Universidad Nacional Autónoma de México

FACULTAD DE CIENCIAS

AISLAMIENTO DE COLISIONES PP RARAS EN ALICE LHC USANDO TÉCNICAS DE APRENDIZAJE DE MÁQUINA

TESIS

QUE PARA OPTAR POR EL GRADO DE:

Físico

PRESENTA:

ISAÍ ROBERTO SOTARRIVA ÁLVAREZ

DIRECTOR DE TESIS:

DR. ANTONIO ORTIZ VELÁSQUEZ

CIUDAD UNIVERSITARIA, Cd. Mx., 2019

AGRADECIMIENTOS

A mi familia por estar siempre a mi lado y apoyarme durante toda la licenciatura.

A mi asesor el Dr. Antonio Ortiz Velásquez por el tiempo y atención que me ha brindado para poder concluir este proyecto de tesis, así como también durante el servicio social donde aprendí las herramientas básicas que luego aplicaría en esta tesis.
A Luciano Díaz del departamento de cómputo por atenderme en mis dudas y problemas técnicos. Gracias por todo su apoyo con el clúster, donde su ayuda con el uso de librerías y terminal me ahorro días de frustración.
A Omar Vásquez Rueda por enseñarme a utilizar Root y Pythia. También muchas gracias por resolver todas mis dudas con prontitud.
A Sergio Iga por enseñarme a enviar trabajos al clúster y por pasarme sus programas para el uso del clúster en forma automática.
A la UNAM y especialmente al Instituto de Ciencias Nucleares por permitirme usar sus instalaciones y equipo de cómputo.

Al programa de apoyo a proyectos de investigación e innovación (PAPIIT) por la beca otorgada correspondiente al proyecto IN102118 titulado “Búsqueda del plasma de quarks y gluones en colisiones protón-protón y desarrollo de nuevos detectores”.

Isaí Roberto Sotarriva Álvarez

RESUMEN

Hasta el momento se habían considerado a las colisiones protón-protón (pp) como referencia para iones pesados (p-A y A-A) pues se consideraba que estas no presentaban tiempos de termalización o volúmenes de interacción lo suficientemente grandes para permitir la formación del plasma de quarks y gluones (QGP). Sin embargo, datos recientes (2010) provenientes del LHC revelan similitudes en los datos pp que parecen sugerir la posible formación del QGP en pp[40]. En MC la inclusión de las múltiples interacciones partónicas (MPI) combinado con la reconexión por color han podido reproducir patrones de flujo radial similares a los encontrados en los datos, por lo que es de interés aislar y analizar eventos con alta presencia de MPI donde los efectos del evento subyacente (UE) son más intensos.
Hasta el momento el método utilizado para aislar eventos con alto número de MPI ha consistido en seleccionar aquellos eventos con valores altos de multiplicidad. El objetivo principal de este trabajo es investigar y mostrar como los algoritmos de aprendizaje de máquina pueden mejorar la forma en que aislamos eventos con alto N_{MPI}, mejorando la calidad de la selección, así como aumentando la eficiencia de selección del del proceso a la vez que se disminuyen los sesgos en las selecciones. Para ello se analizaron simulaciones de colisiones pp a $\sqrt{s} = 13\,\text{TeV}$ generadas en Pythia las cuales incluyen simulación de la reconstrucción en los detectores del experimento ALICE del LHC mediante GEANT. Para la realización del análisis de desempeño de los métodos de aprendizaje de máquina se utilizaron dos simulaciones oficiales de ALICE de propósito general ambas correspondientes a la corrida analizada (LHC16k).

En este trabajo se presenta una potencial aplicación de las nuevas tecnologías y algoritmos a la resolución de problemas de clasificación de eventos en colisiones pp a energías del LHC. Se explora su utilidad a la hora de clasificar de acuerdo a variables "verdaderas" a nivel generador, partiendo únicamente de variables reconstruidas tras simular los efectos de detector (Ver Sección 2.4 "Efectos de detector"), proveyendo de una herramienta eficaz para la reconstrucción de cantidades de interés físico las cuales se ven afectadas por efecto de los detectores. De forma específica se trabajó sobre la reconstrucción del número de múltiples interacciones partónicas la cual es una cantidad "verdadera" a nivel generador, sin contraparte reconstruida.

En esta tesis se muestra el criterio usado para seleccionar el método de aprendizaje, así como las variables de entrenamiento. Posteriormente se muestra el proceso de optimización de los parámetros y valores de corte de cada método con el objetivo de incrementar su efectividad y optimizar el uso de recursos de cómputo. Finalmente se muestran las distribuciones de N_{MPI} generadas por los métodos de aprendizaje de máquina utilizados, así como las distribuciones de esferocidad y cocientes $\frac{p}{\pi}$.

En Pythia 8 para un corte mediante el método tradicional con ($< N_{MPI} > = 15{,}01$) y LD (discriminador lineal) con ($< N_{MPI} > = 15{,}11$) se logró un aumento en la eficiencia de señal de 0.182 (método tradicional) a 0.478 mediante el método LD basado en 4 variables en vez de 2. Esto representa un aumento en la eficiencia de 262.6 %, lo cual se traduce una disminución de 62 % en el número de eventos sin clasificar requeridos para obtener la misma cantidad de eventos seleccionados como de alto N_{MPI}.

Índice general

Índice

1. Introducción

Este proyecto requiere de la física de partículas (de donde surge el problema a resolver), la computación (la herramienta) y la probabilidad (fundamenta y permite evaluar los algoritmos). Debido a que involucra varias disciplinas, es necesario conocer las bases de cada una de ellas para poder entender su aporte en la solución planteada.

Desde hace ya varios años en la comunidad de altas energías y partículas fundamentales se viene haciendo uso de los generadores de eventos Monte Carlo (MC). Este tipo generadores permite la simulación de colisiones de partículas mediante el uso de los conocimientos previos que tenemos sobre la física de altas energías y distribuciones de probabilidad. Además de simular las colisiones, las técnicas MC también permiten simular el transporte de las partículas a través de los detectores, así como su reconstrucción. En adelante, al evento MC sin los efectos del detector se le llamará evento verdadero y a la simulación que incluye los efectos del detector se le llamará evento reconstruido.
Hasta el momento se habían considerado a las colisiones pp como referencia para las de iones pesados (p-A y A-A) pues se consideraba que no presentaban tiempos de termalización o volúmenes de interacción lo suficientemente grandes para permitir la formación del plasma de quark-gluones (QGP), sin embargo, datos recientes provenientes del LHC revelan similitudes en los datos pp que parecen sugerir la posible formación del QGP en pp[38]. En MC la inclusión de las múltiples interacciones partónicas MPI combinado con la reconexión por color han podido reproducir los patrones de flujo radiales encontrados en los datos, los cuales se han asociado con la hidrodinámica del denso QGP, por lo que es de interés aislar y analizar eventos en los que el efecto de las MPI y la reconexión por color sean prominentes (alto número de MPI). La solución planteada en este trabajo se basa en utilizar aprendizaje de máquina, el cual considera variables en el evento reconstruido para aislar eventos con $N_{[}MPI] > 15$. El aprendizaje de máquina permite calcular la probabilidad de que un evento en particular con valores reconstruidos $x_1, ..., x_n$ se encuentre en la región de interés. Para obtener la función que lleva desde las n variables a la probabilidad de encontrar al evento en la región deseada hay decenas de métodos propuestos, la mayoría de los cuales siguen el principio de aproximadores universales es decir una función de $\mathbb{R}^m \times \mathbb{R}^n$ a $\mathbb{R}$ la cual permite aproximar cualquier función de $\mathbb{R}^m$ a $\mathbb{R}$ con precisión arbitraria mediante n parámetros fijos con n suficientemente grande. Los valores fijos de las n variables se conocen como pesos de la función.

Los algoritmos o métodos para aprendizaje de máquina que estaremos utilizando entran en una clasificación conocida como aprendizaje asistido, lo cual indica que los pesos se calculan automáticamente en base a una serie de ejemplos correctamente clasificados con los cuales se entrena el algoritmo. La forma de ajustar los pesos suele estar basada en una función error entre la salida de los algoritmos y el resultado esperado según la muestra de entrenamiento. Cada uno de los algoritmos existentes tiene problemas donde es más efectivo, por ejemplo, los algorítmos SVM (máquina de soporte vectorial) lineal o LD (discriminador lineal) permiten separar fácilmente y con gasto mínimo de poder de cómputo distribuciones sencillas divididas por un hiper-plano (clasificación lineal), sin embargo, fallan para separar distribuciones con formas complicadas. Mientras tanto una red neuronal más compleja y la cual requiere una muestra más grande de entrenamiento, pero resuelve bien distribuciones no lineales, debido a esto es necesario elegir y optimizar los mejores métodos para nuestro problema en particular.

Para poder identificar cual es el mejor método para nuestro problema se requiere la ayuda de la probabilidad. En nuestro caso nos interesa aumentar tanto la pureza como la eficiencia sin aumentar una en prejuicio de la otra. Para poder identificar que algoritmos permiten esto, se utilizan las integrales ROC en el plano eficiencia-pureza (Para la definición de la curva ROC ver 3.1.1.3.2). Para los métodos seleccionados se optimizaron los parámetros de ajuste y el valor de los cortes en base a las curvas ROC, significancia y curvas de sobre entrenamiento (prueba de Kolmogórov-Smirnov 3.1.1.3.4).

1.1. El modelo estándar y la física de partículas

El modelo estándar de la física de partículas, si bien hay razones para creer que es una versión a bajas energía de una teoría más fundamental, ha demostrado mediante experimentos que es capaz de producir predicciones con sorprendente precisión siendo hasta el momento nuestro mejor modelo fundamental sobre la fenomenología en física de partículas fundamentales.[24] El modelo estándar se puede considerar como la tabla periódica de la física de partículas en el sentido que permite clasificar los constituyentes de la materia (Ver Figura 1), así como también realizar predicciones de sus interacciones y reglas de operación. En el modelo estándar las partículas se dividen en 3 grupos fundamentales (quarks, leptones y bosones de norma) de los cuales los quark y leptones presentan 3 generaciones cada una de masa superior a la anterior. Adicionalmente el modelo estándar permite la existencia de la antimateria por lo que cada partícula posee una compañera con las mismas propiedades pero con números cuánticos de signo opuesto a la cual se le llama antipartícula (en ocasiones la antipartícula de una partícula puede ser ella misma como es el caso del fotón) .

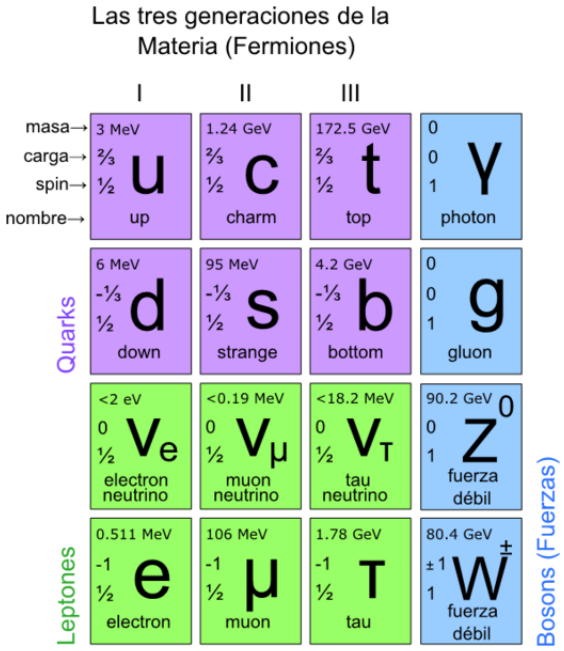


Figura 1: Las partículas integrantes del modelo estándar organizadas en quarks, leptones y bosones, los leptones y quarks adicionalmente se agrupan en 3 generaciones con dos partículas y sus anti partículas cada generación.[25]

En realidad, el modelo estándar es mucho más que una simple tabla de partículas pues incluye un modelo de interacciones entre partículas basado en la teoría cuántica de campos la cual fundamenta la interacción fuerte mediante la QCD, y las interacciones electromagnética y débil mediante la teoría electrodébil.

Desde el punto de vista de los grupos de Lie el modelo estándar está formado por

- El grupo $SU(3)$ de la interacción fuerte también conocido como $SU(3)_c$ pues representa las interacciones de color.
- El grupo $SU(2)_L \times U(1)_Y$ el cual explica las interacciones electro-débiles. Donde la L significa que el grupo $SU(2)$ solo actúa sobre los componentes izquierdos de los campos y la Y de $U(1)_Y$ significa que es de Yang-Mills y está relacionado con la hipercarga $Y = Q - T_3$ la cual tiene además la propiedad de conmutar con los generadores del grupo $SU(2)_L$

De forma que el grupo del modelo estándar es $G_{SM} = SU(3)_c \times SU(2)_L \times U(1)_Y$ y este engloba las fuerzas fuerte y electro-débil. [7]

De forma similar es posible escribir el modelo estándar en forma lagrangiana como campos de partículas interactuando entre sí, formulación que se puede ver en la siguiente ecuación[2]:

$$\mathcal{L} = -\frac{1}{4}F_{\mu\nu}F^{\mu\nu} + i\bar{\psi}\not{D}\psi + (i\bar{\psi}\not{D}\psi)^{\dagger} + \psi_i y_{ij}\psi_j\phi + (\psi_i y_{ij}\psi_j\phi)^{\dagger} + |D_\mu\phi|^2 - V(\phi) \tag{1}$$

1.1.1. Interacción electro-débil

El modelo electrodébil se encuentra basado en tres bosones W^i_μ los cuales obedecen al grupo $SU(2)$ y un bosón B_μ que obedece al grupo $U(1)$. Es importante no confundirlos con los bosones $W^{\pm}$, Z y γ del modelo estándar. La relación de los campos bosónicos W^i_μ y B_μ con el fotón y los bosones $W^{\pm}$, Z es la siguiente:

$$W^{\pm} \equiv \frac{W^1 \mp iW^2}{\sqrt{2}} \tag{2}$$

$$Z \equiv -BSen(\Theta_W) + W^3 Cos(\Theta_W) \tag{3}$$

$$\gamma \equiv BCos(\Theta_W) + W^3 Sen(\Theta_W) \tag{4}$$

Donde Θ_W es el ángulo débil o ángulo de Weinberg, su valor está relacionado con el cociente de masas entre el bosón Z y los bosones W $Cos(\Theta_W) = \frac{M_W}{M_Z}$ y con el valor de la carga del positrón $e = gSen(\Theta_W)$. También es importante recalcar que debido a que las interacciones débiles no conservan paridad, esté modelo distingue a las partículas izquierdas de las derechas.

1.1.2. Interacción fuerte

El modelo de interacción fuerte está basado en las interacciones de color y está formado por 3 cargas (R, G, B) y sus correspondientes anticargas. A bajas energías la interacción fuerte conduce al confinamiento, es decir la formación de hadrones neutros sin color, siendo el menor arreglo posible el formado por 2 quarks, razón por la cual los quarks no pueden observar solos en los experimentos. La partícula mediadora de la fuerza fuerte es el gluon el cual transporta carga de color y por lo tanto a diferencia del fotón si modifica la carga de las partículas afectadas por la fuerza, además de poder ver afectado por la misma fuerza que transporta. En el modelo de quarks se definen números cuánticos asociados a cada quark (Isospín, extrañeza, encanto, superioridad e inferioridad) de la siguiente forma[7]:

Propiedad \Quark	**d**	**u**	**s**	**c**	**b**	**t**
$I - Isoepín$	$\frac{1}{2}$	$\frac{1}{2}$	0	0	0	0
I_z-Proyección del isospín en z	$-\frac{1}{2}$	$\frac{1}{2}$	0	0	0	0
S-Extrañeza	0	0	-1	0	0	0
C-Encanto	0	0	0	+1	0	0
B-Inferioridad	0	0	0	0	-1	0
T-Superioridad	0	0	0	0	0	+1

De las cantidades mostradas en el Cuadro 1 es posible calcular la hipercarga $Y = \beta + S - \frac{C-B+T}{3}$ y la carga eléctrica $Q = I_z + \frac{\beta+S+C+B+T}{2}$ por medio de la fórmula generalizada de Gell-Mann-Nishijima, donde β es el número bariónico el cual vale $\frac{1}{3}$ para quarks y $-\frac{1}{3}$ para anti-quarks. La carga eléctrica es conservada de forma general mientras que la hipercarga es conservada únicamente en interacciones fuertes (las interacciones débiles pueden no conservar la hipercarga).

La teoría de campo que explica las interacciones fuertes es la cromodinámica cuántica (QCD) y corresponde al grupo $SU_c(3)$, cuyo lagrangiano es:

$$\mathcal{L}_F = \sum_a \bar{\psi_{q,a}}(i\gamma^\mu\partial_\mu\delta_{ab} - g_s\gamma^\mu t^C_{ab}\mathcal{A}^C_\mu - m_q\delta_{ab})\psi_{q,b} - \frac{1}{4}F^A_{\mu\nu}F^{A\mu\nu} \tag{5}$$

donde los índices repetidos se suman ,γ^μ son las matrices de Dirac y $\psi_{q,a}$ son espinores del campo de quarks con sabor q y masa m_q, el índice a representa el color el cual corre desde 1 hasta 3 (3 colores), y $F^A_{\mu\nu} = \partial_\mu \mathcal{A}^A_\nu - \partial_\nu \mathcal{A}^A_\mu - g_s f_{ABC} \mathcal{A}^B_\mu \mathcal{A}^C_\nu$ donde f_{ABC} son las constantes de estructura del grupo $SU(3)$. La cantidad g_s es la constante de acoplamiento de la cromodinámica cuántica (QCD).

A pesar de que el efecto principal de la fuerza nuclear fuerte es la creación de hadrones como los protones o los neutrones sus efectos también se manifiesta a escalas mayores, donde de forma similar a como los átomos eléctricamente neutros puede interactuar a "grandes distancias" mediante la fuerza de Van der Walls que es un remanente de la fuerza eléctrica de sus constituyentes, también la fuerza fuerte es responsable de mantener unidos a los nucleones en el interior del núcleo por medio de la fuerza nuclear fuerte, la cual se explica cómo "remanentes" de la fuerza fuerte que actúa sobre los quarks constituyentes. El mecanismo de interacción a "larga distancia (dentro del núcleo)" entre los quarks constituyentes es mediante la creación de π y ρ virtuales que transmiten color entre los nucleones.

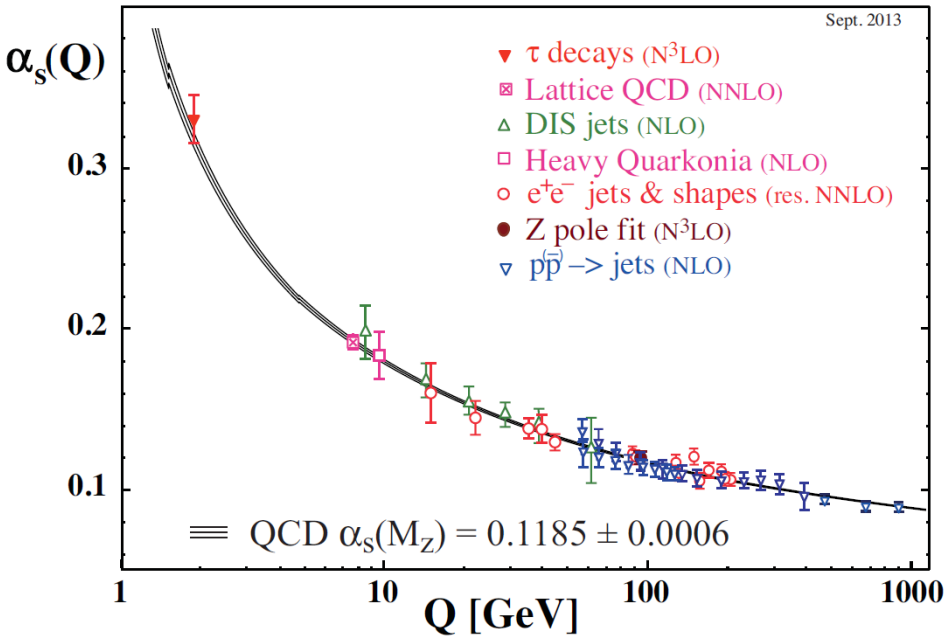


Figura 2: Gráfica de las mediciones de α_s la constante de acoplamiento fuerte en función de la escala energética Q. A baja energía Q el acoplamiento fuerte es grande y da lugar a la hadronización. Imagen tomada de Chinese Physics C[7]

La forma de tratar la QCD depende fuertemente de la constante de acoplamiento de QCD α_{QCD} la cual no es en realidad constante y depende del momento transferido (Ver Figura 2).

