

**MÁSTER UNIVERSITARIO EN CIENCIA DE DATOS**



VNIVERSITAT  
DE VALÈNCIA

**TRABAJO DE FIN DE MÁSTER**

**MACHINE LEARNING EN EL ESTUDIO  $t\bar{t}$  DEL  
EXPERIMENTO ATLAS**

**AUTORA:**  
**ÁNGELA GARCÍA MÍNGUEZ**

**TUTORES:**  
**JORDI MUÑOZ MARI**  
**JOSÉ SALT CAIROL**

**JULIO, 2020**



## MÁSTER UNIVERSITARIO EN CIENCIA DE DATOS

### TRABAJO DE FIN DE MÁSTER

### MACHINE LEARNING EN EL ESTUDIO $t\bar{t}$ DEL EXPERIMENTO ATLAS

**AUTORA:**  
**ÁNGELA GARCÍA MÍNGUEZ**

**TUTORES:**  
**JORDI MUÑOZ MARI**  
**JOSÉ SALT CAIROL**

**JULIO, 2020**

---

#### TRIBUNAL:

PRESIDENTE/A:

VOCAL 1:

VOCAL 2:

**FECHA DE DEFENSA:**

**CALIFICACIÓN:**

# Índice

<b>1. Introducción: Machine learning y física de altas energías</b>	<b>2</b>
<b>2. Física del experimento ATLAS</b>	<b>3</b>
2.1. Introducción a la física de partículas . . . . .	3
2.2. Modelo Estándar . . . . .	5
2.3. Colisión protón protón y nueva física . . . . .	7
2.4. Experimento ATLAS . . . . .	9
<b>3. Problema HEPMASS</b>	<b>12</b>
3.1. Introducción al problema . . . . .	12
3.2. Descripción del Dataset . . . . .	13
3.3. Análisis exploratorio de datos . . . . .	16
3.3.1. Distribuciones estadísticas . . . . .	16
3.3.2. Matriz de correlaciones . . . . .	19
3.3.3. Diferencias en cuanto al etiquetado <i>btag</i> . . . . .	20
3.4. Aplicación de Machine learning . . . . .	24
3.4.1. Resumen teórico de las técnicas de machine learning . . . . .	25
3.4.1.1. Métricas para clasificación binaria . . . . .	25
3.4.1.2. Algoritmos de machine learning . . . . .	26
3.4.1.3. Técnicas de selección de características . . . . .	30
3.4.2. Reducción y selección de características mediante <i>machine learning</i> . . . . .	31
3.4.3. Comparativa de resultados para los diferentes algoritmos . . . . .	34
3.4.4. Error sistemático del b-tag . . . . .	37
3.4.5. Redes Neuronales Parametrizadas . . . . .	39
<b>4. Lineas futuras</b>	<b>42</b>
<b>5. Conclusiones</b>	<b>43</b>
<b>6. Anexo</b>	<b>44</b>
6.1. Tablas de resultados . . . . .	44
6.2. Ejemplo de código utilizado . . . . .	49

## 1. Introducción: Machine learning y física de altas energías

El comienzo del siglo XXI ha supuesto una evidente revolución tecnológica en todas las áreas del conocimiento. En la última década, los conceptos de inteligencia artificial y *machine learning* se han extendido hacia todas las ciencias y se han vuelto indispensables para resolver problemas complejos, desde la clasificación de galaxias en astronomía hasta la determinación de la historia evolutiva de las especies en el campo de la genética. Estas nuevas tecnologías han creado un nuevo paradigma en la física: el uso de la ciencia de datos como complemento a la investigación tradicional.

La investigación en física de partículas requiere de experimentos que operan a altas energías e intensidades. Se producen muestras extremadamente grandes y ricas en información. El uso de técnicas de aprendizaje automático está revolucionando la manera en la que se interpretan las muestras de datos, aumentando enormemente el potencial de descubrimientos en los experimentos.

Algunos ejemplos recientes del uso de la ciencia de datos en la física de altas energías serían:

- Filtro de datos en el LHC *Large Hadron Collider*: análisis en tiempo real de datos para seleccionar los más interesantes. [1]
- Búsqueda de partículas exóticas. [2]
- Análisis de física a partir de los datos recolectados en los detectores. En el complejo del CERN (*Conseil Européen pour la Recherche Nucléaire*) son recogidas cantidades inmensas de señales, que posteriormente necesitan de algoritmos complejos para su análisis. [1]
- Generación de datos simulados mediante redes neuronales generativas (como GANs), los resultados de los experimentos reales se suelen simular a priori. [3]

En este trabajo vamos a aplicar técnicas de aprendizaje automático a la búsqueda de partículas exóticas, concretamente en el estudio  $t\bar{t}$  del experimento ATLAS del LHC. Este experimento trata de encontrar una partícula más allá del *Standard model* que de lugar a los quarks  $t\bar{t}$ .

Hemos dividido el documento en cuatro secciones. En la primera, explicaremos los conceptos básicos de física de partículas, el modelo estándar y las colisiones protón-protón. En la segunda recogeremos el trabajo realizado, éste ha consistido en el análisis y la aplicación de modelos de *machine learning* al conjunto de datos simulados: HEPMASS [4]. Principalmente hemos entrenado clasificadores para diferenciar los sucesos en los que se produce una resonancia o partícula. Adicionalmente hemos estudiado la influencia del etiquetado de jets y el uso de modelos parametrizados. En la tercera sección hemos descrito cuales serían los pasos posteriores a este trabajo, en la cuarta recopilamos las conclusiones. Hemos añadido también una sección de Anexo donde se recopilan todas las tablas de resultados.

## 2. Física del experimento ATLAS

En esta sección analizaremos la física del experimento ATLAS, el cuál trata de la búsqueda de nueva física. Comenzaremos introduciendo conceptos de física de partículas para continuar con la explicación del modelo estándar y otros modelos de nueva física cuando se trata de la colisión protón-protón. Por último explicaremos que es el ATLAS y como qué datos se recogen. Todo esto nos hará entender la física que hay detrás del conjunto de datos HEPMASS, sobre los cuales se ha trabajado y que se analizarán en la segunda parte de la memoria.

### 2.1. Introducción a la física de partículas

La naturaleza de la materia fue uno de los temas que acompañaron al nacimiento de la filosofía. La discusión entre si la materia es continua o está hecha de unidades discretas surgió en muchas culturas antiguas como Grecia (Demócrito, Leucipo), Roma (Lucrécio) e India (los jainistas, las escuelas Nyaya y Vaisheshika) [5]. Sin embargo no fue hasta el siglo XIX cuando los científicos abrazaron y refinaron la idea de que la materia es discreta, posteriormente al modelo atómico de Dalton en el siglo XVIII y con el descubrimiento del electrón por J.J. Thomson en 1887 [6]. Hoy en día la física considera que la materia es discreta y está formada por unidades indivisibles o partículas.

Así pues, **definimos física de partículas al estudio de los componentes fundamentales de la materia y de las interacciones entre ellos. Dado que los experimentos de física de partículas se llevan a cabo en aceleradores de partículas, podemos nombrarla también como física de altas energías.**

En la Figura 1 vemos el campo de estudio al que se dedica la física de partículas: escalas menores a  $10^{-15} m$ . Estas longitudes corresponden a niveles subatómicos, donde encontramos a las partículas fundamentales que componen la materia.

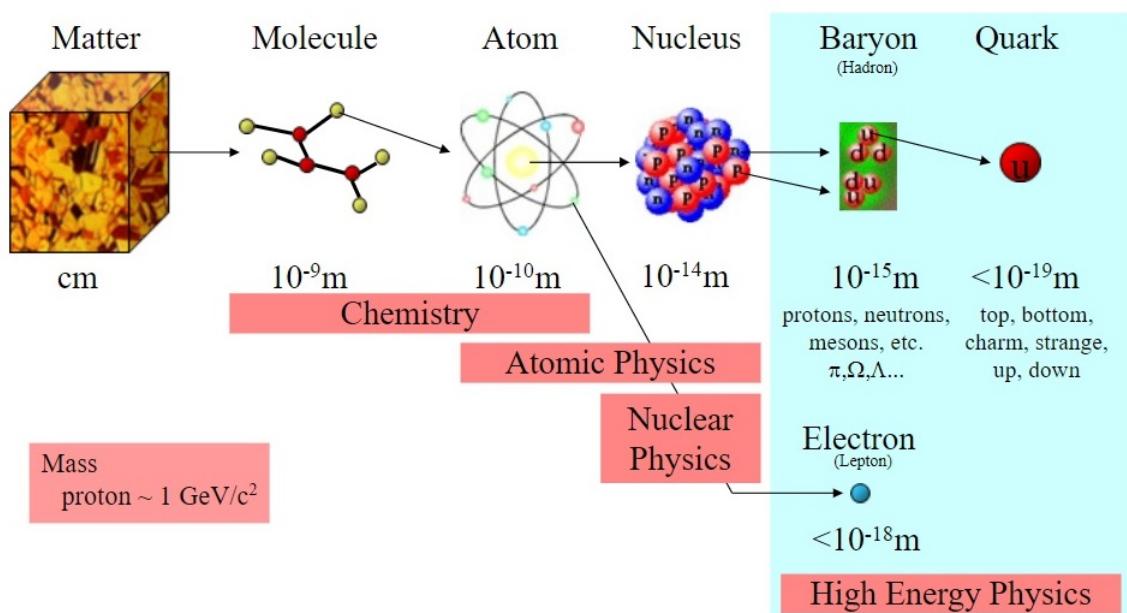


Figura 1: Estructura de la materia y sus campos de estudio [7].

Los experimentos en colisionadores de partículas implican por tanto física a muy bajas escalas y muy altas energías. En este extremo encontramos una dualidad entre materia y energía: la materia se puede transformar en energía y viceversa. Matemáticamente lo vemos en la fórmula relativista [8] :

$$E^2 = (mc^2)^2 + (pc)^2$$

donde :

- $E$  es la energía total de un objeto.
- $m$  es la masa.
- $c$  es la velocidad de la luz,  $mc^2$  sería pues la energía del objeto en reposo.
- $p$  es el momento del objeto, es decir su masa multiplicada por su velocidad,  $(pc)^2$  sería por tanto la energía del objeto asociada a su movimiento.

Esto no es más que la expansión de la famosa ecuación  $E = mc^2$ . A lo largo de todo el trabajo por comodidad vamos a expresar las masas como unidades de energía  $m = E/c^2$ . Por ejemplo, la masa del protón, que es de  $1,67 \cdot 10^{-27}$  kg, sería en estas unidades 0,94 GEVs. Con este sistema, podemos expresar la masa como:

$$m^2 = E^2 - p^2 \quad (2.1)$$

que es lo que en física de partículas se conoce como masa invariante, su definición es la siguiente: *medida de la masa de un objeto, la cual es la misma para todos los observadores inerciales. Para cualquier sistema de referencia la masa invariante se determina mediante un cálculo que incluye la energía total del objeto y su momento* [9].

Una vez explicada la transformación entre física y energía, introducimos dos conceptos importantes:

- Desintegración: proceso espontáneo en el que una partícula se transforma en otras partículas más ligeras.
- Hadronización: fenómeno que se produce cuando tras una interacción en física de altas energías se crea un quark nuevo. Puesto que los quarks no pueden estar libres, se generan en el vacío pares de quarks-antiquarks que van reagrupando y combinando en hadrones (partículas subatómicas resultantes de la unión de quarks). Esto desencadena en la producción de colas o jets de partículas.

Antes de continuar con el siguiente apartado, vamos a definir también cuales son las fuerzas fundamentales de la física:

- Fuerza fuerte: fuerza que mantiene los núcleos unidos.
- Fuerza débil: responsable de las desintegraciones radioactivas.
- Fuerza electromagnética : mantiene unidos los átomos (los electrones al núcleo). Las partículas interactúan por medio de los campos eléctrico y magnético.
- Fuerza gravitatoria, las partículas con masa atraen a otras partículas de acuerdo a la relatividad general de Einstein.

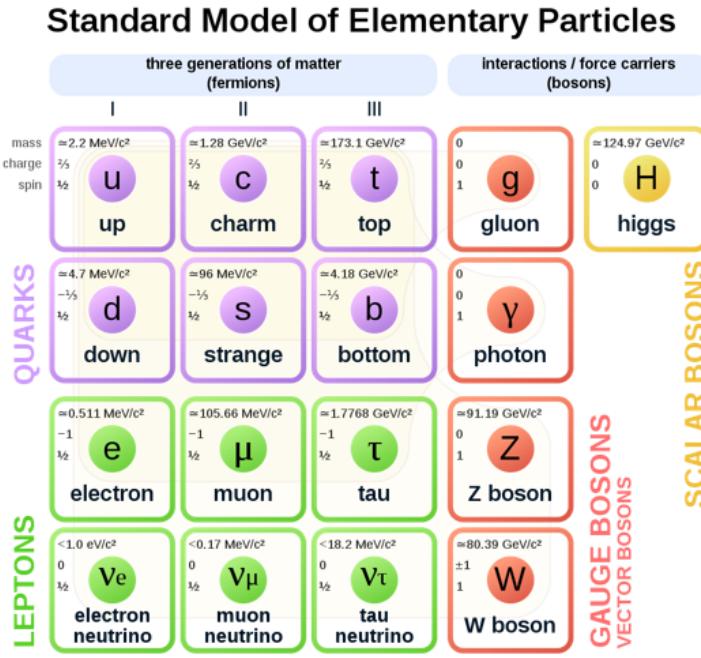


Figura 2: Partículas del modelo estándar.

## 2.2. Modelo Estándar

El modelo estándar (SM) es la teoría más aceptada en la física de partículas. En él se encuentran clasificadas todas las partículas elementales conocidas y se describen las interacciones entre ellas.

Es un modelo que empezó a desarrollarse a partir de la segunda mitad del siglo XX, finalizándose su formulación actual en la década de los 70, con la confirmación experimental de la existencia de los quarks. Desde entonces se han ido descubriendo todas las partículas predichas por el modelo como el tau neutrino (2000) o el bosón de Higgs (2012). El modelo es una teoría cuántica de campos constituido por una colección de teorías relacionadas que incluyen la electrodinámica cuántica, la teoría electrodébil de Glashow, Weinberg y Salam y la cromodinámica cuántica [10].

En la Figura 2 aparecen todas las partículas que componen al modelo estándar. Vemos que el modelo separa a las partículas entre materia (fermiones) e interacción (bosones).

En primer lugar tenemos las partículas correspondientes a la materia o fermiones. Estas partículas tienen espín semientero, siguen el principio de exclusión de Pauli y vienen descritas por la estadística de Fermi-Dirac. A cada fermión le corresponde su antipartícula con la misma masa pero las cargas físicas (como la carga eléctrica) opuestas.

Los fermiones son clasificados a su vez en leptones y quarks:

- Leptones. Son partículas elementales caracterizadas por poder observarse tanto en estado libre como ligado (por ejemplo los electrones). No sienten la interacción fuerte y tienen masa no nula. Hay tres tipos o “sabores” conocidos de leptones: el electrón, el muón y el leptón tau. Cada sabor está representado por un par de partículas. La primera es la partícula cargada masiva que lleva el mismo nombre que su sabor

(como el electrón). La otra es una partícula neutra casi sin masa y sin carga llamada neutrino (como el neutrino electrónico).

- Quarks. Son los fermiones que interactúan fuertemente, no pueden observarse en estado libre, sino que se encuentran en confinados formando los hadrones. Existen 6 tipos de quarks conocidos, que, junto con los leptones pueden agruparse en 3 generaciones (Figura 2, izquierda). Las generaciones van ordenadas por masa, así en la tercera generación encontramos los quarks más masivos y por tanto, menos estables. La primera generación está compuesta por los quarks *up* y *down*, y es la única que constituye la energía ordinaria (protón y neutrón). La segunda generación está formada por los quarks *charm* y *strange* y solo es existente en la radiación cósmica o en los procesos de desintegración. Por último, la tercera generación está compuesta por los quarks *up* y *bottom* que solo aparecen en condiciones creadas artificialmente, como en los instantes posteriores a las colisiones desencadenadas en los aceleradores de partículas.

Por otra parte encontramos a los bosones, o partículas que siguen la estadística de Bose-Einstein. Las fuerzas fundamentales de la física vienen descritas por el modelo estándar como el resultado del intercambio de partículas (emisión y absorción). Las partículas que se intercambian son lo que llamamos bosones o partículas mediadoras de fuerza. Los bosones tienen espín entero y no siguen el principio de exclusión de Pauli. Es decir, los bosones no tienen un límite teórico en su densidad (número por volumen). Encontramos:

- Gluones: partículas sin masa responsables de la interacción fuerte entre quarks. Los gluones y sus interacciones son descritas por la cromodinámica cuántica.
- Fotones: partículas sin masa responsables de la fuerza electromagnética entre partículas cargadas.
- $W^+$ ,  $W^-$  y  $Z$  son responsables de interacción débil. Estos tres bosones son masivos, siendo  $Z$  el más masivo. Junto a los fotones se agrupan como mediadores colectivos de la interacción electrodébil.

Entre las partículas de interacciones tenemos un tipo especial de bosón: el bosón de Higgs, una partícula masiva que no tiene espín (y por ello es clasificado como bosón con espín entero). El rol del bosón de Higgs es la explicación de por qué otras partículas elementales son masivas. En particular explica porqué el fotón no tiene masa, mientras que los bosones  $Z$  y  $W$  son muy masivos [10].

Aunque el modelo estándar describe una amplia gama de fenómenos y ha proporcionado buenas predicciones experimentales, deja algunos fenómenos sin explicación y no es una teoría completa de interacciones fundamentales. Sus principales limitaciones son [11]:

- Solo explica tres de las cuatro fuerzas fundamentales. La fuerza gravitacional queda fuera del modelo tal como se describe en la relatividad general, ya que hasta la fecha no se ha conseguido una formulación cuántica de la gravedad.
- No explica la asimetría bariónica, es decir no explica el desequilibrio entre materia y antimateria. El universo está formado mayormente por materia con un pequeño porcentaje de antimateria, lo cual no entra dentro de las predicciones del SM.
- No explica la existencia de la materia oscura. Para describir la rotación de las galaxias y la expansión acelerada del universo la cosmología observational incluye

lo que conocemos como materia oscura, ya que la materia ordinaria es insuficiente para explicar estos fenómenos. Ninguna partícula del modelo estándar es viable como componente de este tipo de materia.

- No incorpora las oscilaciones de neutrinos ni explica que sus masas sean distintas de cero, esto entra en conflicto con las numerosas evidencias experimentales de que los neutrinos tienen masa y además oscilan.

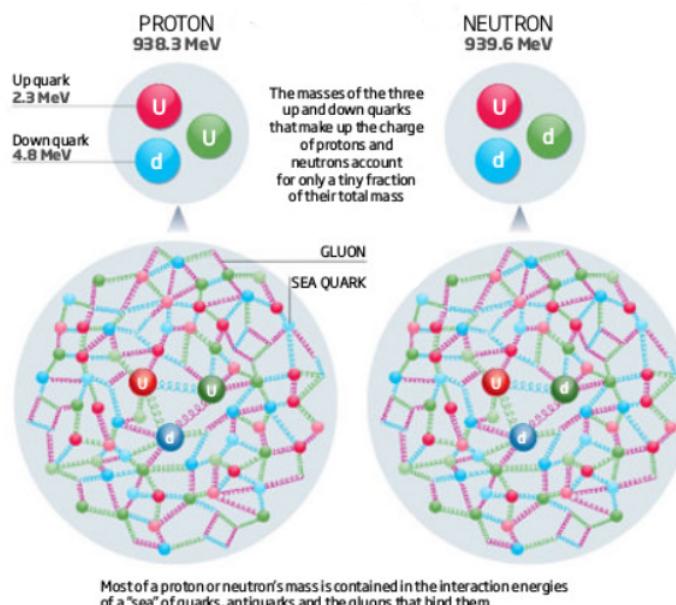
Además de estas cuestiones surgen otras como el por qué hay tres familias de fermiones o por qué la carga eléctrica va como múltiplos de  $1/3$ , a esto le añadimos que el SM tiene 19 parámetros libres, que pueden ser demasiados.

