 0 y 0.5 se considera que la distribución es simétrica, entre 0.5 y 1 ligeramente simétrica y mayores que 1 asimétrica.

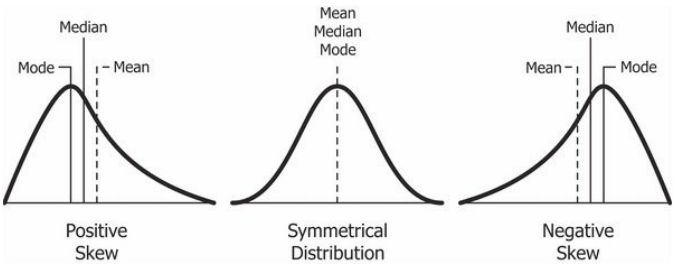


Figura 5.1: Esquema de cómo es la forma de las distribuciones asociadas a diferentes valores de skewness. Fuente: [33].

5. Curtosis (kurtosis)[33]: Magnitud que mide la importancia de las colas para distribuciones de probabilidad que toman valores en los números reales. Conjuntamente con la asimetría, nos describe la forma de la distribución estadística, como podemos ver en la representación gráfica de la figura 5.2.

   En este caso, el valor que se toma como referencia es usualmente 3: si

tiene una $kurtosis = 3$ la distribución se conoce como mesocúrtica y es similar a la normal. Si fuese mayor, la distribución sería leptocúrtica, sus colas serían más prominentes y el pico más pronunciado. Por último, si fuese menor, sería platicúrtica, con colas menos gruesas que la normal y un pico menos apuntado.

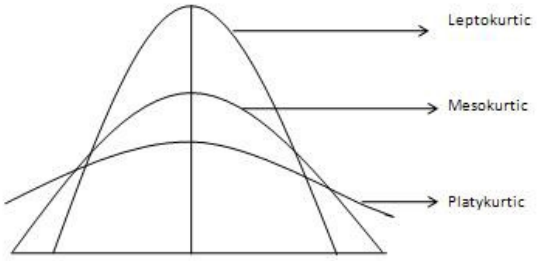


Figura 5.2: Esquema de cómo es la forma de las distribuciones asociadas a cada tipo de curva en función de su kurtosis. Fuente: [33].

Seguimos en esta sección comentando una de las principales características de nuestro problema: el desequilibrio de los datos. Esto es, una clase es notablemente más numerosa que el resto. En nuestro caso, son los datos buenos, que representan casi un 90 % de los datos totales, como se puede ver en la tabla 5.1. Recordemos que esta clasificación ha sido realizada por expertos en la materia. Buscamos entonces un modelo que nos proporcione un porcentaje de acierto superior al anterior, pues queremos que clasifique mejor que el estimador *"la clase mayoritaria"*. Este último estimador, como su propio nombre indica, asocia siempre las muestras a la clase más numerosa. Una forma de atacar este problema será añadir pesos a las clases en los algoritmos o realizar pruebas de cross-validation.

Grupo	G	B	NL	Total
Cantidad datos	1362	129	44	1535
Porcentaje (%)	88.7	8.4	2.9	100

Tabla 5.1: Resumen de la cantidad de datos por clase. G corresponde a la clase de los datos buenos (good), B a los malos (bad) y NL a los no etiquetados (no label).

5.1.1. Preparación de los datos para su estudio

Antes de aplicar los modelos de aprendizaje automático tenemos que preparar nuestros datos. Para ello seleccionaremos las filas de datos que nos interesan y modificaremos algunos valores. Empezaremos eliminando los conjuntos de datos con entries nulos ya que, es evidente, que si ninguna partícula ha participado en el proceso no podemos tener ningún tipo de información relevante. No obstante, tenemos que conseguir que nuestro algoritmo clasifique aquellos conjuntos de datos con entries nulos como datos malos. Esto último se conseguirá porque, como veremos en la siguiente sección, la diferencia entre los entries de las clases buena y mala es bastante llamativa.

Después, tenemos un problema en las columnas de skewness y kurtosis, en las que algunos valores no han sido registrados y aparecen como NaN. Estas filas están también asociadas a los entries nulos, por lo que en el primer paso de preparación ya nos hemos librado de ellos. Por otro lado, hay valores de $\pm\infty$ que nos suponen otro problema al no poder realizar ningún tipo de operaciones con ellos. Aquí tenemos dos opciones: eliminar las filas que presentan esta característica o cambiar estos valores por un número lo suficientemente alto para que el algoritmo lo diferencie con facilidad. Por ejemplo, por 9999. Escogeremos el enfoque en el que se eliminan también las filas con valores infinitos.

Una vez hemos adecuado nuestros datos según lo anterior, les añadimos una nueva variable que será dicotómica. Es decir, es una variable que solo puede tomar dos variables categóricas. En nuestro caso, estas serán la clase de datos buenos y la de datos malos. Para poder trabajar de forma numérica con esta variable, asignaremos el valor 1 a los datos clasificados por los expertos como buenos y el 0, al resto. A partir de esto ya podemos extraer los conjuntos de entrenamiento y evaluación. Usaremos una muestra del 10 % en los datos malos y un 35 % en los buenos para evaluar el modelo y, el resto, para entrenarlo. La diferencia entre los porcentajes es una consecuencia de la diferencia entre la cantidad de datos de cada tipo disponibles. Queremos tener suficientes datos malos para poder entrenar el modelo y, por ello, sacamos menos cantidad del conjunto inicial para evaluar. Obviamente los datos no etiquetados no entran en el entrenamiento del modelo pero sí lo harán en la evaluación. Esencialmente, esperamos que sean etiquetados como bad por las condiciones en las que fueron tomados.

```
In [80]: df_g.head()
```

Out[80]:

run	GlbMuon_Glb_chi2OverDf_entries	GlbMuon_Glb_eta_entries	GlbMuon_Glb_p_entries	GlbMuon_Glb_phi_entries	GlbMuon_Glb_pt_entries	GlbMuon_Glb_ch
254459	6741051.0	6741051.0	6741051.0	6741051.0	6741051.0	
254905	2114154.0	2114154.0	2114154.0	2114154.0	2114154.0	
254906	7715028.0	7715028.0	7715028.0	7715028.0	7715028.0	
254907	532419.0	532419.0	532419.0	532419.0	532419.0	
254914	5084976.0	5084976.0	5084976.0	5084976.0	5084976.0	

5 rows × 26 columns

Figura 5.3: Cabecera de nuestro dataframe, donde se puede ver que los datos se agrupan por run y las columnas corresponden a las variables de entrenamiento y la etiqueta.

Tras todas estas modificaciones nuestros datos quedarían organizados como en la imagen 5.3, donde se muestra la cabecera del dataframe. Tenemos los datos ordenados por run y, para cada run, tenemos 26 columnas. Estas

corresponden a las cinco magnitudes (media, RMS, skewness...) asociadas a cada variable física (pseudorapidez η, ángulo ϕ,...) y, por último, a la columna de etiquetado. Con todo esto, ya podríamos aplicar los algoritmos. En cambio, vamos a empezar con un pequeño análisis estadístico de cada variable por grupos para ver las diferencias iniciales.

5.1.2. Análisis estadístico

Con la opción de pandas de describe() vamos a obtener el número de datos, la media, la desviación estándar, el máximo y el mínimo para cada una de las variables físicas y sus características. Tomaremos por separado las clases para poder compararlas entre ellas y detectar en qué variables destaca más la diferencia. Usaremos, para cada caso, una tabla resumen y un histograma. Debido a su tamaño, las tablas se recogen en el apéndice A.

Entries

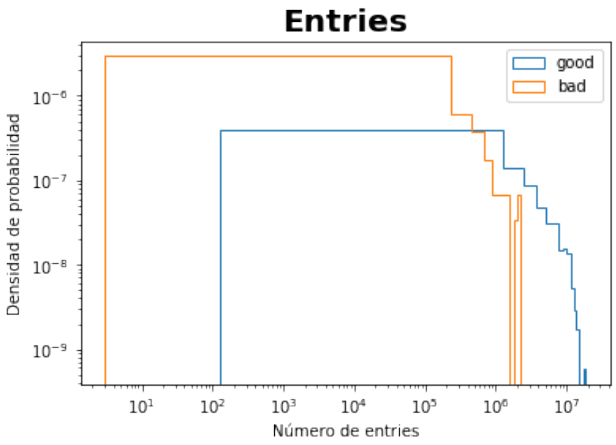


Figura 5.4: Histograma de entries, distinguiendo las clases de datos buenos (good) y malos (bad).

Los datos asociados a las entries son iguales en todas las variables, pues corresponden a la cantidad de partículas involucradas en cada medida y ellas

se miden al mismo tiempo para cada suceso. Podemos observar en la figura 5.4 que los datos malos tienen una distribución mucho más numerosa en el cero y con una cola más pequeña. En cambio, aunque el mayor número de datos buenos esté en valores pequeños, su cola es mucho más larga y, una cantidad alta de entries debería ser clasificada como buena.

En el resumen numérico (primeras columnas de cualquiera de las tablas del apéndice A) podemos ver que las clases de los datos malos y sin etiquetar son bastante similares entre sí, mientras que la clase de los datos buenos suele tener un orden de magnitud más en sus valores.

En definitiva, como las clases se diferencian bien, el número de entries debería ser un buen discriminante para nuestro modelo. En especial, lo será cuando el número de entries sea muy alto o muy bajo.

χ^2 sobre los grados de libertad

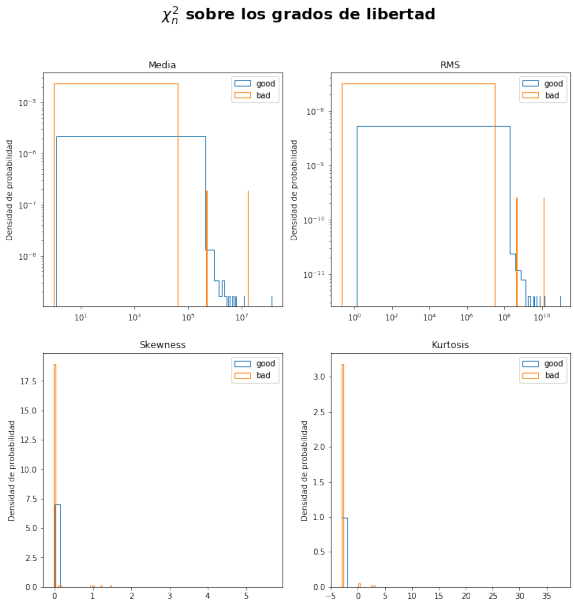


Figura 5.5: Histograma de χ^2 sobre los grados de libertad, distinguiendo las clases de datos buenos y malos.

En este caso tenemos, por un lado los histogramas de la media y el RMS y, por otro, los de la skewness y la kurtosis. En los primeros, podemos observar como los datos buenos tienen colas más largas y suelen tomar valores más altos. En los otros, se observa que los datos malos tienen un pico muy numeroso junto a una pequeña cola. En cambio, los datos buenos solo muestran el pico, con poca dispersión.