A distancias cortas o altas tasas de transferencia de momento (como en las colisiones de hadrones en el LHC) nos encontramos en la región conocida como libertad asintótica donde las interacciones entre color se desvanecen y por lo tanto los resultados obtenidos por teoría de perturbaciones se vuelven más exactos, requiriendo ordenes cada vez menores en los términos de interacción, debido a esto, la zona de libertad asintótica es la única región que puede modelarse en base a primeros principios.[7] A energías como las del LHC donde cada una de las partículas generadas puede alcanzar momentos del orden de cientos de MeVs, ocurre la transición de fase de la $\text{QCD}_{\text{perturbativa}}$(Ver Figura 3) permitiendo que los quarks actúen como partículas "libres".

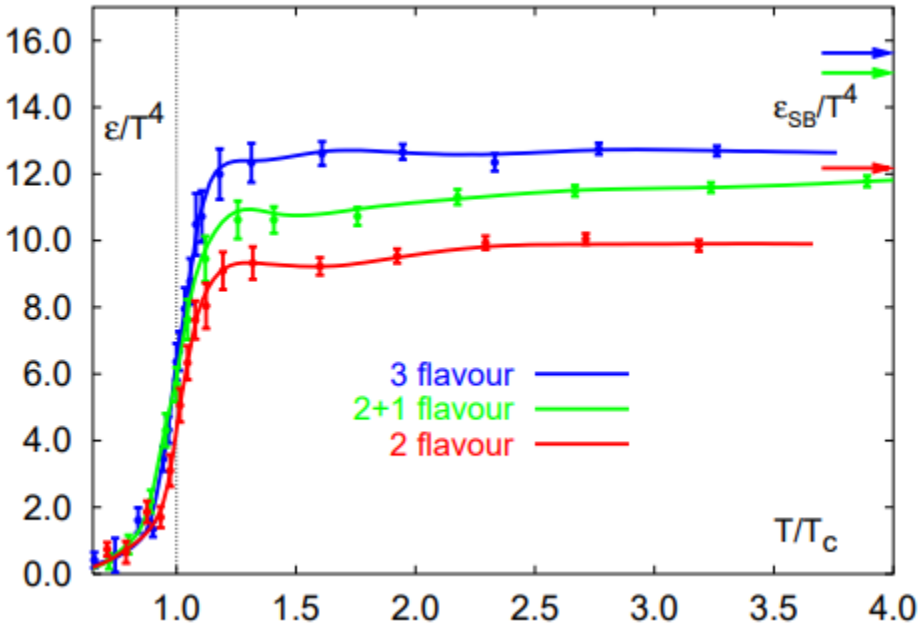


Figura 3: Transición de fase de $\text{QCD}_{\text{perturbativa}}$, la temperatura crítica T_C para QCD de 3 sabores por medio de *improved staggered fermions* se calcula en $T_C = (154 \pm 8)\,\text{MeV}$. Esta gráfica así como el valor de T_C provienen de [31]

Al disminuir la tasa de transferencia de momento entre las partículas los quarks "libres" comienzan a forman hadrones neutros(sin color), a esta región se le conoce como la región de confinamiento. Esta región no permite la utilización de $\text{QCD}_{\text{perturbativa}}$ y por lo tanto no está basada en primeros principios, sin embargo, sus modelos están inspirados en QCD. En los MC esta región se modela de forma fenomenológica mediante los modelos de hadronización siendo los más populares el modelo de cuerdas y el de clusters.

1.2. Generadores Monte Carlo

1.2.1. Concepto

Las técnicas Monte Carlo (MC) son a menudo la única forma práctica para evaluar integrales difíciles o para muestrear variables aleatorias gobernadas por complicadas funciones de probabilidad. En un comienzo todo MC comienza con la generación de números aleatorios en distribución plana en un intervalo usualmente $[0, 1)$, posteriormente esta distribución se transforma a la forma deseada según el proceso a modelar, lo cual en física de partículas corresponde en su mayoría a funciones de distribución partónicas (PDF) y funciones de fragmentación.

Los generadores de eventos MC como HERWIG, Pythia y SHERPA hacen uso de estas técnicas MC para simular colisiones de alta energía. Estos generadores están compuestos de múltiples partes cada una de las cuales simula un aspecto específico de la física en juego comenzando desde distancias cortas (menores al femtómetro) donde la QCD es débilmente interactuante y se puede resolver por medio de teoría de perturbaciones, hasta la hadronización y la formación del evento subyacente (UE)[7]. Entre los generadores de eventos MC estos se dividen en aquellos especializados en colisiones de partículas fundamentales y hadrones (incluyendo pp) como Pythia, Herwig y SHERPA y aquellos diseñados para iones pesados (A-A y p-A principalmente) como HIJING. En ALICE el generador MC por excelencia para pp es Pythia y para iones pesados se suele usar HIJING.

1.2.1.1. Física de corta distancia Los componentes de un generador MC enfocados en física de corta distancia se encargan de calcular el proceso principal, los decaimientos de partículas de corta vida y la generación de la radiación de QCD y QED. El proceso central en esta etapa del generador es el cálculo por medio de QCD perturbativa para momentos $Q > \Lambda_{\mathrm{QCD}} \equiv 1\,\mathrm{GeV}$ correspondientes a distancias menores a aproximadamente un femtómetro. Observables suaves y colinealmente seguras tales como los anchos totales de decaimiento o las secciones eficaces incluyentes se calculan hasta orden n mediante QCD perturbativa. Esta sección también incluye la generación de lluvias de partículas y algunos procesos y correcciones los cuales se mencionarán brevemente a continuación[7]:

- **Correcciones angulares**: En el proceso de división de gluones ($g- > q\overline{q}$,$g- > gg$) propaga correlaciones de espín y correlaciones acausales del tipo EPR. En Pythia, actualmente las correlaciones acausales se desprecian por considerarse pequeñas.
- **Radiación de estado inicial**: Esta corresponde a la radiación generada por las partículas cargadas antes del proceso de dispersión duro.
- **Emisiones suaves y coherencia QCD**: Sin este efecto la multiplicidad de partículas crecería demasiado rápido con la energía.
- **Quarks masivos**: La masa de los quarks actúa como un corte en las singularidades colineales. En quarks masivos se espera menos actividad colineal que en los más ligeros razón por la cual los quarks pesados llevan una mayor proporción del momento adquirido en el proceso duro.
- **Información de color**: Rastrea la información de color durante el desarrollo de la lluvia.
- **Correcciones electromagnéticas**: Permite generar el equivalente a las lluvias de partículas para fotones provenientes de partículas ligeras. Se encuentra implementado en Hergwig y Sherpa.
- **Física más allá del modelo estándar**: Es posible incluir procesos para física más allá del modelo estándar en los generadores.
- **Cadenas de decaimiento y anchos de partícula**: En la mayoría de los proceso más allá del modelo estándar y en algunos del modelo estándar un aspecto importante es la simulación de partículas con decaimiento de vida corta como quarks top y el bosón de Higgs.
- **Emparejamiento con elementos de matrices**: Pythia y HERWIG han incluido desde hace tiempo las matrices de corrección de elementos(*MEC-matrix element corrections*), las MEC corrigen la emisión del jet más duro a ángulos grandes. En la última década se ha mejorado la descripción de las lluvias de partículas en colisiones duras en dos direcciones: La primera llamada elementos de matriz y emparejamiento de lluvia de partones conocida como $(ME + PS)$ y la segunda es el emparejamiento de cálculos próximos al orden líder (NLO) y lluvia de partones.

1.2.1.2. Modelos de hadronización Los modelos de hadronización son aquellos que a partir de la lluvia de partones con color permiten generar hadrones primarios neutros en color los cuales pueden a su vez decaer y generar hadrones secundarios. Existen dos modelos principales para generar los hadrones el modelo de cuerdas y el modelo de clúster.

El modelo de cuerdas puede ser explicado si consideramos el caso de un par $q\bar{q}$, al separarse los quarks el potencial (la energía en la cuerda) $V = kr$ crece y la creación de pares quark- anti-quarks no perturbativa puede romper la cuerda mediante el proceso $(q\bar{q})- > (q\bar{q'}) + (q'\bar{q})$. El proceso de rompimiento de las cuerdas esta casualmente desconectado por lo que no es necesario considerar un orden temporal específico. El modelo de hadronización por cuerdas más usado actualmente es el modelo de Lund el cual se encuentra implementado en Pythia.[7]
El modelo de clúster se basa en el preconfinamiento, es decir que la evolución de la estructura de color en una lluvia en QCD perturbativa a cualquier escala Q_0 es tal que los subsistemas de color neutro ocurren con una

distribución de masa invariante universal que solo depende de la escala Q_0 y de Λ_{QCD} y no en la escala de comienzo $Q >> Q_0 >> \Lambda_{\mathrm{QCD}}$[7]. En el modelo de clústeres si un clúster neutro tiene una masa invariante mayor a cierto valor el clúster se rompe en dos a lo largo de un eje de forma similar a como se rompen las cuerdas en el modelo de cuerdas. Este proceso se continúa hasta que los clústeres generados tengan una masa invariante menor a el valor de corte. Los modelos Sherpa y Herwing utilizan este modelo de hadronización.

Finalmente, los modelos de hadronización consideran que algunos de los hadrones generados (ya sea por cuerdas o clústeres) son inestables y por lo tanto decaerán, los decaimientos de estos hadrones se modelan hasta obtener partículas con un tiempo de vida relevante para la medición ($\tau > \frac{10mm}{c}$).[7]

1.2.1.3. Modelos de interacciones suaves hadrón-hadrón Esta etapa de simulación incluye difracción, evento subyacente (UE), MPI, Bose-Einstein y efectos de reconexión por color.[7]

El modelado de los procesos difractivos en hadrón-hadrón (hh) se divide en: dispersión elástica, disociación difractiva simple, disociación difractiva doble y dispersión no difractiva inelástica.
Las correlaciones de Bose-Einstein en e^-e^+ ha representado una fuente de error para la medición de alta precisión de la masa de W en el LEP sin embargo en hh se usan las correlaciones Bose-Einstein para estudiar la estructura espacio temporal de la materia hadronizante mediante la técnica llamada femtoscopía.[7]

El evento subyacente (UE) denota cualquier actividad adicional más allá del el proceso básico y sus asociadas radiaciones de estado inicial y final. La contribución dominante se cree que proviene de intercambios de color adicionales entre los haces de partículas que pueden ser representados como múltiples interacciones partónicas (MPI) o como los así llamados pomerones de corte. El principal efecto del UE es la aparición del pedestal de jet el cual consiste un aumento en la producción de partículas en la región transversa es decir fuera de los jets principales.

1.2.2. AliRoot

La simulación y reconstrucción de eventos son desarrolladas completamente dentro del marco de trabajo AliRoot desarrollado como parte del proyecto ALICE offline. Los eventos aquí provienen de los generadores Pythia 8 y Pythia 6 mismos que pasan la información sobre las partículas en estado final o no decaídas de larga vida al código de transporte (GEANT3) el cual simula su trayectoria, sus decaimientos, así como también sus interacciones con los materiales del detector y genera datos de conteo representando la respuesta del detector. Los datos de conteo se usan para generar un proceso de digitalización de datos crudos simulados. Con el objetivo de poder mezclar la señal y el evento subyacente, se producen "dígitos sumables" como un paso intermedio. Estos son esencialmente dígitos antes de agregar ruido y antes de la supresión de ceros. De esta forma cada subconjunto de 5 eventos de jet es mezclado con el mismo evento de fondo resultando en una estadística final la cual es 5 veces más grande que el número de eventos de fondo generados. Posteriormente, los eventos mezclados son digitalizados y pasados a el algoritmo de reconstrucción de AliRoot para producir el compendio de datos de eventos ESD por sus siglas en ingles el cual es usado para el análisis.[12]

1.2.3. Pythia

El programa Pythia es una herramienta estándar para la generación de eventos en colisiones de alta energía entre partículas elementales. Actualmente la comunidad de usuarios más grande del programas se puede encontrar entre los experimentalistas del LHC, pero el programa se puede usar para una multitud de otros estudios fenomenológicos o experimentales.[22] Pythia 8 es la primera compilación completa en ser liberada totalmente en C++, Pythia 6 aún contenía rutinas de FORTRAN a pesar de ser mayoritariamente C++. Pythia ha sido desarrollado con la intención de poder ser utilizado en solitario, sin embargo, también permite su interacción con otros programas e interfaces como LHEF, LHA y ROOT los cuales le permiten ampliar sus funciones.

Actualmente el programa solo trabaja con colisiones ya sea hadrón-hadrón o electrón-electrón incluyendo entre los hadrones al $\bar{p}, \bar{n}, \Pi$ y como caso especial el pomerón. No están permitidas las colisiones hadrón-leptón, o para haces de fotones entrantes, tampoco se permiten las colisiones protón -núcleo o núcleo-núcleo[22]. En Pythia las partículas resultantes son producidas en vacío y la simulación de la interacción de estos productos con los materiales del detector no está incluida.

1.2.3.1. Interacciones partónicas múltiples en Pythia 6 En Pythia 6 existen tres modelos de MPI conocidos como el modelo antiguo, el modelo nuevo y el modelo intermedio, de los cuales utilizaremos el modelo nuevo. La principal diferencia entre los modelos es que en el modelo antiguo la radiación inicial (ISR) y final (FSR) de fondo asociada con cada MPI solo se consideraba para la dispersión más dura, sin embargo, en los modelos intermedio y nuevo también se consideran las siguientes dispersiones más suaves.
El mayor cambio del modelo intermedio al modelo nuevo es que en el modelo nuevo las lluvias son ordenadas en función de su p_{T} de forma que el p_{T} se convierte en la escala de evolución común tanto para las MPI como para la ISR y FSR, lo cual permite a la ISR y a las MPI competir en el mismo marco de p_{T} por el momento de los haces.

Entre las características del nuevo modelo se encuentra que las lluvias de MPI llevan a mayor grado de decorrelación y desbalance de p_{T} entre los mini jets producidos por el evento subyacente, en contraste al modelo antiguo donde eran casi exactamente balanceados (pares de mini jets generados en direcciones opuestas (back-to-back)). El nuevo modelo también exhibe menor cantidad de correlaciones de largo alcance, ya que en modelos antiguos de MPI cuando ocurría una fluctuación en un extremo del detector era probable encontrar una similar en el otro extremo.[21]

El ajuste (tune) utilizado para realizar las simulaciones de Pythia 6 fue Perugia 2011 el cual fue preparado en primavera 2011 basándose en el tune anterior Perugia 2010 y correcciones basadas en resultados del LHC para *minimum bias* y evento subyacente a 900 GeV y 7 TeV. Perugia 2011 y los ajustes derivados de este (Perugia 11 M, Perugia 11 C, Perugia 11mpiHi, etc.) pertenecen a la cuarta generación de ajustes en el nuevo modelo de MPI basado en ordenamiento en p_{T}, los parámetros de MPI para Perugia 2011 fueron ajustados en base a los datos obtenidos en el LHC para 7 TeV.[27]

1.2.3.2. Interacciones partónicas múltiples en Pythia 8 El modelo de MPI incluido en Pythia 8 cuenta al igual que versiones anteriores con ordenamiento en p_{T} , secciones transversas basadas en QCD perturbativa amortiguada en el límite $p_T \to 0$ y un formalismo de parámetro de impacto variable. En Pythia 8 estas características ha sido extendidas para abarcar un amplio rango de posibles procesos del UE, incluyendo todos los procesos $2 \to 2$ de QCD[22].

Este modelo fue desarrollado en el marco de las lluvias partónicas ordenadas en momento transverso (p_{T}) donde la escala de evolución basada en p_{T} es común para la ISR, FSR y MPI. Esta evolución común es de suma importancia, pues la ISR y las MPI compiten entre sí por el momento de los haces. En resumen podríamos decir que es una versión mejorada del modelo nuevo presentado en Pythia 6, sin embargo hay que recordar que hubo un cambio importante en el resto del generador entre Pythia 6 y Pythia 8 y fue necesario volver a hacer los ajustes (*tunes*) por completo.[10]

El ajuste utilizado fue Monash 2013 el cual se construyó para dar una descripción razonable tanto de la física suave inclusiva ("*minimum-bias*") como del tipo de observables del UE. En Monash 2013 el ajuste para MPI genera una distribución más ancha en el N_{MPI} comparado a otros ajustes de Pythia 8 (Ver Figura 4), la razón de esto es una combinación entre una distribución de materia ligeramente más localizada combinada con un valor p_{T0} de referencia relativamente bajo.[26]

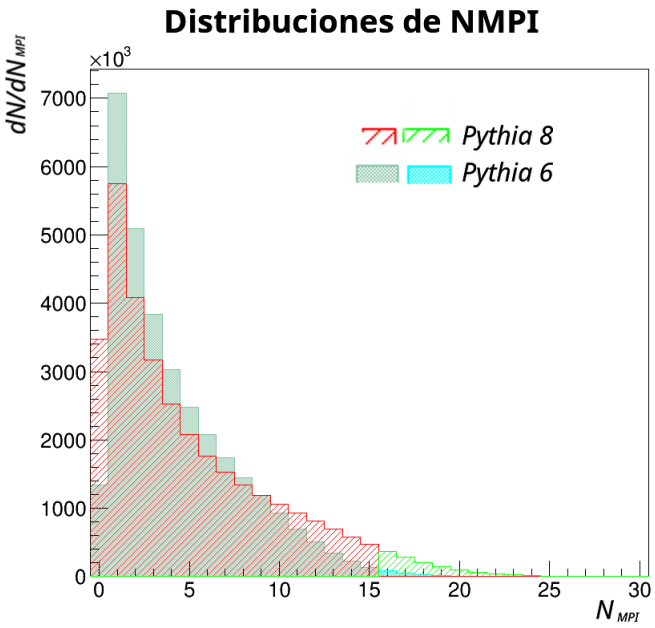


Figura 4: Las distribuciones de N_{MPI} difieren substancialmente entre los generadores Pythia 6 y Pythia 8, siendo la segunda más ancha. Esto toma vital importancia si tratamos de separar alto N_{MPI} pues Pythia 6 genera pocos eventos con $N_{MPI} > 15$ y prácticamente ninguno con $N_{MPI} > 20$.

La distribución más amplia de MPI en Monash se traduce en un espectro de multiplicidad para partículas cargadas más amplio, a pesar de que el efecto es modulado por el modelo de reconexión por color.[26]

1.2.4. GEANT3

GEANT3 es un sistema de descripción de detectores y herramientas de simulación. La versión 3 de GEANT representa un cambio substancial respecto a la versión previa, en esta nueva versión la descripción de la configuración geométrica y de los detectores sensibles es nueva, la estrategia de rastreo ha sido replanteada. También la mayoría de las rutinas de física ha sido mejoradas, y para los procesos hadrónicos, se han agregado la interfaz con el programa GEISHA.[28]

Las principales aplicaciones del programa son realizar el seguimiento de partículas a través de una configuración experimental para estudios de aceptación o simulación de respuesta del detector y la representación gráfica del arreglo y las trayectorias de dichas partículas.