Todas las razones expuestas incitan a la búsqueda de modelos de nueva física que resuelvan las limitaciones del SM. Normalmente, el SM es utilizado como base para construir modelos con partículas exóticas (partículas hipotéticas que no han sido descubiertas), el ejemplo más representativo sería la supersimetría [16]. Por otra parte también han surgido teorías completamente novedosas como la teoría de cuerdas o la teoría M [17]. Sea como sea, no se puede negar que el modelo estándar supuso una revolución en la forma de ver la física a escalas subatómicas, y aún 50 años después sigue siendo la teoría más cercana a una “teoría del todo”.

### 2.3. Colisión protón protón y nueva física

Una vez que hemos visto los aspectos más fundamentales del modelo estándar, podemos ya describir el experimento en particular que nos ocupa: las colisiones protón protón.

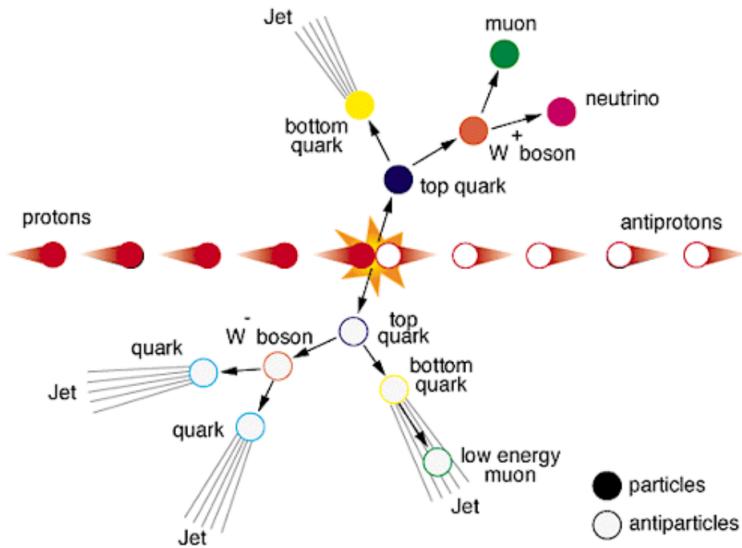
Empezamos describiendo al protón (Figura 3). Un protón es un hadrón formado por tres quarks principales: dos *up* y uno *down*. Estos quarks representantes están “sumergidos en un mar” de gluones y parejas transitorias de quarks y antiquarks. Los gluones son los responsables de la interacción fuerte que produce un estado de confinamiento haciendo posible que el protón sea una partícula estable.



**Figura 3:** Esquema del protón y del neutrón [13].

En el LHC dos protones son acelerados casi a la velocidad de la luz y chocan con energías de entre 7 y 14  $TeV$ . La colisión produce una serie de quarks y desintegraciones que dan lugar a cientos de nuevas partículas, debido a que la energía de la colisión se transforma en masa. Lo que sucede según la cromodinámica cuántica (QCD) es que cuando los protones están muy cerca, los partones que los componen (quarks y gluones) interactúan entre sí dando lugar a nuevos quarks y gluones. Los quarks producidos son de diferente tipo, a mayor masa más difícil será su producción.

Nosotros nos vamos a centrar en la producción de los quarks más pesados (top y anti-top). Su masa de  $172\ GeV$  hace que la producción de un par oscile entre probabilidades de  $1/1000$  o  $1/10000$  frente a la producción de otros quarks más frecuentes [12]. El par  $t\bar{t}$  produce una compleja cadena de desintegraciones (Figura 4).



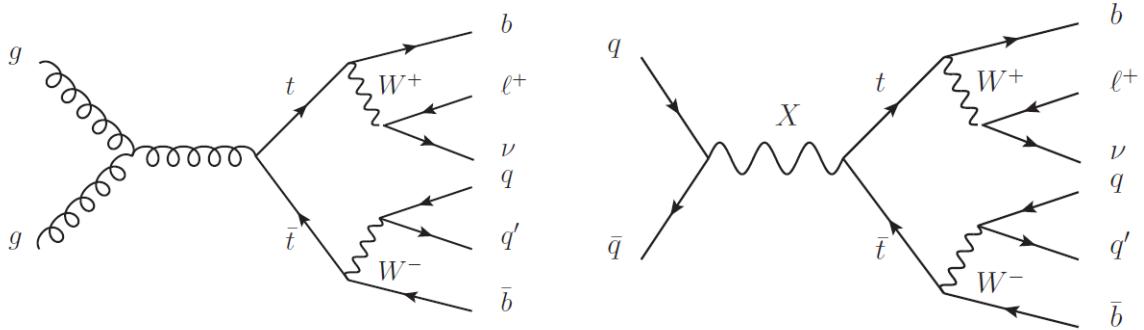
**Figura 4:** Colisión protón-protón y producción del par top-antitop [14].

Así el acoplamiento de dos gluones genera la desintegración  $t\bar{t}$ , tal como vemos en el diagrama de Feynmann (Figura 5, izquierda). El quark top decae en un quark  $b$  y un bosón  $W^+$ , que es el mediador de la interacción débil. Éste último puede decaer en varias formas, aquí se ha considerado que se desintegra en un leptón (electrón o muón) y su respectivo neutrino. El quark  $b$  tiene una vida media muy corta y decae en una serie de partículas dando lugar a una cola o jet. Por otra parte, el quark anti-top se descompone en un quark anti- $b$  y un bosón  $W^-$ . Éste último sufre una desintegración hadrónica en un par de quarks. Como resultado tenemos dos jets procedentes de la desintegración de  $W^-$  y uno de anti- $b$ . El resultado final son cuatro jets, un leptón y un neutrino.

Como hemos visto anteriormente, el modelo estándar presenta varias limitaciones que hacen necesario el desarrollo de nuevos modelos con partículas hipotéticas. Llamamos a estos modelos de *nueva física BSM (beyond standard model)*. Vamos a considerar un modelo que incorpora una nueva partícula  $X$ , este modelo sería un derivado de las teorías de Kaluza-Klein y la nueva partícula podría ser candidata a la materia oscura [18].

Según este tipo de modelo, en la colisión protón protón se produce una resonancia que da lugar a la partícula  $X$ , una partícula masiva con vida media muy corta (alrededor de  $10^{-23}$  segundos). Esta partícula se desintegraría en un par top-antitop, dando lugar

a la misma cadena de desintegraciones que hemos visto [19]. En la Figura 5 aparece la comparacion SM vs BSM.



**Figura 5:** Diagramas de Feynman de la colisión protón-protón para el Standard Model (SM), izquierda; y Beyond Standard Model (BSM) derecha [15].

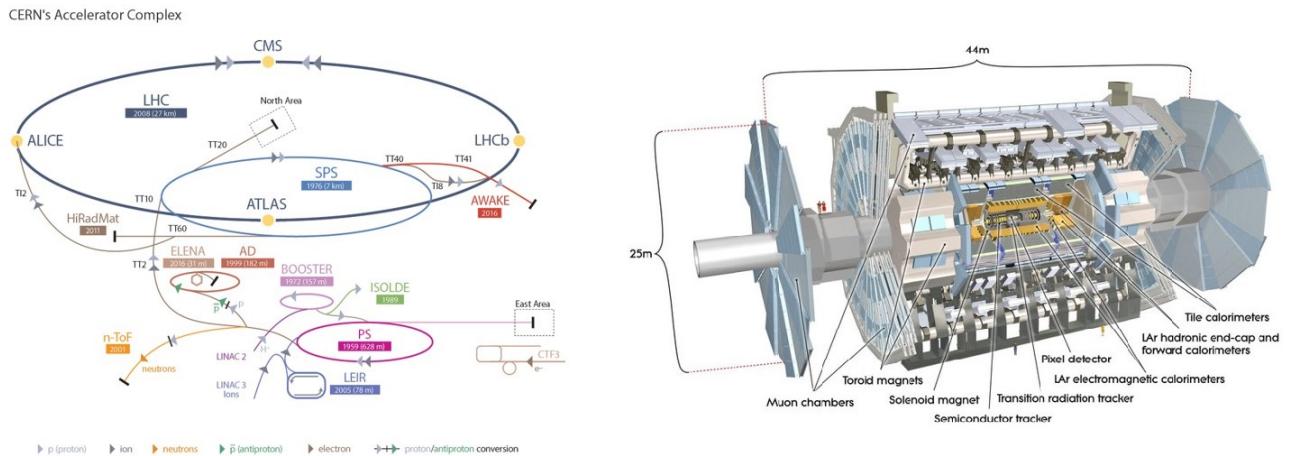
## 2.4. Experimento ATLAS

La búsqueda de experimental de partículas exóticas y nueva física se lleva a cabo en aceleradores como los que se encuentran en el CERN. Tal como se observa en la Figura 6 (izquierda) el CERN alberga un complejo sistema de colisionadores y detectores que lo han convertido en uno de los centros de investigación más importantes del mundo. Se encuentra en Suiza cerca a la frontera con Francia y es un modelo de colaboración científica internacional.

El colisionador de partículas más grande del CERN es también el más grande y energético del planeta. El LHC (*Large Hadron Collider*) tiene 27 km de diámetro y produce ingentes cantidades de datos. Dentro del colisionador se aceleran haces de protones a velocidades próximas a la de la luz, a energías del orden del *TeV* [20].

El LHC es en realidad el último elemento de una cadena de aceleradores. Los protones son producidos aplicando un intenso campo eléctrico y así ionizando átomos de hidrógeno, átomos que pasan por una serie de aceleradores hasta llegar al LHC. Una vez allí los haces son acelerados por última vez y colisionan produciendo altísimas energías que permiten recrear las condiciones de los instantes posteriores al Big Bang. Las partículas resultantes de cada choque son detectadas y estudiadas. Para ello se han construido principalmente cuatro grandes detectores: ATLAS, CMS, LHCb y ALICE.

En la Figura 6, derecha, aparece detector ATLAS (*A large Toroidal LHC Apparatus*) que fue diseñado expresamente para medir la mayor cantidad de señales diferentes, es decir su objetivo principal es la búsqueda de partículas exóticas. Su tamaño es de 46 metros de largo y 25 de diámetro y está formado por un complejo sistema de capas (Figura 7). Se trata de una serie de cilindros concéntricos en torno al punto de colisión de los haces. En cada capa se detecta un tipo de partícula diferente. Las únicas partículas que no pueden detectarse directamente son los neutrinos, debido a que no interactúan con los detectores. Su presencia se infiere midiendo un desequilibrio de momento, es decir, asociándoles la diferencia de momento entre el inicio de la colisión y la suma de momentos de las partículas detectadas resultantes.

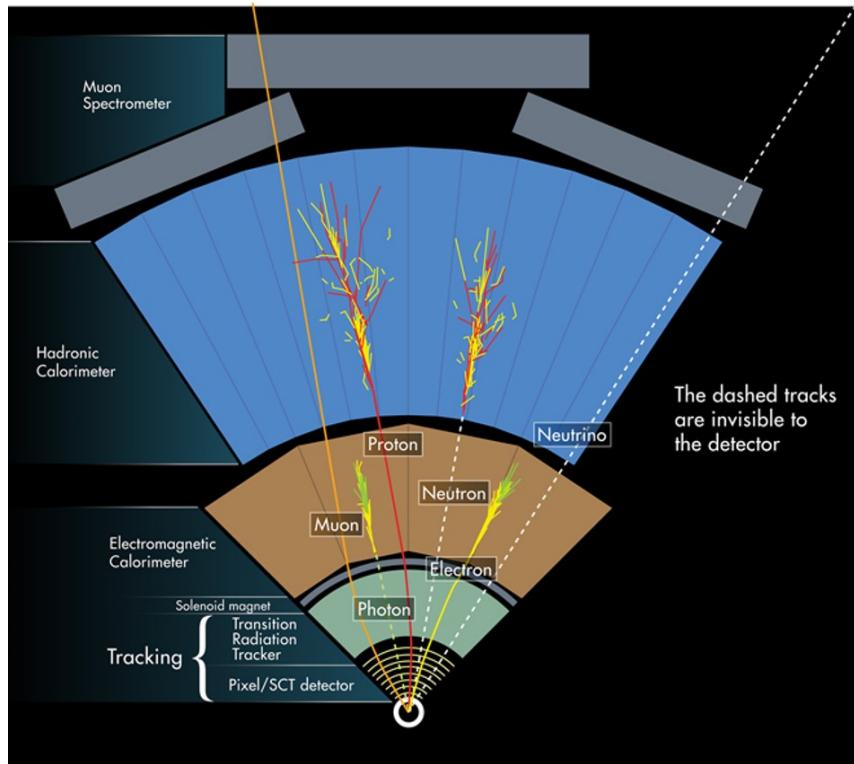


**Figura 6:** Complejo CERN (izquierda), detector ATLAS (derecha).

A continuación describimos más detalladamente cada una de las capas que componen ATLAS:

- Detector interno. Es la capa más interna, se encuentra rodeando al punto de impacto y está compuesto a su vez de tres subdetectores. Su función principal es la detección de partículas cargadas, las cuales interactúan con la materia en puntos concretos. Los puntos iniciales de las trayectorias dan información acerca del tipo de partícula y a través de ellos se puede hacer un etiquetado *btag* [21]. Mediante el campo magnético solenoideal que rodea al detector interno se curva la trayectoria de las partículas cargadas, esto permite la medida de sus momentos y cargas.
- Calorímetros. Los calorímetros están situados fuera del imán solenoideal. Su propósito es medir la energía y localización de las partículas absorbiéndolas mediante metales muy densos. Hay dos sistemas básicos de calorímetro: un calorímetro electromagnético interno que absorbe energía de los electrones y fotones (interaccionan electromagnéticamente); y un calorímetro hadrónico externo, que absorbe energía de las partículas que pasan a través del calorímetro EM, pero interactúan a través de la fuerza fuerte. Estas partículas son principalmente protones y neutrones.
- Espectrómetro de muones. Se trata de un sistema de seguimiento extremadamente grande y complejo que tiene como objetivo la detección de muones. Los muones son partículas que consiguen escapar a los calorímetros debido a su baja interacción con la materia. El espectrómetro, junto a los toroides externos, funciona de manera similar al detector interno y permite identificar energías, trayectorias y momentos de los muones.
- Sistema de imanes. El ATLAS utiliza un complejo sistema de imanes para curvar la trayectoria de las partículas cargadas a partir de la fuerza de Lorentz. Esto permite conocer su momento midiendo la curvatura (a mayor momento menor curvatura), y su carga eléctrica mediante dirección de las partículas. El sistema de imanes está compuesto por un solenoide interno que rodea al Detector Interno y dos toroides externos, situados en el exterior de los calorímetros y dentro del espectrómetro muónico.

Para recolectar los datos ATLAS cuenta con un complejo sistema computacional que analiza en tiempo real los datos “en crudo” y selecciona los eventos más interesantes para un análisis posterior. Se estima que tras este filtrado se producen unos  $100\text{ Mb}$  de datos por segundo, un total de  $1\text{ Pb}$ .



**Figura 7:** Sistema de capas de ATLAS.

### 3. Problema HEPMASS

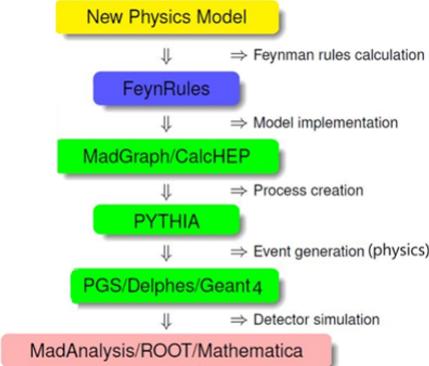
En esta sección se expondrá el trabajo realizado para resolver con el conjunto de datos HEPMASS. Se trata de un problema binario cuyo objetivo es la construcción de un algoritmo de *machine learning*, éste debe ser capaz de distinguir entre sucesos correspondientes a señal (BSM) y ruido (SM). Todo el estudio se realizará con los datos simulados HEPMASS del repositorio *UC Irvine Machine Learning Repository* [4].

En primer lugar presentaremos el problema describiendo el conjunto de datos y variables. Despues se dará paso a un análisis exploratorio de datos y en la última sección se estudiará el problema a través de una perspectiva de *machine learning*.

#### 3.1. Introducción al problema

Las simulaciones son una herramienta muy usada en la búsqueda de nuevas partículas en el LHC. A través de éstas se consiguen datos similares a los producidos en experimentos reales. Además, nos sirven para conocer a priori el resultado experimental de modelos hipotéticos como un modelo BSM.

El conjunto de datos HEPMASS ha sido creado mediante simulaciones Montecarlo. Se trata de datos correspondientes a colisiones protón-protón según el Standard Model y Beyond Standard Model. El resultado para cada evento son una serie de características cinemáticas como variables input, y la etiqueta ruido (SM) o señal (BSM) como variable output.



**Figura 8:** Proceso de creación de datos simulados para un modelo de física de partículas.

Se ha utilizado el generador *MADGRAPH5* [22] para simular la interacción entre los partones de los dos protones en la colisión. Con *PYTHIA* [23] se han simulado los procesos de desintegración y hadronización que se dan en los instantes posteriores a la colisión. Por último, con *DELPHES* [24] se ha generado la respuesta experimental que resultaría en los detectores de ATLAS. De esta manera las variables obtenidas se corresponden con cantidades que se pueden medir al realizar un experimento real, y por tanto este trabajo podría usarse con datos experimentales provenientes de ATLAS. En la Figura 8 se muestra todo el proceso de generación de datos.

### 3.2. Descripción del Dataset

Hemos recogido las variables en la Tabla 1. En total tenemos unas 28 variables input que describiremos en los siguientes párrafos, además de una variable objetivo binaria, y el parámetro masa. Éste parámetro es la masa de la partícula  $X$  que se le ha dado a las simulaciones. Se trata de una masa desconocida pero mayor que la de otras partículas. Esta masa se estima a partir de modelos teóricos del tipo extradimensions (Kaluza-Klein) como ya hemos visto anteriormente. Suele abarcar desde 500  $GeV s$  hasta 5000  $GeV s$ .

En este caso se han supuesto masas de 500 $GeV s$ , 750 $GeV s$ , 1000 $GeV s$ , 1250 $GeV s$  y 1500 $GeV s$  con lo que se ha recorrido un rango amplio.

FEATURE	TIPO	DESCRIPCIÓN	PARTÍCULA
<i>label</i>	binaria	Variable output señal vs ruido	-
<i>lep_pt</i>	float	Momento transversal del leptón	Leptón
<i>lep_eta</i>	float	Pseudorapíldity	
<i>lep_phi</i>	float	Azimut	
<i>met_miss</i>	float	Momento transversal faltante	Neutrino
<i>met_phi</i>	float	Azimut faltante	
<i>jets_no</i>	entero	Número de jets de la colisión	Jets resultantes
<i>jet1_pt</i>	float	Momento transversal	Jet más energético
<i>jet1_eta</i>	float	Pseudorapidity	
<i>jet1_phi</i>	float	Azimut	
<i>jet1_btag</i>	binaria	Correspondencia con un quark $b$	
<i>jet2_pt</i>	float	Momento transversal	Segundo jet más energético
<i>jet2_eta</i>	float	Pseudorapidity	
<i>jet2_phi</i>	float	Azimut	
<i>jet2_btag</i>	binaria	Correspondencia con un quark $b$	
<i>jet3_pt</i>	float	Momento transversal	Tercer jet más energético
<i>jet3_eta</i>	float	Pseudorapidity	
<i>jet3_phi</i>	float	Azimut	
<i>jet3_btag</i>	binaria	Correspondencia con un quark $b$	
<i>jet4_pt</i>	float	Momento transversal	Cuarto jet más energético
<i>jet4_eta</i>	float	Pseudorapidity	
<i>jet4_phi</i>	float	Azimut	
<i>jet4_btag</i>	binaria	Correspondencia con un quark $b$	
<i>m_jj</i>	float	Masa invariante	$W^-$
<i>m_jjj</i>	float	Masa invariante	$\bar{t}$
<i>m_lv</i>	float	Masa invariante	$W^+$
<i>m_jlv</i>	float	Masa invariante	$t$
<i>m_wbbb</i>	float	Masa invariante	$X$
<i>mass</i>	float	Masa de la partícula $X$ en la simulación (0 si es ruido)	$X$

Tabla 1: Features del dataset HEPMASS, azul: low-level, rojo: high-level.