Por otro lado, tenemos los datos recogidos en la tabla A.1. Respecto a la media se observa que tanto los datos buenos como los malos tienen un valor medio muy similar, con aproximadamente el doble de varianza para los datos buenos. Esto es, están más dispersos. Lo mismo ocurre con el resto de las magnitudes, excepto en skewness. En esta última variable la media de la clase de los datos buenos es un orden de magnitud menor que la de los malos. Podríamos pensar que la skewness es un buen discriminante basándonos en este hecho, en cambio, basta fijarnos de nuevo en el histograma de la figura 5.5 para ver que los datos de ambas clases se acumulan muy cerca. Por lo tanto, este orden de magnitud corresponde a los datos que se encuentran en las colas y no a que sean distribuciones muy diferentes.

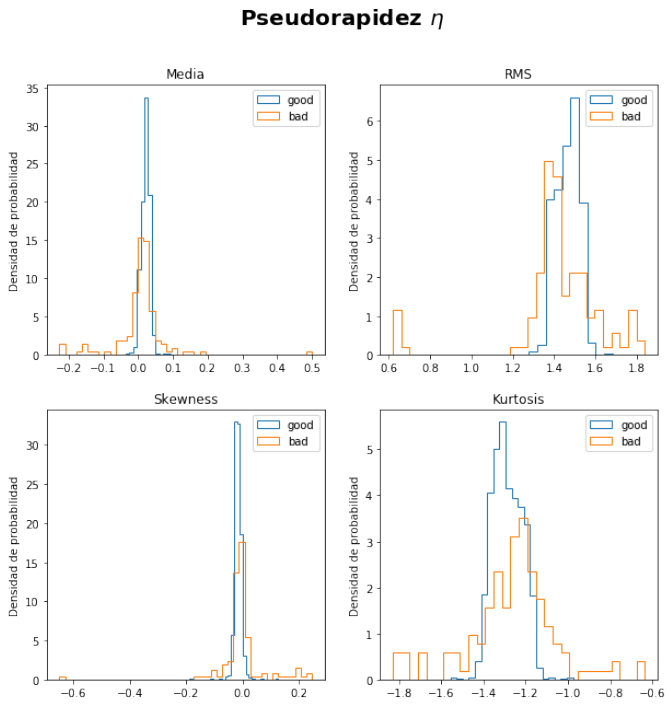


Figura 5.6: Histograma de las variables asociadas a la pseudorapidez η, distinguiendo las clases de datos buenos(good) y malos (bad).

Como podemos observar en la figura 5.6, en todas las variables la clase de datos malos está más dispersa que la de los buenos.

En el resumen numérico (tabla A.2) de la media, podemos ver que todos los estadísticos tienen el mismo orden de magnitud, a excepción de la media. Esta se diferencia en un orden de magnitud entre las clases, con ambos valores muy cercanos al cero.

Desde el punto de vista físico los datos nos dicen que las partículas están, de forma mayoritaria, contenidas en el plano transverso y aquellas que más se alejan pertenecen a la clase bad. Además, fijándonos en los valores de los estadísticos asociados a la skewness y los máximos y mínimos de la media, se observa que están distribuidos de una forma simétrica respecto al plano

transverso.

En este caso, lo que más diferencia a ambas clases es la longitud de las colas en el caso de los bad. También podría ser distintiva la posición de los picos centrales para el RMS y la kurtosis.

Ángulo ϕ

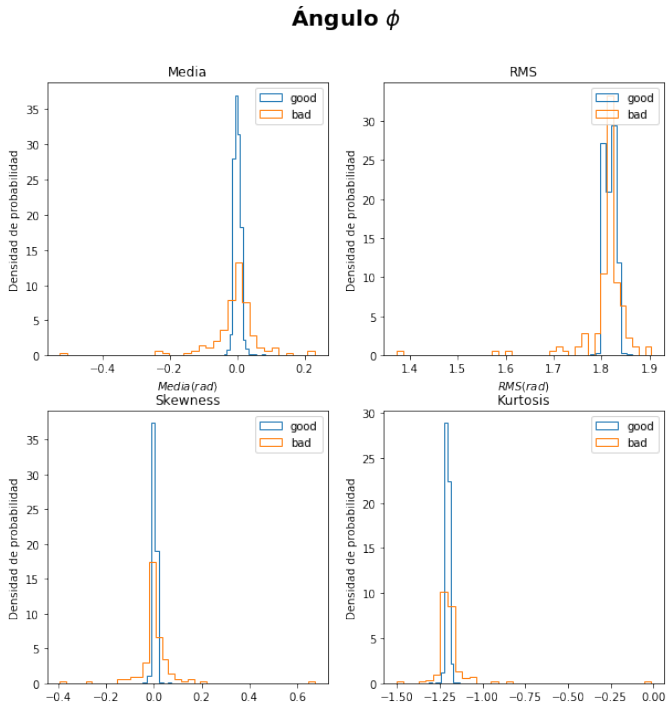


Figura 5.7: Histograma de las variables asociadas al ángulo ϕ, distinguiendo las clases de datos buenos (good) y malos(bad).

Obtenemos un resultado similar al que teníamos con la pseudorapidez η: la principal diferencia entre clases es la mayor dispersión en los datos malos. En consecuencia, valores que estén muy alejados de los picos centrales de cada una de las variables deberían ser clasificados como malos.

En el resumen numérico (tabla A.3) podemos comprobar que, efectivamente, los datos entre clases son bastante similares. En especial en media

y desviación típica. La principal diferencia radica en la amplitud de los intervalos (valor máximo y mínimo) donde se encuentran los datos, que son siempre mayores para la clase de datos malos. Esto justifica lo que decíamos antes: cuando haya un valor de cualquiera de las variables alejado del pico central, será probablemente malo.

Momento lineal p

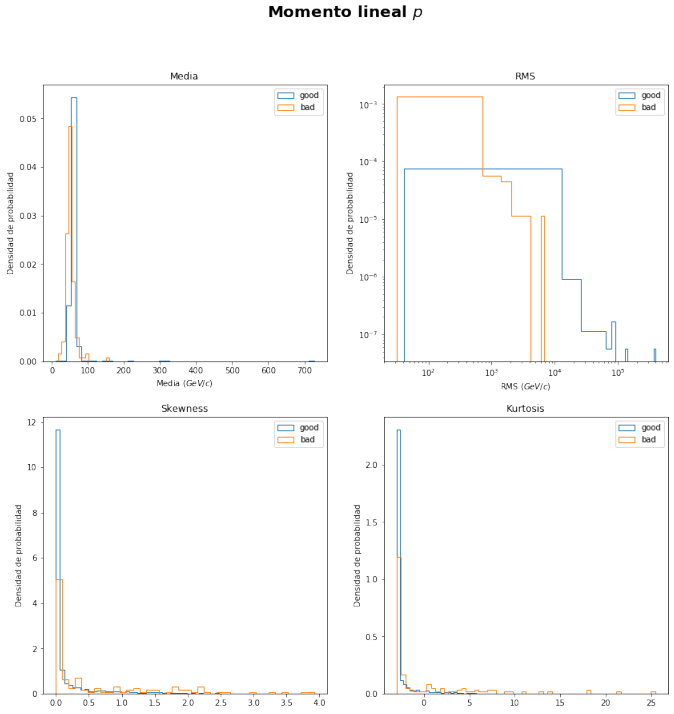


Figura 5.8: Histograma de las variables asociadas al momento lineal, distinguiendo las clases de datos buenos (good) y malos (bad).

En la figura 5.8 podemos observar que la media y la skewness concentran los picos bastante cercanos. La principal diferencia en ambos casos es la longitud de las colas. En el caso de la media, como podemos ver en la tabla A.4, el valor más alto de momento lineal corresponde a un dato bueno. En cambio, para la skewness, el dato asociado es malo.

Por otro lado, tanto en el RMS como en la kurtosis sí que diferencian de forma considerable ambas clases. En concreto, los datos malos toman valores más pequeños en RMS y más altos en kurtosis.

Momento transverso p_T

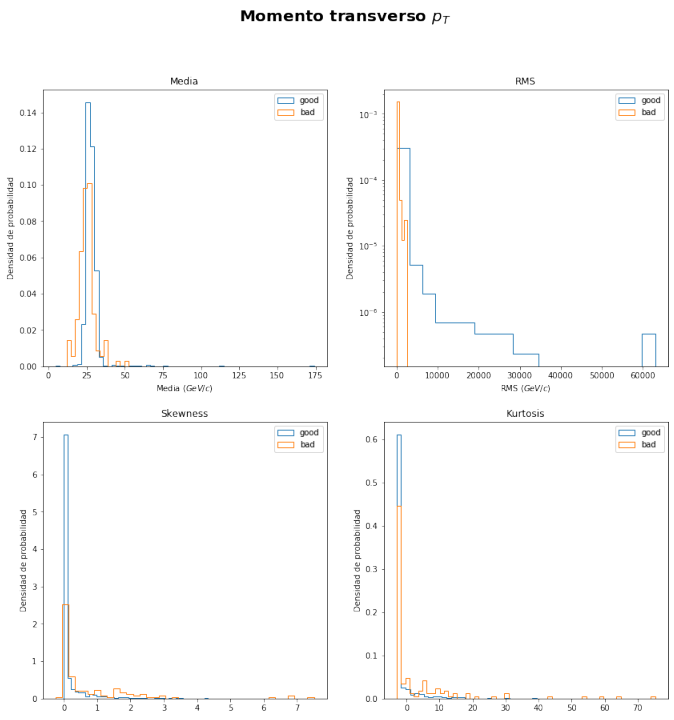


Figura 5.9: Histograma de las variables asociadas al momento transverso, distinguiendo las clases good y bad.

Para este caso tenemos la figura 5.9 y la tabla A.5. Podemos ver que, en la media, los picos están bastante solapados y la mayor variabilidad ocurre para valores altos donde, generalmente, los datos son buenos. Las distribuciones de RMS se diferencian fácilmente, siendo la de los datos malos más concentrada que la de los buenos. En este caso, sí que podría ser un buen discriminante.

Respecto a la skewness tenemos dos distribuciones bastante similares,

con formas que recuerdan a una distribución χ^2. En el caso de los malos, tenemos una cola más larga y menor probabilidad en el origen. Por otro lado, la kurtosis se acumula en el mismo punto pero, en este caso, sí que se ve una diferencia sustancial para valores altos.

5.2. Implementación de modelos y resultados

En esta sección se presentarán los resultados obtenidos al aplicar cada uno de los modelos introducidos en la sección 4.1. Empezaremos con los más simples, dejando para el final los que creemos más prometedores. Para cada modelo vamos a presentar la gráfica de la curva ROC con su AUC, las matrices de confusión para los conjuntos de entrenamiento, evaluación y para los datos sin etiquetar. Además, usaremos una gráfica que representa la probabilidad de que nuestro modelo clasifique cada dato como bueno. Esta última gráfica es bastante intuitiva, consiste de un histograma en el que se representan, por colores, las clases de datos buenos y malos del conjunto de entrenamiento y las clases de datos sin etiquetar, buenos y malos del conjunto de evaluación. Un buen modelo asignaría a los datos buenos una probabilidad cercana a 1 y, por tanto, estos se concentrarían a la derecha del histograma. En cambio, a los datos malos y sin etiquetar les asignaría una probabilidad baja, apareciendo así a la izquierda de la gráfica.