Entre las acciones permitidas por el programa se encuentra[28]:

- La definición de volúmenes y materiales de interacción.
- La generación de eventos desde generadores MC.
- Controlar el transporte de partículas a través de varias regiones del arreglo, tomando en cuenta las fronteras geométricas del volumen y todos los efectos físicos debido a la naturaleza propia de las partículas como sus interacciones con la materia y con el campo magnético.
- Registrar los elementos de la trayectoria de las partículas y sus interacciones con el detector.
- Visualizar ya sea interactivamente o remotamente el detector y la trayectoria de las partículas.

2. Experimento ALICE

El experimento ALICE se encuentra ubicado dentro del marco del LHC y su objetivo es explorar la región de bajo momento transverso (p_{T}) en colisiones de alta multiplicidad principalmente en iones pesados $A-A$, sin embargo también ha realizado experimentos en pp y p-A.

ALICE es un experimento de iones pesados de propósito general diseñado para estudiar la física de materia fuertemente interactuante y el plasma de quark-gluones en colisiones núcleo-núcleo en el LHC. El detector ALICE está diseñado para trabajar con las altas multiplicidades esperadas en colisiones Pb-Pb y ha estado en operación desde la puesta en marcha del LHC. Adicionalmente a los sistemas pesados, la colaboración ALICE también estudia colisiones de iones de menor masa, y protones (tanto pp como p-A), las cuales tienen como propósito principal servir como referencia para colisiones núcleo-núcleo. Adicionalmente la información de pp recolectada también permite una cierta cantidad de estudios de física completamente de pp.[12]

El detector ALICE realiza mediciones evento por evento de hadrones electrones, fotones y muones (Ver Figura 5). La parte central se encuentra embebida en el gran magneto solenoidal $L3$ y cubre los ángulos polares de 45° a 135° y está integrada por: el sistema de seguimiento interior (ITS) formado por detectores de silicio de alta resolución; la cámara cilíndrica de proyección temporal(TPC); el calorímetro electromagnético (PHOS) y tres arreglos de identificación de partículas (el detector de tiempo de vuelo (TOF), el anillo de captura de imagen Cherenkov de brazo simple (HMPID) y el detector de radiación de transición (TRD)).
El brazo frontal de muones (cubre los ángulos polares $180° - \Theta = 2° - 9°$) y consiste de arreglos complejos de absorbedores, un gran dipolo magnético, catorce planos de seguimiento y cámaras de disparo. Varios detectores pequeños (ZDC,PMD,FMD,$T0$,$V0$) para la caracterización global del evento y disparo están localizados en los ángulos frontales. Un arreglo de centelladores ($ACORDE$) en la parte superior de imán $L3$ también es usado como disparador por rayos cósmicos.

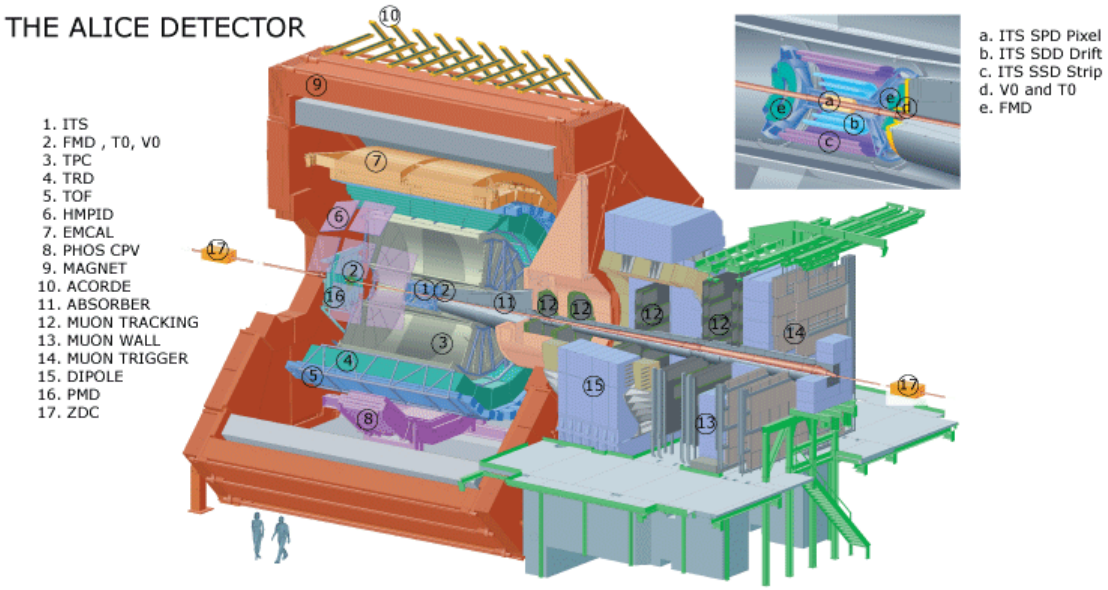


Figura 5: Diagrama mostrando el arreglo de detectores en ALICE. Imagen tomada de [38]

Como parte de los trabajos de análisis de los datos obtenidos por la colaboración se desea estudiar las colisiones pp con alto N_{MPI} con el fin de entender el UE y la nueva física en la región de bajo p_{T}. Como parte de la exploración de la física del UE se explora las MPI, la reconexión por color (CR) y su relación con la posible formación del QGP (plasma de quarks y gluones) en las colisiones pp. Debido a la importancia que ha mostrado el modelo de MPI en Pythia para la explicación del flujo radial observado en datos, se desea aislar los eventos donde sus efectos

sean significativos (alto N_{MPI}).

2.1. Detectores y reconstrucción de eventos en ALICE

En ALICE los detectores y procesos de reconstrucción se dividen en [12]:

- Reconstrucción del vértice y trayectoria en la parte central de ALICE.
- Reconstrucción de trayectorias en el espectrómetro de muones
- Seguimiento en línea y filtrado de eventos.
- Identificación de partículas para partículas cargadas y neutrales.

2.1.1. Reconstrucción de trayectorias con los detectores centrales y reconstrucción del vértice primario

Los detectores centrales de ALICE incluyen:

- El sistema de seguimiento interno (ITS)
- La cámara de proyección temporal (TPC)
- El detector de radiación de transición (TRD)
- El detector de tiempo de vuelo (TOF)
- El detector de identificación de partículas de alto momento (HMPID)
- Espectrómetro de fotones (PHOS).

El proceso de reconstrucción se lleva a cabo en las coordenadas del sistema coordenado global de ALICE con centro en la intersección del eje z (del haz) con el plano de la membrana central de la TPC.

Los elementos en el proceso de reconstrucción se agrupan de la siguiente forma:

- Dígitos: Señal digitalizada correspondiente a la respuesta de un detector a un cierto tiempo.
- Clúster: Grupo de dígitos adyacentes (en espacio y/o tiempo) los cuales se presume fueron generados por la misma partícula al cruzar el detector.
- Punto espacial reconstruido: Estimación del punto donde la partícula cruzo el detector. Usualmente se calcula como el centro de gravedad del clúster.
- Trayectoria reconstruida: Consta de 5 parámetros y sus matrices de covarianza estimadas.

El vértice primario del evento se reconstruye los detectores de silicio más centrales de la ITS, y las trayectorias de las partículas se reconstruyen entre los detectores TPC, ITS y TRD por medio de la aplicación de filtros de Kalman. ALICE también es capaz de identificar a partir de las trayectorias reconstruidas los vértices secundarios correspondientes a grupos de trayectorias que coinciden en un punto presumiblemente decaimientos de partículas.

2.1.2. Espectrómetro de Muones

El espectrómetro de muones está basado en cámaras de "almohadillas" catódicas (*CPC-cathode pad chambers*) con lecturas en ambos planos catódicos. Hay dos métodos que se pueden utilizar:

- El método tradicional: Consiste en reconstruir los clústeres y a partir de ellos las trayectorias de las partículas. Este método tiene la desventaja que su desempeño tiende a disminuir con altas ocupaciones en el detector (alta multiplicidad).
- El método EM: Este método está basado en la técnica de deconvolución llamada *Maximum Likelihood-Expectation Maximization* ($MLEM$ o EM). La cual involucra un proceso iterativo, que en comparación con el método tradicional resulta más lento, sin embargo, tiene la ventaja de mejorar la segmentación del detector.

El método tradicional de forma similar a otros detectores en ALICE también está basado en los filtros de Kalman para la reconstrucción de las trayectorias.

2.2. Identificación de partículas cargadas

La identificación de partículas cargadas en ALICE cubre partículas con momentos desde cerca de 0,1 GeV/c hasta algunos GeV/c, combinando distintos sistemas que son eficientes en regiones limitadas pero complementarios del espectro de momentos.

Los detectores que participan en la identificación de partículas cargadas en ALICE son:

- ITS: Mediante la medición de la perdida de energía en los detectores de silicio se puede identificar las partículas en el régimen no relativista.
- TPC: En la TPC se puede combinar la información del momento de la partícula con la medición de $< \frac{dE}{dx} >$ para obtener la identidad de la partícula.
- TRD: Ayuda a la TPC con la discriminación electrón/pion para momentos mayores a 1 GeV/c
- TOF: Mediante la combinación de la longitud de trayectoria registrada, el tiempo de vuelo y el momento de la partícula es posible calcular su masa, lo cual puede ayudar a identificar la identidad de la partícula.
- HMPID: Se enfoca a la identificación de hadrones (específicamente π, K y p) en la región de alto momento transverso $p_T > 1\,\mathrm{GeV}$.

2.3. Identificación de partículas neutras

La reconstrucción de partículas neutras en ALICE se lleva a cabo en el espectrómetro de fotones (PHOS) el cual tiene como objetivo identificar los fotones reales y medir con alta resolución su cuadrímomento.

Entre los obstáculos que enfrente PHOS se encuentra identificar de forma inequívoca los fotones entre la gran cantidad de partículas que llegan al detector, así como separar las señales producidas por fotones directos y aquellos provenientes de decaimientos.

PHOS también permite realizar mediciones de mesones neutrales principalmente π^0 y η.

2.4. Efectos de detector

Consideremos la analogía entre los detectores de partículas (principalmente cargadas) y una cámara fotográfica (detector de fotones). Una cámara ideal permitiría registrar toda la información disponible sobre la imagen sin

afectarla, sin embargo, ningún sensor es perfecto, por ejemplo, las cámaras fotográficas tienen ruido electrónico, difracción, aberraciones cromáticas, etc. Los cuales afectan la calidad de la imagen, la cual se parece a la realidad a fotografiar, sin embargo, no es una representación exacta de ella. Es importante indicar que al igual que en una fotografía, en los detectores de partículas, también pueden ocurrir defectos en la reconstrucción los cuales pueden afectar la "imagen" capturada. Uno muy común es la pérdida de partículas en la imagen capturada por los detectores, sin embargo, no es el único, por ejemplo, en ocasiones una partícula puede ser confundida por otra, o pueden observarse dos partículas viajando juntas como si fueran una sola. Llamamos cantidades reales a las generadas por el MC basadas en la teoría subyacente la cual trata de reproducir el fenómeno tal y como creemos que sucedería en la realidad. Posteriormente a la creación de todo el evento (sin considerar el detector) existe otra capa la cual representa nuestro detector y su interacción con nuestro evento. Esta capa tiene como propósito simular la respuesta del detector y por lo tanto predecir los valores que se espera observar al realizar el experimento en el laboratorio. A los valores obtenidos tras la simulación de los efectos de detector se les llama cantidades reconstruidas. Pasar de cantidades reales a reconstruidas es sencillo pues solo implica simular el detector a usar mediante un programa como GEANT. Sin embargo, el proceso contrario es complicado ya que siempre existe pérdida de información en los detectores. Actualmente es posible realizar un desdoblamiento mediante las matrices de detector, las cuales permiten pasar de cantidades reconstruidas a cantidades reales mediante distribuciones de probabilidad donde se indica la probabilidad de que un valor reconstruido corresponda a un valor real dado. Sin embargo, estas matrices no se pueden ocupar para valores reales sin contraparte reconstruida como es el caso de N_{MPI} y es por lo tanto de interés explorar las posibilidades que provee el aprendizaje de máquina para reconstruir su valor (regresión) y/o separar regiones de interés de esta variable.

3. Técnicas de análisis multivariable en física de altas energías.

3.1. Aprendizaje de máquina

El aprendizaje de máquina también conocido como aprendizaje automático o *machine learning* permite a las computadoras determinar por ellas mismas la mejor forma de tratar y resolver un problema ya sea de clasificación o de regresión. El aprendizaje de máquina representa un cambio de paradigma en la forma de analizar la información, ya que hasta antes de su existencia el programador necesitaba poseer un conocimiento profundo del problema y saber plantear una solución a este; sin embargo, con la llegada del aprendizaje de máquina la computadora vino a ser la encargada de estudiar el problema, determinar el modelo y proponer su solución.

Los algoritmos de aprendizaje de máquina han sido diseñados para resolver una gran variedad de problemas distintos y pueden tener diferentes enfoques como, por ejemplo: ser de fácil interpretación, ser ligeros o lograr el mejor desempeño posible (máxima pureza, eficiencia, etc.). Una vista a la taxonomía de los algoritmos de aprendizaje de máquina se puede encontrar en la sección 8.1 Anexo: "Tipos de algoritmos de aprendizaje de máquina".

La gran mayoría los algoritmos de aprendizaje de máquina están basados en el concepto de aproximadores universales, por lo que es de importancia entender que es un aproximador universal antes de poder discutir cada uno de los algoritmos. Matemáticamente un aproximado universal es una función de $\mathbb{R}^m \times \mathbb{R}^n$ a $\mathbb{R}$ la cual para n lo suficientemente grande permite reproducir con precisión arbitraria a una función de $\mathbb{R}^m$ a $\mathbb{R}$. Los parámetros en $\mathbb{R}^n$ se conocen en el mundo de aprendizaje de máquina como pesos, sin embargo, en otras áreas pueden tener otros nombres, por ejemplo, las series de Taylor son aproximadores universales ya que pueden reproducir cualquier función de $\mathbb{R}^m$ a $\mathbb{R}$ sin embargo sus parámetros de ajuste se les suele conocer como coeficientes en vez de pesos.

Tomemos por ejemplo la red neuronal más simple que existe compuesta por una única neurona con función de respuesta identidad. Esta red neuronal es equivalente a un modelo lineal, lo cual en el caso de regresión se puede escribir como $Y_\alpha = A^\beta_\alpha X_\beta + b_\alpha$ con A una matriz de $n * m$, Y un vector de salida de longitud m y X un vector de entrada de longitud n. Para problemas no lineales no basta con agregar más neuronas pues aplicar consecutivamente el operador $A^\beta_\alpha(\cdot)_\beta + b_\alpha$ (neuronas en serie) ó sumar con pesos más de una neurona $Y_\alpha = B^\beta_\alpha Y_\beta$ sigue resultando en relaciones lineales entre las variables, en las redes neuronales la no linealidad se integra me-

diante funciones de respuesta que actúan en la salida de cada neurona y permiten que al apilar neuronas en serie o paralelo se puedan generar modelos de mayor orden.

Además de un aproximado universal se requiere de una función que permita determinar el error cometido por el aproximador y de un algoritmo que permita determinar cómo mover los pesos para minimizar el error a esta función se le conoce como optimizador[30]. La función de error tiene como objetivo calificar al algoritmo con base a sus predicciones y definir un objetivo de búsqueda para el optimizador[30]. En el caso de una regresión donde se busca calcular un valor numérico como salida, la función de error puede ser por ejemplo la diferencia entre el valor esperado como respuesta y la respuesta dada por el método. Algunas de las funciones de error más populares son para trabajos de clasificación la entropía cruzada, y para regresión el error mínimo absoluto (MAE) o el error cuadrático medio (ECM o también conocido como mínimos cuadrados).[32]

Los optimizadores tienen el objetivo de minimizar la función de error de forma global es decir deben encontrar los parámetros correspondientes a los pesos tal que las predicciones del modelo ajusten de la mejor forma posible con los ejemplos de entrenamiento. El primer método que se ocupó históricamente para este objetivo fue el gradiente descendiente el cual se basa en calcular en el espacio de pesos la dirección en la que disminuye más rápido el error y ajustar los pesos un paso en esa dirección hasta llegar al mínimo error posible.

Además del gradiente descendiente, el cual presenta una tasa de convergencia limitada, se han desarrollado otros algoritmos como Adam, Adaboost, Adadelta, adagrad y momentum (Ver Figura 6) los cuales presentan tasas de convergencia superiores y solucionan algunos otros problemas que presentaba el gradiente descendiente como por ejemplo en casos patológicos como la silla de montar donde el gradiente descendiente podía quedarse montando la silla en lugar de resbalar por uno de sus costados.

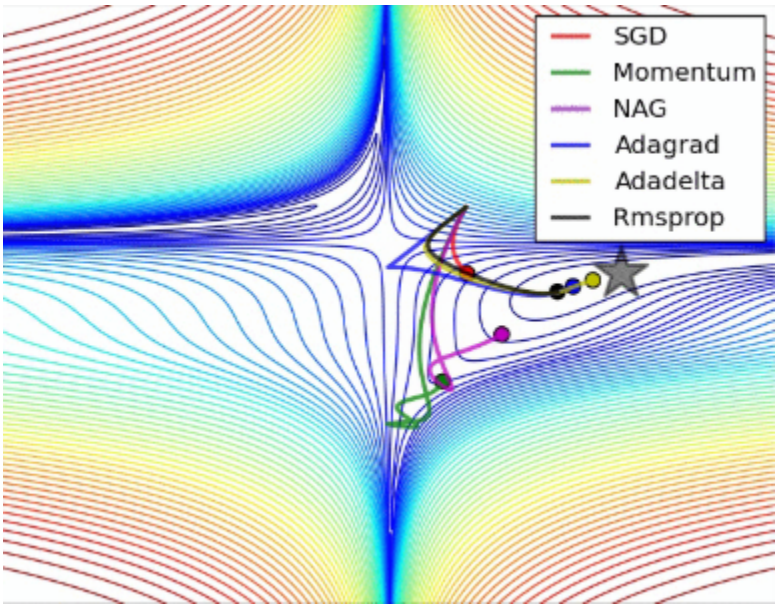


Figura 6: Trayectoria de varios optimizadores en el espacio $Pesos(x, y)$ *vs.* $Error(z)$. Los optimizadores están iterando para disminuir el error, el cual tiene su mínimo en el punto indicado por la estrella [1]

En breve y de forma general el proceso interno con él trabaja un algoritmo de aprendizaje de máquina suele ser como sigue:

- Se cargan y revuelven aleatoriamente los ejemplos de entrenamiento.
- Se inician los pesos con valores aleatorios o si se tiene alguna idea del valor esperado también se puede iniciar a partir de este valor.

- Se aplica de forma iterativa el algoritmo optimizador para ajustar los pesos en forma que se reduzca la función de error.
- Al llegar a cierto número de iteraciones o al ser detenida por un monitor de sobre entrenamiento externo, se paran las iteraciones, guardan los pesos y exporta el modelo.
- Se prueba el modelo generado mediante una colección de eventos independiente y se determina su eficacia en predecir la cantidad esperada.