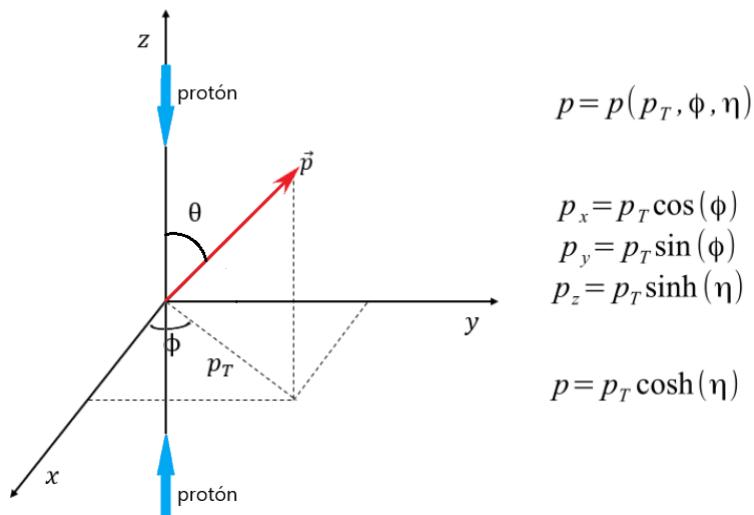


Entre las variables input podemos distinguir entre 21 *low-level features*, que resultarían

de las mediciones de propiedades de las partículas resultantes del experimento; y 5 características *high-level*, resultantes de la combinación de las características *low-level* y que proveen información de las partículas intermedias que se han desintegrado y no se pueden observar directamente.

### Low-level features

Las características de bajo nivel son la reconstrucción de los vectores resultantes de la colisión. Cada vector se puede describir de acuerdo a su momento transversal (momento en el plano transversal a la colisión) y los ángulos que forma con respecto a los ejes de coordenadas (Figura 9).



**Figura 9:** Coordenadas de un vector (partícula o jet) resultante de la colisión protón protón.

Así tenemos:

- Momentos transversales de los cuatro jets más energéticos, el leptón y el neutrino (*Missing Transverse Energy*).
- Azimut  $\phi$  de los cuatro jets más energéticos, el leptón y el neutrino.
- Pseudorapidity calculada con la expresión  $\eta = -\ln(\tan(\frac{\theta}{2}))$  para los cuatro jets más energéticos y el leptón [25].
- $b\_tag$  de cada jet.
- Número total de jets.

Como ya hemos dicho anteriormente el neutrino no es una partícula detectable. Tanto su momento transversal como su azimut se obtienen aplicando los principios de conservación de momento y energía entre el inicio y el final de la colisión. Se plantea el sistema de ecuaciones correspondiente y se despejan las variables analíticamente. Su *pseudorapidity* también se podría obtener a partir de la conservación de energía-cantidad de movimiento,

sin embargo esta variable no aparece en el conjunto de datos porque tendría un poder discriminatorio muy bajo.

### High-level features

Como variables de alto nivel tenemos las masas invariantes de las partículas intermedias de la colisión, que se desintegran después de formarse. Estas masas se tienen que conservar en las desintegraciones. Cuando una partícula  $a$  se desintegra en dos partículas  $b$  y  $c$  su masa invariante se puede calcular mediante la siguiente expresión [26]:

$$M^2 = m_a^2 = m_{b+c}^2 = m_{b+c}^2 = (E_b + E_c)^2 - |(\vec{p}_b + \vec{p}_c)|^2$$

que se puede expresar como función de las características de bajo nivel:

$$M^2 = 2p_T a p_T b (1 - \cos(\phi)) \quad (3.2)$$

Aclaradas las variables solo nos queda presentar el tamaño del dataset (Tabla 2).

Originales			Normalizados		
Ruido	17821710		Ruido	5249876	
Señal	500	6014342	34105228	500	1049486
	750	8084733		750	1049426
	1000	7255944		1000	1050776
	1250	7507750		1250	1050965
	1500	5242459		1500	1049471
Total		51926938	5250124		10500000

**Tabla 2:** Número de registros en los datasets para cada *label* y *masa*.

Tenemos pues dos conjuntos de datos:

- Normalizados: son los procesados que se encuentran directamente en el repositorio. Primero se ha aplicado  $\log(X + 10^{-5})$  a los momentos transversales y las masas invariantes. Después se han estandarizado todas las variables. Nosotros usaremos éstos en la construcción de los modelos, así tendremos las variables en una escala común, sin distorsionar las diferencias en los intervalos de valores.
- Sin normalizar: es el output resultante de las simulaciones. Estos no provienen del repositorio sino que se han añadido por completitud para tener una compresión de la parte de la física; se pueden coger como punto de partida y hacer la normalización que se considere más oportuna. Nosotros usaremos éstos en el análisis exploratorio de datos, cuando sea necesario conocer el significado físico de las variables.

Vemos que hay un menor número de datos normalizados, esto es porque los autores han hecho una selección previa de datos a la normalización. En nuestro caso los 10 millones de datos normalizados son más que suficientes para la construcción de

modelos de *machine learning*, por lo que utilizaremos los dos conjuntos de datos sin realizar ninguna modificación.

### 3.3. Análisis exploratorio de datos

Esta sección consistirá en un análisis exploratorio del conjunto de datos. A lo largo del análisis utilizaremos los datos sin normalizar, ya que nuestro objetivo es entender las variables físicas que conforman el dataset. Así, intentaremos encontrar posibles tendencias y patrones que nos puedan dar una primera intuición antes de trabajar en el modelado. No hemos incluido las visualizaciones de los datos normalizados por su similitud con las de los datos sin normalizar. Las visualizaciones se han realizado con las librerías de *Python*: *Pandas* [27], *Matplotlib* [28] y *Seaborn* [29].

Es importante decir que el dataset no presenta problemas de missing values o datos erróneos. Las visualizaciones consistirán en las distribuciones de las variables, la matriz de correlaciones lineales y la diferencia teniendo en cuenta el etiquetado *btag*.

#### 3.3.1. Distribuciones estadísticas

Como primera aproximación al problema hemos dividido el conjunto de datos en grupos de variables. A continuación, mostramos la distribución estadística de estas variables. De esta manera vemos como cada una de ellas afecta (por separado) a la clasificación señal-ruido.

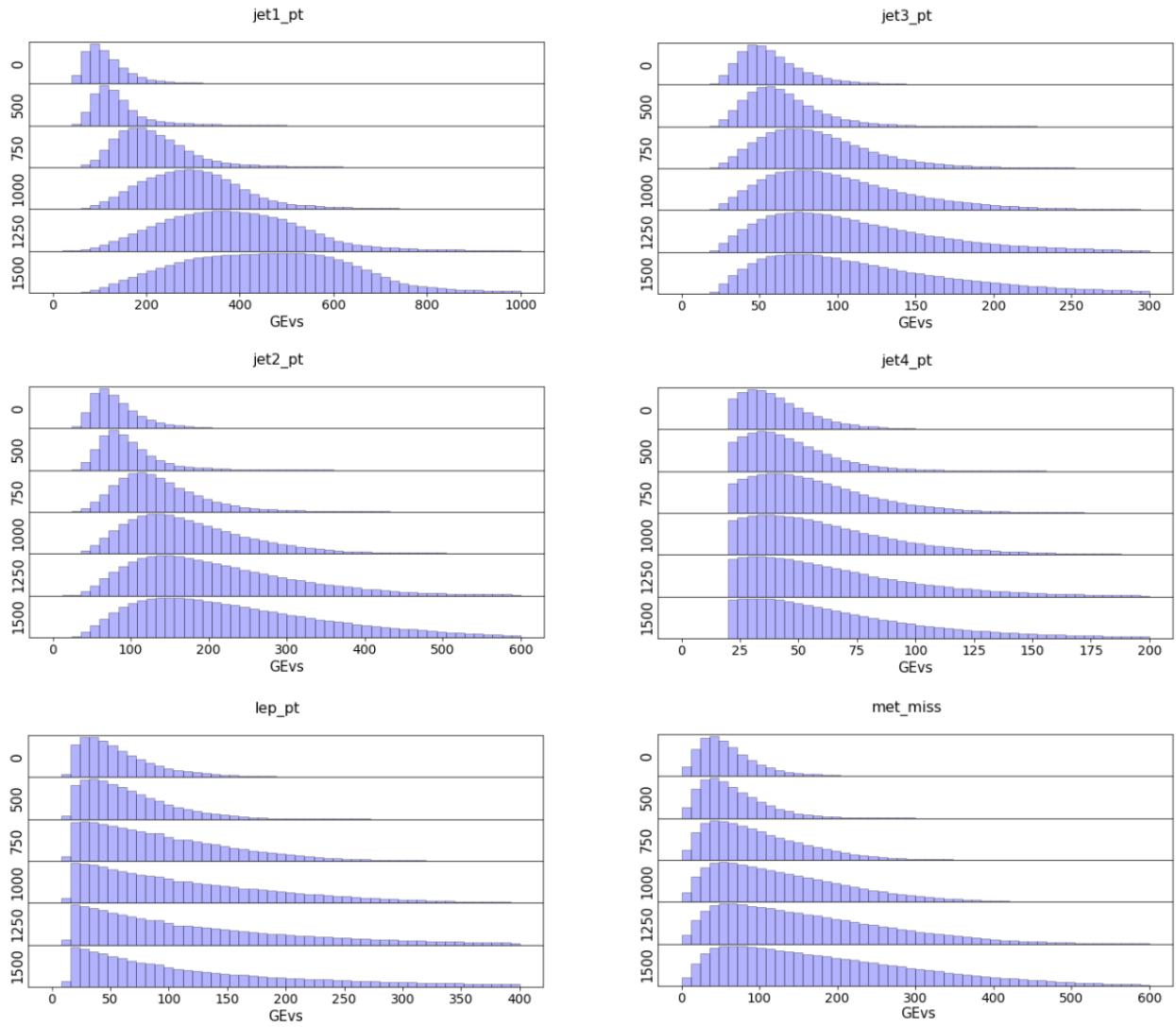
En todas las visualizaciones hemos representado los datos de ruido (en los ejes de las figuras nos referimos a ellos como 0) frente a los datos de señal: masas de 500,750,1000,1250 y 1500 *GEVs* .

#### Momentos transversales

En la Figura 10 vemos la distribución de los momentos transversales. Lo que más llama la atención es que a medida que la masa va siendo mayor las distribuciones crecen hacia la derecha. Esto es lógico ya que cuanto mayor masa tenga la partícula *X* mayor masa (energía) tendrán las partículas resultantes de la cola de desintegraciones (los jets tendrán mayor momento transversal).

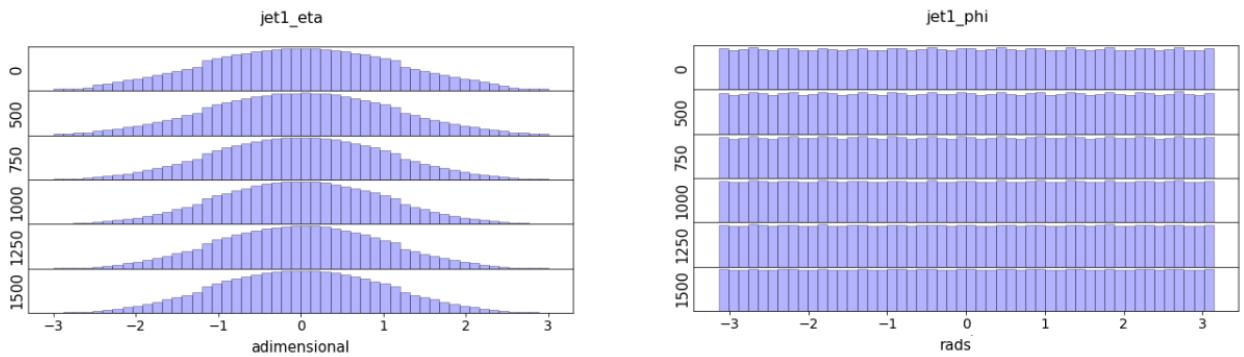
Vemos también como parece que los momentos transversales de los jets son variables muy discriminantes a masas altas. Por otra parte, cuando tenemos una masa de 500 *GEVs* la distribución es muy similar al caso de ruido.

Por último, destacamos que la distribución del momento transversal del cuarto jet parece cortada en la parte de la izquierda. Esto se debe a las condiciones iniciales que se le han puesto a la simulación, que obligan a que la energía del cuarto jet sea mayor de 25*GeV*s.



**Figura 10:** Distribución del los momentos transversales de los jets, del leptón, y masa faltante del neutrino.

### Azimut y pseudorapidity



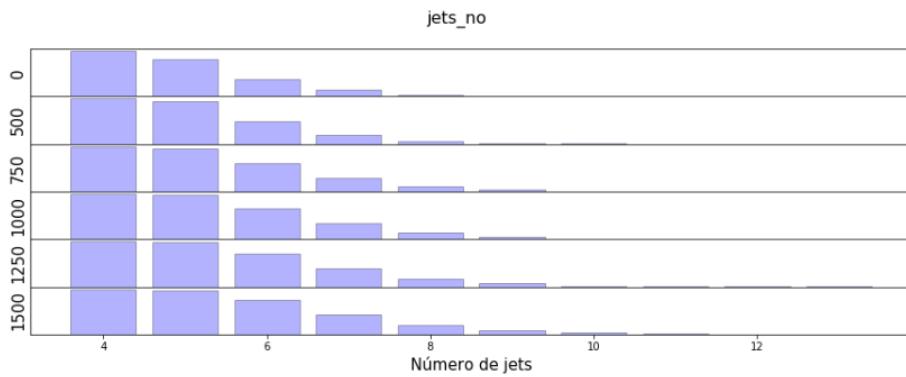
**Figura 11:** Azimut y *pseudorapidity* del primer jet.

En lo que se refiere a las variables tipo ángulo (azimut y *pseudorapídy* encontramos la

visualización en la Figura 11. Solo hemos mostrado el resultado para el primer jet, ya que en los demás casos la distribución es similar. Como vemos ninguna de las dos variables parece discriminante. Además la distribución de la variable azimut es totalmente uniforme.

### Número de jets

Mostramos también la visualización para el número de jets (Figura 12). En este caso vemos que lo más común es que se produzcan 4 o 5 jets de partículas. Siendo más improbable que se de un número mayor, el número máximo lo encontramos en 12 o 13 jets. Destacamos aquí que a mayor masa de la partícula  $X$  más probabilidad hay de que se produzca un mayor número de jets, lo cual parece lógico, pues a mayor energía mayor es el número de jets que se puede producir.

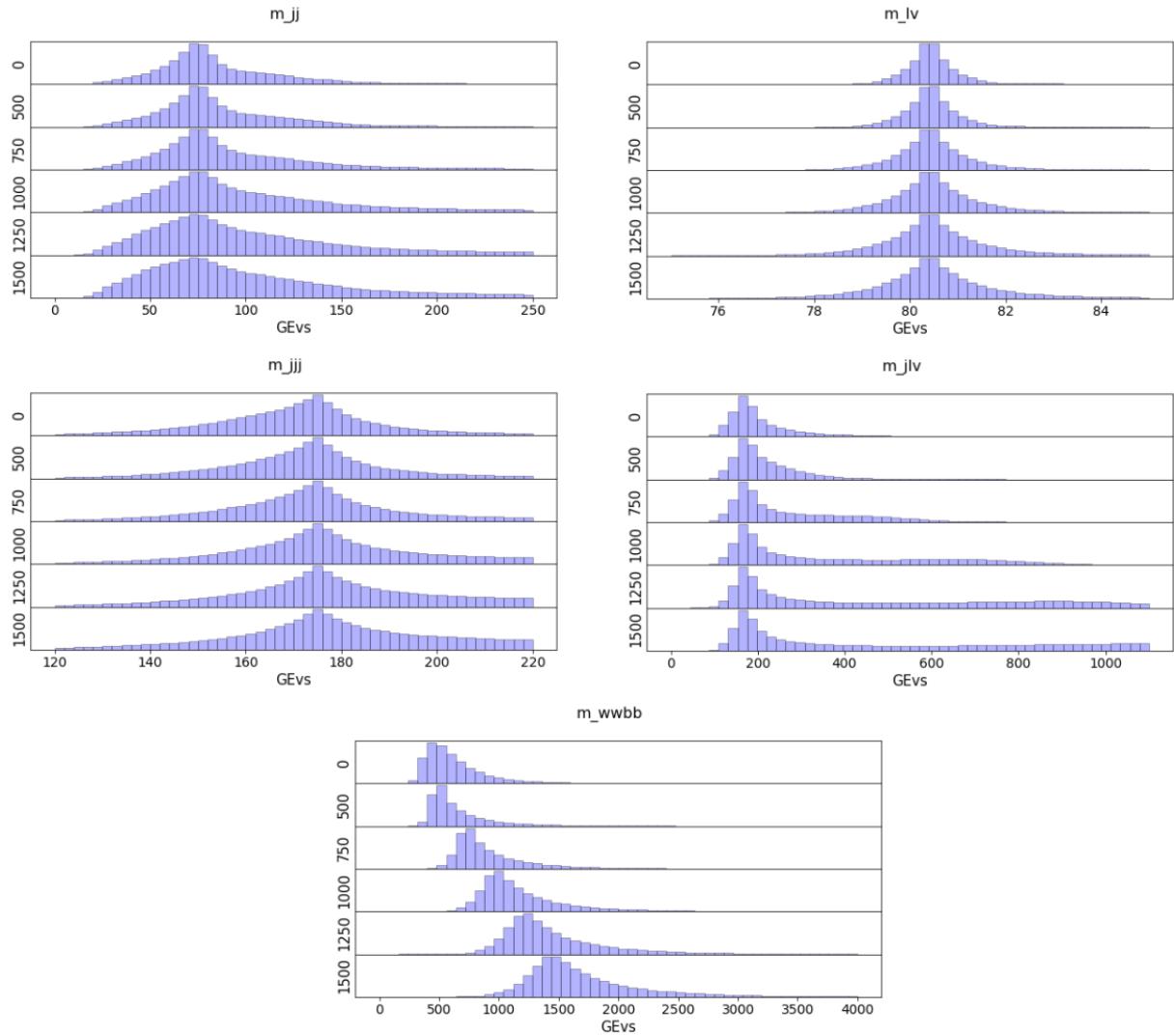


**Figura 12:** Distribución del número de jets.

### High-level features: masas invariantes

Por último representamos las masas invariantes. En los tres primeros casos apenas vemos diferencias en las visualizaciones. Lo único a destacar sería que las distribuciones para masas altas son más anchas a ambos lados (sobre todo  $m_{jj}$ ). En cuanto a  $m_{jlv}$  vemos que para masas altas la cola de la distribución es más larga hacia la derecha, teniendo incluso una pequeña joroba entre los 800 – 1000 GEVs.

Claramente la masa invariante final  $m_{wwbb}$  es una de las variables con mayor poder de discriminación. Da una condición de cierre aunque sea una característica calculada a partir de otras variables. En las visualizaciones vemos que, para los sucesos correspondientes a señal, el pico de la distribución casi se corresponde con la masa de la simulación. Éste es ligeramente menor debido a errores de reconstrucción del jet, ya que faltan partículas que no se han detectado. En el caso de ruido (SM) la distribución es parecida a de masa 500, sin embargo es más abierta. La cromodinámica cuántica es capaz de explicar la distribución para el caso de ruido sin que se de la condición de resonancia, es decir sin que haya sido generada por una partícula  $X$ .