5.2.1. Regresión logística

La regresión logística puede ser estudiada balanceada con pesos o de forma simple, donde todos los datos tienen la misma importancia en el modelo. Empezaremos con el segundo caso, que nos proporciona el modelo más básico posible que podemos aplicar sobre nuestros datos. Además, desde un principio se sabe que este modelo no va a ser una buena opción. Esto es una consecuencia del desequilibrio entre clases de nuestros datos, la razón

principal por la que nuestro problema es tan complejo.

Para la implementación usaremos el paquete *sklearn.linear_model* y la función *LogisticRegression*. Respecto a los parámetros de la función vamos a escoger una penalización de tipo l_1, que toma el modelo Lasso para su función de coste. En el segundo caso, imponemos que los pesos sean balanceados con el parámetro *class_weight* de la función empleada. Este parámetro pondera las clases considerando su tamaño, corrigiendo así el desbalance inicial de nuestros datos.

Regresión logística simple

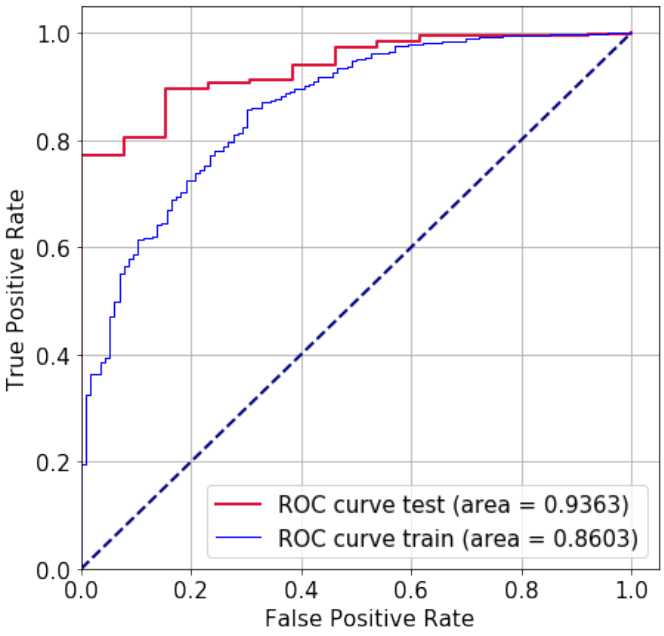
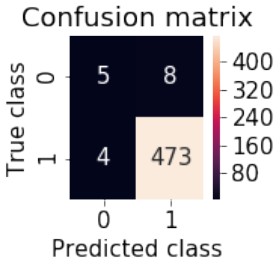
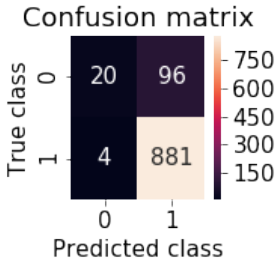


Figura 5.10: Curva ROC con su AUC para el modelo de regresión logística simple.

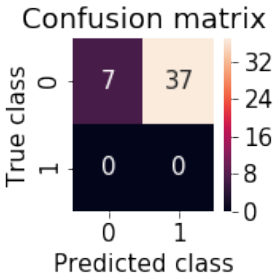
El poder predictivo, representado por el valor del área AUC de la curva ROC es bastante alto en ambas clases, siendo incluso mayor para el conjunto de evaluación. Podríamos pensar que un valor predictivo de $AUC_{test} = 0.93$



(a) Conjunto de evaluación.



(b) Conjunto de entrenamiento.



(c) Datos sin etiquetar.

Figura 5.11: Matrices de confusión para el modelo logístico sin pesos.

nos proporciona un buen modelo. En cambio, basta fijarnos en la matriz de confusión 5.11b para ver que el modelo no se ajusta bien ni a los datos de entrenamiento. Clasifica bien los datos buenos porque esencialmente lo que hace es clasificar casi todos como buenos, de forma similar al estimador 'la clase mayoritaria'.

Obtenemos, para los datos asociados al test, la tabla resumen 5.2. Tendremos una exactitud de 0.98, sin esto significar que el modelo funciona bien. Simplemente, este valor viene dado porque hay 477 datos de la clase de datos buenos bien clasificados. En cambio, si nos fijamos en la primera fila, la cantidad de datos malos mal clasificados casi duplica a los bien clasificados. O, lo que es lo mismo, tenemos un recall para los datos malos muy bajo. Por lo tanto, el modelo no clasifica bien estos datos.

Clase	***Precisión***	***Recall***	F_1	***Cantidad datos***
0	0.56	0.38	0.45	13
1	0.98	0.99	0.99	477
		Exactitud	0.98	490

Tabla 5.2: Medidas de evaluación asociadas a la matriz de confusión de los datos de evaluación (Figura 5.11a).

Relacionada con la clase no etiquetada está la matriz de confusión de la figura 5.11c y la tabla 5.3. Claramente no es un buen estimador para esta clase, pues tiene una tasa de falsos positivos altísima y, así, un recall muy bajo. Recordemos que, para aceptar el modelo, queremos que clasifique los datos no etiquetados como malos.

Clase	***Precisión***	***Recall***	F_1	***Cantidad datos***
0	1.00	0.16	0.27	44
		Exactitud	0.16	44

Tabla 5.3: Medidas de evaluación asociadas a la matriz de confusión de los datos no etiquetados (Figura 5.11c).

Por último, veamos en la figura 5.12 la probabilidad asociada a cada dato de pertenecer a la clase buena según nuestro modelo. Observamos que, este modelo, proporciona una distribución bastante diferente a la ideal, pues casi todos los datos se agrupan por encima del 0.5 sin importar su clase de origen.

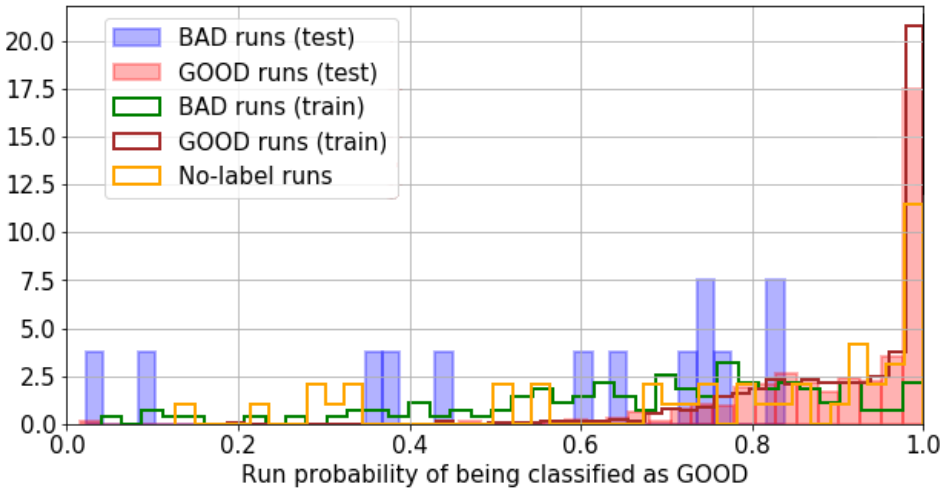


Figura 5.12: Probabilidad de clasificar los datos como buenos para el modelo logístico no balanceados.

En resumen, el modelo logístico sin pesos no es bueno para nuestra tarea. Clasifica mayoritariamente los datos como buenos, lo que hace que su actuación no sea buena en las clases de los datos malos y los no etiquetados.

Regresión logística balanceada con pesos

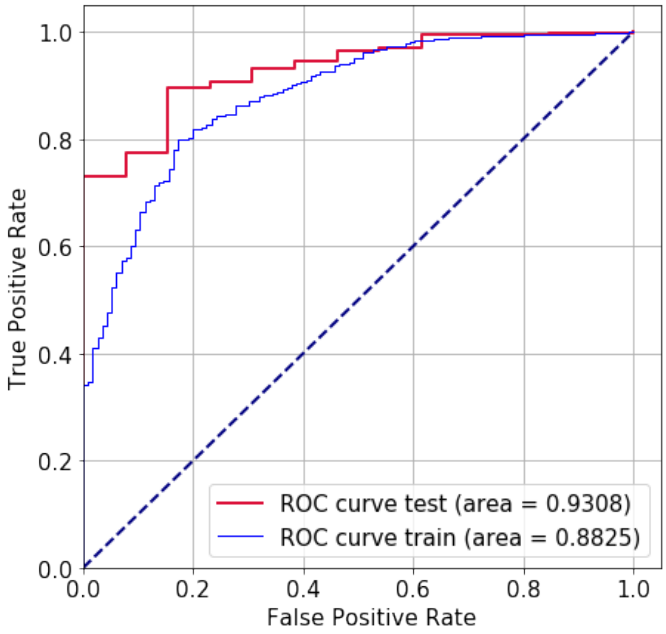
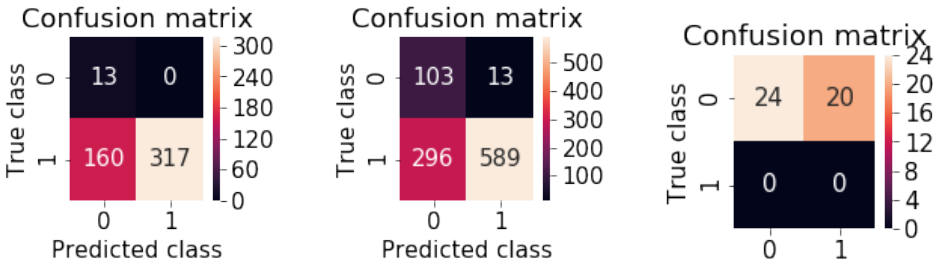


Figura 5.13: Curva ROC con su AUC para el modelo de regresión logística balanceado con pesos.

Fijándonos en la figura 5.13 y comparándola con la figura 5.10 podemos ver que añadir pesos aumenta la capacidad de clasificación del modelo para los datos de entrenamiento. En cambio, apenas influye en los datos de evaluación. Si miramos ahora las matrices de la figura 5.14 podemos ver que el efecto de añadir los pesos es una mejoría en la clasificación de los datos malos. Sin embargo, ha perdido exactitud sobre los buenos de una forma significativa.

En efecto, la exactitud asociada al conjunto de evaluación ha disminuido

hasta 0.67, como se puede ver en la tabla 5.4. Esto se debe a la tasa de falsos negativos que proporciona este modelo, que hace tomar relevancia a la diagonal negativa de la matriz. Disminuye así la exactitud global y la precisión de la clase de datos malos.



(a) Conjunto de evaluación.

(b) Conjunto de entrenamiento.

(c) Datos sin etiquetar.

Figura 5.14: Matrices de confusión para el modelo logístico balanceado con pesos.