3.1.1. Etapas

Se puede dividir el proceso de trabajo en aprendizaje de máquina en cuatro etapas, las cuales son:

3.1.1.1. Preparación de los árboles de eventos Esta es posiblemente la etapa más importante a la hora de trabajar con cualquier información, ya que el resultado final dependerá de la calidad de los eventos generados y de que estos estén correctamente optimizados para el aprendizaje de máquina.

El primer paso en esta etapa consiste en medir o generar los eventos a analizar. Es necesario garantizar que los eventos tengan un formato estandarizado y en el caso de Root que estos están organizados en árboles de eventos (entrenamiento, evaluación y prueba), los cuales a su vez contengan ramas con las mismas variables de entrenamiento y que cada una de las variables sea legible para los algoritmos (Los algoritmos solo aceptan variables numéricas en Root).

El segundo paso consiste en filtrar la información mediante criterios de calidad, lo cual en física de partículas consiste en aplicar cortes cinemáticos a los datos, multiplicidad mínima requerida, clústeres por traza, etc.

El tercer paso consiste en normalizar las variables de entrada, este paso es necesario para garantizar que los pesos asignados a cada variable sean proporcionales a su importancia y no a su escala. Tomemos por ejemplo la multiplicidad contra la esferocidad, la primera tiene un rango que llega hasta un poco más de 60, mientras que la segunda solo varía entre 0 y 1, si aplicáramos directamente los algoritmos a estos eventos, la multiplicidad dominaría inmediatamente sobre la esferocidad únicamente por cuestión de escala no porque realmente fuera más importante. La forma correcta de normalizar una variable es restarle su promedio y posteriormente dividir el valor resultante entre su varianza promedio (desviación estándar), de forma que este centrada en 0 y en promedio varíe entre -1 y 1 [33].

En ocasiones está etapa también puede incluir decorrelacionar las variables de entrada[38], sin embargo en TMVA la decorrelación se realiza como elemento interno de cada algoritmo durante el entrenamiento, evaluación y aplicación[17, 18].

3.1.1.2. Entrenamiento En esta etapa, se muestra a los algoritmos ejemplos resueltos es decir parejas de variables de entrada X y salida esperada Y. Los detalles internos del funcionamiento de cada método en esta etapa son altamente dependientes del método elegido, pero en lo general están relacionados a optimizar los pesos para minimizar el error cometido.[4]

Esta etapa se puede comparar a tomar apuntes en las clases y lecturas al estudiar en la universidad. En la mayoría de los algoritmos esta etapa también viene acompañada por ejercicios de autoevaluación equivalente a resolver problemas del libro y luego consultar el solucionario para comprobar si lo que creímos haber aprendido está bien. Usualmente este proceso se repite varias veces hasta lograr resolver los ejercicios sin ver las respuestas.

Las principales razones para terminar esta etapa son[33, 30] :

- Alcanzar la precisión deseada antes del tiempo determinado.
- Agotamiento del tiempo determinado o número de ciclos de entrenamiento.
- Muestra de signos de sobre entrenamiento.

La primera de las razones se puede considerar como una ejecución exitosa, mientras que la segunda dependiendo de la precisión alcanzada se puede considerar exitosa o fallida.

Para entender que es el sobre entrenamiento, como afecta al aprendizaje de máquina y por qué es importante considerarlo, es necesario acordarnos de las series de Taylor y los grados de libertad[30]. Tómese por ejemplo los datos de la Figura 7:

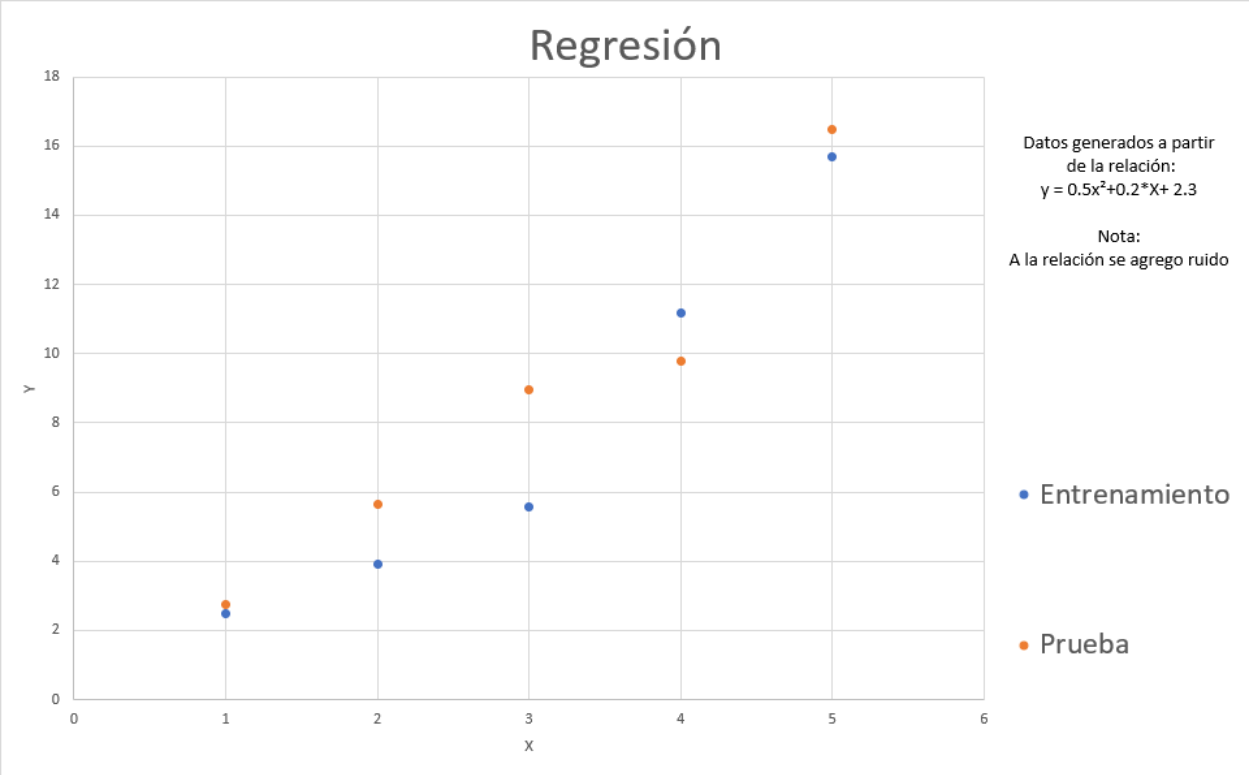


Figura 7: Distribución de puntos de entrenamiento y prueba para un problema de regresión de segundo grado

La distribución proviene de una parábola a la cual se le agregó un poco de ruido, de forma que el ajuste no fuese exacto. Es posible ajustar una parábola y reproducir de la mejor forma posible la distribución (Ver Figura 8). Sin embargo, el error no será cero debido al ruido incluido.

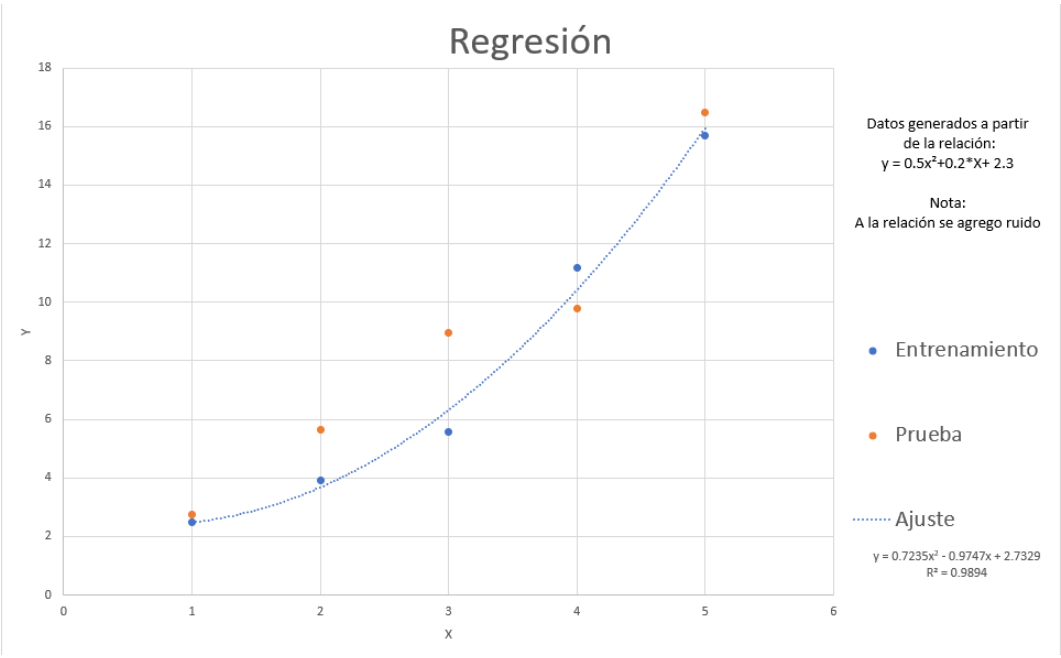


Figura 8: Ajuste de segundo grado por medio de mínimos cuadrados para los puntos del problema de regresión

La muestra de validación siempre es generada con el mismo algoritmo que la de entrenamiento, sin embargo, la única diferencia es que esta nunca se usa para entrenar al método.

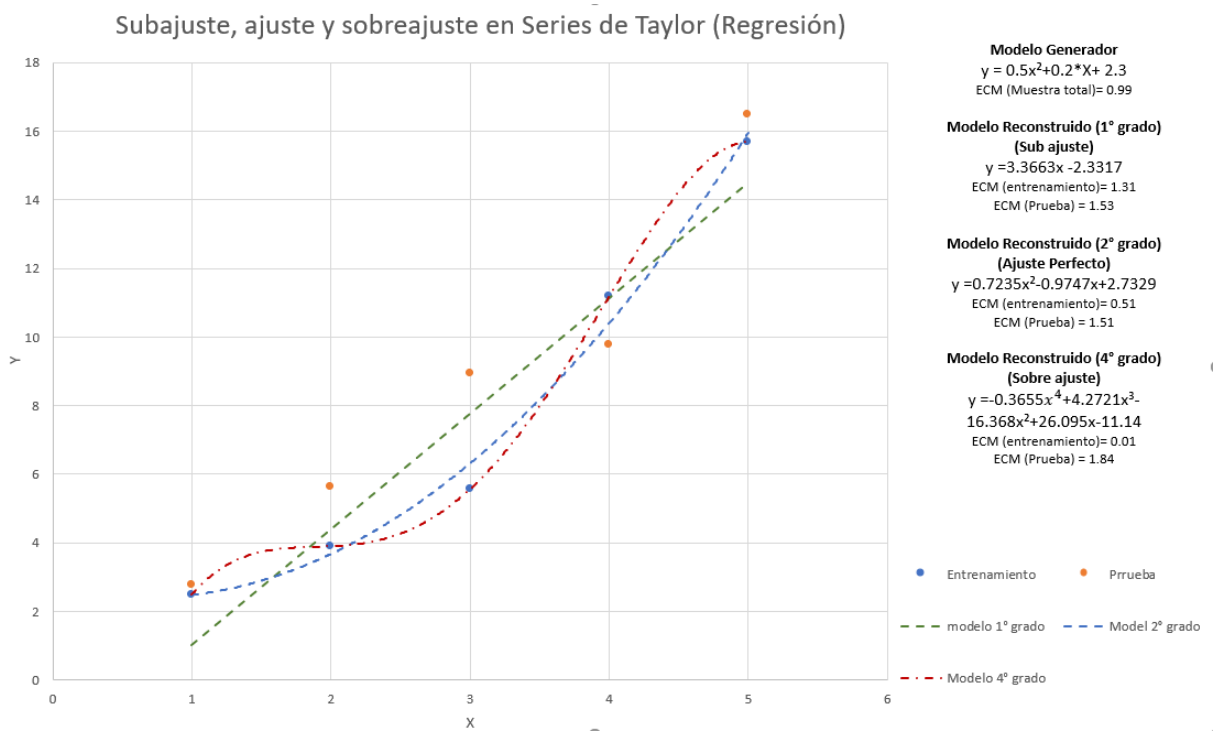


Figura 9: Comparación de ajustes polinómicos de 1, 2 y 5 grado sobre los datos de entrenamiento. Se muestra su error cuadrado medio (ECM) de los ajustes tanto en la muestra de entrenamiento como en la de prueba.

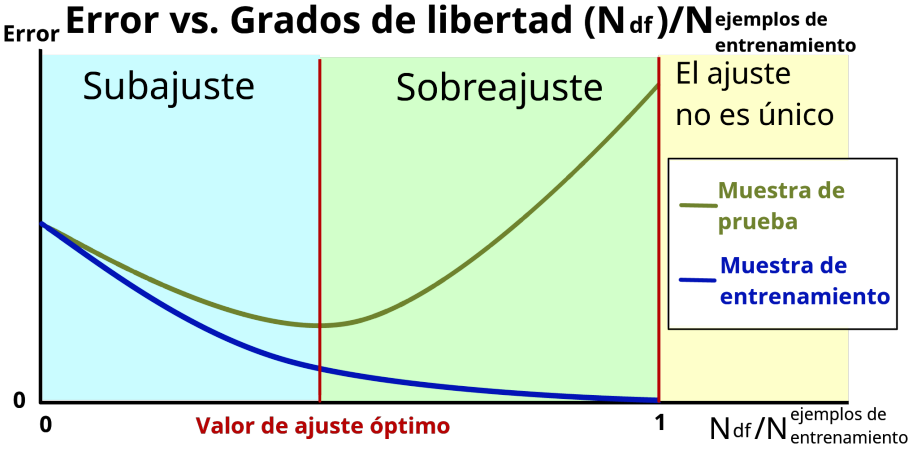


Figura 10: Curva típica de error en función del número de grados de libertad del modelo. A la izquierda se ubican modelos sobre simplificados y a la derecha modelos con complejidad excesiva para el problema a resolver.[30]

En la Figura 9 se calculó el error cuadrático medio para cada ajuste con respecto a los puntos de entrenamiento y a los de prueba. En circunstancias normales se espera que el error de prueba siempre sea mayor que el de entrenamiento.

En error cuadrático medio (ECM) de los puntos respecto a la función generadora nos da una idea del nivel de ruido agregado por el generador y se ubica en $ECM = 0{,}99$. Es decir, el valor mínimo que se puede esperar para el ECM de la muestra de prueba es $ECM = 0{,}99$ que es en el caso de una reproducción total y perfecta de la función generadora.

Es posible llevar el ECM de entrenamiento hasta cero si aumentamos el número de grados de libertad hasta que este sea igual al número de ejemplos en la muestra de entrenamiento (modelo de 4° grado), sin embargo, es el caso más extremo de sobreajuste posible. Es importante observar que al aumentar los grados de libertad de forma innecesaria nuestro modelo se está sobre esforzando por seguir los puntos de entrenamiento y por lo mismo se está alejando de la función generadora, lo cual se ve reflejado en un peor desempeño sobre la muestra de prueba.

La forma típica de las curvas de error en función del número de grados de libertad del modelo de aprendizaje de máquina suele tener la forma de Figura 10 [30]:

En el diagrama anterior se puede observar que la curva de error en la muestra de prueba tiene un mínimo, mientras que la curva de error en la muestra de entrenamiento continúa descendiendo. A la región donde el error de prueba crece al mismo tiempo que el error de entrenamiento decrece se le conoce como región de sobreentrenamiento o sobreajuste.[30]

Debido a que usualmente no se conoce de antemano el número exacto de grados de libertad del problema, se suele elegir un modelo ligeramente sobrado en grados de libertad (y por lo tanto con posibilidad de sobreentrenamiento) el cual se poda posteriormente o se monitorea cuidadosamente mediante un programa de detección de sobreentrenamiento. Un ejemplo de podado es el caso de BDT el cual genera cientos de arboles cada uno con decenas de hojas, al término del entrenamiento el programa de podado busca aquellas hojas y arboles de menor peso y los elimina reduciendo de forma efectiva el número de grados de libertad del modelo y por lo tanto el nivel de sobre entrenamiento.

Los programas de detección de sobreentrenamiento actúan de forma distinta y se aplica a redes neuronales donde no es posible cambiar el número de grados de libertad o ejemplos durante ejecución. Su funcionamiento está basado en la curva de error en función de número de ciclos de entrenamiento y sorprendentemente está tiene un comportamiento similar a la de la figura 10, por lo que evaluando el error en la muestra de prueba en función del número de ciclos de entrenamiento pueden determinar el mejor punto para detenerse antes de incurrir en sobreentrenamiento. Como comprobación para asegurar que no se incurrió en sobreentrenamiento es necesario realizar la prueba de Kolmogórov-Smirnov de bondad de ajuste entre las curvas de respuesta en los datos de

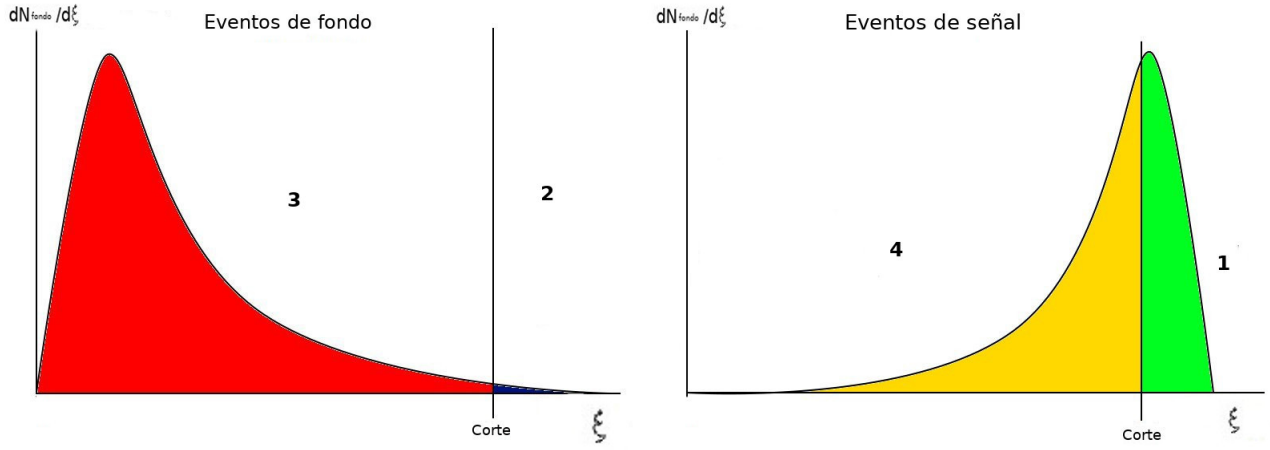
entrenamiento contra las mismas curvas en los datos de prueba, si ambas distribuciones (entrenamiento y prueba) difieren substancialmente entre sí el modelo esta sobreentrenado[17, 18].

3.1.1.3. Prueba o validación En esta etapa se ocupan eventos generados ya sea por el mismo generador que el entrenamiento o por un generador similar. Estos eventos son distintos de los usados en el entrenamiento y por lo tanto constituyen una muestra independiente la cual se espera sea lo más similar posible a los datos reales. Un punto importante es que los eventos de validación deben contener tanto variables de entrada X como predicción esperada Y[33].

En esta etapa se grafican valores tales como las curvas *ROC*, las curvas de señal- ruido, eficiencias, purezas, etc.[17, 18]

En esta etapa se calculan los valores óptimos de corte y se determina si existe algún sobre entrenamiento mediante la prueba de Kolmogórov-Smirnov. También se puede estimar en esta etapa la dependencia del modelo al generador usado, así como estimar el error esperado en datos reales y las contaminaciones de la señal y fondo.