**Figura 13:** Distribución de las masas invariantes.

### 3.3.2. Matriz de correlaciones

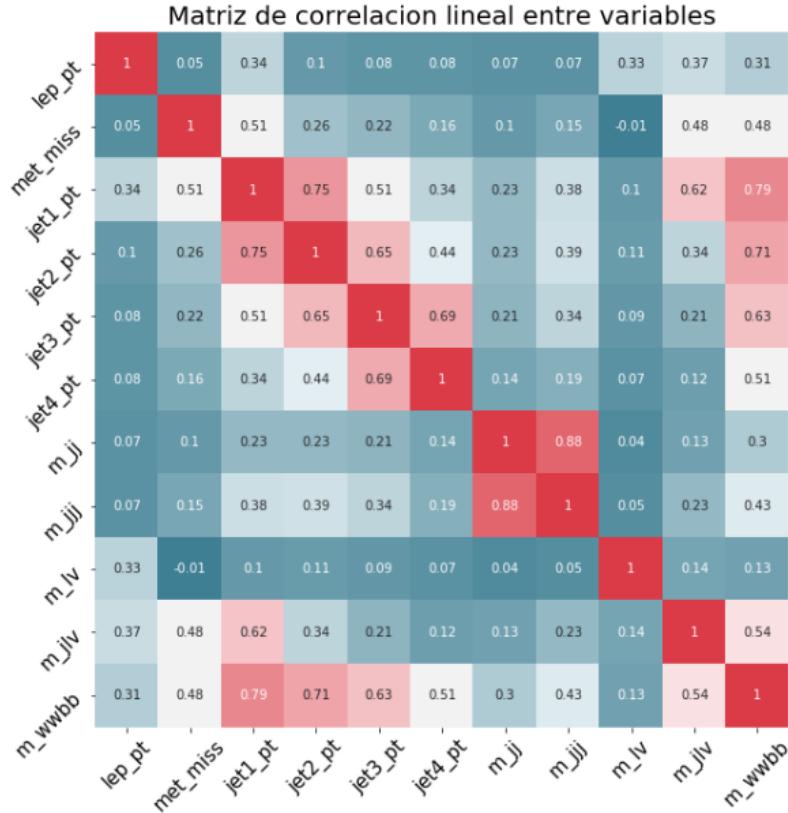
En la Figura 14 podemos ver la matriz de correlación lineal de Pearson entre variables [30]. Se han representado las variables referentes a momentos transversales y masas invariantes. Para simplificar la figura hemos eliminado los *btags*, el número de jets y las variables referentes a ángulos por no guardar correlaciones con otras variables. Aunque mostremos las correlaciones para los datos sin normalizar, hemos comprobado que son similares a las de los datos normalizados (en ningún caso hay más de 0,1 de diferencia).

Observamos más de un 0,7 de correlación entre las siguientes variables:

- $jet1\_pt$  y  $jet2\_pt$
- $jet1\_pt$  y  $m\_wwbb$
- $m\_jj$  y  $m\_jjj$

Es lógico pensar que a mayor momento transversal de los jets, mayor sea la masa invariante final. Tiene sentido también la alta correlación entre  $m\_jj$  y  $m\_jjj$  ya que con el diagrama de Feynmann (Figura 5) reconstruimos una a partir de la otra.

Estas correlaciones pueden suponer un problema a la hora de construir modelos de *machine learning* [31]. En la sección de aplicación de *machine learning* utilizaremos técnicas como *PCA* y *LDA* para eliminar la correlación lineal y comparar los resultados de los algoritmos cuando no se elimina esta correlación.



**Figura 14:** Matriz de correlación lineal entre variables.

### 3.3.3. Diferencias en cuanto al etiquetado *btag*

Aunque este problema no presenta datos erróneos como tal, si que encontramos errores derivados del etiquetado *btag* que introducen un error sistemático al reconstruir las masas invariantes.

Veíamos en la Figura 5 que hay dos jets tipo *b* en el estado final, estos jets deberían aparecer etiquetados en los datos. Al contar con datos simulados sabemos a ciencia cierta que la etiqueta que se le ha puesto a los jets es correcta. Estadísticamente los jets *b* tienen los momentos más altos pero es una distribución estadística y hay otras combinaciones pueden darse. En dichas combinaciones los jets *b* no corresponderían a los dos más energéticos. Como nos quedamos con los cuatro jets principales lo único que pedimos es que dos de esos jets sean *b* independientemente del lugar que ocupen en el orden en energías. Por tanto cuando tenemos uno o ningún jet etiquetado como *b* consideramos que se introduce un error en los datos.

En datos reales el tratamiento sería diferente porque no se sabe con exactitud si el jet es *b* o no. Al reconstruir los jets se utilizan una serie de variables asociadas con los jets que permiten obtener una probabilidad de que sean *b*. Eso hace más complicada la reconstrucción del esquema de desintegraciones y disminuye la eficiencia.

jet1_btag	1				0			
jet2_btag	1		0		1		0	
jet3_btag	1	0	1	0	1	0	1	0
jet4_btag	1	0	1	0	1	0	1	0
% eventos	0	0	0	32.0	0	17.5	9.2	4.4
					0	14.1	7.7	3.8
						5.6	3.0	2.2
							0.6	

**Tabla 3:** Distribución del etiquetado de los jets en en dataset.

En nuestro dataset (Tabla 3) tenemos:

- ~86 % sucesos con dos jets etiquetados como  $b$ .
- ~13 % sucesos con un jet etiquetado como  $b$ .
- ~1 % sucesos con cero jets etiquetados como  $b$ .

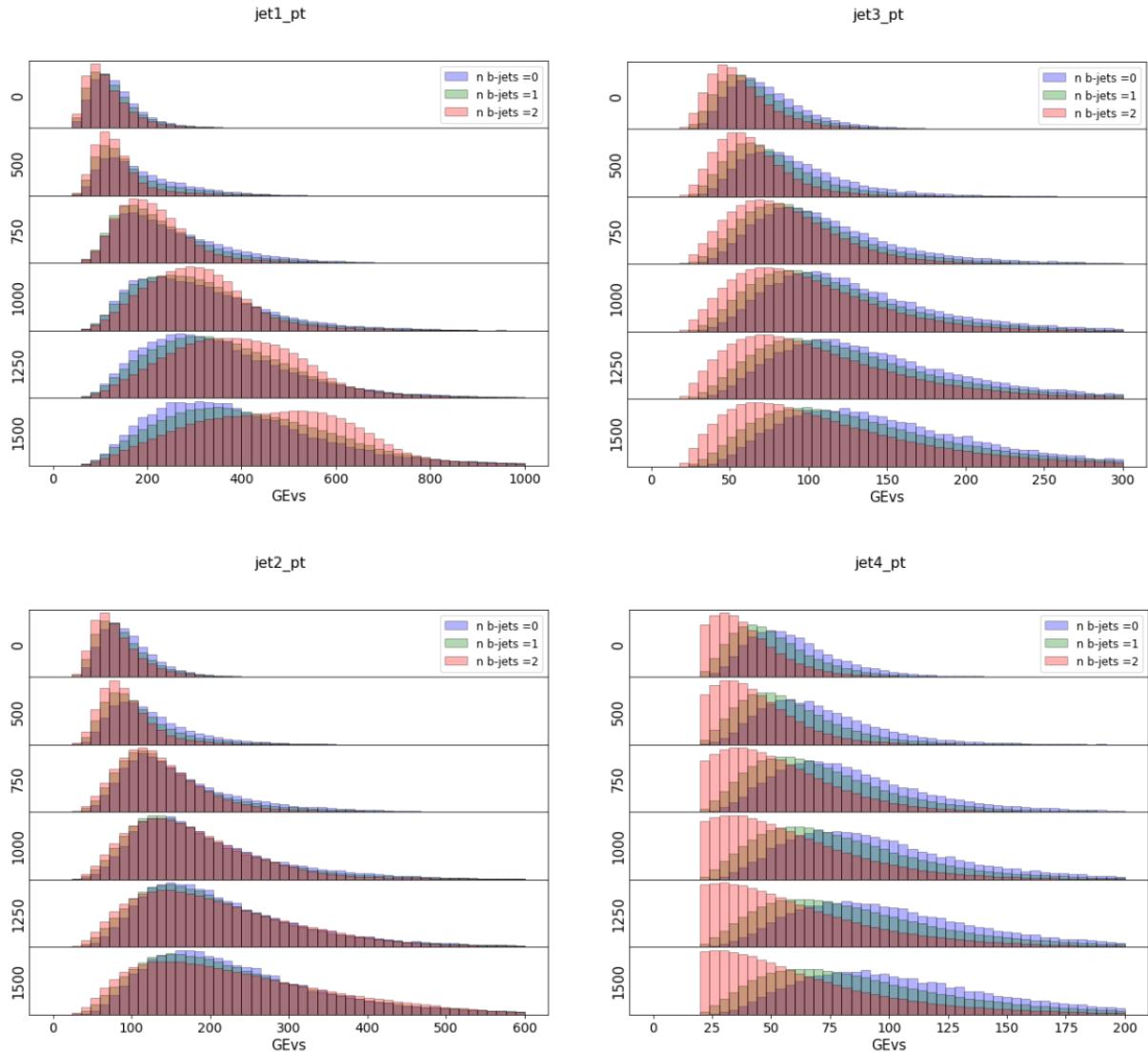
Como hemos dicho, estos datos son demasiado idílicos. La eficiencia con datos simulados oficiales de ATLAS [32] donde están incluidos todos los posibles factores de mala identificación/reconstrucción es del 65 %. El resto (35 %) estaría mal identificado. Por tanto, en datos reales un etiquetado  $btag$  se parecería más a:

- ~65 % sucesos con dos jets etiquetados como  $b$ .
- ~33 % sucesos con un jet etiquetado como  $b$ .
- ~2 % sucesos con cero jets etiquetados como  $b$ .

El etiquetado erróneo de los jets  $b$  afecta a las demás variables. Las diferencias causadas por el etiquetado se pueden ver a simple vista mediante distribuciones estadísticas. Hemos dividido los datos según su etiquetado  $btag$  (2, 1 o ningún jet  $b$ ) y hemos escogido el mismo número de datos para cada uno de los etiquetados. A continuación mostramos una selección de las variables que se ven más afectadas.

En la figura 15 aparecen las distribuciones de los momentos transversales de los jets. Vemos que a masas altas, cuando hay dos jets etiquetados como  $b$ , para el jet más energético las distribuciones se corresponden con las de energía más alta (comparados con los casos en los que hay uno o ninguno jet  $b$ ), mientras que para los momentos de los demás jets la distribución parece que se “desplaza” a la izquierda, hacia bajas energías.

Sucede exactamente lo opuesto cuando el número de jets  $b$  es igual a 0. Lo que está ocurriendo es que cuando tenemos dos jets etiquetados como  $b$ , estos dos jets suelen ser los dos más energéticos, es como si “capturaran” la mayor parte de la energía, mientras que los demás jets (el tercero y el cuarto) se han quedado con menos energía. Por otra parte cuando ninguno de los cuatro jets aparece etiquetado como  $b$  el reparto es más equitativo. El caso intermedio lo encontramos cuando solo hay un jet etiquetado como  $b$ .

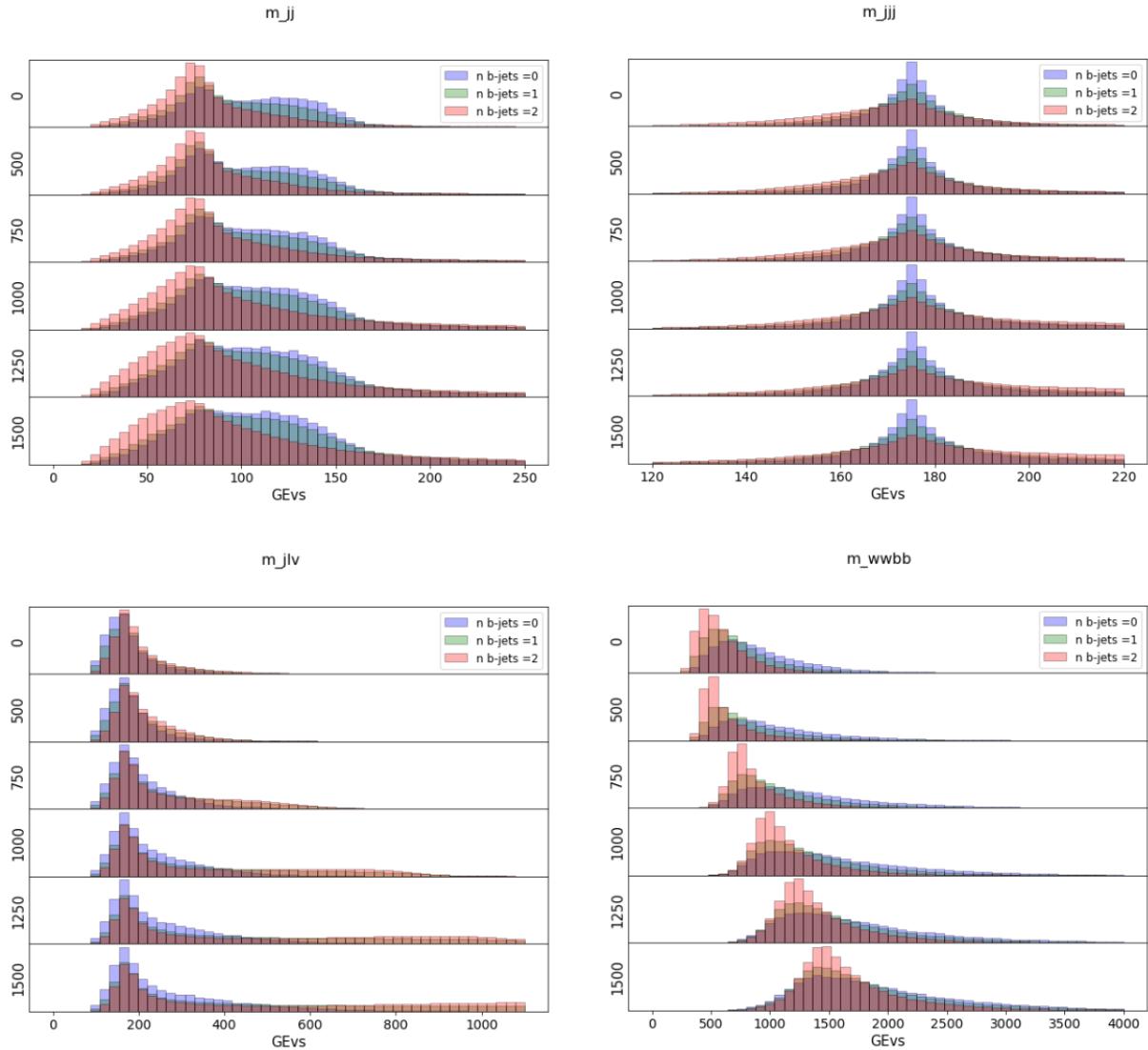


**Figura 15:** Distribución del los momentos transversales de los jets dependiendo de etiquetado *btag*.

En la figura 16 vemos como el etiquetado *btag* afecta a las masas invariantes (no se ha mostrado la masa  $m_{lv}$  porque apenas se veía afectada ya que es la masa reconstruida a partir del leptón y el neutrino).

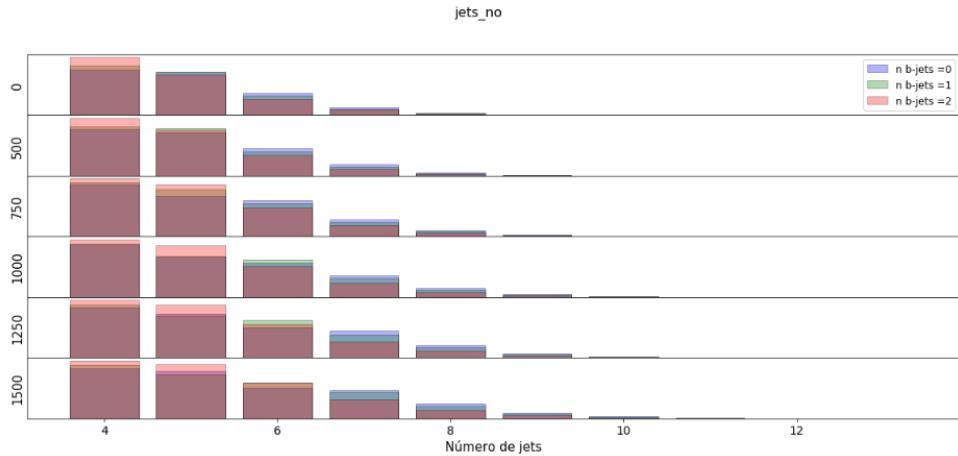
Observamos que el comportamiento de las masas invariantes según el etiquetado *btag* difiere de una masa a otra. La mayor diferencia la encontramos en la distribución de la masa invariante final  $m_{wubb}$ . Esto tiene sentido ya que es la masa que se ve afectada por todos los jets. Vemos que es más estrecha cuando el número de jets *b* es 2.

Por otra parte, la masa  $m_{jjj}$  es la segunda más afectada (3 jets) seguida de las masas  $m_{jj}$  (2 jets) y  $m_{jlv}$  (1 jet)



**Figura 16:** Distribución de las masas invariantes dependiendo de etiquetado *btag*.

Por último hemos representado también las distribuciones del número de jets (Figura 17). Lo que ocurre aquí es que si tenemos dos jets etiquetados como  $b$  el número total de jets es generalmente menor, mientras que si ningún jet es  $b$  el número de jets es mayor. La explicación a esto iría en la línea de lo ocurrido con los momentos transversales. Los jets  $b$  son más energéticos y “acaparan” una mayor proporción de la energía total. Mientras que si no hay jets  $b$  hay más energía disponible para formar otros jets.



**Figura 17:** Distribución del número de jets dependiendo de etiquetado *btag*.

### 3.4. Aplicación de Machine learning

En esta sección nos adentraremos en la parte de modelización del conjunto de datos.

Empezaremos mostrando un pequeño resumen teórico de los algoritmos de machine learning utilizados, junto con las métricas que se han usado para comparar el resultado de los modelos. Además describiremos las técnicas de selección de características *PCA* y *LDA*.

A continuación mostraremos los resultados de la selección de características realizadas. Las técnicas utilizadas han incluido la transformación de los datos mediante *PCA* y *LDA*, y la creación de un ranking de características mediante modelos Random Forest. De esta manera veremos que características son las más importantes para la construcción de modelos.

Después expondremos el resultado de los diferentes algoritmos aplicados al problema. Se realizará una comparación entre los modelos utilizados donde se verá cuales son los que mejor se adaptan a los datos.

Por último veremos dos estudios adicionales: el error sistemático producido por el etiquetado *btag* y el caso de aplicación de redes neuronales parametrizadas.

Cabe destacar que **en toda esta sección se han usado los datos normalizados**. En el conjunto de datos normalizado ya venía definido un conjunto de datos entrenamiento y un conjunto de datos de test, con proporción 66 % – 33 %. Por tanto se han entrenado los modelos con los datos correspondientes al train mientras que se han testeado con los de test. Debido al gran número de datos se ha seleccionado una fracción de ellos. Hay que tener en cuenta también que como la masa de la partícula *X* en los datos de señal es una constante, hemos separado el conjunto de datos de señal en diferentes masas y entrenado un clasificador para cada una de las masas. **Los conjuntos de entrenamiento han contado en todos los experimentos realizados con clases totalmente balanceadas con señal vs ruido 50 % – 50 %.**

Todos los algoritmos y métricas utilizadas provienen de la librería *Scikit-learn* [33] de *Python*, a excepción de las redes neuronales profundas que se han configurado a través de la librería *Keras* [34].

### 3.4.1. Resumen teórico de las técnicas de machine learning

**3.4.1.1. Métricas para clasificación binaria** Al ser un problema de clasificación binaria, el objetivo de los modelos es predecir la distribución de probabilidad de la variable objetivo (señal o ruido) a través de las variables predictoras. Para medir la calidad de las predicciones se ha calculado la matriz de confusión resultante de evaluar los datos de test. Los modelos asignan una probabilidad de que el evento sea positivo (señal), si esa probabilidad es mayor de un umbral de decisión (0.5 por defecto) el caso es etiquetado como señal, en caso contrario se etiqueta como ruido.

La matriz de confusión construida consta de los siguientes elementos:

- TP: verdaderos positivos, sucesos correspondientes a señal bien etiquetados.
- TN: verdaderos negativos, sucesos correspondientes a ruido bien etiquetados.
- FP: falsos positivos, sucesos correspondientes a ruido etiquetados como señal.
- FN: falsos negativos, sucesos correspondientes a señal etiquetados como ruido.