Clase	***Precisión***	***Recall***	F_1	***Cantidad datos***
0	0.08	1.00	0.14	13
1	1.00	0.66	0.80	477
		Exactitud	0.67	490

Tabla 5.4: Medidas de evaluación asociadas a la matriz de confusión de los datos de evaluación (Figura 5.14a).

Por otro lado, los datos no etiquetados están mejor clasificados ahora que en el caso simple, como podemos ver en la tabla 5.5. Obtenemos un recall de 0.55 o, lo que es lo mismo, un 55 % de los datos no etiquetados bien clasificados. En comparación al 16 % del modelo anterior, es una mejora sustancial. Este incremento era de esperar porque, como acabamos de comentar, el modelo clasifica mejor los casos negativos.

Clase	***Precisión***	***Recall***	F_1	***Cantidad datos***
0	1.00	0.55	0.71	44
		Exactitud	0.55	44

Tabla 5.5: Medidas de evaluación asociadas a la matriz de confusión de los datos no etiquetados (Figura 5.14c).

Por último, la figura 5.15 nos muestra la probabilidad de que el modelo clasifique como buenos cada uno de los datos. Se observa, de nuevo, que clasifica mejor los casos malos y sin etiquetar que el modelo anterior, pero clasifica bastante peor los datos buenos. Por lo tanto, tampoco es un buen modelo para nuestro caso.

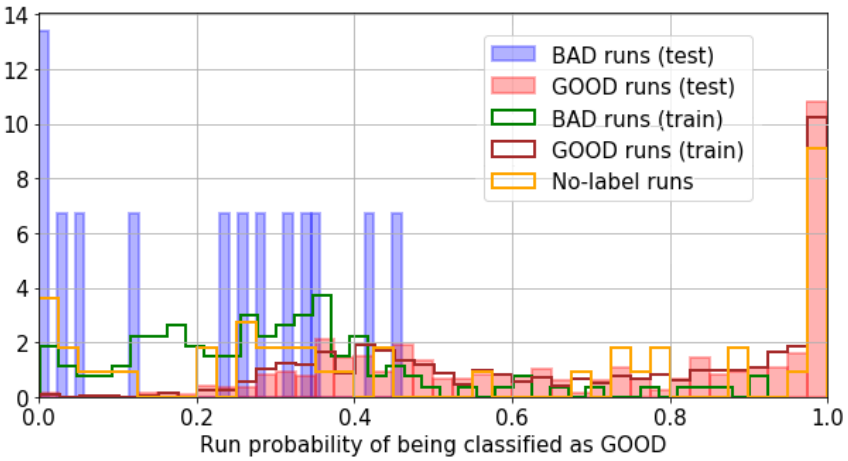


Figura 5.15: Probabilidad de clasificar los datos como buenos para el modelo logístico balanceado con pesos.

5.2.2. K-nearest neighborgs (KNN)

Para el modelo de KNN emplearemos la función $KNeighborsClassifier$ del paquete $sklearn.neighbors$. El parámetro más importante es el número de vecinos considerado y, con el objetivo de encontrar el mejor, realizamos un barrido de parámetros con pruebas de cross-validation. En el barrido, también conocido como Grid Search, evaluamos el modelo en diferentes combinaciones de parámetros para encontrar aquella que nos devuelva el mejor resultado. En concreto, probaremos varias opciones de números impares en torno al valor $\sqrt{N}/2$ con N el número de muestras disponibles. Esta elección es un convenio en el aprendizaje automático para escoger el número de vecinos. Usaremos, a su vez, tres conjuntos de cross-validation.

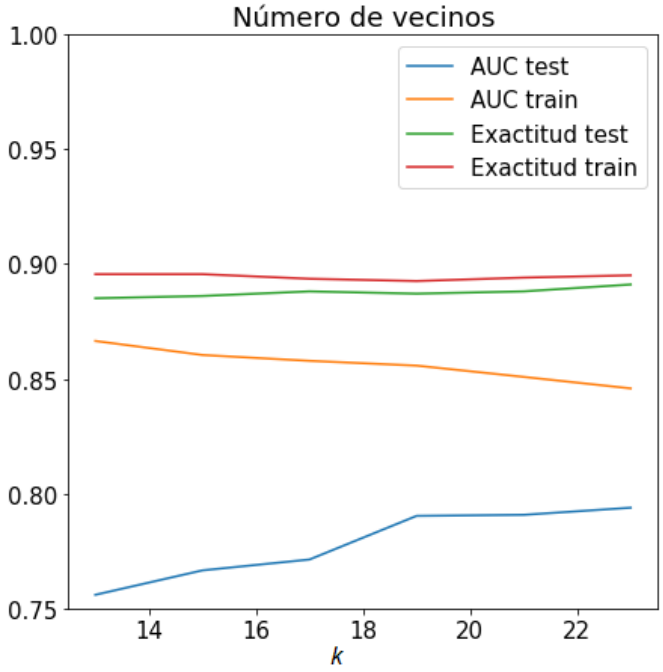


Figura 5.16: Resultado del barrido de parámetros para el modelo de KNN en función del número de vecinos.

El valor es $\frac{\sqrt{N}}{2} = \frac{\sqrt{1001}}{2} = 15.82$, con N la cantidad de datos del con-

junto de entrenamiento. El resultado del barrido de parámetros se presenta en la figura 5.16, donde se observa que no hay grandes variaciones en los valores de la exactitud y, a medida que el número de vecinos aumenta, disminuye el valor de AUC_{train} y aumenta el de AUC_{test}. Probamos entonces empíricamente en torno a los valores del barrido y escogemos el mejor resultado, llegando así a que el caso $k = 15$ nos proporciona el mejor modelo. Por ello, los resultados mostrados en esta sección corresponden al modelo con 15 vecinos.

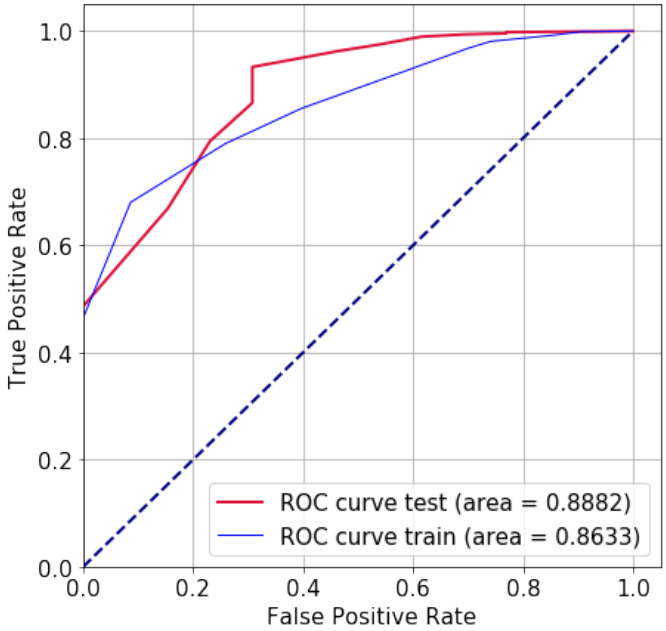
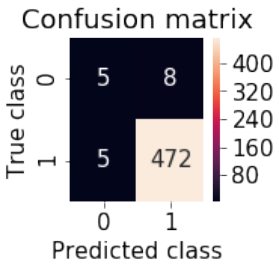
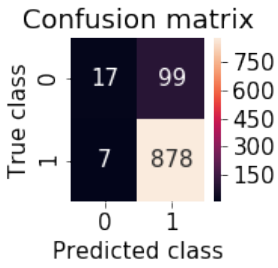


Figura 5.17: Curva ROC con su AUC para el modelo KNN.

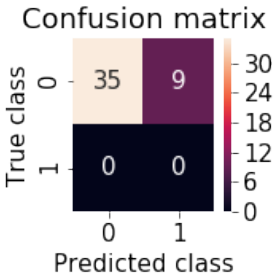
Empezando con el poder predictivo tenemos la figura 5.17. Podemos ver que es el que peor poder predictivo posee hasta el momento, con un ligero overfitting. Es un modelo demasiado básico para poder tratar nuestro problema.



(a) Conjunto de evaluación.



(b) Conjunto de entrenamiento.



(c) Datos sin etiquetar.

Figura 5.18: Matrices de confusión para el modelo KNN.

En la figura 5.18b observamos que el modelo clasifica muy mal los datos malos y muy bien los buenos. Sobre nuestra muestra de evaluación nos devuelve una tasa muy alta de falsos positivos y, en consecuencia, un recall para la clase de datos malos muy bajo. La exactitud del modelo sobre la muestra, como se puede ver en la tabla 5.6, es muy alta. En cambio, no es muy representativa porque el modelo no es bueno, ya que no clasifica bien los datos de la clase mala. Ocurre lo mismo que en el caso del modelo balanceado sin pesos: como la clase de los datos buenos es mucho más numerosa que la de datos malos, clasificar moderadamente bien esta clase dispara el valor de la exactitud.

Clase	***Precisión***	***Recall***	F_1	***Cantidad datos***
0	0.50	0.38	0.43	13
1	0.98	0.99	0.99	477
		Exactitud	0.97	490

Tabla 5.6: Medidas de evaluación asociadas a la matriz de confusión de los datos de evaluación (Figura 5.18a).

Por otro lado, los datos no etiquetados están sorprendentemente bien

clasificados (tabla 5.7), con un recall de 0.80, el más alto hasta ahora. Esto puede deberse a que los datos no etiquetados estén más cerca de datos malos que de datos buenos. Es una consecuencia de cómo funciona este modelo: clasifica la probabilidad en función de la clase de los k vecinos del conjunto de entrenamiento más cercanos. Es decir, si para un dato concreto de los no etiquetados tenemos 10 vecinos malos y 5 buenos en el conjunto de entrenamiento, asignará una probabilidad de 10/15 de pertenecer a la clase de datos malos.

Clase	***Precisión***	***Recall***	F_1	***Cantidad datos***
0	1.00	0.80	0.89	44
		Exactitud	0.80	44

Tabla 5.7: Medidas de evaluación asociadas a la matriz de confusión de los datos no etiquetados (Figura 5.18c).

Por último, la figura 5.19 nos muestra la probabilidad de clasificar los datos como buenos. Observamos que los datos de la clase buena están generalmente bien clasificados, así como los datos no etiquetados. En cambio, en la clase de los datos malos, se observa que ni los del conjunto de entrenamiento (línea verde) tienen una probabilidad menor de 0.5. Por lo tanto, no es una buena opción.

Cabe mencionar el porqué del aspecto "discreto"de las probabilidades de la figura 5.19. Es una consecuencia de cómo funciona el modelo. Al asignar las probabilidades en función de los k vecinos más cercanos, estas probabilidades solo podrán tomar valores en el conjunto $\left\{\frac{i}{k} \mid 0 \leq i \leq k\right\}$.