3.1.1.3.1. Definiciones importantes Tras clasificar una colección de eventos mediante un parámetro de retorno propio del método al cual llamaremos ξ se obtienen dos curvas, una para la señal y otra para el ruido. Dependiendo donde se corte en el parámetro de clasificación ξ será la calidad de la clasificación final.



Hay 4 casos posibles de eventos a encontrar[4]:

1 - Verdaderos positivos

2 - Falsos positivos

3 - Verdaderos negativos

4 - Falsos negativos

A partir del número de eventos en cada categoría se definen valores que indican la calidad de un corte[17, 18].
Pureza: La pureza se define como el número total de verdaderos positivos dividido entre el número total de positivos. $Pureza = \frac{1}{1+2}$
Eficiencia Se define como el número de verdaderos positivos sobre el total de eventos de tipo señal. Eficiencia= $\frac{1}{1+4}$
Rechazo de fondo Se define como el número de verdaderos negativos sobre el total de eventos de tipo fondo. Rechazo de fondo= $\frac{3}{2+3}$
Significancia La significancia se define como el número de verdaderos positivos sobre la raíz cuadrada del número de eventos clasificados como señal. Significancia= $\frac{1}{\sqrt{1+2}}$.

3.1.1.3.2. Curvas *ROC* Las curvas *ROC* son las gráficas del rechazo de fondo contra la eficiencia de señal para cada método [17, 18]. Hay que recordar que cada uno de estos valores graficados esta parametrizado por un valor de corte propio del método el cual puede ser muy diferente dependiendo del método que se trate, sin embargo, mediante el uso de las curvas *ROC* y el área bajo ellas es posible comparar métodos muy distintos bajo un marco común. En las curvas *ROC* un método ideal que clasificara perfectamente los eventos estaría representado por un cuadrado de área 1 en esta gráfica, mientras que un método que fallara en todo tendría área 0. El método del área bajo la curva (Ver Figura 11) nos permite determinar a grandes rasgos cuando un método es útil en una gran variedad de situaciones (cortes) distintas para nuestro problema en particular, sin embargo, es posible que estemos interesados solamente en una característica especial del método como por ejemplo una alta pureza sin importar la eficiencia, en ese caso el área bajo la curva no sería un indicador de importancia para nuestro objetivo.

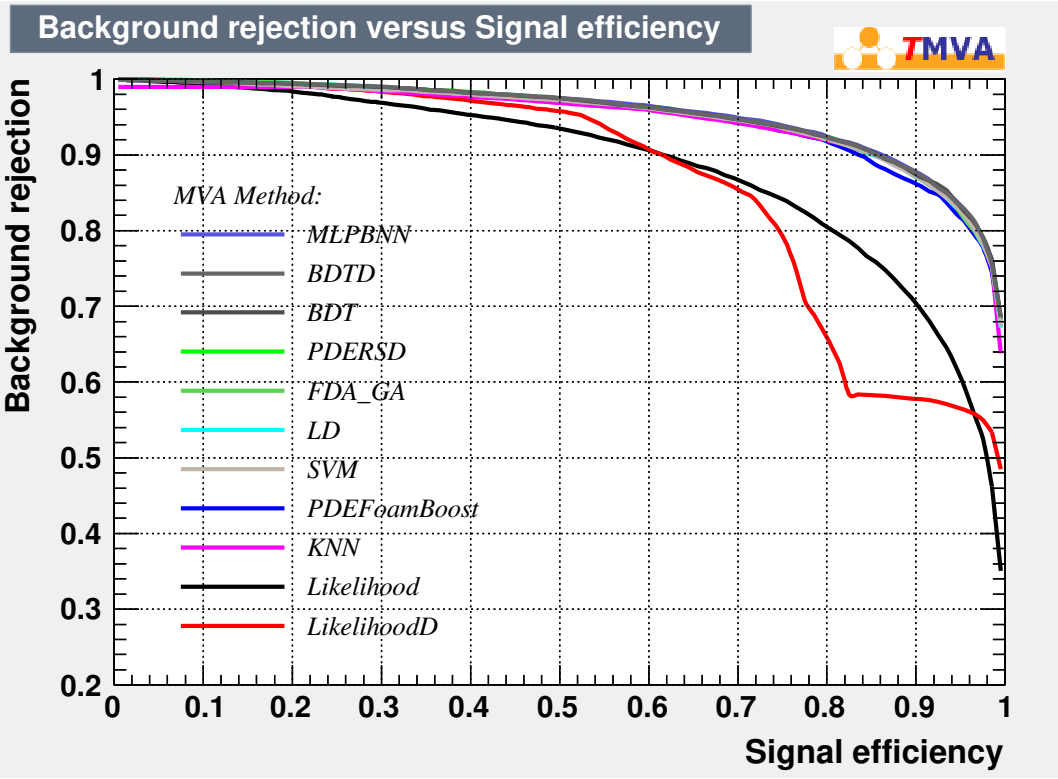


Figura 11: Curvas ROC de los 11 métodos de aprendizaje de máquina a utilizar.

3.1.1.3.3. Curva de significancia La curva de significancia puede ser utilizada para determinar el mejor punto para realizar un corte sobre el parámetro único del método ξ así como ayudar durante el proceso de optimización del método[17, 18].
En caso de utilizarse la significancia como parámetro de selección, se deberá optimizar los parámetros de los métodos tal que maximice la significancia en el punto de corte ξ, el cual también se puede elegir tal que maximice la significancia total del corte.

3.1.1.3.4. Kolmogórov-Smirnov (sobre entrenamiento) La prueba de Kolmogórov-Smirnov es una prueba de bondad de ajuste de distribuciones probabilísticas. También es comúnmente utilizada para comparar la distribución de señal y ruido sobre la variable ξ usadas durante el entrenamiento con una muestra independiente clasificada por el mismo algoritmo. Si se obtiene que ambas distribuciones difieren substancialmente entre sí (no ajustan) el valor de la prueba de bondad de ajuste caerá lo cual indica un sobre entrenamiento del método[34]. Los valores de ajuste inferiores a 0.01 tanto para señal como para ruido se consideran pruebas de sobre entrenamiento.

3.1.1.4. Evaluación o aplicación Esta etapa corresponde a la aplicación del modelo de aprendizaje de máquina en datos reales donde no se cuenta con ninguna forma de respaldar las predicciones del aprendizaje de máquina. En esta etapa solo existen las entradas X sin ningún valor Y esperado, lo que se desea en esta etapa es usar los valores Y predichos por el método.

Para esta etapa se ocupan estimaciones de la forma de las señales generadas y contaminación esperada basadas en los resultados de la etapa de validación.

3.1.2. Pesos por evento

El sistema de pesos se puede aplicar a una colección de eventos a clasificar permitiendo que algunos eventos específicos tengan mayor influencia que otros a la hora de mover los pesos del algoritmo de aprendizaje de máquina. Es decir, los eventos con pesos pequeños tendrán poca influencia sobre el resultado final del entrenamiento, mientras que los eventos con pesos grandes serán más importantes en el resultado[36].
Existen muchos enfoques posibles a la hora de asignar peso a los eventos uno de ellos está orientado a disminuir ya sea falsos positivos o falsos negativos y es aplicado por ejemplo en aplicaciones médicas del aprendizaje de máquina, donde es preferible un falso positivo que puede ser descartado cuando un doctor analice al paciente a tener falsos negativos donde el paciente puede morir por falta de atención. Este método consiste en asignar pesos dispares a la señal y el ruido, si el objetivo es disminuir los falsos negativos, se puede aplicar un peso grande a la señal y bajo al fondo de forma que la máquina se vea fuertemente penalizada por cada elemento de señal mal clasificado, pero no así por elemento de fondo incorrectamente clasificado.
Otra forma de asignar los pesos está basada la conveniencia de entrenar usando el mismo número de eventos de señal como de fondo. En este caso los pesos se asignan para simular las abundancias de eventos señal-fondo en la muestra real y se asignan peso $W_S = 1$ para señal y $W_B = \frac{N_{B_T}}{N_{S_T}}$ para fondo con N_{B_T} el número total de eventos de fondo y N_{S_T} en número total de eventos de señal en la muestra real.
En esta forma de asignar pesos la idea es tomar ventaja de que se pueden reproducir las características de la muestra de fondo o de señal completa tomando solo una muestra representativa de sus integrantes. A la vez que se aprovecha que algunos algoritmos convergen más rápido para muestras simétricas señal-fondo es decir muestras con el mismo número de ejemplos de señal y de fondo.
Durante este proyecto de tesis se entrenó con una muestra simétrica de eventos señal-fondo y se ocupó la asignación de pesos $W_S = 1$ para señal y $W_B = \frac{N_{B_T}}{N_{S_T}}$ para fondo, con el objetivo de recuperar la abundancia relativa de los eventos en la muestra. El parámetro $\frac{N_{B_T}}{N_{S_T}}$ como se verá más adelante puede ser dependiente del generador de eventos utilizado y es importante observar como su cambio entre generador a generador puede afectar significativamente el desempeño de los algoritmos.

3.1.3. Algoritmos utilizados

Se utilizaron los siguientes de aprendizaje de máquina para el análisis y clasificación de eventos:

- MLPBNN
- BDT
- BDTD
- LD
- Likelihood
- LikelihoodD
- PDERSD

- PDEFoamBoast
- SVM
- FDA_{GA}
- KNN

Para más información sobre estos métodos o su implementación consulte la sección de anexos 8.2 “Métodos” o directamente el manual de TMVA [17, 18].

4. Aislamiento de colisiones pp con alto número de MPI

4.1. Búsqueda del plasma de quarks y gluones en colisiones pp

En los generadores de eventos MC es posible dividir las colisiones pp en dos componentes: la física dura correspondiente a la dispersión partónica dura de alto momento y el UE[7]. El UE contiene radiación de estado inicial (ISR) y final(FSR), MPI y remanentes resultantes de la hadronización de constituyentes partónicos que no participaron en ninguna dispersión. Adicionalmente el UE contiene información sobre la CR la cual describe las interacciones entre los campos de color durante la transición de hadronización.[7]

En un principio en el LHC las simulaciones MC del UE fueron generadas basadas en afinaciones (*tunes*) provenientes de los datos del Tevatron. Estas simulaciones mostraron grandes discrepancias con los datos en las primeras medidas del experimento ATLAS, sin embargo con la inclusión de nuevas afinaciones (*tunes*) a los generadores la descripción mejoró[7], sin embargo, es importante mencionar el descubrimiento de forma inesperada de la presencia de un comportamiento similar al observado en iones pesados en los datos la cual no era descrita por los generadores MC basados en un modelo tradicional del UE. [16]

Se ha encontrado que al agregar una combinación de CR y MPI es posible producir patrones de flujo radial vía cuerdas de color aumentadas compatibles con la descripción de los datos, la cual no ha podido ser reproducida por el modelo tradicional[16].

4.2. Interacciones partónicas múltiples

Se ha mostrado que una combinación de MPI con CR puede producir efectos de flujo similares a los observados en los datos. [16] Entre los principales efectos observables de las MPI se encuentran la generación de pares de jets opuestos (*back-to-back*) de bajo p_{T}. Sin embargo, la fracción de MPI que conduce a jets reconstruibles es pequeña [7].

Las MPI suaves que no llevan a la producción de jets observables son mucho más frecuentes y son responsables de correcciones importantes en el flujo de color y distribución de la energía del evento, lo cual afecta la actividad de estado final en forma global aumentando la multiplicidad, distribuciones sumadas de E_T y contribuyendo al rompimiento de los remanentes del haz en la dirección delantera (forward)[7].

Para más detalles sobre su implementación en Pythia consulte la sección 1.2.3

4.3. Aislamiento de eventos con número de interacciones partónicas elevadas

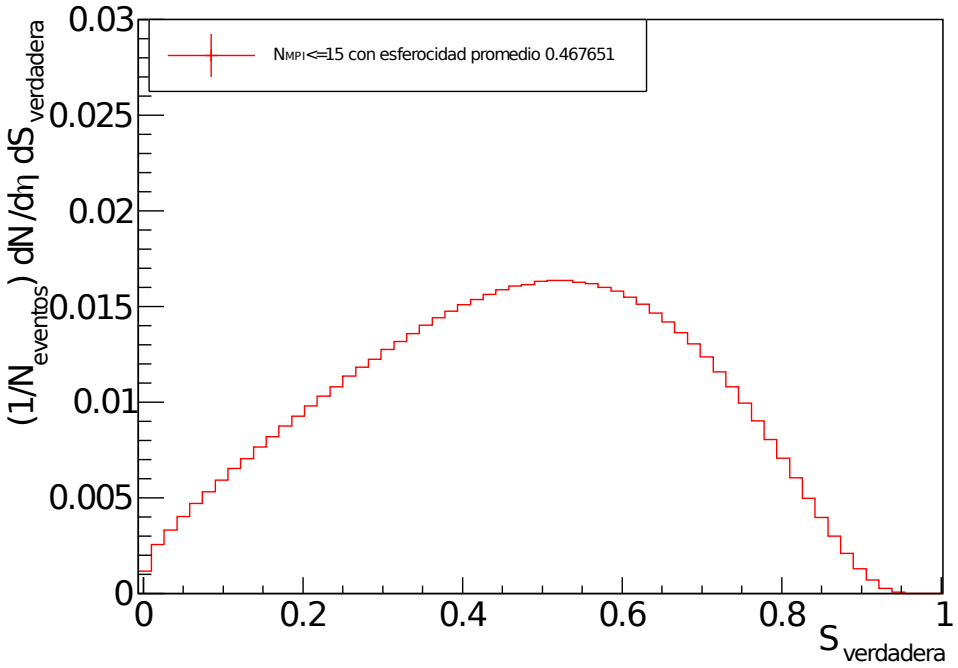
El problema por resolver es el siguiente, de una muestra de eventos que incluye todo tipo de eventos, se desea separar aquellos que tienen un alto N_{MPI}. La complicación viene cuando al igual que como sucedería con datos reales decidimos restringirnos solo a trabajar con las variables reconstruidas en vez de variables verdaderas, es decir, como poder cortar en una cantidad real de la forma más precisa posible sin poder acceder a ella directamente, sino solo a sus efectos medidos a través de un sistema imperfecto como los detectores de partículas.

4.3.1. Método tradicional

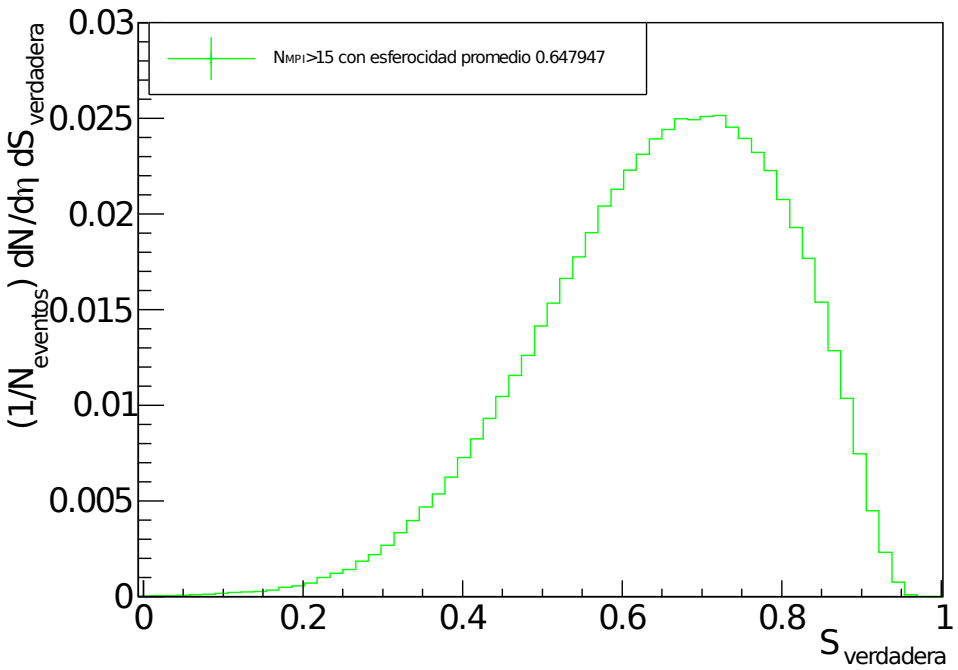
La solución que se ha venido utilizando hasta este momento consiste en utilizar un par de variables (esferocidad y multiplicidad) en su versión reconstruida las cuales se sabe por la teoría que en su versión real están relacionadas con el N_{MPI}[37].

La idea tras este método es que los eventos con alto número de MPI tienden a generar muchas partículas en el evento subyacente, por lo que si aislamos eventos de alta multiplicidad en el evento subyacente habremos seleccionado eventos de alto número de MPI.

La forma de aislar aquellos eventos donde se puede ver claramente el UE sin contaminación de los jets consiste en calcular la esferocidad total del evento. Debido a que el UE no tiene un eje preferencial, es posible seleccionar eventos sin jets prominentes al seleccionar eventos con una alta esferocidad, en contraste con eventos de baja esferocidad los cuales consisten principalmente en jets de partículas (Ver Figura 12).



(a) $N_{MPI} <= 15$



(b) $N_{MPI} > 15$

Figura 12: Los eventos de alto N_{MPI} están acompañados de mayor esferocidad promedio.

Una vez seleccionados los eventos dominados por el UE, es posible utilizar directamente el criterio de selección de alta multiplicidad para quedarnos únicamente con eventos de alta multiplicidad en el UE correspondientes a un alto número de MPI.

El estándar de corte en multiplicidad y esferocidad puede variar, sin embargo, en adelante se comparará como método tradicional al corte $Esferocidad > 0{,}75$ y $Multiplicidad > 30$ como selección generada por el método tradicional.

4.3.2. Método propuesto

Se propone el uso de algoritmos de aprendizaje de máquina para realizar un análisis multidimensional del problema y determinar cuál es la mejor forma de cortar sobre las variables propuestas como de posible relevancia en la determinación del número de MPI.

Mediante el uso del aprendizaje de máquina se planea analizar en un principio 7 variables de posible relevancia para el problema de clasificación. El objetivo es no limitarnos a 2 variables y sus correlaciones como en el método tradicional, sino mediante análisis de las correlaciones entre las 7 variables y la determinación de su importancia relativa en el resultado final se pueda determinar desde el punto de vista estadístico que variables son las más importantes para la clasificación.

También se planea utilizar la capacidad de los algoritmos de aprendizaje de máquina en el análisis multivariable para maximizar la extracción de información por cortes al combinar decenas o cientos de decisiones de corte en 2 o más variables en vez de limitarnos a un único corte en 2 variables como el método tradicional.

5. Trabajo realizado, metodología y resultados

5.1. Obtención de las simulaciones Monte Carlo usadas en este trabajo

Los eventos analizados provienen de las simulaciones de propósito general oficiales de ALICE generadas mediante Pythia 6 o Pythia 8 y como sistema de simulación de detector GEANT 3. Para más detalles refiérase a las secciones 1.2.2, 1.2.3 y 1.2.4 .

5.2. Generación de los árboles de entrenamiento y evaluación en ROOT

Las simulaciones MC generan colecciones de árboles de eventos enormes las cuales se encuentran partidas en archivos los cuales deben analizarse individualmente y extraer de ellos aquellas cantidades que son de relevancia para nuestro análisis. Los arboles provenientes de los MC contienen variables globales de evento como el número de MPI, o banderas que permiten aceptar o desechar eventos fácilmente basado en criterios globales del evento. Cada evento contiene adicionalmente 2 subárboles uno correspondiente a las partículas MC y otro correspondiente a las partículas reconstruidas tras la simulación del detector. Estos árboles de partículas cuentan con información como el pdg (permite identificar el tipo de partícula Ej. Pion, Protón, positrón, fotón, etc), el cuadrimomento de la partícula y en el caso de las reconstrucciones información detallada sobre la traza reconstruida tal como el número de clúster generados en el detector.