:

A partir de la matriz se pueden construir diferentes métricas [35] que permiten evaluar la bondad de los modelos, en este trabajo hemos escogido las siguientes:

- **Accuracy.** Es el porcentaje de acierto del modelo, calculado como:  $(TP+TN)/(TP+TN+FP+FN)$ . Su rango va de 0 a 1 siendo 1 la mejor medida.
- **Precisión.** Evalúa el porcentaje de señal que el modelo identifica correctamente  $TP/(TP+FP)$ . Su rango va de 0 a 1 siendo 1 la mejor medida.
- **Recall o sensibilidad.** Evalúa el porcentaje de señal que el modelo es capaz de identificar entre todos los sucesos que realmente son señal:  $TP/(TP+FN)$ . Su rango va de 0 a 1 siendo 1 la mejor medida.
- **F1-score.** Es la media armónica entre precisión y sensibilidad:

$$2 \cdot \frac{\text{precisión} \cdot \text{sensibilidad}}{\text{precisión} + \text{sensibilidad}}$$

. Su rango va de 0 a 1 siendo 1 la mejor medida.

- **Kappa de Cohen:** Es una medida estadística que se basa en comparar la concordancia observada en un conjunto de datos respecto a la que podría ocurrir por mero azar. En el caso binario se calcula de la siguiente manera:

$$\frac{(TP+FP) \cdot (TP+FN) + (TN+FP) \cdot (TN+FP)}{TP+TN+FP+FN}$$

Es efectiva para medir los casos de clases desbalanceadas. Su rango va del -1 (peor medida) al 1 (mejor medida) [36].

- **AUC (area under the curve).** La curva ROC (receiver operator curve) nos muestra el rendimiento de un modelo mediante una representación gráfica bidimensional. Se calcula el área bajo la curva de la representación de la tasa de verdaderos positivos frente a la tasa de falsos positivos según se varía el umbral de discriminación. Va de 0 a 1 siendo 1 la mejor medida.

La métrica que más hemos usado ha sido *F1-score*, ya que contiene a la *precisión* y sensibilidad. Por tanto hemos considerado que aporta más información que otras métricas como *accuracy* o *AUC*.

**3.4.1.2. Algoritmos de machine learning** Hemos escogido los siguientes modelos para clasificación: regresión logística, Random Forest, SVM y redes neuronales.

Para configurar los hiperparámetros internos de cada modelo hemos utilizado *cross validation* e *hyperparameter tuning*. Por un lado *cross-validation* es un proceso que separa los datos de entrenamiento en subsets construyendo varios subconjuntos en los que se puede entrenar y evaluar el modelo. Por otro lado, el ajuste de hiperparámetros consiste en probar diferentes parámetros internos y seleccionar los que maximicen la precisión del clasificador. Mediante la combinación de ambos procesos se prueban los hiperparámetros (fijados en el código) en los diferentes subsets y se escogen los que mejor resultado tienen, después se entrena con el train completo y se testea sobre los datos de test.

A continuación explicamos brevemente cada uno de los modelos escogidos.

**Regresión logística** La regresión logística es uno de los algoritmos más sencillos en clasificación. Matemáticamente, se modeliza el logaritmo de la distribución de probabilidad de la variable objetivo como una combinación lineal de las variables predictoras [37]. Es decir:

$$\ln \frac{p}{1-p} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \dots \quad (3.3)$$

En el ajuste de hiperparámetros se han probado configuraciones de los términos:

- C, término de regularización (penalización adicional a la función de coste para que le modelo generalice mejor).
- penalty, norma utilizada en el ajuste (l1 o l2).
- solver, algoritmo utilizado en la optimización.

## Random Forest

El Random Forest consiste en realidad en un conjunto de modelos, concretamente un conjunto de árboles de decisión. [38]

Los árboles se inicializan de manera distinta para aprender así de forma diferente. Cada árbol se entrena con distintos subconjuntos del dataset (seleccionados aleatoriamente y con reemplazamiento). Además, al seleccionar la partición óptima, se tiene en cuenta sólo una porción de los atributos, elegidos también aleatoriamente. Los árboles se evalúan de forma independiente y el resultado final es la moda de todas las predicciones.

En este modelo se han ajustado los siguientes hiperparámetros:

- n\_estimators: número de árboles de decisión utilizados.
- criterion: es el criterio usado para medir la calidad de las divisiones que se realizan en las hojas de los árboles de decisión. El criterio puede ser “gini” (impureza de Gini) o “entropía” (ganancia de información) [39].

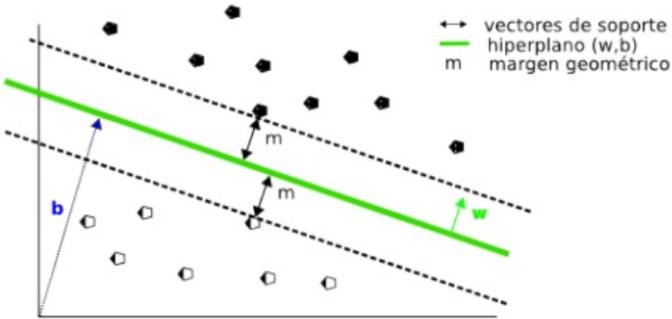
- `max_features`: número de características a considerar cuando se busca la mejor división.
- `min_samples_leaf`: número mínimo de muestras requeridas en un nodo.

Además de como modelo de clasificación los Random Forest se pueden utilizar para construir un ranking de características. En un árbol de decisión el lugar en el que una característica es utilizada se puede usar para evaluar su importancia relativa (con respecto a la predicción de la variable objetivo). Así, las características utilizadas en la parte superior del árbol contribuyen más a la decisión final, por tanto tienen una importancia mayor. A partir de los diferentes árboles (con diferentes características) de decisión que utiliza el Random Forest se puede calcular la importancia de cada variable y elaborar un ranking de características.

## SVM

Un clasificador SVM (*support vector machine*) es un modelo que representa los datos en un espacio de alta dimensionalidad, pudiendo ser incluso infinita [40]. En este espacio se separan las clases en dos subespacios lo más amplios posible mediante un hiperplano de separación. Cuando las muestras se ponen en correspondencia con dicho modelo, en función del espacio al que pertenezcan, pueden ser clasificadas a una o la otra clase. Se denominan vectores de soporte a los puntos que conforman las dos líneas paralelas al hiperplano, siendo la distancia entre ellas (margen) la mayor posible. Por tanto las muestras que constituyen los vectores soporte son las muestras con mayor poder de discriminación. (Figura 18)

Con este tipo de modelo, aparte de dar una predicción se tratará de obtener cuales son los vectores soporte y por tanto las muestras más decisivas en la clasificación.



**Figura 18:** Hiperplano de separación construido por un SVM, los vectores soporte son las muestras a partir de las cuales se construye el plano.

En nuestro caso hemos ajustado el Kernel (función de transformación) además del parámetro  $C$ , que al igual que en la regresión logística modula la fuerza de la regularización.

## Redes neuronales simples

Definimos red neuronal o perceptrón multicapa como algoritmo de aprendizaje supervisado consistente en un conjunto de unidades [41], llamadas neuronas artificiales, conectadas entre sí para transmitirse señales. La información de entrada atraviesa la red neuronal, donde se somete a diversas operaciones, produciendo unos valores de salida. En el caso de clasificación, los valores de salida para cada uno de los eventos se corresponden con la clase a la que pertenece dicho evento.

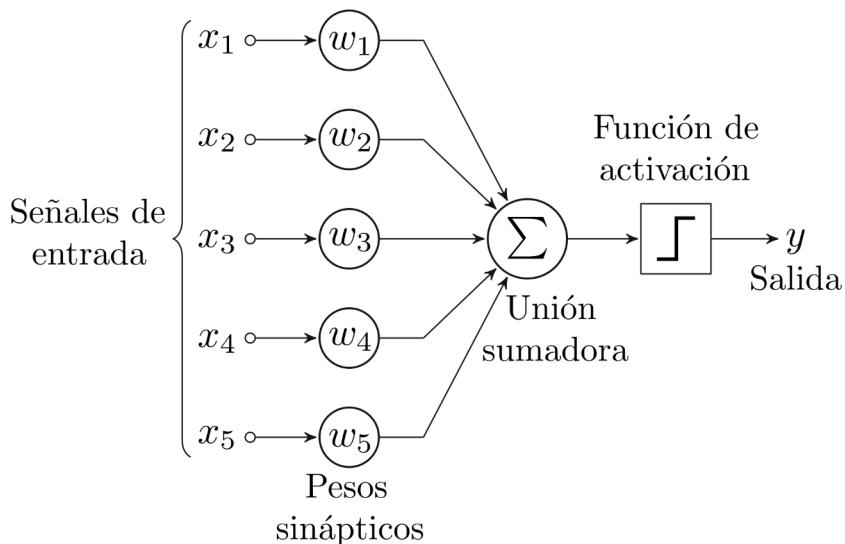
La arquitectura más simple de una red neuronal Figura 19 en un problema binario se trata de una estructura formada por tres capas. Al inicio hay una capa de entrada cuyo tamaño es el número de variables de entrada. A continuación, se encuentra una capa oculta con  $n$  neuronas. Cada neurona tiene asociado un valor descrito por la suma de los valores de las neuronas de la capa anterior multiplicado por un peso. Por último, a la salida, una única neurona clasifica cada dato en señal-ruido.

Los pesos se inician de forma aleatoria, pero a través del aprendizaje por *backpropagation* [42] se van modificando en las sucesivas iteraciones. Además, existe una función de activación en cada neurona que decide si la información de la neurona pasa a la capa siguiente o no.

La estructura de una red neuronal es altamente modificable a través de hiperparámetros. En nuestro caso hemos ajustado los siguientes hiperparámetros:

- solver: método que se utiliza a la hora de optimizar los pesos.
- max\_iter: número máximo de iteraciones.
- alpha: término de regularización
- hidden\_layer\_sizes: número de neuronas en la capa oculta.
- activation: función de activación que se utiliza en la capa oculta.

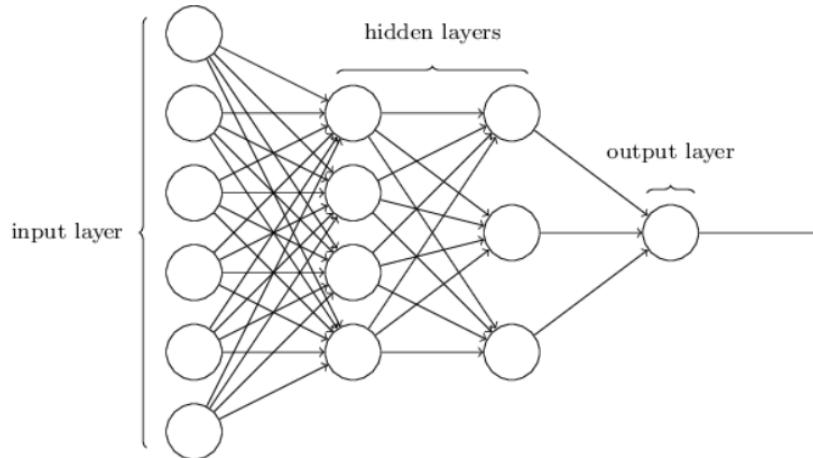
Además, en este caso, previo al entrenamiento se han vuelto a normalizar todos los datos con la función *MinMaxScaler*. Se resta la media y se divide por el rango (max valor - min valor). De esta manera se preservan las distribuciones y los datos transformados están en el rango  $[0, 1]$ , lo cual es más adecuado a la hora de trabajar con redes neuronales [43].



**Figura 19:** Esquema de una red neuronal simple.

## Redes neuronales profundas

Decimos que una red neuronal es profunda o compleja cuando está formada por más de una capa oculta (Figura 20). En este caso el número de parámetros a aprender (pesos) va creciendo con el número de capas. La idea es utilizar el aprendizaje profundo como herramienta para mejorar los resultados de los modelos anteriores. Nuestro conjunto de datos es muy apropiado para el uso de redes neuronales profundas, ya que todas las variables son numéricas y contamos con un gran número de datos.



**Figura 20:** Esquema de una red neuronal compleja con dos capas ocultas.

Hemos construido una arquitectura de varias capas y neuronas. Se ha usado la función de activación ReLU en las capas intermedias (por ser la menos costosa computacionalmente), mientras que se han ajustado:

- solver: método que se utiliza a la hora de optimizar los pesos.
- hidden\_layers: número de capas ocultas.
- neurons: número de neuronas en cada capa.
- activation (última capa): función de activación que se utiliza en la última capa.
- learning\_rate: tamaño del paso en cada iteración mientras el algoritmo de optimización se mueve hacia un mínimo de una función de coste.
- loss: función de coste utilizada.
- batch\_size: número de muestras que se utilizan para entrenar la red en cada iteración.
- epochs: número de veces que todo el conjunto de entrenamiento es utilizado hasta que el algoritmo finaliza.
- dropout: porcentaje de neuronas eliminadas aleatoriamente en cada capa oculta. Con esto se consigue que ninguna neurona memorice parte de la entrada y no se produzca sobreajuste.
- alpha: término de regularización.

Para acelerar el tiempo de cálculo hemos fijado la función de activación de las capas intermedias a ReLu. Antes de construir la red también hemos vuelto a normalizar los datos con *MinMaxScaler*.

### 3.4.1.3. Técnicas de selección de características

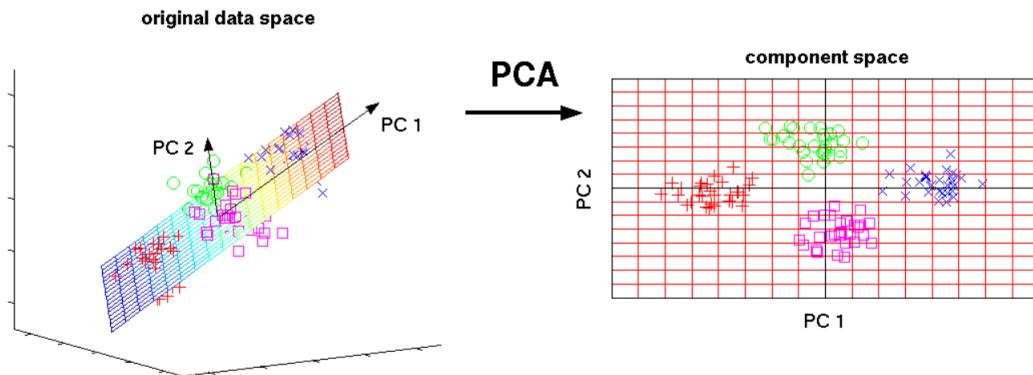
#### PCA

El Análisis de Componentes Principales (*PCA*) es una técnica utilizada para describir un conjunto de datos en términos de nuevas variables no correlacionadas linealmente [44], llamadas componentes principales. Las nuevas variables se ordenan por la cantidad de varianza original que describen. Se trata de simplificar la complejidad de los datos pero conservando la información. La técnica es útil tanto para eliminar dependencia lineal de las variables como para reducir la dimensionalidad del conjunto de datos (Figura 21).

A partir de las características originales se genera la matriz de correlaciones, después se obtienen sus autovalores a partir de sus autovectores. Las componentes principales del sistema se expresan como:

$$Z_i = \phi_{1i}X_1 + \phi_{2i}X_2 + \phi_{3i}X_3 + \dots$$

Con  $Z_i$  las componentes principales,  $X_p i$  las variables originales y  $\phi_{pi}$  los autovalores obtenidos de la matriz de correlaciones.



**Figura 21:** Reducción de tres a dos dimensiones a partir de PCA.

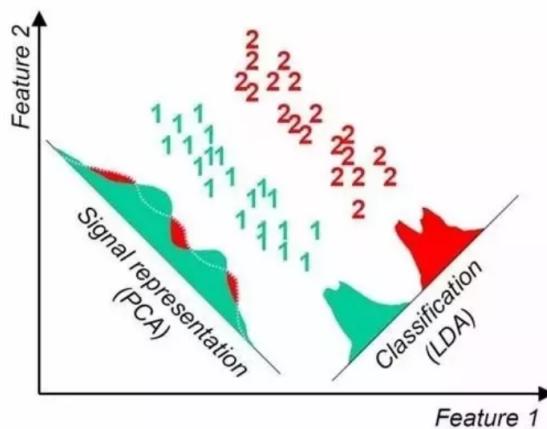
Se pueden construir tantas componentes principales como variables originales tiene el dataset, ya que están ordenadas por importancia, para reducir la dimensionalidad se coge un número de ellas.

#### LDA

Al igual que la PCA, el Análisis Discriminante Lineal (LDA) consiste en una técnica de transformación mediante combinación lineal de las variables originales del sistema [45]. Mientras que el PCA es una técnica no supervisada, el LDA se caracteriza por ser un modelo de reducción de dimensionalidad supervisado buscando la máxima división de las clases del problema.

En este caso se trata maximizar la distancia entre las medias de cada clase, que se hace también a través de un problema de autovectores y autovalores. El inconveniente de este método es que el número de nuevas características debe ser igual o menor al número de clases. Por tanto mediante este método podemos reducir nuestro conjunto de datos a solo dos dimensiones.

En la Figura 22 podemos ver la comparación entre PCA y LDA en un caso de reducción de dos a una dimensión.



**Figura 22:** Reducción de dos a una dimensión para los métodos de LDA y PCA.

### 3.4.2. Reducción y selección de características mediante *machine learning*

En esta subsección mostraremos los resultados que se han obtenido para las técnicas de reducción de variables explicadas en la subsección anterior. Los criterios usados han sido el Ranking de Características de Random Forest junto a PCA y LDA.

En todos los modelos entrenados se han seleccionado 100000 datos para train y 50000 datos para test para cada una de las masas.

#### Ranking de características Random Forest

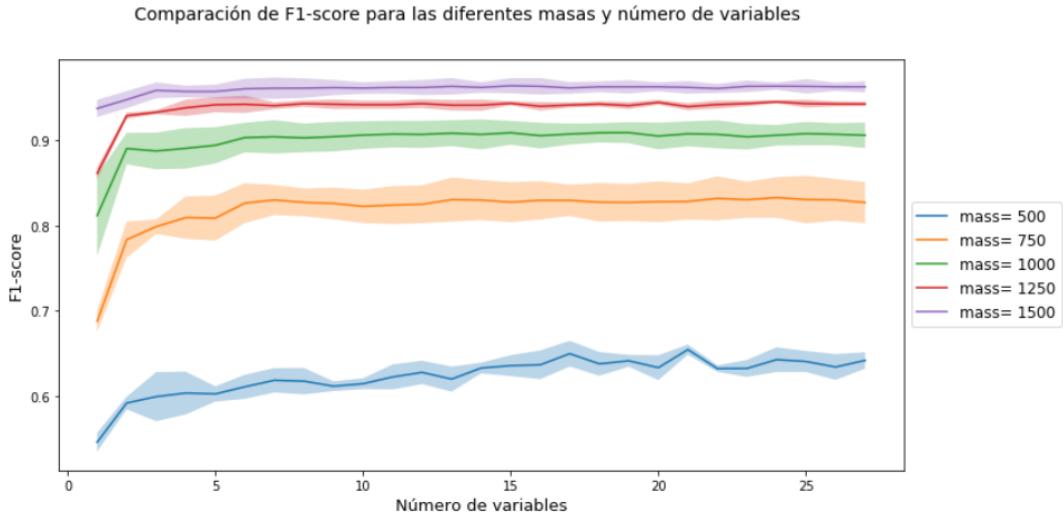
En primer lugar hemos construido el ranking de características que resulta de la aplicación del algoritmo Random Forest. Hemos seguido el siguiente procedimiento:

- Entrenamiento de un modelo Random Forest con las variables originales.
- Construcción del Ranking de Características a partir de la importancia que da el modelo a las diferentes características.
- Selección de las  $n - 1$  características con más importancia.
- Repetición de los pasos anteriores hasta llegar a una única característica.
- Repetición de todo el algoritmo 5 veces para que sea estadísticamente significativo.
- Promedio de la importancia de las características.

En la Figura 23 vemos los resultados de *F1-score* para las diferentes masas. No vamos a entrar en detalle acerca del resultado concreto del modelo (lo haremos en la siguiente

subsección cuando hagamos la comparativa entre modelos). Nuestro objetivo aquí es seleccionar las características más importantes con las que entrenaremos los demás modelos.