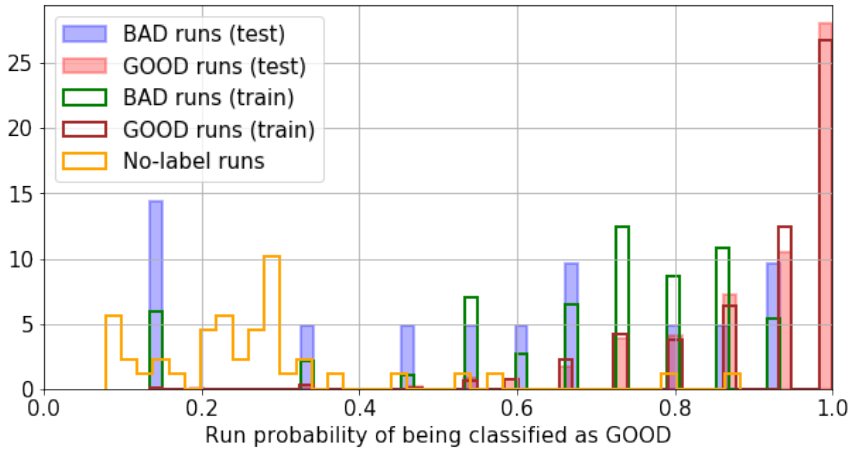


Figura 5.19: Probabilidad de clasificar los datos como buenos para el modelo KNN.

En conclusión, el modelo KNN no es una buena opción para nuestro problema. Al tener una muestra con poco equilibrio entre clases, obtenemos demasiado overfitting cuando consideramos el número de vecinos a partir del criterio estándar. En cambio, varias opciones de k entre 2 y 10 han sido probadas, obteniendo resultados igual de malos. Necesitamos un modelo más complejo.

5.2.3. Árboles de decisión

En esta sección reuniremos los tres modelos asociados a árboles de decisión. Empezaremos por el caso simple, que sólo considera un árbol, y continuaremos con las implementaciones de bagging (Random Forest) y boosting (Boosted Decision Trees).

Usaremos los paquetes *sklearn.tree* y *sklearn.ensemble*. Los parámetros más relevantes en estos modelos son la profundidad del árbol, el número de muestras mínimas para hacer una división (crear un nuevo nodo) y el mínimo de muestras necesarias para que un nodo sea un nodo final, esto es, una hoja. Estos parámetros sirven para controlar el overfitting del modelo y obtener

mejores resultados.

Árbol de decisión simple

Para obtener el mejor modelo hemos realizado un barrido de parámetros con tres conjuntos de cross-validation para 2304 posibles combinaciones de hiperparámetros. En concreto, los valores probados son: $max\_depth \in [2, 10]$, $min\_samples\_leaf \in [2, 20]$ y $min\_samples\_split \in [4, 20]$. Haciendo, tras este barrido, un análisis empírico en torno a la solución proporcionada, llegamos a que el mejor modelo toma los parámetros: $max\_depth = 3$, $min\_samples\_leaf = 10$ y $min\_samples\_split = 8$.

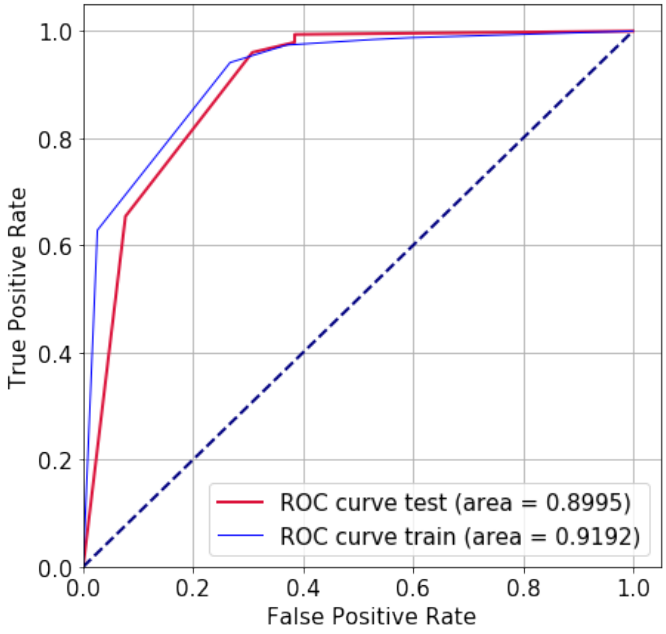
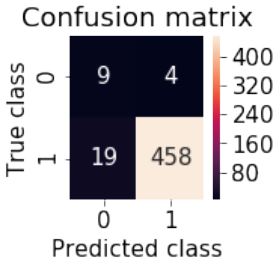
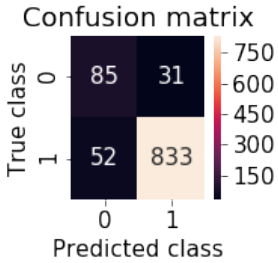


Figura 5.20: Curva ROC con su AUC para el árbol de decisión simple.

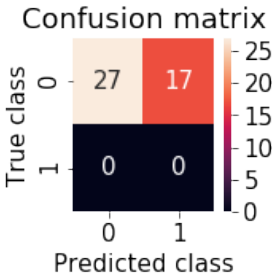
Empezando con el poder predictivo, es el modelo que menor diferencia tiene entre los conjuntos de entrenamiento y evaluación, apenas un 2 %. Además de eso, ambos valores son bastante altos. En consecuencia, este modelo generaliza bien nuestro problema.



(a) Conjunto de evaluación.



(b) Conjunto de entrenamiento.



(c) Datos sin etiquetar.

Figura 5.21: Matrices de confusión para el árbol de decisión.

Fijándonos en las matrices de confusión, en especial en la figura 5.21a podemos ver que clasifica bastante bien los datos de ambas clases. Como se puede ver en la tabla 5.8, el recall de los datos malos es cercano al 70 % y, el de los datos buenos, superior al 95 %. La exactitud del modelo sobre estos datos de entrenamiento es del 95 %, lo que lo hace el mejor hasta el momento. Hemos encontrado un modelo que clasifica bien ambas clases y no solo una de ellas, como nos pasaba hasta ahora.

Clase	***Precisión***	***Recall***	F_1	***Cantidad datos***
0	0.32	0.69	0.44	13
1	0.99	0.96	0.98	477
		Exactitud	0.95	490

Tabla 5.8: Medidas de evaluación asociadas a la matriz de confusión de los datos de evaluación (Figura 5.21a).

En relación a los datos no etiquetados, tabla 5.9, tenemos una exactitud del 61 %, que es menor que la del modelo KNN. Esta última es la mejor hasta el momento.

Clase	***Precisión***	***Recall***	F_1	***Cantidad datos***
0	1.00	0.61	0.76	44
		Exactitud	0.61	44

Tabla 5.9: Medidas de evaluación asociadas a la matriz de confusión de los datos no etiquetados (Figura 5.21c).

En la figura 5.22 podemos ver que las contribuciones más importantes están en 0 y 1. En cambio, también hay dos grupos de datos malos por encima de 0.5 y unos pocos datos buenos en probabilidades muy pequeñas. Por último, en la figura 5.23 incluimos un esquema del árbol asociado a este modelo. Podemos ver que las magnitudes más importantes son el número de entradas de χ^2_n/n, la skewness en la variable ϕ y la media del momento lineal p.

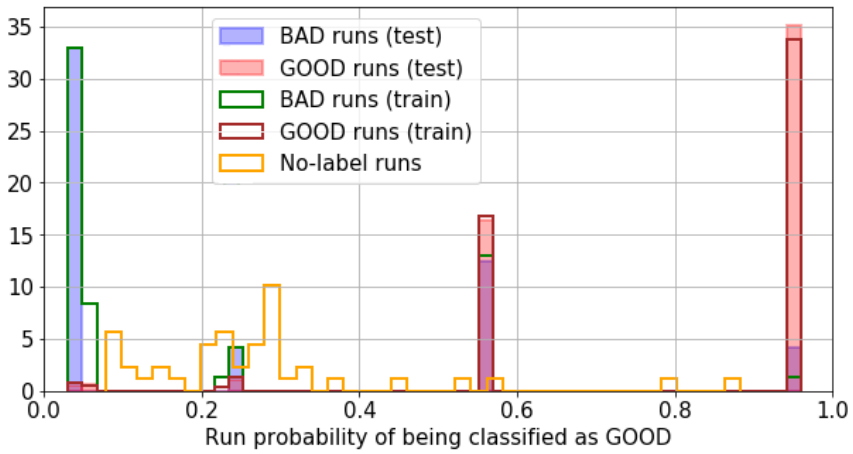


Figura 5.22: Probabilidad de clasificar los datos como buenos para el árbol de decisión

Necesitamos entonces un modelo que mantenga la buena clasificación que tiene el árbol de decisión en los datos buenos y malos pero mejore su actuación sobre los datos sin etiquetar. Probaremos, entonces, con métodos de ensemble relacionados con este modelo.

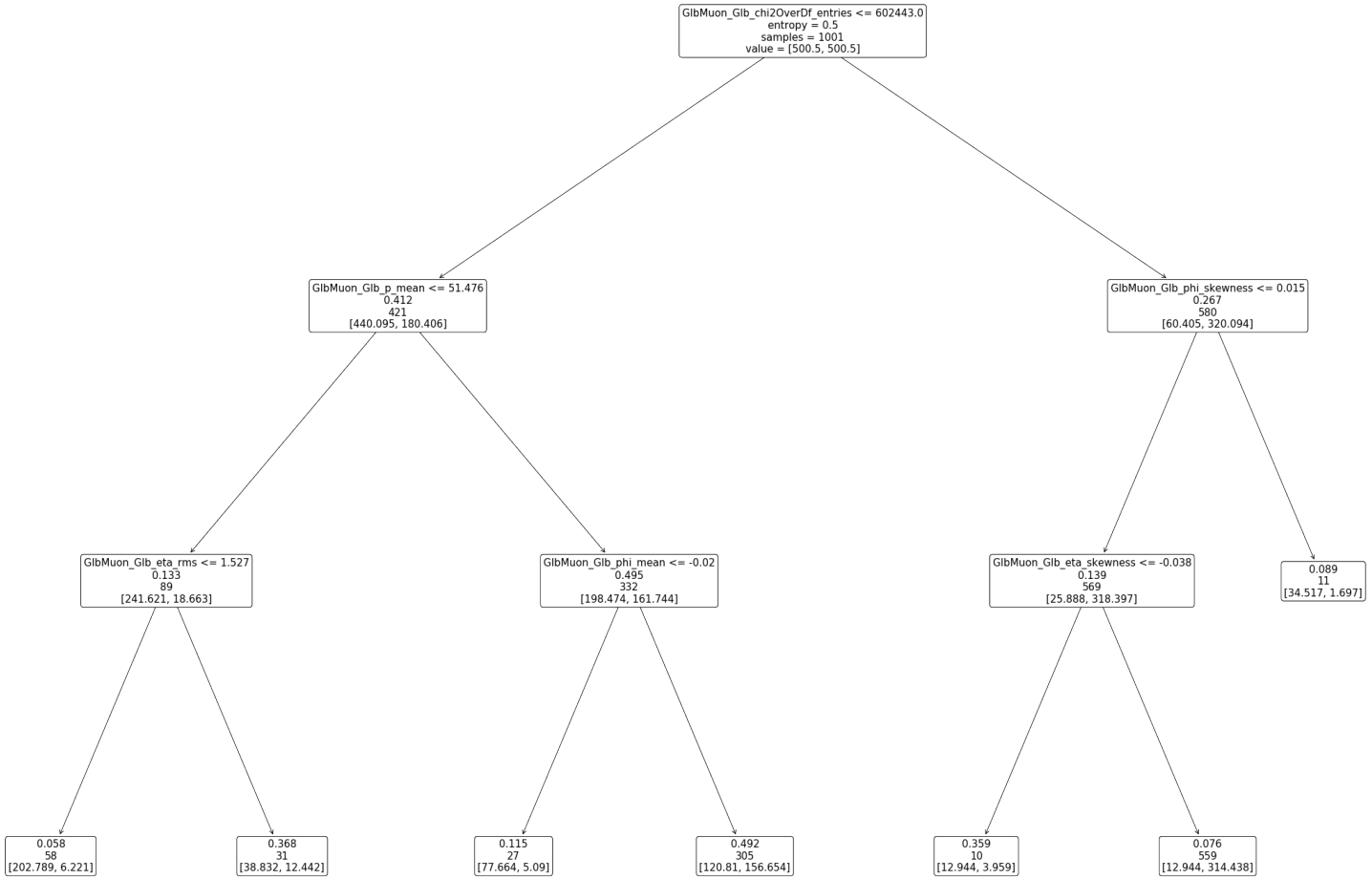


Figura 5.23: Árbol de decisión asociado a nuestro modelo.