Los arboles provenientes de los MC contienen todo tipo de eventos y trazas registradas por los detectores, por lo que es necesario identificar primero las trazas bien reconstruidas. Para que una traza se considere bien reconstruida se requiere:

- Más de 70 clusters.
- Ajuste con $\chi^2 < 4$.
- Distancia mínima de aproximación al vértice principal en el eje Z menor a 3,2 cm.
- Y para el plano XY menor a 2,4 cm.
- No debe depender del ITS para su reconstrucción.

A las trazas aceptadas se les aplican los cortes cinemáticos $0{,}15\,\mathrm{GeV/c} < p_\mathrm{T} < 10^8\,\mathrm{GeV/c}$ y $-0{,}8 < \eta < 0{,}8$ los cuales corresponden a la ventana de aceptación de ALICE. Si el evento aún posee 3 o más partículas correctamente reconstruidas y en la ventana de aceptación se aprueba si no se descarta.

Usando únicamente los eventos aprobados por las condiciones anteriores y usando únicamente las trayectorias y partículas correctamente reconstruidas se procedió a calcular las variables de entrada para el clasificador (Ver 5.3.1). Para el caso de la generación de árboles de entrenamiento se crean dos árboles uno de señal y uno de fondo, ambos se llenan con sus correspondientes eventos de acuerdo al corte en N_{MIP} deseado. Cada evento almacenado incluye los árboles de partículas con sus cuadrimomentos y pdg correspondientes, además del N_{MIP} y cantidades globales reconstruidas usadas por los métodos durante el entrenamiento. Para el caso de los árboles para evaluación se realiza exactamente lo mismo, sin embargo, todos los eventos se guardan en un solo árbol en vez de separarlos en señal o fondo. Con el objetivo de realizar cómputo en paralelo en el clúster, en vez de crear un solo árbol ese árbol se divide en 50 partes cada una que será analizada por un núcleo diferente en paralelo a la hora de la evaluación.

En su posible aplicación a los datos experimentales, los cortes y el proceso de selección serían idénticos, con la excepción de que no habría una parte "real" proveniente del MC por lo que las líneas del programa que requieran valores MC para funcionar se ignorarían. Es importante recordar que N_{MPI} es una cantidad "real" proveniente del MC por lo que no es posible entrenar usando datos reales sino solo evaluar usando un algoritmo previamente entrenado en algún MC.

Es importante mencionar que el algoritmo anterior es posible correrlo en paralelo y posteriormente unificar los arboles generados mediante la función *merger* de Root la cual permite unir archivos Root con la misma

estructura en un solo archivo Root. Para el clúster se separó el total de los archivos a analizar en 73 paquetes de 48 simulaciones cada uno para Pythia 8 y 139 paquetes de 25 simulaciones cada uno para Pythia 6 paquetes los cuales se analizaron por separado y posteriormente se fusionaron para producir un único archivo Root para entrenamiento y otro para evaluación. El número de simulaciones corresponde al número de hilos usados durante el proceso de simulación y corresponden a semillas distintas del generador de números aleatorios de Pythia, cada simulación contiene alrededor de 20000 eventos no filtrados.

5.3. Selección de variables relevantes

El proceso de selección de variables relevantes para la clasificación es de gran importancia pues determinan la complejidad y efectividad de los algoritmos. Una mala elección de variables puede conducir a un subdesempeño de los métodos, así como un uso excesivo de memoria y tiempo de cómputo.

Los requisitos de las variables a ser usadas son:

- Deben estar poco correlacionadas entre sí
- Deben ser globales al evento.
- Deben ser numéricas
- Deben ser no vacías: Es decir por ninguna razón puede existir un evento en el cual la variable quede sin asignársele valor alguno.
- La forma de calcularlas debe estar bien definida por un algoritmo.

En nuestro caso también se debe agregar una condición más, queremos que las variables tengan relevancia física, pues los métodos nos proveerán de información sobre su relación con el N_{MPI} relación que puede resultar de interés y relevancia en estudios futuros.

5.3.1. Variables propuestas

Se propusieron 7 variables que podrían resultar de importancia para nuestra clasificación, las cuales se sabe están altamente relacionadas con el N_{MPI}.

5.3.1.1. Multiplicidad La multiplicidad es el número de partículas finales producidas en nuestro evento y en el caso de la multiplicidad reconstruida es el número de partículas correctamente reconstruidas mediante sus trazas. Los eventos con alto N_{MPI} suelen estar relacionados con alta producción de partículas y por lo tanto con alta multiplicidad sobretodo en la región transversa.

5.3.1.2. Esferocidad La esferocidad es una medida para cuantificar la homogeneidad o heterogeneidad en la distribución espacial de partículas y p_{T} en un evento; toma valores entre 0 y 1, donde 1 corresponde a un evento homogéneamente distribuido en todas direcciones (esférico) y 0 corresponde a un evento heterogéneo donde las partículas se concentran en jets.

Un criterio importante para seleccionar eventos de alta multiplicidad que también correspondan a alto N_{MPI} es requerir que la alta multiplicidad no sea generada en jets de partículas, es decir eventos con esferocidad alta.

5.3.1.3. Retroceso El retroceso o en ingles *recoil* indica el desbalance de momento lineal en el evento y se define como:

$$R = \frac{|\sum_i \vec{P}_i|}{\sum_i |\vec{P}_i|} \tag{6}$$

Es decir la norma de la suma vectorial de los momentos de las partículas entre la suma de las normas de los momentos.

El valor del retroceso se encuentra en un rango entre 0 y 1, para alto N_{MPI} se esperan eventos balanceados con retroceso cercano a 0 mientras que eventos con dispersiones duras tendrían valores mayores.

5.3.1.4. Multiplicidad en la región transversa La región transversa definida como aquella entre 60° y 120° (en ϕ no en η) respecto a la partícula líder es altamente sensible a lo que se conoce como evento subyacente (UE) , por lo tanto se espera que también sea la más sensible al N_{MPI}.

La ventaja de observar la multiplicidad en esta región respecto a la multiplicidad global es que nos encontramos alejados de los jets principales y por lo tanto una alta multiplicidad en esta región no puede provenir de fragmentaciones de los jets.

5.3.1.5. p_{T} líder El p_{T} líder corresponde al mayor p_{T} encontrado entre las partículas registradas y define la partícula líder así como el eje del jet principal. Un p_{T} líder alto suele corresponder a un jet bien definido, mientras que un p_{T} líder bajo corresponde a un evento donde las partículas se repartieron de forma más equitativa el momento y por lo tanto el jet principal no toma la misma prominencia.

En los eventos con alto N_{MPI}, el momento se reparte de forma equitativa entre todas las partículas constituyentes por lo que el p_{T} líder es menor que en un evento dominado por jets.

5.3.1.6. p_{T} promedio Los eventos de interés cuentan con una multiplicidad alta y no tienen presencia de jets, lo cual implica que hay muchas partículas entre quienes repartir la energía y el momento de forma equitativa, resultando en p_{T} promedios bajos, mientas que los eventos duros, poseen pocas partículas muy energéticas y por lo tanto p_{T} promedios altos.

5.3.1.7. p_{T} del jet líder Mediante el algoritmo de búsqueda K_{T} incluido en FastJet [19] se ubicaron aglomeraciones de partículas en el espacio de momentos, correspondientes a jets. Los parámetros de búsqueda fueron radio de 0.4 y p_{T} mínimo de 5 GeV/c. De todos los jets encontrados se ubica el jet con mayor p_{T} y esté se define como jet líder. Similar al p_{T} líder en motivación, pero usando un algoritmo de búsqueda de jets y p_{T} total del jet en vez de p_{T} por partícula.

5.3.2. Variables no incluidas

Hay variables que no se incluyeron por estar excesivamente correlaciones con otras de las variables propuestas o por su poca relevancia.

Entre estas variables se encuentra la esfericidad, la cual está demasiado emparentada con la esferocidad y no aporta información extra al análisis debido a su alta correlación.
Otras variables que se incluyeron en primeros análisis, pero posteriormente fueron desechadas debido a su baja relevancia fueron: el número promedio de clústeres por traza y el p_{T} promedio de los jets encontrados.

5.3.3. Correlaciones entre variables

La importancia relativa de cada variable se obtiene de dos fuentes, la primera es el coeficiente de correlación con otras variables tanto en la señal como en el fondo. Si el coeficiente de correlación entre dos variables es alto significa que la información que aportan estas variables es redundante y no ayuda a mejorar la selección, incluso algunos métodos se desempeñan por debajo de su nivel óptimo en presencia de correlaciones entre las variables como por ejemplo $MLPBNN$ o *likelihood*. Hasta el punto de que se sugiere eliminar una de las dos variables y quedarnos solamente con la más significativa de las dos.

En caso de que no se pueda o no se desee eliminar las variables redundantes, se puede utilizar algoritmos de decorrelación de variables como preparación previa al entrenamiento y evaluación. Esto agrega complejidad al proceso, sin embargo, es necesario para el correcto funcionamiento de algunos métodos pues trabajan bajo el supuesto que las variables de entrada son independientes y se desempeñan muy por debajo de su capacidad cuando las variables están correlaciones.

5.3.3.1. Matrices de correlación En las figuras 13 y 14 se muestran las matrices de correlación para las variables de clasificación para el grupo de eventos señal y para el grupo de evento de fondo.

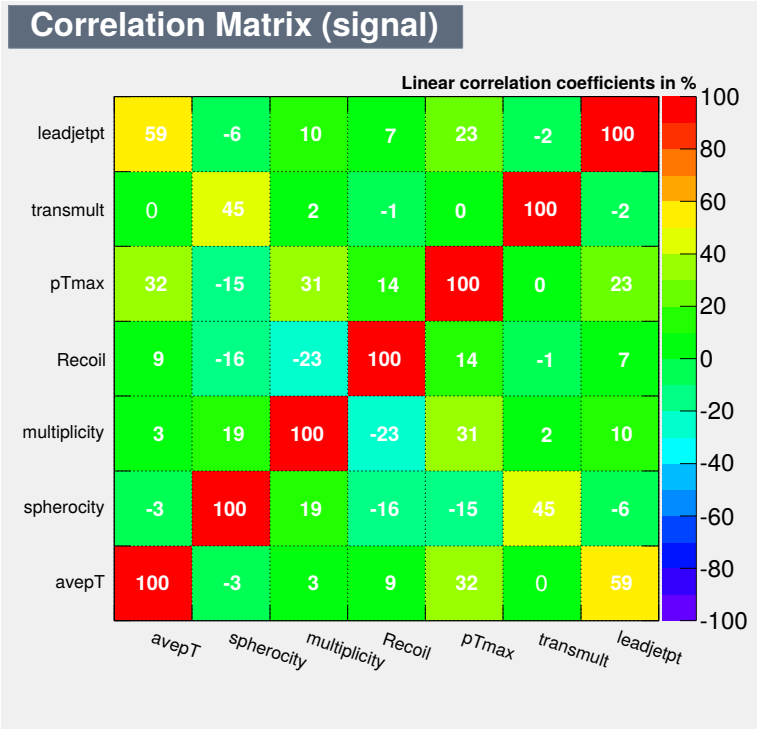


Figura 13: Matriz de correlaciones para la señal ($NMPI > 15$)

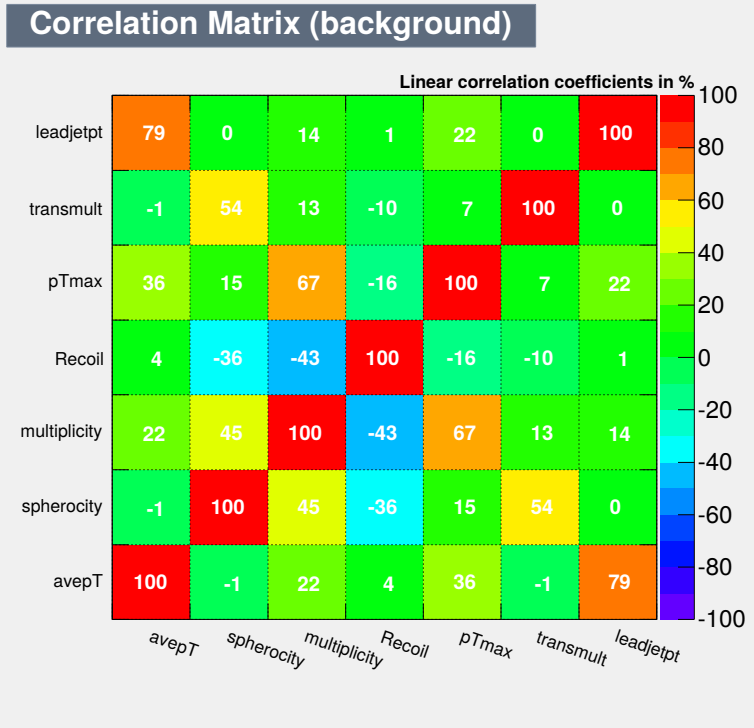


Figura 14: Matriz de correlaciones para el fondo ($NMPI <= 15$)

Es posible ver que las variables p_{T} del jet líder y el p_{T} promedio se encuentran fuertemente correlacionadas asimismo las variables multiplicidad en la región transversa y esferocidad también están muy correlacionadas.

La variable p_{T} líder normalizado esta correlacionada con la multiplicidad, pero también con el p_{T} promedio, por lo que usualmente es preferible eliminarlo a eliminar las otras dos variables que no están correlacionadas entre sí de forma tan importante.

Hay tanto correlaciones como anti-correlaciones importantes entre el retroceso, la multiplicidad y la esferocidad para el fondo, sin embargo, no son tan importantes para la señal. También es de notar que son menores en importancia a las otras correlaciones entre mencionadas.

5.3.4. Separación de una variable

Es posible eliminar variables poco relevantes para nuestro proceso de clasificación tras detectar una redundancia en la información o una variable inútil sin poder de clasificación. Para determinar que variable debe quedarse y cual eliminarse se recurre a una cantidad llamada separación la cual indica que tanto se traslapan las curvas de señal y fondo en la variable a analizar. Si las curvas se traslapan completamente (variable inútil) la separación resultante es cero, mientras que si las curvas están completamente separadas (variable ideal) y se puede separar señal de fondo usando únicamente esta variable, entonces la separación resultante es 1.

La forma de calcular la separación $< S^2 >$es:

$$< S^2 >= \frac{1}{2} \int \frac{(Y_S(X) - Y_B(X))^2}{Y_S(X) + Y_B(X)} dX \tag{7}$$

Con X la variable en cuestión, $y_S(X)$ y $y_B(X)$ el número de eventos de señal o fondo con ese valor de la variable X (histogramas) normalizados a área 1.

Los valores de separación calculados para las variables en orden descendente son:

Variable	Separación $< S^2 >$
Multiplicidad	0.6647
p_{T} líder normalizado	0.4025
p_{T} promedio	0.2961
Retroceso	0.2261
Esferocidad	0.2207
Multiplicidad en la región transversa	0.1867
p_{T} del jet líder	0.0883

Cuadro 1: Valores de separación $< S^2 >$ de las 7 variables propuestas par a la clasificación en N_{MPI}.

5.3.5. Reducción del número de variables

Se decidió realizar el entrenamiento y la obtención de curvas de pureza, eficiencia y rechazo de fondo para todos los métodos utilizando tanto las 7 variables originalmente propuestas como usando un conjunto de solo 4 variables. Se compararán las curvas al final y se observará si hubo algún cambio en el desempeño tras eliminar aquellas variables juzgadas como menos significativas o redundantes.

La primera variable eliminada fue p_{T} del jet líder debido a su alta correlación con el p_{T} promedio y baja separación. De forma similar la multiplicidad en la región transversa se eliminó en favor de la esferocidad por su menos separación.

Finalmente, la siguiente correlación importante es entre el p_{T} líder normalizado y la multiplicidad. Hay dos razones para eliminar el p_{T} líder normalizado, la primera es que tiene una separación menor que la multiplicidad, y la segunda es porque también guarda correlaciones importantes con el p_{T} promedio, sin embargo la multiplicidad está correlacionada con todo por lo que hay razones para eliminarla. E nuestro caso decidimos quedarnos con la multiplicidad y eliminar el p_{T} líder normalizado para poder comparar con el método tradicional y por tener mayor poder de separación que cualquier otra variable individual. El grupo de 4 variables propuesto como las más significativas para la clasificación es:

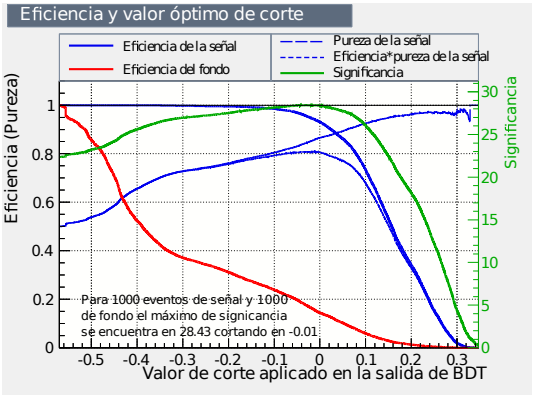
- Multiplicidad
- p_{T} promedio
- Retroceso
- Esferocidad

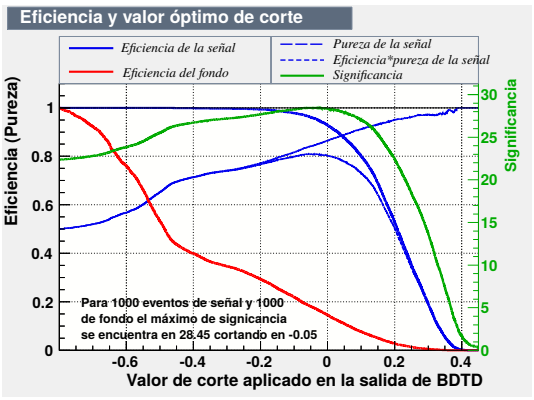
5.4. Entrenamiento de los algoritmos de aprendizaje

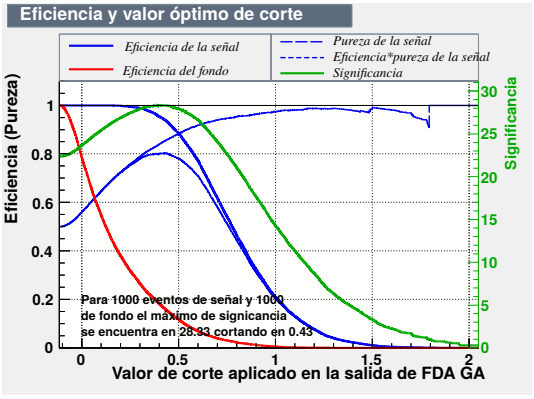
5.4.1. 4 o 7 variables

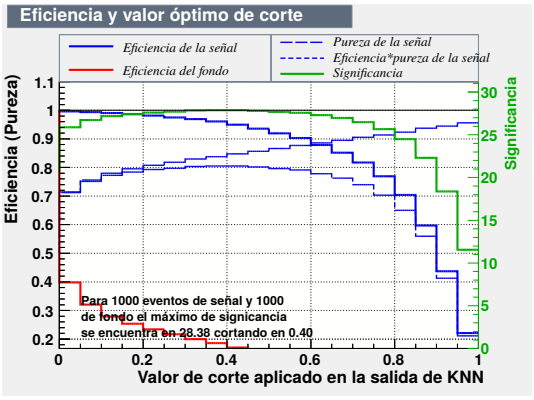
La figura 15 muestra las curvas de eficiencia y pureza correspondientes a cada uno de los métodos en función de su valor de respuesta ξ esto para 4 y 7 variables. Los pesos durante el entrenamiento se ajustaron en 24,871 para señal y 1,000 para fondo, el número de eventos de entrenamiento fueron $20K$ de señal y $20K$ de fondo y para prueba $10K$ eventos de señal y $10K$ de fondo sin pesos. Se puede observar a la izquierda la gráfica para 4 variables y a la derecha para 7 variables.