Para masas superiores a 500  $GeV$ s entre 5 y 10 características son suficientes para que el resultado del F1-score se estabilice. Por otra parte, para la masa de 500  $GeV$ s el resultado es peor y necesitamos entre 10 y 20 características.



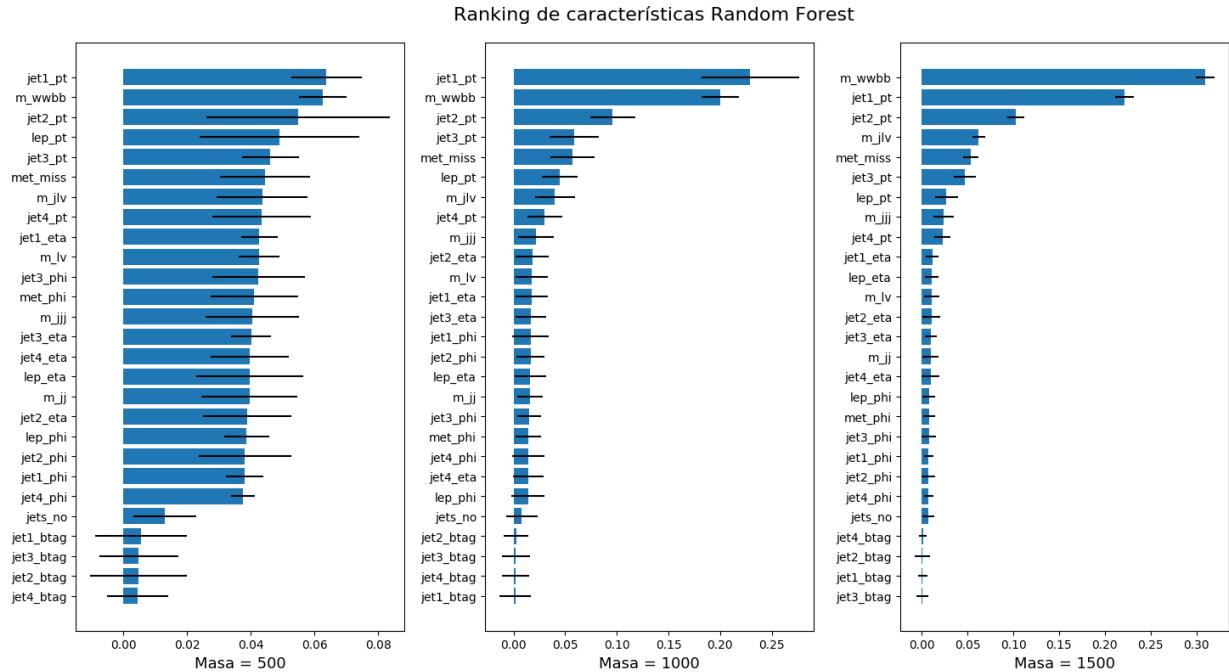
**Figura 23:** Resultados de Random Forest variando el número de características en el entrenamiento.

Por otra parte, en la figura 24 hemos construido el ranking de características para las masas de 500, 1000 y 1500  $GeV$ s (los resultados de 750 y 1250  $GeV$ s se encontrarían en el medio). Nos llama la atención lo siguiente:

- En la masa de 500  $GeV$ s hay más incertidumbre, lo cual era de esperar viendo el resultado de la Figura 23. La diferencia entre la importancia relativa de características es menor que en masas mayores.
- En la parte más alta del ranking aparecen el momento transversal del primer jet y la masa invariante final, resolviendo prácticamente el problema para masas altas.
- A continuación de estas dos características aparece el momento transversal del segundo jet, los demás momentos y algunas masas invariantes.
- El orden de aparición de las masas invariantes es  $m_{wbb}$ ,  $m_{jlv}$ ,  $m_{jjj}$ ,  $m_{lv}$  y  $m_{jj}$ . Lo cual tiene sentido pues es más importante la masa reconstruida de la partícula  $X$ , seguida de los quarks top y antitop y por último de las partículas  $W$ .
- Las *pseudorapidity* aparecen en la parte media del ranking, mezclándose con las masas invariantes de menor importancia.
- En la parte más baja aparece siempre la etiqueta *btag* (ésto es lógico pues solo nos dice el tipo de quark que ha generado el jet), y el número de jets. Así pues, **parece que el número de jets no es importante para el modelo a la hora de clasificar**. Esto puede ser debido a que es una variable entera que solo tiene 10 valores diferentes.

- Inmediatamente antes que el número de jets aparecen los ángulos  $\phi$ , cuya distribución es uniforme. Aunque su importancia no es muy diferente de la de otras características (como las variables de *pseudorapidity* o algunas masa invariantes).

En los siguientes párrafos realizaremos una eliminación de variables de acuerdo a estos resultados.



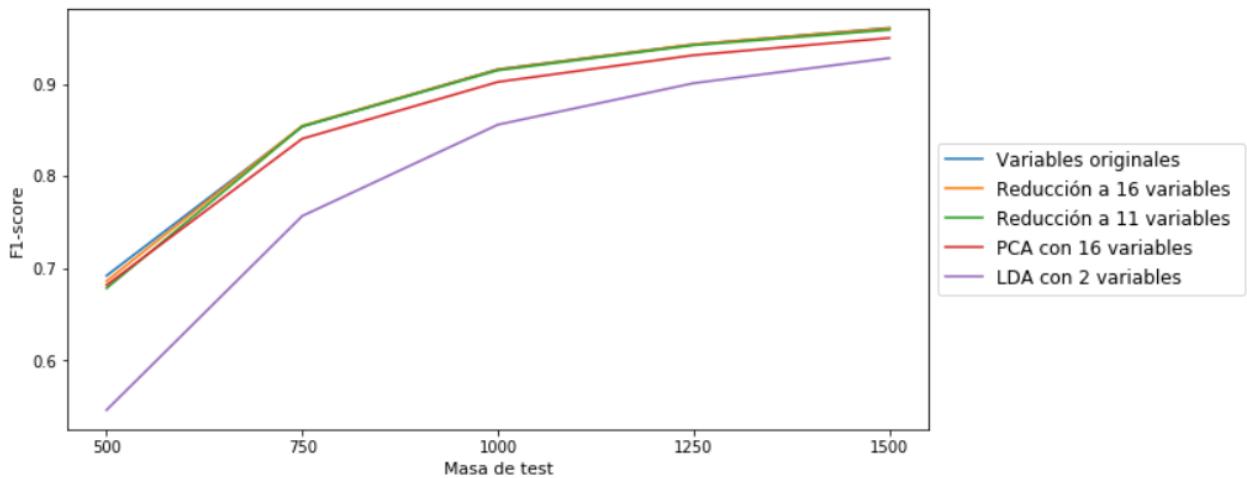
**Figura 24:** Ranking de características Random Forest.

### Comparación de los resultados de de Random Forest.

El siguiente experimento hemos realizado la comparación de las técnicas de reducción de variables. Hemos utilizado los modelos Random Forest para comparar las siguientes configuraciones:

- Dataset con las variables originales.
- Dataset con 16 variables, eliminando las etiquetas *btag*, el número de jets y los ángulos azimut. Se han eliminado estas variables por ser las que aparecían en último lugar del ranking de características.
- Dataset con 11 variables, eliminando también las *pseudorapidity*. Aunque no aparecían los últimos lugares en el ranking de características, en las visualizaciones no parecían ser muy discriminantes.
- Extracción de características con PCA, quedándonos con 16 variables ortogonales.
- Extracción de características con LDA, quedándonos con 2 variables, ya que es el máximo en un problema binario.

Comparación de F1-score para las diferentes técnicas de reducción de características



**Figura 25:** Aplicación de técnicas de reducción de características. En la reducción a 16 variables se han eliminado las etiquetas *btag*, el número de jets y los ángulos azimut. En la reducción a 11 variables se han eliminado también las *pseudorapidity*

En la Figura 25 podemos ver los siguientes resultados:

- Las reducciones a 11 y 16 variables dan los mismos resultados que el dataset completo. A excepción de la resolución del problema para la masa de 500 *GeV*s, en cuyo caso el resultado es ligeramente peor.
- PCA no nos aporta nada nuevo, sus resultados son ligeramente inferiores a los casos en los que se han usado las variables originales, por lo tanto no la consideraremos.
- Los resultados de LDA son los peores. Esto ya se esperaba debido a que solo usa dos variables para clasificar. Podría ser un método útil en el caso de tener un problema multiclasa.

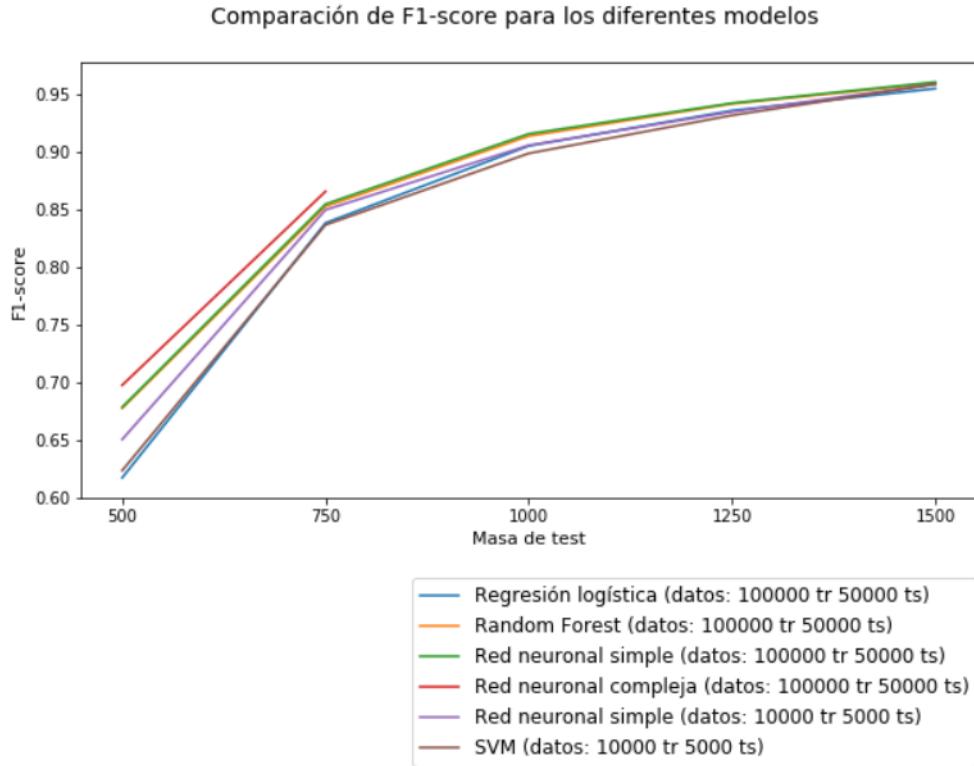
A partir de estos resultados hemos decidido reducir el conjunto a 11 variables, ya que la diferencia con la reducción a 16 variables es mínima. En la Tabla 5 del Anexo vemos que la diferencia de *F1\_score* entre las características originales y la reducción a 11 es de  $0,691 - 0,678$ , para la masa de 500 *GeV*s. Le hemos dado más peso a la posibilidad de eliminar más variables que a esa décima de diferencia. Por tanto en los demás modelos prescindiremos de las etiquetas *btag*, el número de jets, los ángulos azimut y las *pseudorapidity* del dataset. Sin embargo, debemos recordar que los ángulos azimut se utilizan también para la construcción de las masas invariantes.

### 3.4.3. Comparativa de resultados para los diferentes algoritmos

Ahora que conocemos los clasificadores que vamos a usar y las características del dataset con las que vamos a trabajar, vamos a presentar una comparativa entre los diferentes modelos.

De nuevo trabajaremos con 100000 datos de train y 50000 de test separando por masas. En todos los algoritmos hemos usado los mismos datos. A excepción del SVM, por ser más costoso computacionalmente, hemos trabajado con 10000 datos para train y 5000

para test, y lo hemos comparado con una red neuronal simple para la que hemos usado los mismos datos.

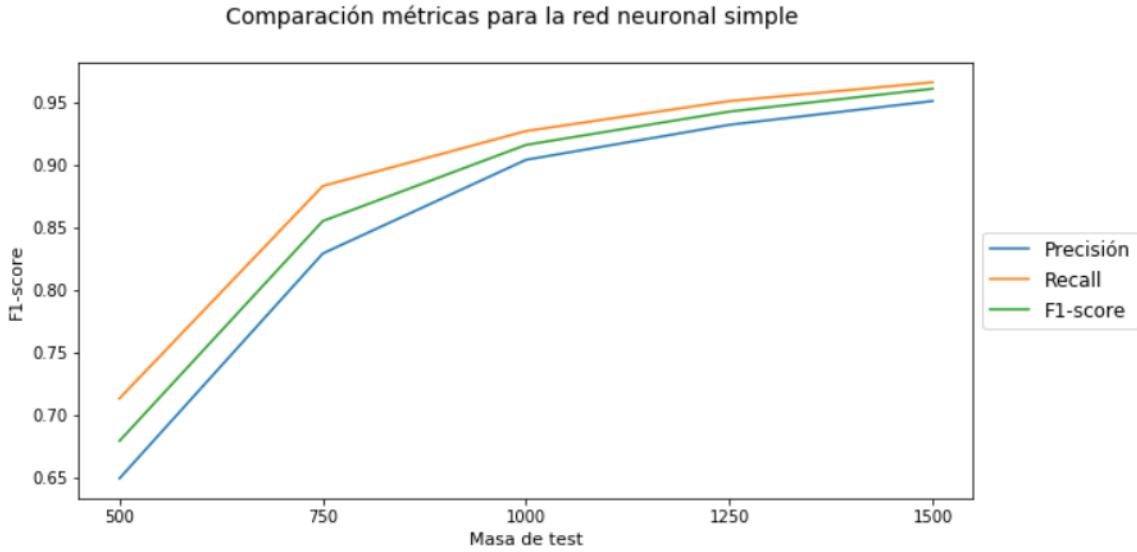


**Figura 26:** *F1-score* resultante para los diferentes algoritmos.

En la Figura 26 podemos ver los resultados para la métrica *F1\_score*. Observamos lo siguiente:

- A masa más baja peor resultado. El poder de clasificación de los algoritmos es más bajo a bajas energías. Esto ya lo veíamos en las distribuciones estadísticas, donde la distribución de ruido era parecida a la de señal a 500 *GeV*s. Así, a la región de bajas energías la llamamos *boosted region* (masas de 500 y 750 *GeV*s) y a la región de altas energías *resolved region* (masas superiores a 1000 *GeV*s). En estos casos el problema está resuelto con *F1\_score* superiores a 0,9
- El algoritmo que mejor resultados presenta es la red neuronal compleja (construida con la librería *Keras*). Ésta solo se ha lanzado para los caso de masa de 500 y 750 *GeV*s debido al gran tiempo de cálculo que requería.
- A la red neuronal compleja le siguen las redes neuronales simples y los Random Forest. Aún así la regresión logística queda muy cerca de ellos a masas altas. Hemos seleccionado los Random Forest y las redes neuronales simples para trabajar en los siguientes experimentos ya que ofrecen un buen resultado y son mucho más rápidos que la red neuronal construida con *Keras*.
- Cuando bajamos el número de datos de entrenamiento el resultado es ligeramente peor. Además el SVM es peor que la red neuronal simple. Hemos calculado el número de vectores soporte y van desde 1500 en la masa más alta a 8500 en la masa de 500

*GEVs* (Tabla 7 del Anexo). Estos números son muy altos comparados con el número de muestras de entrenamiento, con lo que descartamos el modelo SVM.



**Figura 27:** *F1-score*, precisión y sensibilidad (recall) para la red neuronal simple.

Por otra parte hemos representado la precisión, la sensibilidad y *F1\_score* para la red neuronal simple (Figura 27. Hemos construido también la matriz de confusión (Tabla 4)

Masa	VP	VN	FP	FN
500	35.7	30.6	19.3	14.4
750	44.5	40.5	9.1	5.9
1000	46.1	45.4	4.9	3.6
1250	47.5	46.7	3.5	2.4
1500	48.2	47.6	2.5	1.7

**Tabla 4:** Matriz de confusión para la red neuronal simple, resultados en porcentajes.

Observamos que:

- La Precisión es inferior la sensibilidad. Siendo la diferencia algo más de 5 centésimas en la masa de 500 *GEVs* y disminuyendo a medida que vamos a altas energías.
- El modelo predice más resultados positivos (señal) que negativos (ruido). Siendo un 55 % VS 45 % en para la masa de 500 *GEVs* y disminuyendo para las demás.
- Para la masa de 500 *GEVs* el modelo falla un 34 % de las veces, de las cuales un 20 % son falsos positivos. Para 750 *GEVs* el porcentaje de fallo es de un 15 % de las veces, con 9 % de falsos positivos, mientras que para las demás masas el porcentaje de fallo es menor al 10 % con diferencias muy pequeñas entre falsos positivos y negativos.

Por tanto podemos concluir que en general los modelos tienden a predecir más casos de señal, generando así un número mayor de falsos positivos que de negativos. Lo cual se hace relevante en el régimen de bajas energías.

En la sección de Anexo mostramos todos los resultados numéricos (Tablas 6, 7), así como los hiperparámetros escogidos para cada uno de los algoritmos (Tabla 8).

#### 3.4.4. Error sistemático del b-tag

Como ya hemos visto en el apartado de visualizaciones el etiquetado de *btags* influye en la distribución de los datos. Esto hace que a la hora de clasificar se introduzca un error sistemático. Para comprobar el error introducido hemos construido redes neuronales simples y Random Forest entrenados con los siguientes datos:

- Train con proporción de 86-13-1 (2 jets *b*, 1 jet *b* , 0 jets *b*). Es el caso del conjunto de datos.
- Train con proporción 65-33-2, sería más cercano al caso real.
- Train con datos cuando siempre hay 2 *btags*.
- Train con datos cuando siempre hay 1 *btag*.
- Train con datos cuando siempre hay 0 *btags*.

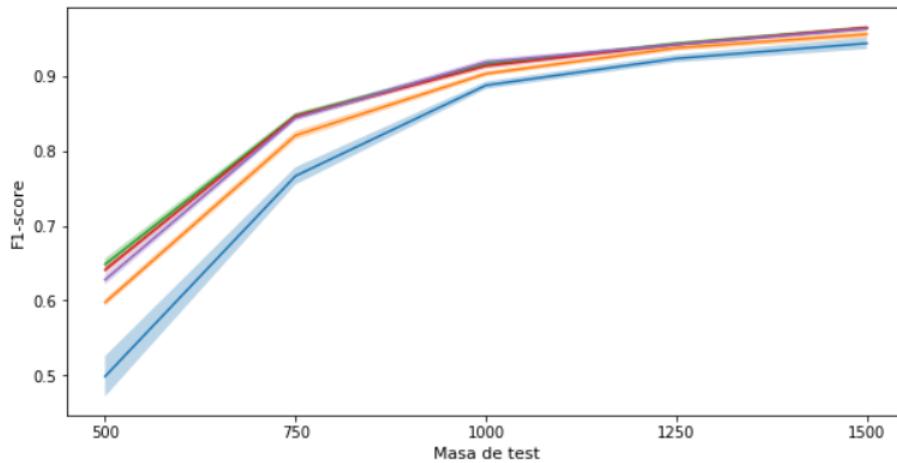
En todos los casos se ha entrenado sobre el mismo train y testeado sobre el mismo test, para el cual todos los datos son correctos (2 *btags*). La proporción utilizada ha sido de 9300 datos de train y 4650 datos de test. Esto es debido a que 9300 es aproximadamente el número de datos de train para los que no hay ningún jet *b*. Al utilizar pocos datos de entrenamiento se han realizado 10 iteraciones en las que se ha ido variando la inicialización de la semilla aleatoria de los modelos. Es decir, la inicialización de los pesos en la red neuronal y los conjuntos de datos y variables en los árboles de Random Forest. Al final se ha calculado la media y la desviación típica. Los resultados numéricos de las medias para los coeficientes *F1-Score* aparecen en la Tabla 9 del Anexo, la representación se muestra en la Figura 28.