Boosted Decision Trees (BDT)

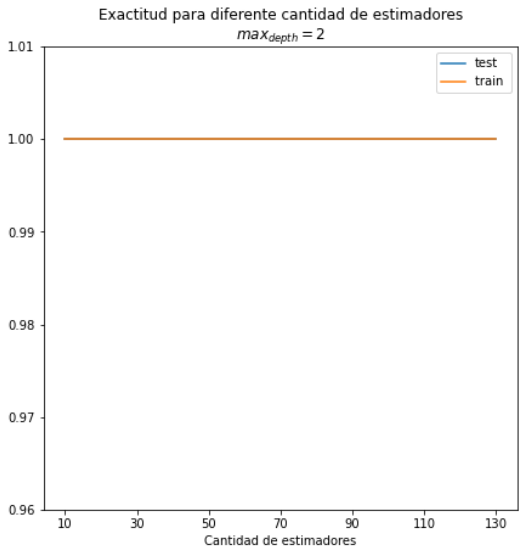


Figura 5.24: Resultado del barrido de parámetros para el modelo BDT.

En este caso realizamos el barrido de parámetros en función del número de estimadores porque empíricamente hemos visto que una profundidad máxima mayor o igual a tres producía demasiado overfitting a la muestra de entrenamiento. En la figura 5.24 podemos ver que este método de búsqueda no nos proporciona información útil, así que nos vemos obligados a probar diferentes combinaciones. Cuando el número de estimadores es muy pequeño, las probabilidades asociadas a la figura 5.27 están más concentradas y no toman los valores extremos de 0 y 1. Por otro lado, si el número de estimadores es muy alto, el AUC_{train} tomaba valor 1 y el de evaluación se aleja cada vez más de la unidad. Al final, hemos llegado a la conclusión de que los mejores parámetros son $n_{estimadores} = 30$ y $depth_{max} = 2$.

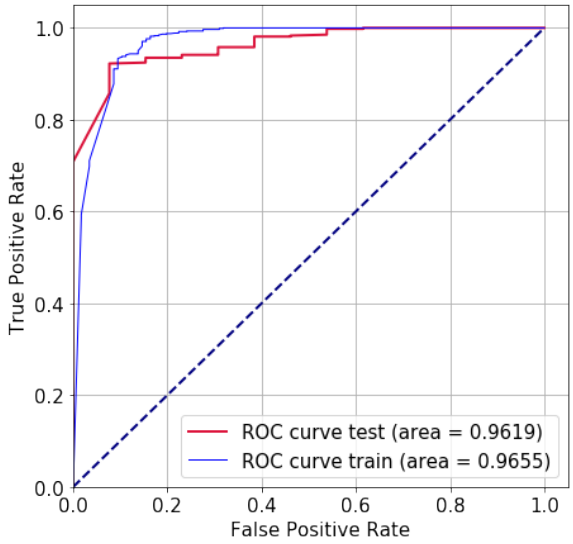
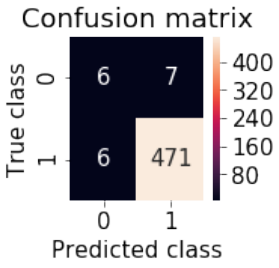
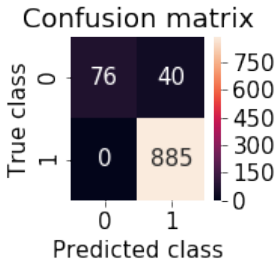


Figura 5.25: Curva ROC con su AUC para el BDT.

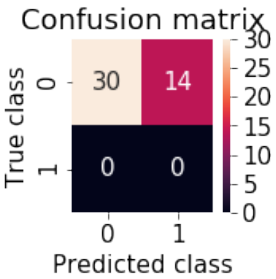
El modelo BDT mejora el poder predictivo del árbol de decisión y reduce el overfitting, como podemos ver en la figura 5.25. Respecto a las matrices de confusión tendremos que, aunque clasifica relativamente bien los datos buenos, ha bajado el recall en la clase de datos malos (tabla 5.10). Por otro lado, sí ha mejorado la clase de los no etiquetados, como podemos ver en la tabla 5.11 y la figura 5.26c. No obstante, no ha conseguido superar al modelo KNN en la clasificación de los mismos.



(a) Conjunto de evaluación.



(b) Conjunto de entrenamiento.



(c) Datos sin etiquetar.

Figura 5.26: Matrices de confusión para el BDT.

Clase	***Precisión***	***Recall***	F_1	***Cantidad datos***
0	0.50	0.46	0.48	13
1	0.99	0.99	0.99	477
		Exactitud	0.97	490

Tabla 5.10: Medidas de evaluación asociadas a la matriz de confusión de los datos de evaluación (Figura 5.26a).

Clase	***Precisión***	***Recall***	F_1	***Cantidad datos***
0	1.00	0.68	0.81	44
		Exactitud	0.68	44

Tabla 5.11: Medidas de evaluación asociadas a la matriz de confusión de los datos no etiquetados (Figura 5.26c).

Por último, podemos ver en la figura 5.27 que efectivamente clasifica bien los datos buenos, estando casi todos en torno a la probabilidad 1. En cambio, también se ve que los datos malos no tienen una distribución muy polarizada, ni en los del conjunto de evaluación ni en los de entrenamiento. Y, para los datos sin etiquetar, tenemos que también se clasifican mayoritariamente a la izquierda del 0.5.

En conclusión, el modelo BDT mejora el poder predictivo, la clasificación de los datos buenos y la de los no etiquetados respecto al árbol de decisión. En cambio, clasifica peor los datos malos.

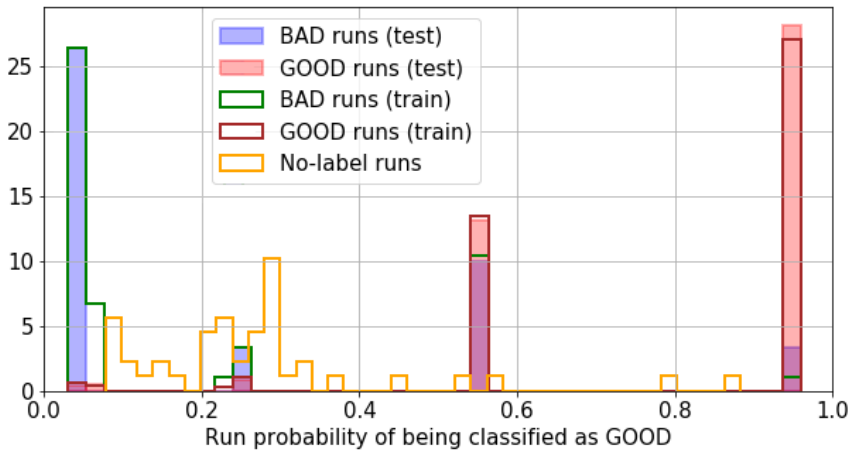
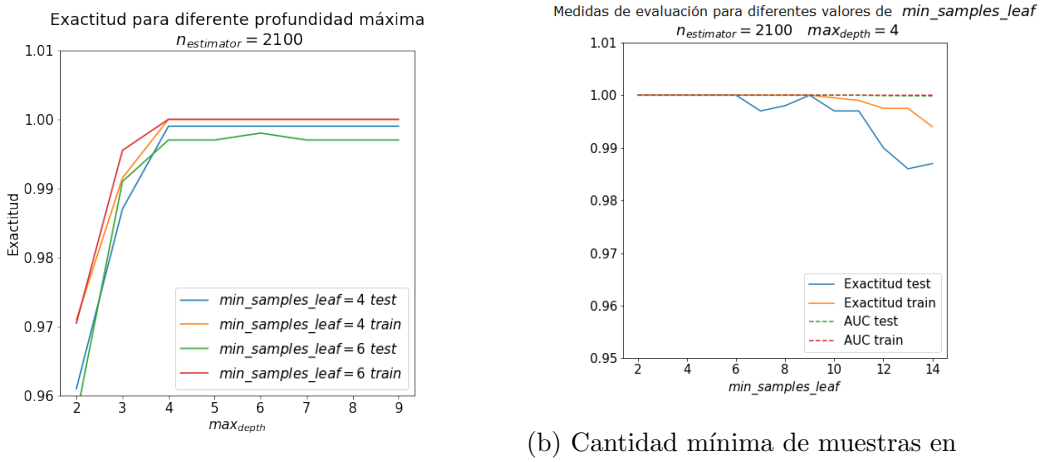


Figura 5.27: Probabilidad de clasificar los datos como buenos para el BDT

Random Forest (RF)



(a) Profundidad máxima.

(b) Cantidad mínima de muestras en una hoja.

Figura 5.28: Grid Search en función de la profundidad máxima y de la cantidad mínima de muestras en una hoja (nodo final).

En este caso, el estudio de parámetros lo hemos hecho con tres conjuntos de cross-validation y diferentes combinaciones de los hiperparámetros profundidad máxima ($max_{depth} \in [4, 15]$), muestras mínimas para que un nodo

pueda ser una hoja ($min\_samples\_leaf \in [5, 20]$) y número de estimadores ($n\_estimators$ de 100 a 2000 en pasos de 10). Se presentan en la figura 5.28 cómo evoluciona la exactitud del modelo con la profundidad máxima para diferentes valores de $min\_samples\_leaf$ (figura 5.28a) y cómo varían la exactitud y el AUC en función de $min\_samples\_leaf$ con una profundidad máxima fija (5.28b). En la primera gráfica vemos que, a partir de $max_{depth} = 4$, la exactitud no varía demasiado. Por lo tanto, tomaremos esta profundidad como referencia para la búsqueda manual posterior. Por otro lado, la segunda gráfica nos dice que la exactitud disminuye con valores altos de $min\_samples\_leaf$. Empezaremos entonces en 6 en la búsqueda posterior. Por último, hemos escogido en torno a 2000 el número de estimadores por ser la propuesta inicial del barrido de parámetros. Al final, los parámetros del modelo escogidos son: $n_{estimators} = 2100$, $max_{depth} = 4$ y $min\_samples\_leaf = 13$.