Entrenamiento usando 4 variables

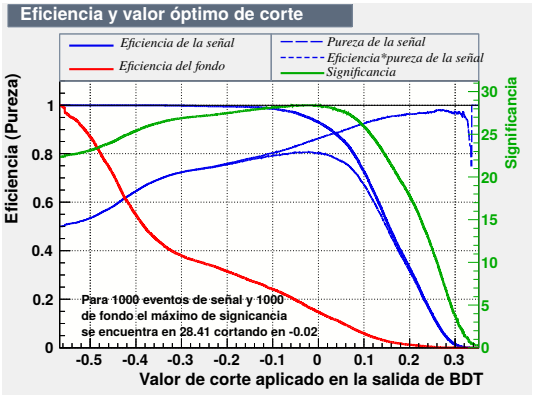


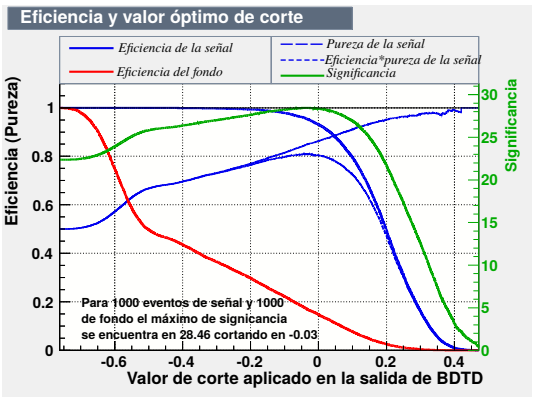


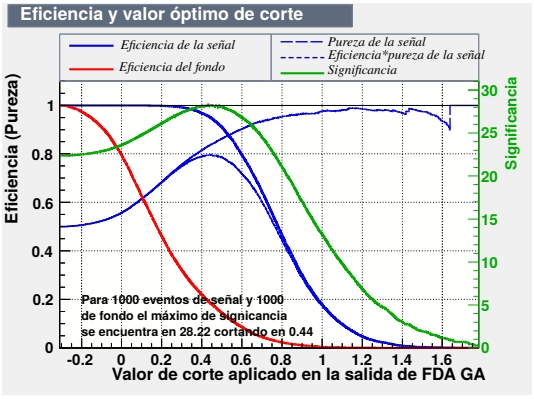


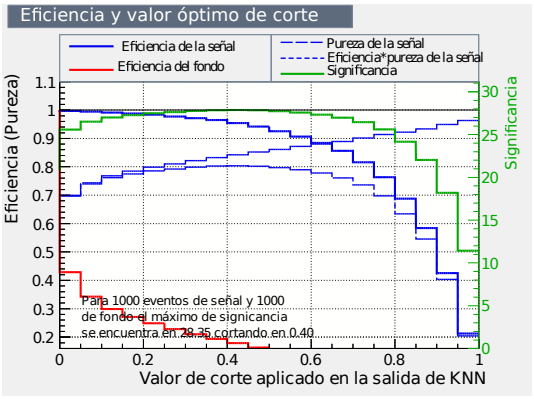


Entrenamiento usando 7 variables

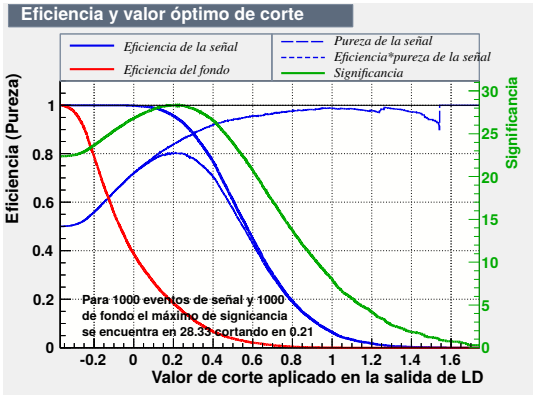


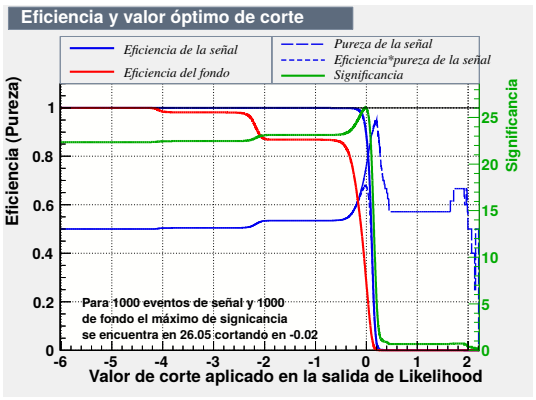


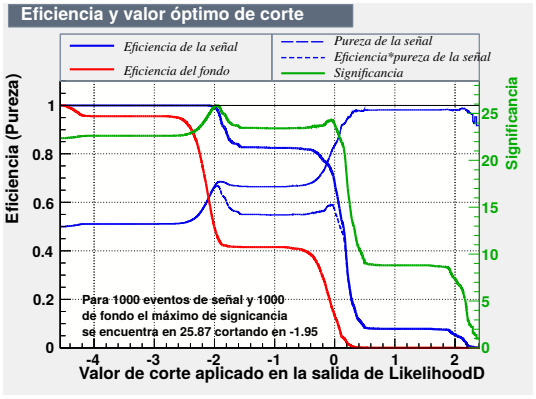


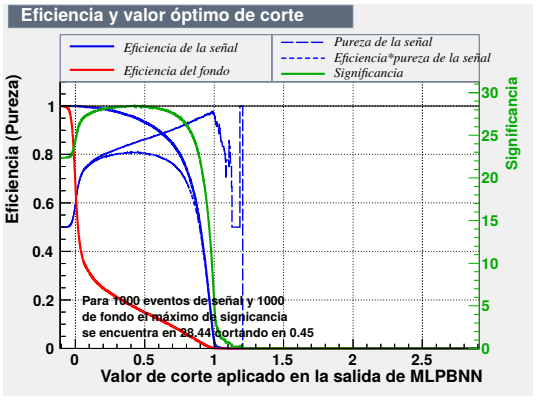


Entrenamiento usando 4 variables

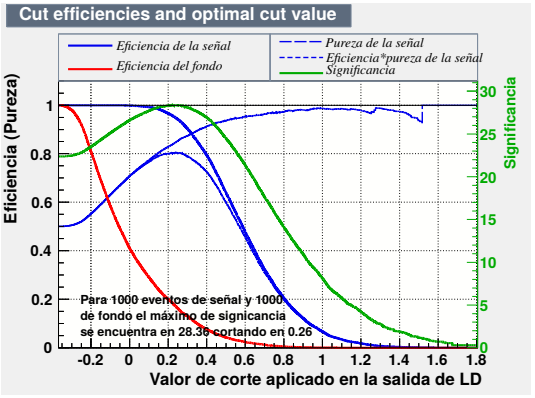


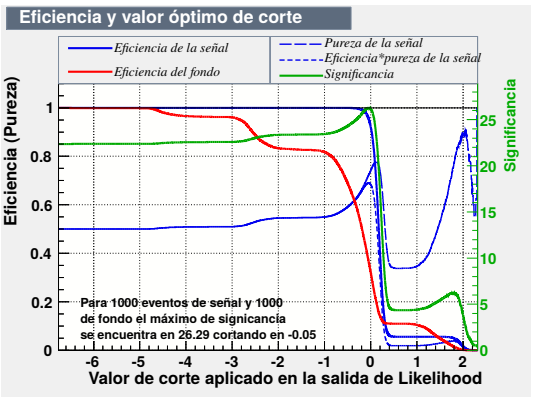


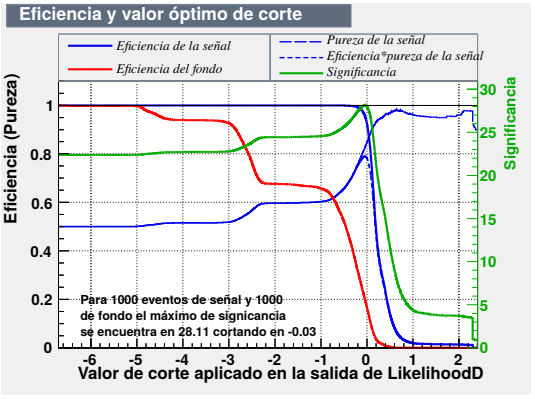


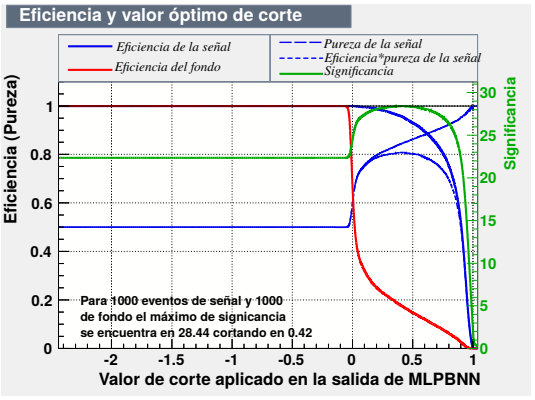


Entrenamiento usando 7 variables









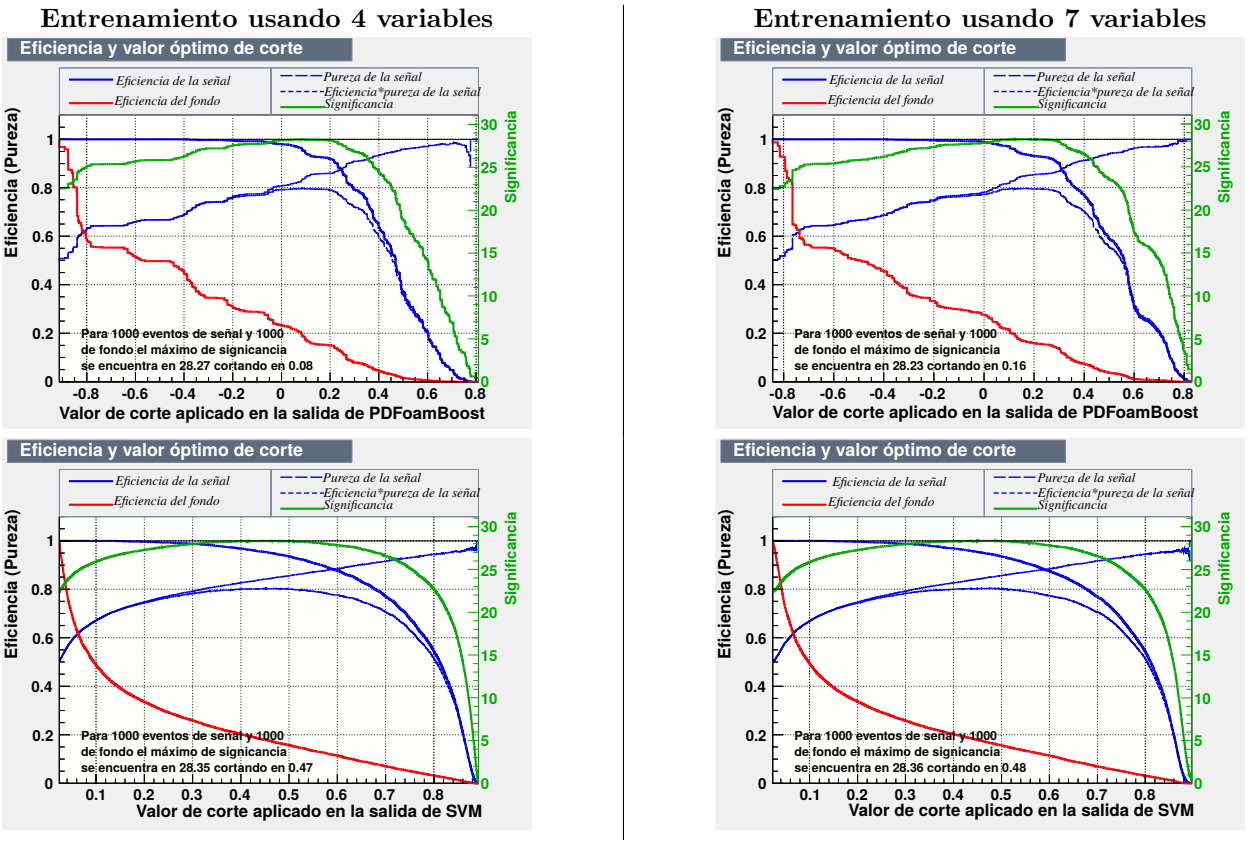


Figura 15: Curvas ROC para 4 y 7 variables

También se graficaron las curvas *ROC* las cuales al no depender del parámetro de corte de forma explícita, permiten comparar distintos métodos en un marco común.

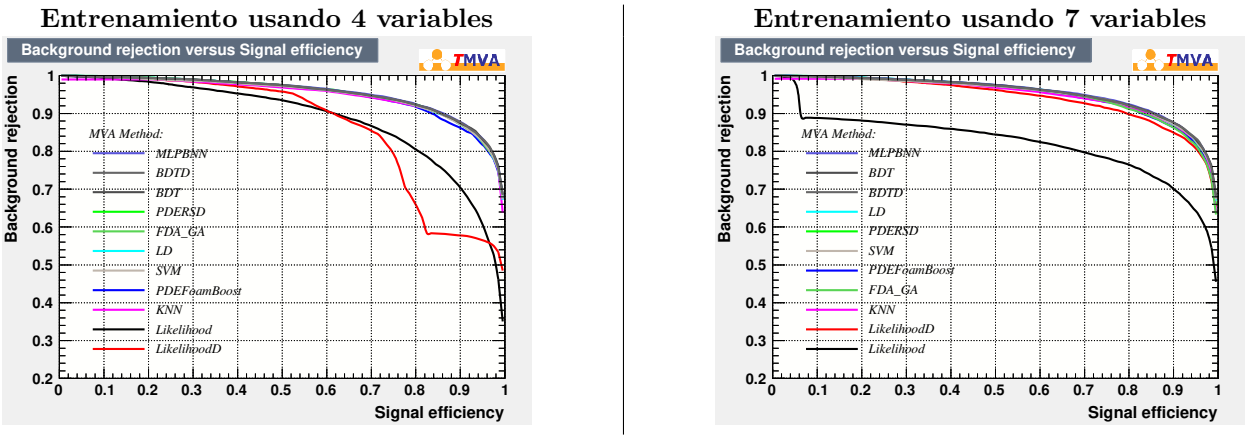


Figura 16: Curvas ROC para 4 y 7 variables

Método	*ROC* **para 4 variables**	*ROC* **para 7 variables**
MLPBNN	0.953	0.953
BDT	0.953	0.953
BDTD	0.953	0.952
LD	0.952	0.952
PDERSD	0.953	0.951
SVM	0.950	0.950
PDEFoamBoost	0.949	0.949
FDA$_{GA}$	0.952	0.948
KNN	0.947	0.947
LikelihoodD	0.885	0.940
Likelihood	0.866	0.821

Cuadro 2: Comparación de los valores de las integrales ROC para los modelos entrenados usando 4 o 7 variables.

Con base en las curvas de la figura 16 es posible determinar si existe perdida significativa en el poder de discriminación al retirar las 3 variables consideradas menos significativas para nuestra tarea de clasificación. Si las variables retiradas eran redundantes como propusimos, el poder de clasificación no se verá negativamente afectado al retirarlas, sino por el contrario en ocasiones podría haber mejoras en el poder de clasificación.

Como se puede observar en el cuadro 2 el valor de la integral ROC registró una mejora al retirar las 3 variables para los métodos BDTD (+0.001), FDA$_{GA}$ (+0.004) ,Likelihood (+0.045) y PDERSD (+0.002) mientras que el único método que empeoro significativamente fue LikelihoodD con (-0.055).
Debido a que las curvas ROC no revelan una perdida importante en el poder de clasificación para los mejores métodos si no por el contrario una ligera mejora en la clasificación; podemos reducir nuestro grupo de variables de clasificación de 7 a 4, sin peligro de afectar el desempeño de los métodos.

5.5. Selección de los mejores métodos

Hasta el momento hemos realizado la clasificación usando todos los métodos con parámetros por defecto de TMVA, sin optimizarlos de forma específica para nuestro problema. Para optimizarlos primero debemos identificar en cuales vale la pena trabajar y en cuáles no.

Los métodos se pueden clasificar en lineales y no lineales (Ver Sección 8.1). Los problemas lineales son más sencillos de analizar y se pueden utilizar algoritmos ligeros y de fácil interpretación, mientras que los problemas no lineales requieren de métodos más complejos, con más profundidad y etapas intermedias los cuales pueden resultar más complicados de analizar sin embargo son capaces de resolver problemas de clasificación complejos.

Es importante notar que se requiere hacer las corridas con estadística suficiente de entrenamiento para asegurar la convergencia tanto de los métodos lineales como no lineales, pues los métodos lineales convergen más rápidamente que los no lineales Para 7 variables se ha observado que se requieren al menos 1K eventos para asegurar convergencia de los métodos lineales y más de 10K para los no lineales. Las corridas de entrenamiento de la sección 3.3.5 se realizaron con 20K eventos de entrenamiento por lo que se cuenta con estadística suficiente para que los métodos convergieran.

Los problemas lineales buscan a lo más correlaciones entre variables las cuales sean de primer grado, es decir, la señal y el fondo se pueden separar perfectamente cortando el espacio por medio de un hiperplano en dos regiones una de señal y una de fondo. Comparando el desempeño de los métodos lineales y no lineales es posible determinar rápidamente si el problema a resolver es en su mayoría lineal o si difiere grandemente de la hipótesis lineal.

En nuestro caso podemos comparar los métodos lineales como LD o FDA_{GA} (en configuración lineal) con los métodos más generales no lineales como MLPBNN, BDT, SVM y KNN. Si su desempeño es similar o si los métodos lineales fueron ligeramente superiores significa que el problema es en su mayoría lineal. Para nuestro problema las curvas generadas son casi idénticas y las integrales *ROC* son del mismo orden 0.947 a 0.953 para no lineales y 0.952 para lineales lo cual apunta a un problema prácticamente lineal.
Debido a la simplicidad y rapidez de los métodos lineales, nos gustaría quedarnos con ellos por eso continuaremos con LD y FDA_{GA}. De los métodos no lineales podemos tomar los dos mejores los cuales son MLPBNN y BDT. Adicionalmente podemos tomar el método KNN ya que a pesar de mostrar una *ROC* menor y ser no lineal por naturaleza el modelo es sencillo y al igual que FDA puede ajustarse para asemejarse más a un modelo lineal o a un modelo no lineal dependiendo de un parámetro K.

De esta forma nos quedaremos con 5 métodos los cuales optimizaremos para su desempeño en nuestro problema particular. Los métodos por conservar son:

- LD
- FDA_{GA}
- MLPBNN
- BDT
- KNN

5.6. Optimización de los parámetros

En muchos de los algoritmos de *machine learning* existen parámetros libres los cuales pueden modificarse de acuerdo al problema a enfrentar y los objetivos deseados. Los objetivos principales del proceso de optimización suelen ser aumentar el poder de discriminación y/o disminuir la carga computacional del algoritmo.

En nuestro caso nos interesa disminuir la carga computación sin afectar de forma significativa el poder de discriminación. Por lo que nos dedicaremos a podar los métodos removiendo aquellos grados de libertad internos con poco peso sobre el resultado final.