Llegamos a las siguientes conclusiones:

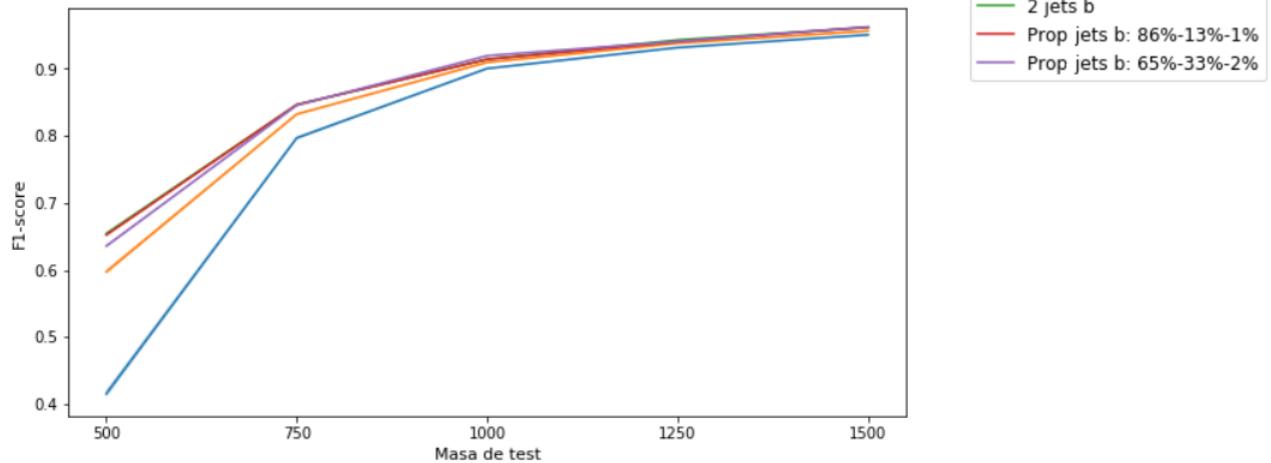
- Como era de esperar los modelos clasifican mejor cuando se entrena con todos los datos correctos (2 jets *b*). Mientras que para el caso más incorrecto (0 jets *b*) encontramos los peores resultados. Para el caso de 0 jets *b* los resultados son menores de 0,5 de *F1-score*. Lo cual supone decir que cuando no hay ningún jet etiquetado como *b* los resultados son peores que si tirásemos una moneda al aire en masas muy bajas.
- Por otra parte conforme avanzamos a masas superiores, el resultado va siendo mejor incluso aunque los jets estén mal etiquetados. Esto se debe a que la etiqueta de los jets afecta principalmente a las masas invariantes, ya que causa que no se reconstruyan correctamente. En secciones anteriores hemos visto que el **el momento transversal del primer jet es una variable muy discriminante, al no verse afectada por el etiquetado de los jets, por si sola es capaz de resolver el problema dando muy buenos resultados en masas altas**.
- Si comparamos Random Forest con la red neuronal, vemos que éste es ligeramente superior en algunos casos. Sin embargo todos los resultados son muy similares exceptuando 0 jets *b* donde la red neuronal es superior (0,5 frente a 0,4).

- Las desviaciones típicas son mayores en la red neuronal que en Random Forest. Esto se debe principalmente a la variabilidad la red neuronal en función de la inicialización de la semilla aleatoria. El Random Forest es un modelo que tiene poca varianza ya que es el resultado de la combinación de un número de árboles de decisión (500 en nuestro caso), como se ha lanzado 10 veces el algoritmo los resultados son el promedio de 5000 árboles de decisión, lo que hace que la varianza sea pequeña. Observando la desviación típica de la red vemos que **la red neuronal es más sensible a su inicialización aleatoria cuando el etiquetado de los datos es incorrecto.**
- Los casos en los que se entrena con el 100 %, 86 % o 65 % de datos bien etiquetados apenas presentan diferencias. Además su desviación típica es pequeña. Por tanto podemos concluir que **si en un caso real entrenamos con menos del 35 % de datos incorrectamente etiquetados los resultados no se deberían ver afectados.**

Comparación de F1-score para las diferentes configuraciones de b-tag (RED NEURONAL)



Comparación de F1-score para las diferentes configuraciones de b-tag (RANDOM FOREST)

**Figura 28:** Comparativa de la métrica *F1-score* para diferentes configuraciones de etiquetado *btag*. Red neuronal (arriba), Random Forest (abajo).

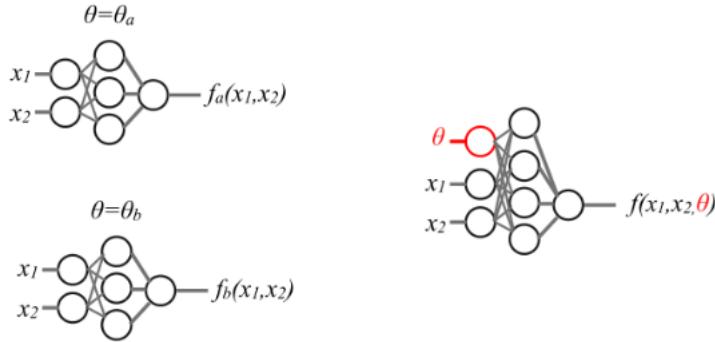
### 3.4.5. Redes Neuronales Parametrizadas

Hasta ahora hemos estudiado el conjunto de datos separándolo según la masa de resonancia de la señal, y para cada una de las masas hemos construido y evaluado diferentes clasificadores. Esto supone un problema ya que a priori no conocemos la masa de la partícula  $X$ . En las simulaciones se ha hecho un barrido de valores dentro del rango de energías en el que se sospecha que pueda estar esa masa (500, 750, 1000, 1250 y 1500  $GeV$ s). Lo cual no quiere decir que la masa sea exactamente 500, ya que podría ser 653, 1295 o incluso 1693  $GeV$ s y salirse del rango.

Una de las posibles soluciones a esta problemática sería el uso de los modelos parametrizados, los cuales se basan en incluir la masa como una característica más [15]. En un caso real la idea sería entrenar el modelo con masas provenientes de las simulaciones. A la hora de predecir sobre datos reales se añadiría en estos un parámetro de masa. Se podrían hacer varias pruebas con masas sospechosas a ser la masa de la partícula  $X$ , y que no tienen por qué ser las masas utilizadas para entrenar. Estas posibles masas se podrían estimar por ejemplo de las visualizaciones de la masa invariante final.

Este tipo de modelo presenta la ventaja de que no es necesario construir un clasificador para cada una de las masas simuladas, además de contar con el poder de la interpolación para testear con masas con las que no ha sido entrenado.

El inconveniente principal es que necesitamos dar la masa como parámetro en los datos a predecir. Además la robustez del modelo depende sobre todo de cómo las distribuciones de señal cambian con el parámetro de masa.



**Figura 29:** Estructura de la red neuronal parametrizada

Nosotros hemos seleccionado las redes neuronales simples como modelos a parametrizar (Figura 29). Para comprobar la efectividad de las redes parametrizadas hemos construido redes con las siguientes configuraciones de datos:

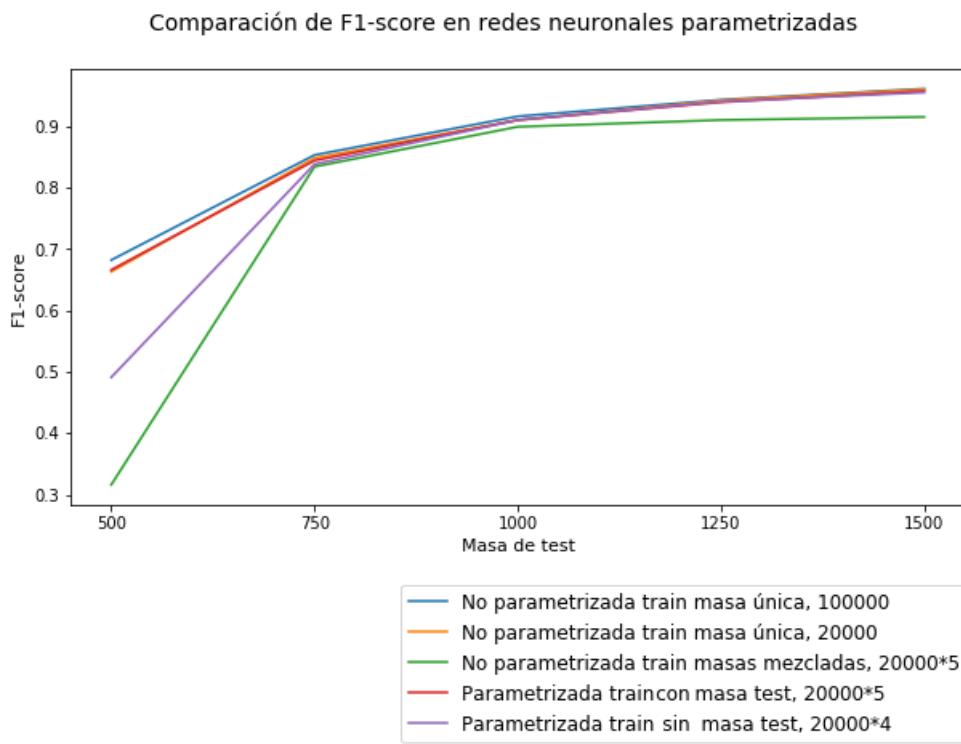
- Red neuronal entrenada con la masa a testear, se escogen los datos para esa masa y se elimina la masa como característica. Este sería el caso que ya hemos visto en el apartado de comparativa de modelos. Hemos construido una red entrenada con 100000 datos y otra con 20000 (para comparar con las redes que describimos a continuación).
- Red neuronal entrenada con datos de todas las masas y testeada con datos de una masa en concreto. El parámetro de masa también es eliminado. La única diferencia

con el caso anterior es que estamos entrenando el modelo sin seleccionar la masa, es decir cogemos la misma cantidad de datos para cada masa y entrenamos.

- Red neuronal parametrizada entrenada con todas las masas y testeada sobre una masa. En este caso la masa entra como parámetro del modelo, se coge una cantidad similar de datos de entrenamiento para cada masa y se prueba sobre una masa en concreto.
- Red neuronal parametrizada entrenada con todas las masas menos con la masa del test. Aquí se entrenaría la red con todas las masas menos una, por ejemplo con 500, 750, 1250, 1500  $GeV$ s y se probaría sobre la masa de 1000  $GeV$ s.

Cabe destacar que en todos los casos hemos utilizado el mismo conjunto de datos de test, por lo que los resultados son comparables. En el caso de las redes neuronales parametrizadas nos vimos en la necesidad de añadirle el parámetro de masa a los datos etiquetados como ruido. Por tanto en el caso del train escogimos como masa aleatoriamente una de las masas que se utilizaban en los datos de señal de train. En el test, asignamos al ruido la masa de los datos de señal.

Hemos construido cinco redes neuronales diferentes, en la Tabla 10 del Anexo mostramos los resultados de este experimento, en la última columna aparecen bien descritos los casos utilizados para entrenamiento y test. Estos resultados los mostramos condensados en la Figura 30.



**Figura 30:** Comparativa de la métrica F1-score en redes neuronales parametrizadas.

Llegamos a las siguientes conclusiones:

- Los mejores resultados son para la red única, entre las dos redes construidas, el

entrenamiento con 100000 datos ofrece resultados muy similares al entrenamiento con 20000 datos.

- Los peores resultados se obtienen en el caso de la mezcla de masas, sobre todo en los extremos (masas de 500 o 1500  $GeV$ s). En los valores centrales de masa el resultado solo es ligeramente peor. Añadir la masa ya sea en una red parametrizada o no mejora la *performance* del modelo.
- Entrenando con 20000 datos y masa única, el resultado es idéntico al obtenido entrenando con la red neuronal parametrizada que incluye la masa de test. En este caso, los datos que contenían masas adicionales no han añadido ninguna información útil.
- En el caso de la red neuronal parametrizada que no entrena con masa de test, encontramos que los resultados son buenos para todas las masas excepto para 500  $GeV$ s. **Por tanto, podemos concluir que las redes parametrizadas interpolan bien excepto para masas muy bajas.**

## 4. Lineas futuras

El paso lógico a este trabajo sería el uso de las redes neuronales que hemos utilizado para el etiquetado de datos reales. Sin embargo es necesario realizar algunos estudios previos al trabajo con datos experimentales.

Por una parte, sería necesario realizar el mismo estudio con datos simulados más detallados. Ahora que sabemos qué variables son las más importantes podríamos simular datos con más detalle. El conjunto de datos utilizado estaba formado por las variables referentes a los cuatro jets más energéticos. Un caso más aproximado a la realidad sería contar con las características de los demás jets, ya que como hemos visto en algunos eventos se producen más de una decena. También sería conveniente contar con un etiquetado *btag* más realista. En datos reales la etiqueta se asigna como una probabilidad de que el jet sea o no *b*, mientras que en el dataset utilizado esto no venía contemplado.

Otro problema que surge es que la masa de resonancia es totalmente desconocida. A pesar de que el uso de redes neuronales parametrizadas permite usar como parámetro cualquier masa, quizás se necesitaría de la simulación de otras masas para una mejor aproximación.

Por otra parte, también sería necesario realizar un estudio sobre clases desbalanceadas. Ya hemos visto en los resultados de los modelos que la métrica *precisión* es menor a la sensibilidad. Es decir, nuestro modelo predice más falsos positivos que negativos. Esto no llega a presentar un problema muy grande porque el dataset está formado por clases balanceadas. Sin embargo, en la realidad, esto no tiene por qué ser así. El bosón de Higgs, por ejemplo, aparece en una colisión una de cada millón de veces [46]. La aparición de la partícula *X*, en caso de que existiese, podría ser incluso menor. Por tanto vemos necesario realizar un experimento que contemple los casos en el que la señal fuese mucho menor que el ruido. Para este estudio se podría trabajar también con datos simulados más detallados que aporten más información que los usados hasta ahora.

## 5. Conclusiones

El estudio  $t\bar{t}$  en el detector ATLAS del LHC es un experimento ambicioso en la búsqueda de partículas exóticas. Su objetivo es la búsqueda de una resonancia o partícula que de lugar a los quarks  $t\bar{t}$  en las colisiones protón-protón. En el caso de encontrarse supondría un importante avance dentro del campo de la física de altas energías, ya que conllevaría la confirmación de modelos de nueva física que van más allá del modelo *Standard*.

A lo largo de esta memoria hemos realizado un riguroso estudio de la aplicación de técnicas de *machine learning* en este experimento. Hemos utilizado datos simulados para construir clasificadores que sean capaces de diferenciar entre señal (*Beyond Standard Model*) y ruido (*Standard Model*). Los datos simulados contaban con diferentes variables cinemáticas correspondientes a eventos de colisión, separados por la masa de la partícula hipotética fijada en las simulaciones.

A través de la visualización de las distribuciones de variables y las técnicas de extracción de características, hemos visto que **las variables más importantes son el momento transversal del jet más energético y la masa invariante final reconstruida**, seguidos de otros momentos transversales y masas invariantes. Hemos comprobado que bastan unas pocas variables para resolver el problema en régimen de altas energías.

Mediante el uso de diferentes clasificadores hemos visto que **se trata de un problema fácilmente resoluble en regímenes de altas energías, con porcentajes de acierto de más del 90 %**. Por otra parte, para bajas energías los resultados eran peores, destacando el caso de masa de 500 $GeV$ s, con un *F1\_score* menor a 0,7. Acerca de los algoritmos utilizados **destacamos las redes neuronales y los Random Forest**, cuyos resultados eran los más altos. Por otra parte, hemos comprobado la diferencia entre falsos positivos y negativos, concluyendo que en este problema **los modelos tienden a predecir más casos de señal generando un número mayor de falsos positivos**.

Por otra parte hemos realizado experimentos con diferentes etiquetas para los jets  $b$ , concluyendo que **si en un caso real entrenamos con alrededor del 35 % de datos etiquetados incorrectamente los resultados no se deberían ver afectados**. Los resultados empeoran notablemente cuando todos los datos se etiquetan de forma incorrecta.

Por último hemos trabajado con redes neuronales parametrizadas, donde el parámetro de masa se le ha pasado a la red y se ha comprobado su poder de interpolación. El resultado ha sido que **las redes parametrizadas interpolan correctamente excepto para masas muy bajas**.

En definitiva, mediante los diferentes experimentos hemos realizado un estudio minucioso y exhaustivo del problema. El paso siguiente sería trabajar con datos simulados más detallados, con más variables como las de jets menos energéticos o simulaciones adicionales para otras masas. Con dichos datos se podría mejorar este mismo trabajo y expandirlo con experimentos adicionales de clases desbalanceadas, los cuales serían más similares a la realidad en caso que existiera la partícula hipotética  $X$ .

## 6. Anexo

### 6.1. Tablas de resultados

En las siguientes páginas mostramos la recopilación de todos los resultados numéricos de este trabajo.

#### Resultados del experimento de selección de variables

Masa	Red neuronal	F1-score	Kappa
500	Variables originales	0.691	0.347
	Reducción a 16 variables	0.686	0.336
	Reducción a 11 variables	0.678	0.323
	PCA con 16 variables	0.681	0.317
	LDA con 2 variables	0.546	0.090
750	Variables originales	0.853	0.693
	Reducción a 16 variables	0.854	0.696
	Reducción a 11 variables	0.853	0.696
	PCA con 16 variables	0.840	0.663
	LDA con 2 variables	0.756	0.510
1000	Variables originales	0.915	0.830
	Reducción a 16 variables	0.915	0.829
	Reducción a 11 variables	0.914	0.828
	PCA con 16 variables	0.902	0.801
	LDA con 2 variables	0.855	0.713
1250	Variables originales	0.942	0.884
	Reducción a 16 variables	0.942	0.884
	Reducción a 11 variables	0.941	0.883
	PCA con 16 variables	0.941	0.885
	LDA con 2 variables	0.900	0.801
1500	Variables originales	0.960	0.920
	Reducción a 16 variables	0.960	0.919
	Reducción a 11 variables	0.958	0.916
	PCA con 16 variables	0.949	0.898
	LDA con 2 variables	0.927	0.855

**Tabla 5:** Resultados de *F1-Score* del entrenamiento de Random Forest para varias configuraciones de las variables del dataset.