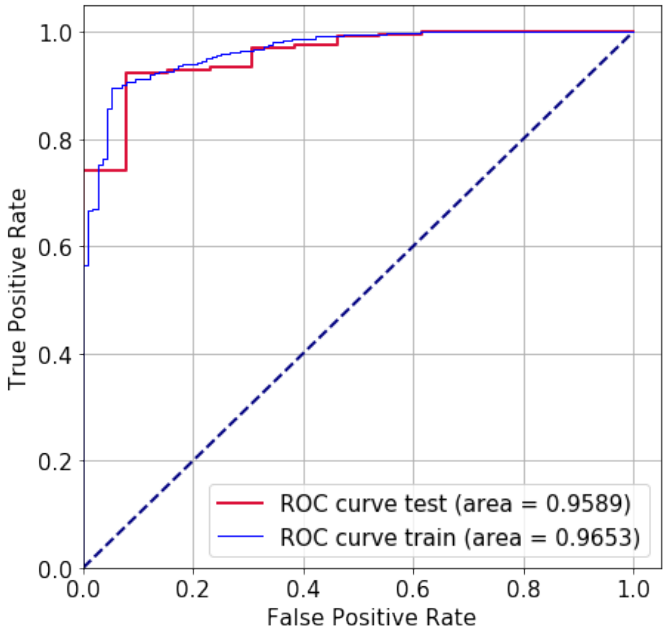
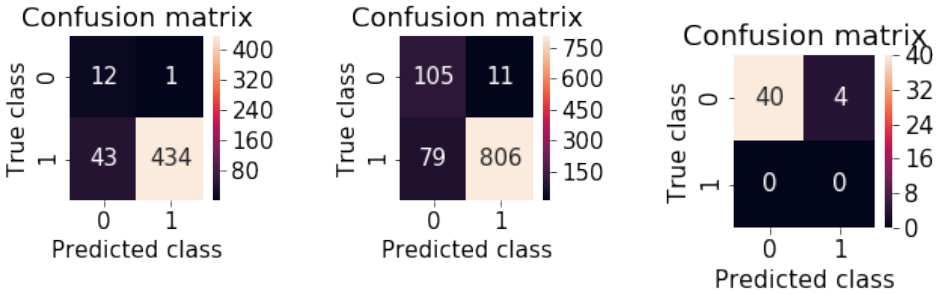


Figura 5.29: Curva ROC con su AUC para el RF.

Como en los casos anteriores, la figura 5.29 nos proporciona el poder

predictivo del modelo. La diferencia entre AUC_{test} y AUC_{train} es de un 0.65 %, lo que quiere decir que el overfitting es despreciable. Obtenemos así una mejoría respecto al árbol simple en este aspecto. Fijándonos en las matrices de confusión de la figura 5.30, en especial en 5.30a, observamos que sólo tenemos un falso positivo. Respecto a los falsos negativos tenemos 43, por lo que el recall de las clases de datos malos y buenos es de 0.92 y 0.91, respectivamente (tabla 5.12). Esto constituye una mejoría de cualquiera de los modelos anteriores para estos parámetros.



(a) Conjunto de evaluación.

(b) Conjunto de entrenamiento.

(c) Datos sin etiquetar.

Figura 5.30: Matrices de confusión para el RF.

Clase	***Precisión***	***Recall***	F_1	***Cantidad datos***
0	0.22	0.92	0.35	13
1	1.00	0.91	0.95	477
		Exactitud	0.91	490

Tabla 5.12: Medidas de evaluación asociadas a la matriz de confusión de los datos de evaluación (Figura 5.30a).

Por otro lado, para los datos no etiquetados tenemos la figura 5.30c y la tabla 5.13. Podemos ver en ellas que solamente 4 datos son mal clasificados, haciendo que el recall sea 0.91. Por lo tanto, este modelo es el que mejor clasifica los datos, sin importar su clase de procedencia.

Clase	***Precisión***	***Recall***	F_1	***Cantidad datos***
0	1.00	0.91	0.95	44
		Exactitud	0.91	44

Tabla 5.13: Medidas de evaluación asociadas a la matriz de confusión de los datos no etiquetados (Figura 5.30c).

Por último, en la figura 5.31, se observa cómo los datos están bien clasificados. Tenemos una barra de datos malos mal clasificados, pero con una probabilidad cercana a 0.6, que no es demasiado alta. Para los datos buenos, podemos ver cómo del 0.5 hacia la izquierda los grupos de datos acumulan menor probabilidad. Los datos sin etiquetar también se agrupan a la izquierda.

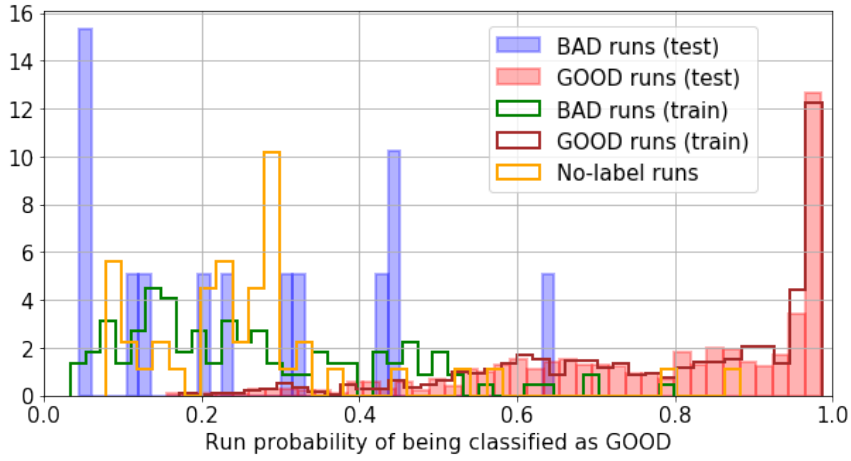


Figura 5.31: Probabilidad de clasificar los datos como buenos para el RF

En resumen, hemos conseguido un modelo con un overfitting despreciable, que clasifica bien los tres grupos de datos. Por lo tanto, este sí que es apropiado para tratar nuestro problema.

5.3. Resumen y comparación de modelos

Recapitulando sobre los modelos, tenemos que:

1. El modelo logístico sin pesos clasificaba los datos mayoritariamente como buenos, siendo una mala opción para datos malos y sin etiquetar. Además, la diferencia entre el poder de predicción era muy grande en los conjuntos de entrenamiento y evaluación.

2. El modelo logístico con pesos mejoraba la actuación del primero respecto a los datos malos y sin etiquetar. En cambio, empeoraba sustancialmente la clasificación de los datos buenos y no mejoraba la diferencia de poder predictivo.

3. El modelo KNN tampoco era apropiado, clasificaba bien los datos buenos y los que están sin etiquetar. En cambio, su actuación no era buena sobre los datos malos.

4. El árbol de decisión simple reducía el overfitting sustancialmente y clasificaba mejor los datos buenos que los malos, pero ambos de manera aceptable. Los datos sin etiquetar eran los que peor clasificados estaban.

5. El modelo BDT eliminaba el overfitting, tenía un buen poder de predicción y clasificaba bien los datos buenos. Mejoraba el modelo del árbol de decisión simple para los datos sin etiquetar pero no era muy bueno sobre los datos malos.

6. El modelo RF eliminaba el overfitting y tenía un buen poder de predicción. Además, clasificaba bastante bien los datos de todas las clases.

Teniendo en cuenta todo lo anterior, el modelo más apropiado para nuestro problema es el Random Forest.

Capítulo 6

Conclusiones

Durante el transcurso del trabajo nos hemos encontrado con varias cuestiones. Para empezar, el principal problema de nuestros datos es el desequilibrio entre clases: el hecho de que tengamos demasiados datos clasificados por los expertos como buenos y pocos datos clasificados por los expertos como malos hace que, de forma general, los modelos tiendan a clasificar bien los datos del primer tipo y no detecten del todo los del segundo. Este problema lo hemos abordado realizando pruebas de cross-validation en el barrido de parámetros. Por otro lado, también es un problema que haya conjuntos de datos sin valores asociados o con valores infinitos. En este caso, lo que hemos hecho es descartar directamente estos conjuntos de datos. Otro acercamiento sería sustituir los casos de $\pm\infty$ por valores del tipo ± 9999. Realizar un análisis sobre este aspecto sería una posibilidad para ampliar y mejorar esta investigación.

Respecto a las conclusiones propias del análisis, llegamos a que el mejor modelo para afrontar el problema es el Random Forest. En concreto, obtenemos un modelo bastante completo con un alto poder predictivo, siendo $AUC_{train} = 0.9653$ y $AUC_{test} = 0.9589$. Además, tiene un recall de los datos buenos de 0.91 y de 0.92 para los malos en el conjunto de evaluación. Para los datos no etiquetados este recall es de 0.91. En definitiva, clasifica bien los datos de todas las clases. Concluimos así que la solución a nuestro problema

es bastante buena. No obstante, siempre podríamos refinarla con los datos análogos de próximos Runs. En concreto, necesitaríamos más datos malos.

Por último, si quisiéramos ampliar el estudio, otra opción para certificación de datos empleando aprendizaje automático sería analizar los datos asociados a los Lumisections de cada run. El enfoque aquí sería diferente porque no podríamos usar técnicas de aprendizaje supervisado, ya que estos periodos de tiempo no están etiquetados por expertos. En este caso, la estrategia que está siendo empleada es generar, con autoencoders, las distribuciones asociadas a cada LS.

Apéndice A

Tablas de los resúmenes estadísticos

- χ^2 sobre los grados de libertad:

χ^2/n	**Entries**			**Media**			**Valor cuadrático medio**			**Skewness**			**Kurtosis**		
Grupo	G	B	NL	G	B	NL	G	B	NL	G	B	NL	G	B	NL
Media	2.32e+06	2.66e+05	5.24e+04	1.34e+05	1.33e+05	153	1.06e+08	9.98e+07	5.10e+04	5.48e-03	0.037	0.275	-2.96	-2.85	-1.48
Desviación estándar	2.81e+06	4.29e+05	2.09e+05	3.66e+06	1.46e+06	842	2.57e+09	1.09e+09	2.58e+05	0.161	0.207	0.733	1.12	0.826	2.95
Mínimo	126	3	2	1.21	0.953	0.804	1.28	0.208	0.0840	-2.78e-03	-0.025	-1.19	-3.00	-3.00	-3.00
Máximo	1.89e+07	2.27e+06	1.14e+06	1.34e+08	1.66e+07	5570	9.33e+10	1.24e+10	1.67e+6	5.67	1.49	3.69	5.67	3.09	15.2

Tabla A.1: Resumen estadístico para la variable χ^2 sobre los grados de libertad. G, B y NL corresponden, respectivamente a good, bad y no label.