### Resultados la comparativa de modelos

Masa	Modelo	Acc	AUC	Prec	Sens	F1-Score	Kappa
500	Regr Logística	0.618	0.659	0.620	0.614	0.617	0.237
	Random Forest	0.661	0.720	0.648	0.711	0.678	0.323
	RN simple	0.663	0.723	0.649	0.713	0.679	0.326
	RN compleja	0.667	0.729	0.631	0.763	0.698	0.334
750	Regr Logística	0.834	0.896	0.823	0.854	0.838	0.237
	Random Forest	0.847	0.914	0.826	0.883	0.853	0.695
	RN simple	0.850	0.916	0.829	0.883	0.855	0.699
	RN compleja	0.852	0.919	0.823	0.904	0.866	0.707
1000	Regr Logística	0.905	0.960	0.897	0.913	0.905	0.668
	Random Forest	0.914	0.967	0.903	0.926	0.914	0.828
	RN simple	0.914	0.968	0.904	0.927	0.915	0.829
1250	Regr Logística	0.935	0.979	0.925	0.946	0.936	0.810
	Random Forest	0.941	0.984	0.930	0.953	0.941	0.882
	RN simple	0.941	0.984	0.932	0.951	0.941	0.882
1500	Regr Logística	0.954	0.988	0.946	0.963	0.955	0.871
	Random Forest	0.958	0.991	0.950	0.966	0.958	0.916
	RN simple	0.958	0.991	0.951	0.966	0.958	0.916

**Tabla 6:** Métricas obtenidas para cada modelo entrenando con 100000 datos y testeando con 50000.

Masa	Modelo	Acc	AUC	Prec	Sens	F1-Score	Kappa	Número vectores soporte (SVM)
500	RN simple	0.647	0.688	0.643	0.660	0.651	0.294	
	SVM	0.612	0.657	0.604	0.645	0.624	0.224	8572
750	RN simple	0.847	0.908	0.826	0.875	0.850	0.694	
	SVM	0.833	0.899	0.814	0.860	0.837	0.667	4604
1000	RN simple	0.907	0.961	0.890	0.922	0.906	0.814	
	SVM	0.901	0.957	0.891	0.910	0.899	0.802	2636
1250	RN simple	0.932	0.978	0.922	0.947	0.935	0.865	
	SVM	0.930	0.974	0.919	0.946	0.932	0.861	1811
1500	RN simple	0.959	0.988	0.955	0.964	0.959	0.918	
	SVM	0.959	0.988	0.953	0.965	0.959	0.918	1424

**Tabla 7:** Comparativa entre red neuronal y SVM entrenando con 10000 datos y testeando con 5000.

### Resultados del ajuste de hiperparámetros para los diferentes modelos

Modelo	Masa	Hiperparámetros escogidos
Regresión Logística	500	C=1, penalty=l2, solver=liblinear
	750	C=0.1, penalty=l1, solver=liblinear
	1000	C=10, penalty=l2, solver=liblinear
	1250	C=10, penalty=l1, solver=liblinear
	1500	C=1, penalty=l1, solver=liblinear
Random Forest	500	n_estimators= 500, min_samples_leaf=1, max_features= sqrt, criterion=entropy
	750	n_estimators= 350 , min_samples_leaf=1, max_features= sqrt, criterion=entropy
	1000	n_estimators= 350, min_samples_leaf=1, max_features= log2, criterion=entropy
	1250	n_estimators= 350 , min_samples_leaf=1, max_features= sqrt, criterion=entropy
	1500	n_estimators= 500, min_samples_leaf=1, max_features= sqrt, criterion=entropy
SVM	500	C=5, kernel = rbf
	750	C=3, kernel = rbf
	1000	C=5, kernel = rbf
	1250	C=5, kernel = rbf
	1500	C=5, kernel = rbf
RN simple	500	max_iter =2000, alpha=1e-05,hid_layer_sizes=35, solver=lbfgs, activation=tanh
	750	max_iter =1500, alpha=1e-04,hid_layer_sizes=35, solver=lbfgs, activation=tanh
	1000	max_iter =1500, alpha=1e-04,hid_layer_sizes=35, solver=lbfgs, activation=tanh
	1250	max_iter =1500, alpha=1e-04,hid_layer_sizes=35, solver=lbfgs, activation=tanh
	1500	max_iter =1000, alpha=1e-05,hid_layer_sizes=15, solver=lbfgs, activation=tanh
RN profunda	500	solver=adam, hidden_layers=7, neurons=200, activation (última capa) =tanh, learning_rate=0.01, loss=binary_cross_entropy, batch_size=10, epochs=50, dropout=0, alpha=1e-04
	750	solver=adam, hidden_layers=3, neurons=50, activation (última capa) =tanh, learning_rate=0.01, loss=binary_cross_entropy, batch_size=40, epochs=50, dropout=0, alpha=1e-04

**Tabla 8:** Resultado del ajuste de hiperparámetros. Las capas intermedias de la red neuronal profunda se han configurado siempre como ReLu. Otros hiperparámetros que no aparecen se han dejado por defecto.

Vemos como en todos los algoritmos (excepto en las redes neuronales) los hiperparámetros son muy parecidos entre sí para las diferentes masas, por ejemplo en el SVM el mejor kernel es el Gaussiano y en los Random Forest el mejor criterio es la entropía.

Las diferencias en las redes neuronales las encontramos en el número de capas (en la red neuronal profunda), de neuronas o en el número de interacciones. Vemos que a mayor complejidad del problema (masa de 500 o 750  $GeV$ s) más compleja es la estructura de la red neuronal. Debemos tener en cuenta también que los hiperparámetros dependen de como se haya inicializado la red. Si la inicialización ya se encuentra cerca del mínimo el número de interacciones o las épocas va a ser bajo.

### Resultados según los diferentes etiquetados $btag$

Masa	Configuración	Red Neuronal	Random Forest
500	0 jets $b$	0.498	0.415
	1 jets $b$	0.597	0.597
	2 jets $b$	0.648	0.654
	Prop jets $b$ : 86%13%1%	0.640	0.652
	Prop jets $b$ : 65%33%2%	0.627	0.635
750	0 jets~ $b$	0.766	0.796
	1 jets $b$	0.820	0.832
	2 jets $b$	0.847	0.846
	Prop jets $b$ : 86%13%1%	0.846	0.846
	Prop jets $b$ : 65%33%2%	0.843	0.845
1000	0 jets $b$	0.887	0.900
	1 jets $b$	0.902	0.909
	2 jets $b$	0.915	0.914
	Prop jets $b$ : 86%13%1%	0.913	0.914
	Prop jets $b$ : 65%33%2%	0.918	0.919
1250	0 jets $b$	0.923	0.931
	1 jets $b$	0.937	0.937
	2 jets $b$	0.943	0.940
	Prop jets $b$ : 86%13%1%	0.941	0.940
	Prop jets $b$ : 65%33%2%	0.941	0.942
1500	0 jets $b$	0.943	0.950
	1 jets $b$	0.955	0.956
	2 jets $b$	0.964	0.961
	Prop jets $b$ : 86%13%1%	0.964	0.961
	Prop jets $b$ : 65%33%2%	0.963	0.962

**Tabla 9:** Promedio de resultados de  $F1$ -Score del entrenamiento de redes neuronales y Random Forest para varios etiquetados  $btag$ , los algoritmos se han lanzado 10 veces.

### Redes neuronales parametrizadas

Masa	Red neuronal	F1-score	Kappa	Datos train	Datos test
500	No param, entrena con la masa de test	0.679	0.326	100000 datos con masa = 500	50000 datos con masa = 500
	No param, entrena con la masa de test	0.661	0.300	20000 datos con masa = 500	
	No param, entrena con todas las masas	0.316	0.061	20000 datos para cada masa (100000 total)	
	Param, entrena con todas las masas	0.661	0.293	20000 datos para cada masa (100000 total)	
	Param, entrena con todas las masas menos con la de test	0.491	0.147	20000 datos para cada masa (80000 total)	
750	No param, entrena con la masa de test	0.855	0.699	100000 datos con masa = 750	50000 datos con masa = 750
	No param, entrena con la masa de test	0.845	0.680	20000 datos con masa = 750	
	No param, entrena con todas las masas	0.834	0.669	20000 datos para cada masa (100000 total)	
	Param, entrena con todas las masas	0.844	0.678	20000 datos para cada masa (100000 total)	
	Param, entrena con todas las masas menos con la de test	0.838	0.668	20000 datos para cada masa (80000 total)	
1000	No param, entrena con la masa de test	0.915	0.829	100000 datos con masa = 1000	50000 datos con masa = 1000
	No param, entrena con la masa de test	0.910	0.818	20000 datos con masa = 1000	
	No param, entrena con todas las masas	0.899	0.789	20000 datos para cada masa (100000 total)	
	Param, entrena con todas las masas	0.910	0.814	20000 datos para cada masa (100000 total)	
	Param, entrena con todas las masas menos con la de test	0.910	0.817	20000 datos para cada masa (8000 total)	
1250	No param, entrena con la masa de test	0.941	0.882	100000 datos con masa = 1250	50000 datos con masa = 1250
	No param, entrena con la masa de test	0.942	0.883	20000 datos con masa = 1250	
	No param, entrena con todas las masas	0.910	0.810	20000 datos para cada masa (100000 total)	
	Param, entrena con todas las masas	0.939	0.875	20000 datos para cada masa (100000 total)	
	Param, entrena con todas las masas menos con la de test	0.940	0.878	20000 datos para cada masa (80000 total)	

1500	No param, entrena con la masa de test	0.958	0.916	100000 datos con masa = 1500	50000 datos con masa = 1500
	No param, entrena con la masa de test	0.959	0.915	20000 datos con masa = 1500	
	No param, entrena con todas las masas	0.915	0.819	20000 datos para cada masa (100000 total)	
	Param, entrena con todas las masas	0.957	0.914	20000 datos para cada masa (100000 total)	
	Param, entrena con todas las masas menos con la de test	0.955	0.911	20000 datos para cada masa (100000 total)	

Tabla 10: Resultado de los experimentos con redes neuronales parametrizadas.

## 6.2. Ejemplo de código utilizado

A continuación presentamos el código que hemos utilizado para construir las redes neuronales simples. La construcción de todo el código utilizado en esta memoria es similar a este ejemplo.

```
# LIBRERÍAS
import pandas as pd
import numpy as np
import warnings
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV
from sklearn import metrics
from sklearn.neural_network import MLPClassifier
warnings.filterwarnings('ignore')
from sklearn.preprocessing import MinMaxScaler
from IPython.display import display

# CARGA DEL DATASET
df=pd.read_pickle('data/normalizados/trainpickle')
df_originaltest=pd.read_pickle('data/normalizados/testpickle')

# DEFINICIÓN DE FUNCIONES

# métricas
def Scores(y_true,y_pred):
    tn, fp, fn, tp = metrics.confusion_matrix(y_true, y_pred).ravel()
    prec=tp / (tp + fp)
    recall= tp / (tp + fn)
    F1_score= 2 * (prec * recall) / (prec + recall)
    acc=metrics.accuracy_score(y_true,y_pred)
    kappa_cohen=metrics.cohen_kappa_score(y_true,y_pred)

    return(tn, fp, fn, tp, acc, prec,recall, F1_score,kappa_cohen)
```

```

# Red neuronal
def model(X_train,y_train,X_test):
    scaler = MinMaxScaler()
    scaler=scaler.fit(X_train)
    X_train = scaler.transform(X_train)
    X_test = scaler.transform(X_test)

    model = MLPClassifier()

    # hyperparameter tuning
    params = {
        'solver': ['lbfgs','adam', 'sgd'],
        'max_iter': [1000,1500,2000],
        'alpha': 10.0 ** -np.arange(4, 6),
        'hidden_layer_sizes':[15,25,35,50],
        'learning_rate':['constant', 'invscaling', 'adaptive'],
        'activation':[ 'identity', 'logistic', 'tanh', 'relu'],
        'random_state':[0]}

    grid = GridSearchCV(estimator=model, param_grid=params, cv=5,verbose=1,
    n_jobs=-1)

    grid.fit(X_train,y_train)
    best_model = grid.best_estimator_
    best_model.fit(X_train,y_train)
    y_pred = best_model.predict(X_test)
    y_pred_proba = best_model.predict_proba(X_test)[:,1]
    return(grid,best_model,y_pred,y_pred_proba)

# ALGORITMO

# Metric df:
df_metrics = pd.DataFrame(index=[500,750,1000,1250,1500],
columns=['tn', 'fp', 'fn', 'tp', 'acc', 'prec','recall','F1_score',
'kappa_cohen','auc'])

for i in (500,750,1000,1250,1500):
    print('mass=', i )

    # train
    dfmass1000=df.loc[df['mass'] ==i]
    dfmass0=df.loc[df['mass'] == 0].sample(random_state=1,n=dfmass1000
    .shape[0])
    dfmass1000=pd.concat([dfmass1000, dfmass0])
    .sample(random_state=1,frac=1)
    .reset_index(drop=True)
    dfmass1000['mass']=dfmass1000['mass'].astype(float)

```

```

dfmass1000=dfmass1000
.sample(random_state=1,n=20000*5)
print('train', pd.crosstab(dfmass1000['label'],dfmass1000['mass']))
dfmass1000=dfmass1000.drop('mass', axis=1)

# test
dfmass1000test=df_originaltest.loc[df_originaltest['mass'] == i]
dfmass0test=df_originaltest.loc[df_originaltest['mass'] == 0]
.sample(random_state=1,n=dfmass1000test.shape[0])
dfmass1000test=pd.concat([dfmass1000test, dfmass0test])
.sample(frac=1).reset_index(drop=True)
dfmass1000test=dfmass1000test.sample(random_state=1,n=10000*5)

print('test', pd.crosstab(dfmass1000test['label'],
dfmass1000test['mass']))
dfmass1000test=dfmass1000test.drop('mass', axis=1)

# Selección de características
drop=['lep_eta', 'lep_phi', 'met_phi', 'jets_no', 'jet1_eta',
'jet1_phi', 'jet1_btag', 'jet2_eta', 'jet2_phi', 'jet2_btag',
'jet3_eta', 'jet3_phi','jet3_btag', 'jet4_eta', 'jet4_phi', 'jet4_btag']

X_train=dfmass1000.drop(['label'], axis=1)
X_train=X_train.drop(drop, axis=1)
y_train=dfmass1000.label

X_test=dfmass1000test.drop(['label'], axis=1)
X_test=X_test.drop(drop, axis=1)
y_test=dfmass1000test.label

# lanzamiento modelo
grid,best_model,y_pred,y_pred_proba = model(X_train,y_train,X_test)
print(best_model)

tn, fp, fn, tp, acc, prec,recall, F1_score,kappa_cohen=
Scores(y_test,y_pred)

df_metrics.tn[i]=tn
df_metrics.fp[i]=fp
df_metrics.fn[i]=fn
df_metrics.tp[i]=tp
df_metrics.acc[i]=acc
df_metrics.prec[i]=prec
df_metrics.recall[i]=recall
df_metrics.F1_score[i]=F1_score
df_metrics.kappa_cohen[i]=kappa_cohen
fpr, tpr, _ = metrics.roc_curve(y_test, y_pred_proba)
auc = metrics.roc_auc_score(y_test, y_pred_proba)

```

```
df_metrics.auc[i]=auc
```

```
# RESULTADOS
```

```
display(df_metrics)
```

## Referencias

- [1] A Radovic, M Williams (2018) Machine learning at the energy and intensity frontiers of particle physics *Nature* 560(7716), 41-48
- [2] P Baldi, P Sadowski (2014) Searching for exotic particles in high-energy physics with deep learning *Nature communications*, 5
- [3] L Oliveira, M Paganini, B Nachman (2017) Learning Particle Physics by Example: Location-Aware Generative Adversarial Networks for Physics Synthesis *Computing and Software for Big Science* 1, 4
- [4] HEPMASS DataSet  
<https://archive.ics.uci.edu/ml/datasets/HEPMASS>
- [5] S Berryman (2018) Ancient Atomism *Stanford Encyclopedia of Philosophy*
- [6] H Kragh (1997) J. J. Thomson, the electron, and atomic architecture *The Physics Teacher* 35, 328-332
- [7] A Brandt, Quantum Chromodinamics  
<https://slideplayer.com/slide/9333510>
- [8] D. McMahon (2008) Quantum Field Theory. DeMystified *Mc Graw Hill* 11-88
- [9] G González-Martin (1994) A geometric definition of mass *General Relativity and Gravitation* 26, 1177–1185
- [10] GS. F. Novaes (2000) Standard Model: An Introduction *Instituto de Física Teórica, UNESP*
- [11] J Ellis (2002) Limits of Standard Model *CERN-TH* [arXiv: hep-ph/0211168]
- [12] M. Czakon , P.Fiedler, A.Mitov (2013) Total Top-Quark Pair-Production Cross Section at Hadron Colliders Through  $O(\alpha_4 S)$  *Phys. Rev. Let.* 110, 252004
- [13] G Filipelli, Dawn of the quark ages  
<https://ulaulaman.wordpress.com/2016/05/25/dawn-of-the-quark-ages/>
- [14] C H. Colwell, Mass of the Top Quark  
[http://dev.physicsslab.org/Document.aspx?doctype=2&filename=AtomicNuclear\\_TopQuarkMass.xml](http://dev.physicsslab.org/Document.aspx?doctype=2&filename=AtomicNuclear_TopQuarkMass.xml)
- [15] P Baldi, K Cranmer (2016) Parameterized Machine Learning for High-Energy Physics *Eur. Phys. J.* C76(5), 235
- [16] H E Haber, G L Kane (1985) The search for supersymmetry: Probing physics beyond the standard model *Physics Reports* 117, 2–4, 75-263
- [17] K Becker, M Becker(2007) String theory and M-theory: A modern introduction *Cambridge University Press 2007*
- [18] G Servant, T Tait (2003) Is the Lightest Kaluza-Klein Particle a Viable Dark Matter Candidate? *Nuclear Physics* B650, 391-419
- [19] R. M. Harris, C. T. Hill, S. J. Parke (1999) Cross-section for topcolor Z-prime(t) decaying to t anti-t: Version 2.6 *Nuclear Physics* B650, 391-419

- R. M. Harris, C. T. Hill, S. J. Parke, 1004 Cross-section for topcolor Z-prime(t) decaying to t anti-t: Version 2.6 (1999) CERN-TH 1005 [arXiv: hep-ph/9911288].
- [20] CERN's accelerator complex  
<https://home.cern/science/accelerators/accelerator-complex>
- [21] L Scodellaro (2017) *b* tagging in ATLAS and CMS *ATLAS and CMS Collaborations* [arXiv:1709.01290]
- [22] J Alwall, M Herquet (2011) MadGraph 5 : Going Beyond *JHEP* , 1106, 128
- [23] T. Sjöstrand, S. Mrenna (2008) A brief introduction to PYTHIA 8.1 *Comput. Phys. Commun.*, 178, 852 [arXiv:0710.3820]
- [24] J. de Favereau (2014) DELPHES 3, A modular framework for fast simulation of a generic collider experiment *JHEP* , 1402, v057
- [25] E. Daw, Lecture 7 - Rapidity and Pseudorapidity  
[http://www.hep.shef.ac.uk/edaw/PHY206/Site/2012\\_course\\_files/phy206rlec7.pdf](http://www.hep.shef.ac.uk/edaw/PHY206/Site/2012_course_files/phy206rlec7.pdf)
- [26] L.D Landau, E.M Lifshitz (1980) The Classical Theory of Fields Course of Theoretical Physics *Butterworth Heinemann*, 36-38
- [27] Pandas  
<https://pandas.pydata.org/>
- [28] Matplotlib  
<https://matplotlib.org/>
- [29] Seaborn  
<https://seaborn.pydata.org/>
- [30] SPSS Tutorials: Pearson Correlation  
<https://libguides.library.kent.edu/SPSS/PearsonCorr>
- [31] D Gujarati (2009) Multicollinearity: what happens if the regressors are correlated?. Basic Econometrics *McGrawHill* 341-386
- [32] W. Barbe et al (2018) A search for top-antitop resonances in the lepton+jets final state using  $36.1fb^{-1}$  of proton-proton collisions at  $\sqrt{s} = 13TeV$  *ATL-COM-PHYS* 2016-1036, 43-44
- [33] Scikit-learn  
<https://scikit-learn.org/>
- [34] Keras  
<https://keras.io/>
- [35] Metrics and scoring: quantifying the quality of predictions  
[https://scikit-learn.org/stable/modules/model\\_evaluation.html](https://scikit-learn.org/stable/modules/model_evaluation.html)
- [36] V.Abraira (2001) El índice kappa Unidad de Bioestadística Clínica. *Hospital Ramón y Cajal, Madrid* 27,5

- [37] Logistic Regression  
[https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.LogisticRegression.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html)
- [38] Random Forest  
<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>
- [39] L Rokach, O Maimon (2008) Data mining with decision trees: theory and applications  
*World Scientific Pub Co Inc*, 61-67
- [40] SVC  
<https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>
- [41] MPL Classifier  
[https://scikit-learn.org/stable/modules/generated/sklearn.neural\\_network.MLPClassifier.html](https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html)
- [42] R Rojas, The backpropagation algorithm of Neural Networks-A Systematic Introduction  
<http://page.mi.fu-berlin.de/rojas/neural/chapter/K7.pdf>
- [43] N.M. Nawi, W.H. Atomi, M.Z.Rehman (2013) The Effect of Data Pre-processing on Optimized Training of Artificial Neural Networks *Procedia Technology* 11, 32–39
- [44] PCA  
<https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html>
- [45] LDA  
[https://scikit-learn.org/stable/modules/generated/sklearn.discriminant\\_analysis.LinearDiscriminantAnalysis.html](https://scikit-learn.org/stable/modules/generated/sklearn.discriminant_analysis.LinearDiscriminantAnalysis.html)
- [46] G. Aad et al (2012) Observation of a new particle in the search for the Standard Model Higgs boson with the ATLAS detector at the LHC *ATLAS Collaboration Physics Letters B* 716, 1-29