- Pseudorapidez η:

Eta η	**Entries**			**Media**			**Valor cuadrático medio**			**Skewness**			**Kurtosis**		
Grupo	G	B	NL	G	B	NL	G	B	NL	G	B	NL	G	B	NL
Media	2.32e+06	2.66e+05	5.24e+04	0.0209	6.35e-03	-0.0185	1.46	1.42	1.13	-0.0174	-3.50e-03	-0.052	-1.28	-1.26	-1.33
Desviación estándar	2.81e+06	4.29e+05	2.09e+05	0.0150	0.0752	0.427	5.62e-02	0.226	0.474	0.0129	0.0879	0.333	0.0699	0.214	0.508
Mínimo	126	3	2	-0.136	-0.226	-1.44	1.28	0.620	0.268	-0.188	-0.649	-0.864	-1.56	-1.83	-2.65
Máximo	1.89e+07	2.27e+06	1.14e+06	0.194	0.500	1.58	1.68	1.84	2.26	0.127	0.245	0.731	-0.976	-0.635	0.278

Tabla A.2: Resumen estadístico para la variable eta. G, B y NL corresponden, respectivamente, a good, bad y no label.

- Ángulo ϕ:

Phi ϕ	**Entries**			**Media (rad)**			**Valor cuadrático medio (rad)**			**Skewness**			**Kurtosis**		
Grupo	G	B	NL	G	B	NL	G	B	NL	G	B	NL	G	B	NL
Media	2.32e+06	2.66e+05	5.24e+04	1.22e-04	-7.91e-03	-0.041	1.82	1.81	1.64	2.93e-03	4.69e-03	0.115	-1.20	-1.18	-1.44
Desviación estándar	2.81e+06	4.29e+05	2.09e+05	0.0125	0.0786	0.419	0.0106	0.0563	0.430	8.66e-03	0.087	0.326	0.013	0.121	0.284
Mínimo	126	3	2	-0.160	-0.525	-1.05	1.76	1.37	0.188	-0.0465	-0.393	-0.717	-1.31	-1.50	-2.02
Máximo	1.89e+07	2.27e+06	1.14e+06	0.082	0.231	1.29	1.87	1.90	2.36	0.126	0.674	1.12	-1.13	-0.13	-0.73

Tabla A.3: Resumen estadístico para la variable phi. G, B y NL corresponden, respectivamente a good, bad y no label.

- Momento lineal p:

p	**Entries**			**Media** (GeV/c)			**Valor cuadrático medio** (GeV/c)			**Skewness**			**Kurtosis**		
Grupo	G	B	NL	G	B	NL	G	B	NL	G	B	NL	G	B	NL
Media	2.32e+06	2.66e+05	5.24e+04	59.2	51.9	71.2	1890	360	73.4	0.153	0.630	1.03	-2.47	-0.016	1.33
Desviación estándar	2.81e+06	4.29e+05	2.09e+05	22.7	14.9	35.4	12600	789	106	0.352	0.929	1.35	1.46	5.33	4.97
Mínimo	126	3	2	10.1	17.9	14.1	40.8	31.3	0.248	-4.4e-05	-7.30e-03	-0.687	-3.00	-3.00	-3.00
Máximo	1.89e+07	2.27e+06	1.14e+06	727	159	220	3.90e+05	6.80e+03	710	2.48	3.92	6.88	11.0	25.5	21.1

Tabla A.4: Resumen estadístico para la variable momento lineal p. G, B y NL corresponden, respectivamente a good, bad y no label.

- Momento transverso p_T:

p_T	**Entries**			**Media**(GeV/c)			**Valor cuadrático medio**(GeV/c)			**Skewness**			**Kurtosis**		
Grupo	G	B	NL	G	B	NL	G	B	NL	G	B	NL	G	B	NL
Media	2.32e+06	2.66e+05	5.24e+04	28.0	25.2	44.0	733	155	33.8	0.183	0.766	1.04	-1.53	3.44	4.19
Desviación estándar	2.81e+06	4.29e+05	2.09e+05	6.02	5.60	28.3	3370	345	34.6	0.459	1.38	1.37	4.54	13.3	10.8
Mínimo	126	3	2	4.62	12.0	9.40	12.2	11.0	0.203	-4.60e-05	-0.258	-1.02	-3.00	-3.00	-3.00
Máximo	1.89e+07	2.27e+06	1.14e+06	174	52.3	132	63000	2500	199	4.33	7.52	4.63	51.8	75.3	51.3

Tabla A.5: Resumen estadístico para la variable momento lineal transverso p_T. G, B y NL corresponden, respectivamente a good, bad y no label.

Bibliografía

[1] CERN. (s.f) The Standard Model. Recuperado el 22 de Marzo de 2021 de https://home.cern/science/physics/standard-model

[2] CERN. (2017) LHC: the guide . Recuperado el 10 de Marzo de 2021 de https://home.cern/resources/brochure/cern/lhc-guide

[3] Povh, B., Lavelle, M. (2009). *Particles and nuclei*. Springer.

[4] Alonso, R. (2012). Sobre el origen de la estructura generacional de las partículas elementales. *UAM Gazette*. Recuperado el 23 Marzo 2021, de https://uam.es/ss/Satellite/es/1242652962055/1242661792404/articulo/articulo/Sobre_el_origen_de_la_estructura_generacional_de_las_particulas_elementales.htm

[5] CMS Collaboration. (s.f) CMS Luminosity. Public Results.Recuperado el 15 de Junio de 2021, de https://twiki.cern.ch/twiki/bin/view/CMSPublic/LumiPublicResults#Multi_year_plots

[6] HiLumi.(s.f) The HL-LHC project. Recuperado el 17 de Junio de 2021 de https://hilumilhc.web.cern.ch/content/hl-lhc-project.

[7] P.A. Zyla et al.(2020) *Particle Data Group*, Prog. Theor. Exp. Phys. 2020, 083C01

[8] Villatoro, F.R (2017). CMS no observa el efecto magnético quiral sugerido por STAR y ALICE. Recuperado el 25 de

Marzo de 2021 de https://francis.naukas.com/2017/06/21/cms-no-observa-efecto-magnetico-quiral-sugerido-star-alice/

[9] CERN. (s.f) *CMS*.Recuperado el 25 de Marzo de 2021, de https://home.cern/science/experiments/cms.

[10] Davis, S. (2016). Interactive Slice of the CMS detector. CERN Document Server. Recuperado el 25 de Marzo de 2021, de https://cds.cern.ch/record/2205172/?ln=es.

[11] CMS Collaboration. (2006). CMS Physics: Technical Design Report Volume 1: Detector Performance and Software. CMS Collaboration.

[12] Cid Vidal, X., Cid Manzano, R. (s.f) LHC p collisions. Taking a closer look at LHC. Consultado el 20 de junio de 2021 en http://lhc-closer.es/taking_a_closer_look_at_lhc/0.lhc_p_collisions.

[13] CMS Collaboration. (s.f) Detector.Recuperado el 21 de Marzo de 2021, de https://cms.cern/detector.

[14] CMS Collaboration. (s.f) Tracking. Recuperado el 21 de Marzo de 2021, de https://cms.cern/detector/identifying-tracks.

[15] CMS Collaboration. (s.f) Energy of electrons and photons (ECAL).Recuperado el 21 de Marzo de 2021, de https://cms.cern/detector/measuring-energy/energy-electrons-and-photons-ecal.

[16] CMS Collaboration. (s.f) Energy of hadrons (HCAL).Recuperado el 21 de Marzo de 2021, de https://cms.cern/detector/measuring-energy/energy-hadrons-hcal

[17] Mc Cauley, T. (2020) Photo of the CMS detector with a rendering of a real event superimposed to-scale. [Fotografía]. Extraído el 21 de Junio de 2021 de https://cds.cern.ch/record/2723292?ln=es.

[18] CMS Collaboration. (s.f) Detecting muons.Recuperado el 21 de Marzo de 2021, de https://cms.cern/detector/detecting-muons

[19] A. Benaglia (2014) The CMS ECAL performance with examples, JINST 9 C02008.

[20] CMS Collaboration (2010) Performance of the CMS Hadron Calorimeter with Cosmic Ray Muons and LHC Beam Data. JINST 5 T03012. Disponible en: https://arxiv.org/pdf/0911.4991.pdf

[21] Constantini, S. (2012) Uniformity and Stability of the CMS RPC Detector at the LHC. CMS Collaboration

[22] Halyo, V. et al. (2013) Massively Parallel Computing and the Search for Jets and Black Holes at the LHC. Nucl. Instrum. Methods Phys. Res., A 744 (2014) 54-60. Disponible en: https://arxiv.org/pdf/1309.6275.pdf

[23] Guio, Federico. (2014). The CMS data quality monitoring software: experience and future prospects. Journal of Physics: Conference Series. 513. 032024. Disponible en: https://doi.org/10.1088/1742-6596/513/3/032024.

[24] Azzolini, V., Broen van, B., Bugelskis, D., Hreus, T., Maeshima, K., & Fernandez, J. et al. (2019). The Data Quality Monitoring Software for the CMS experiment at the LHC: past, present and future. EPJ Web Of Conferences, 214, 02003. Disponible en: https://doi.org/10.1051/epjconf/201921402003

[25] Géron, A. (2019) *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow* O'Reilly. 978-1-492-03264-9.

[26] Alpaydin, E.(2014). *Introduction to Machine Learning* (3ª Ed.)

[27] Pedregosa, F. et al. (2011) Scikit-learn: Machine Learning in Python. JMLR 12, pp. 2825-2830.

[28] L., J. H. Friedman, R. A. Olshen,& C. J. Stone. (1984) *Classification and Regression Trees*. Belmont, CA: Wadsworth International Group.

[29] Quinlan, J. R. (1986). *Induction of Decision Trees*. Machine Learning 1:81?106.

[30] Quinlan, J. R. (1993). *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

[31] Kalman, R. E.,(1960) A New Approach to Linear Filtering and Prediction Problems. Transactions of the ASME - Journal of Basic Engineering Vol. 82: pag. 35-45.

[32] What is Confusion Matrix and Advanced Classification Metrics? .(2019). Data science and machine learning. Consultado el 1 de Mayo de 2021, en `https://manisha-sirsat.blogspot.com/2019/04/confusion-matrix.html`

[33] Diva, D. (2018). Skewness and Kurtosis: 2 Important Statistics terms you need to know in Data Science. Codeburst.io. Consultado el 14 de Mayo 2021, en `https://codeburst.io/2-important-statistics-terms-you-need-to-know-in-data-science-\skewness-and-kurtosis-388fef94eeaa`

[34] Nagpal, A. (2017). L1 and L2 Regularization Methods. Towards data science. Consultado el 27 de ma-

yo de 2021, en https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c.