Los algoritmos de aprendizaje de máquina poseen grados de libertad que se usan para ajustar la función de clasificación desconocida y la forma de manejarlos o llamarlos es diferente para cada algoritmo. Por ejemplo, en el caso de BDT al aumentar el número de árboles del bosque aumenta el número de grados de libertad del ajuste, mientras que en KNN al disminuir la muestra K de comparación el algoritmo aumenta su capacidad de distinguir detalles en la distribución lo cual arroja resultados similares a un método con mayor número de grados de libertad.

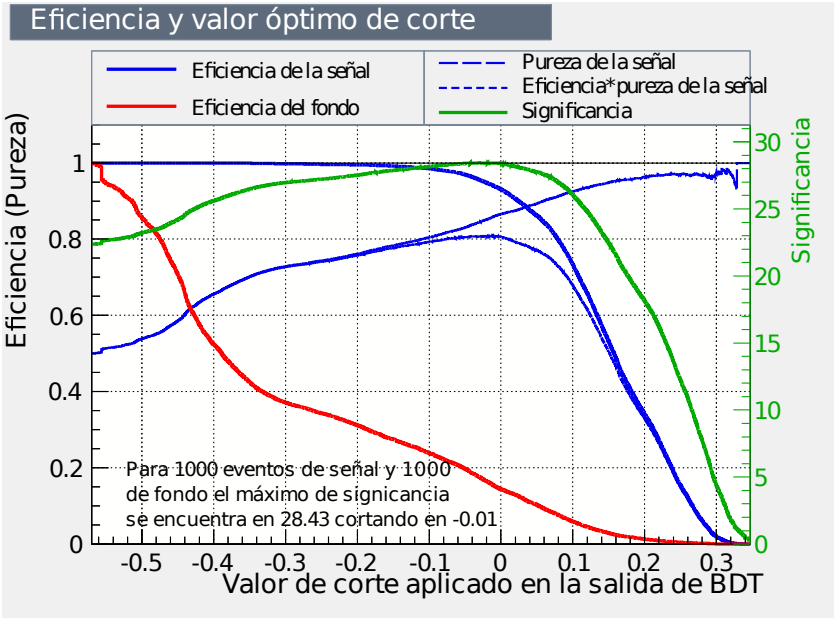
En caso de que exista un método con menos grados de libertad de los necesarios para resolver el problema de forma correcta, se pueden agregar grados de libertad extra, hasta determinar cuántos son los mínimos necesarios para resolver el problema. En nuestro caso, como el problema es en su mayoría lineal, es probable que los algoritmos como BDT y MLPBNN tengan muchos grados de libertad de sobra y no al revés como sucede en problema complejos con relaciones altamente no lineales entre las variables.

La forma de identificar si sobran o faltan grados de libertad es observar los pesos asignados tras el entrenamiento a cada uno de estos grados. Si todos los pesos son similares podría deberse a que el problema es complejo y aún es necesario agregar más grados de libertad para poder resolver correctamente los detalles en la clasificación. Si por el contrario hay un grupo de pesos grandes y el resto son pequeños, esto significa que los detalles más sobresalientes están ya completamente resueltos por los primeros pesos y el resto de los pesos aportan poco o nada al análisis, en cuyo caso la optimización corresponde en eliminar aquellos grados de libertad con pesos reducidos.

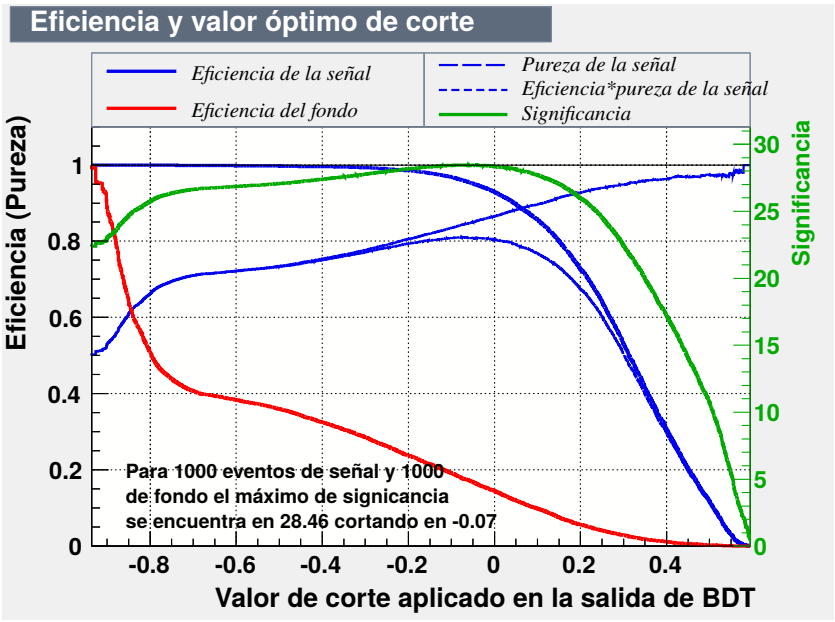
Para el caso de BDT hay tres variables posibles a optimizar, el número de árboles, la profundidad de cada árbol y el tamaño de nodo. Mientras que la profundidad de árbol está relacionada con la complejidad de la cadena de decisiones a tomar, la cantidad de árboles está relacionada con el número de cadenas de decisiones a considerar

y el peso de cada una de estas decisiones, finalmente el tamaño de nodo está relacionado con la cantidad mínima de eventos por hoja necesarios para considerar un patrón relevante en la clasificación.

El primer parámetro por optimizar sería el tamaño de nodo, el cual se sugiere de acuerdo al manual de TMVA en 5.0 % para tareas de clasificación. La profundidad de árbol por defecto es 3 y para relaciones lineales 2 o 3 variables deben ser suficientes para la tarea de clasificación. En la figura 17, se muestra la comparación entre el desempeño a 2 y 3 variables en profundidad de árbol.



(a) Profundidad de árbol =3



(b) Profundidad de árbol =2

Figura 17: Curvas de eficiencia, pureza y significancia para 2 y 3 nodos de profundidad por árbol.

Como se puede observar en la figura 17, las curvas para 2 y 3 nodos de profundidad de árbol son prácticamente idénticas y de hecho existe una pequeña mejora en la significancia máxima al disminuir la profundidad permitida a 2 nodos. Por lo que podemos considerar que reducir el número de nodos no tiene un efecto negativo sobre el proceso de selección, sino por el contrario aumenta la tasa de convergencia (menos parámetros a ajustar) y también hace al modelo más ligero y fácil de interpretar (modelo más pequeño con menos grados de libertad). La figura 18 muestra los pesos de los árboles en función del número de árbol generado, con esta información se puede determinar aproximadamente el número mínimo de árboles necesarios para lograr una separación con el mejor balance entre complejidad de modelo y área ROC.

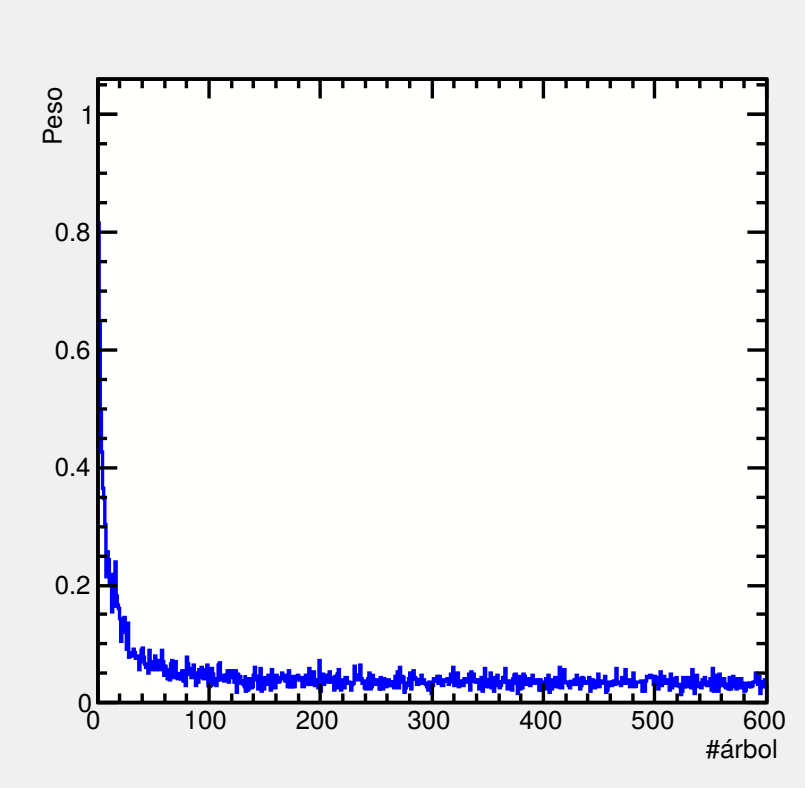


Figura 18: Gráfica de control del algoritmo BDT mostrando el peso en función del árbol.

Es posible notar que los pesos a partir de los arboles 100 a 110 alcanzan un nivel más o menos constante y menor a los primeros pesos, estos árboles aportan relativamente poco al poder de discriminación total. Es posible eliminar la mayoría de estos árboles sin causar daño al poder de discriminación, sin embargo, es recomendable dejar un porcentaje extra para permitir un reacomodo de los pesos en el nuevo ciclo de entrenamiento con menos árboles. En la versión optimizada se utilizarán 175 árboles para asegurar que no estamos deforestando demasiado nuestro bosque y no afectemos el poder de discriminación del método BDT.

Para el algoritmo KNN el proceso de optimización tiene dos elementos, el tamaño de la muestra de entrenamiento y el parámetro K. El tamaño de la muestra está relacionado con el peso final del archivo, la estadística de la selección y el tiempo de cómputo, entre más complicada la forma de la frontera entre señal y fondo mayor será la estadística necesaria para resolverlo propiamente. Si el parámetro K crece de forma proporcional al tamaño de la muestra, el número de detalles que el algoritmo será capaz de resolver será el mismo, sin embargo, con la ventaja de menor fluctuación estadística en el resultado a mayor tamaño de muestra de entrenamiento. Si para un número dado de eventos de entrenamiento se decrementa el valor de K aumentará el número de detalles que se pueden resolver a costa de menor estadística, si aumentáramos K el número de detalles posibles disminuirá, pero tendríamos mayor estadística. En el caso de nuestro problema el cual es prácticamente lineal no necesitamos

resolver detalles complejos por lo que podemos aumentar el valor de K de 20 (valor por defecto) a 200 para contar con mayor estadística.

Para el caso del algoritmo MLPBNN hay tres parámetros posibles a optimizar. El primero es la cantidad de neuronas, y su distribución en capas, el segundo es el número de ciclos de entrenamiento y el tercero es método de entrenamiento.

En nuestro caso el problema a resolver es lineal por lo cual debe bastar con una única capa oculta. Se realizo el entrenamiento con una capa oculta y los parámetros por defecto del método y se obtuvo el diagrama de pesos de la figura 19.

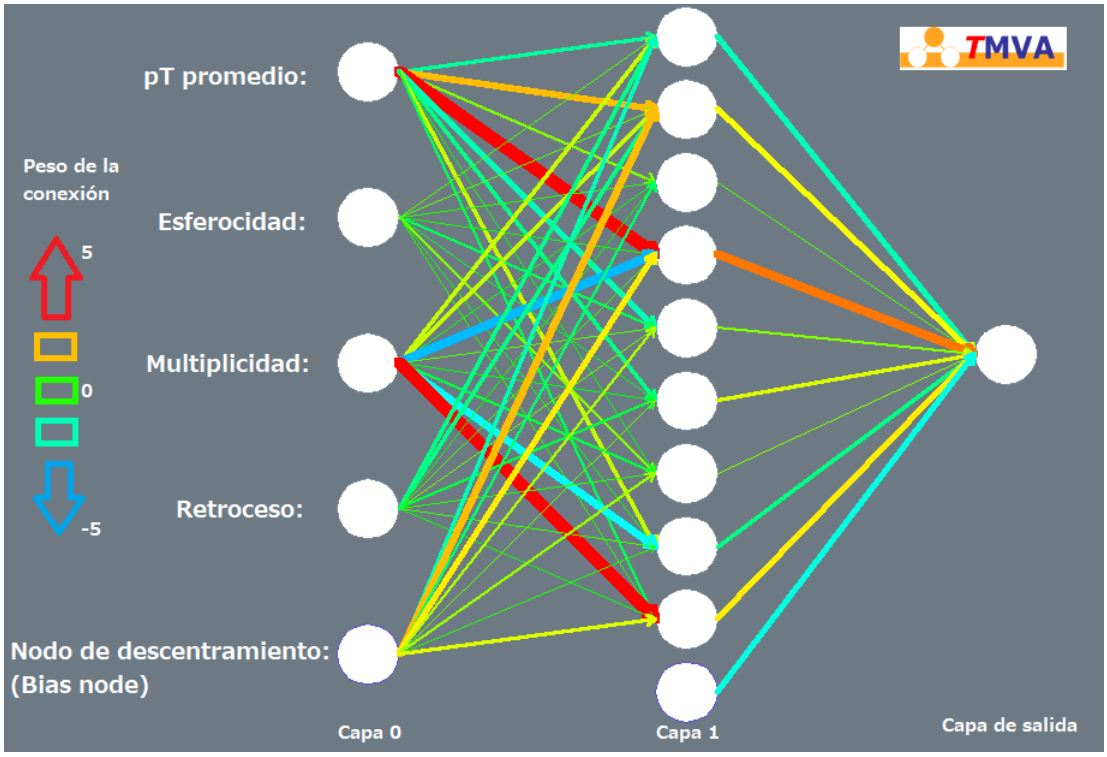


Figura 19: Diagrama ilustrando las conexiones y pesos obtenidos para las diferentes neuronas en el método MLPBNN.

Es posible observar en la figura 19 que la tercera y séptima neurona de la capa 1 cuentan con pesos tanto de entrada como de salida significativamente menores en valor absoluto al del resto de sus compañeras. Por lo que es posible eliminarlas afectando solamente de forma mínima la red y sus predicciones, debido al bajo peso de estas en comparación a las otras neuronas de la red.

En cuanto a la optimización del número de ciclos de entrenamiento TMVA provee un auto regulador, el cual cada cierto número de iteraciones de entrenamiento realiza una prueba de sobre entrenamiento y determina si es prudente continuar o si se debe detener. El número máximo de iteraciones que estamos permitiendo es 600, sin embargo, puede que la red entre a sobre entrenamiento mucho antes de este número en cuyo caso el auto regulador detendría el entrenamiento. La prueba de sobre entrenamiento se está realizando cada 5 ciclos.

Finalmente, el método de entrenamiento es determinante para la velocidad de convergencia y número de ciclos necesarios para ella. Para redes pequeñas como la nuestra el método recomendado en TMVA es BFGS, por lo que es el que se usará.

5.7. Selección del punto de corte

Cada uno de los métodos de clasificación devuelve un número como respuesta esta variable de salida presenta una separación igual o mayor a la de cualquiera de las variables de entrada del método, sin embargo, no es perfecta, por lo que requiere determinar el mejor punto de corte de acuerdo a nuestras necesidades.

5.7.1. Máxima significancia

La significancia se define de la siguiente forma:

$$S = \frac{N_S}{\sqrt{N_S + N_B}} \tag{8}$$

Donde N_S es el número de eventos de señal y N_B es el número de eventos de fondo que pasan un corte dado en forma de señal. En otras palabras, es el número de verdaderos positivos dividido entre la raíz cuadrada del total de positivos.

La motivación subyacente para usar este corte es que queremos ser capaces de minimizar cualquier indicación incorrecta en la señal debida a fluctuaciones estadísticas en la muestra. Como resultado si queremos calcular un valor numérico para S, podemos decir que la significancia esperada para un corte dado es S_σ, asumiendo que el denominador corresponde a una incertidumbre gaussiana en el total de eventos observados.

5.7.2. Máxima pureza

En casos donde se desee reducir el valor de la contribución por parte del fondo en la señal, sin importar la eficiencia del proceso de selección, se puede ocupar el criterio de pureza.

La pureza se define de la siguiente forma:

$$Pureza = \frac{N_{S^+}}{N_{S^+} + N_{S^-}} \tag{9}$$

Donde N_{S^+} es el número de verdaderos positivos y N_{S^-} es el número de falsos positivos.

Es necesario indicar que este criterio puede conducir a cortes en valores de extrema baja eficiencia y por lo tanto incertidumbre estadística alta en el propio valor de la pureza.

5.7.3. Criterio utilizado

Es importante notar que el corte en $N_{MPI} > 15$ es un corte arbitrario sobre una variable discreta y los criterios usuales de clasificación binaria como pureza, eficiencia y significancia aunque útiles solo fueron diseñados para ver en blanco y negro (señal o fondo), continuaremos usándolos pues se decidió utilizar el aprendizaje de máquina como clasificador, sin embargo es posible utilizar métodos de regresión y criterios como el error cuadrado promedio o error absoluto promedio los cuales consideran la distancia entre las predicciones y los observados.

Se ocupó el criterio de máxima significancia, pues se desea que los métodos sean lo más consistentes posibles es decir que las distribuciones de N_{MPI} y otras propiedades no cambien significativamente entre muestra y muestra de eventos.

El criterio de pureza no se ocupó debido a varias razones, la primera es que alta pureza esta usualmente asociada a baja eficiencia y por lo tanto baja estadística lo cual a su vez conlleva a una alta incertidumbre en el mismo valor de la pureza. Este problema se ve agravado con la baja abundancia relativa entre señal y fondo de los eventos de alto N_{MPI} la cual es de 24781 eventos de fondo por cada 1000 de señal.

La segunda razón para no ocupar la pureza es que los eventos seleccionados como señal por los métodos se encuentran distribuidos en N_{MPI} como en la Figura 20 .

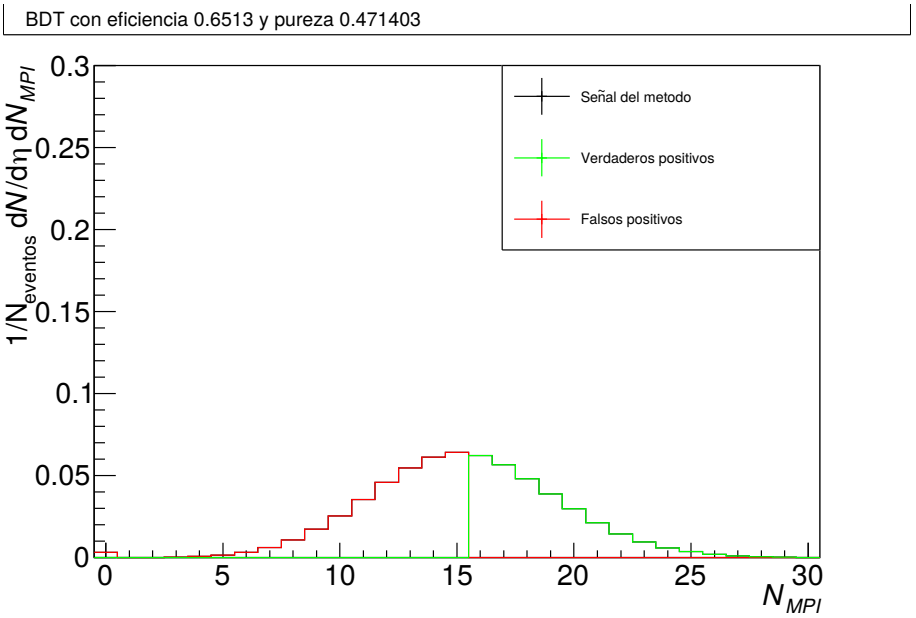
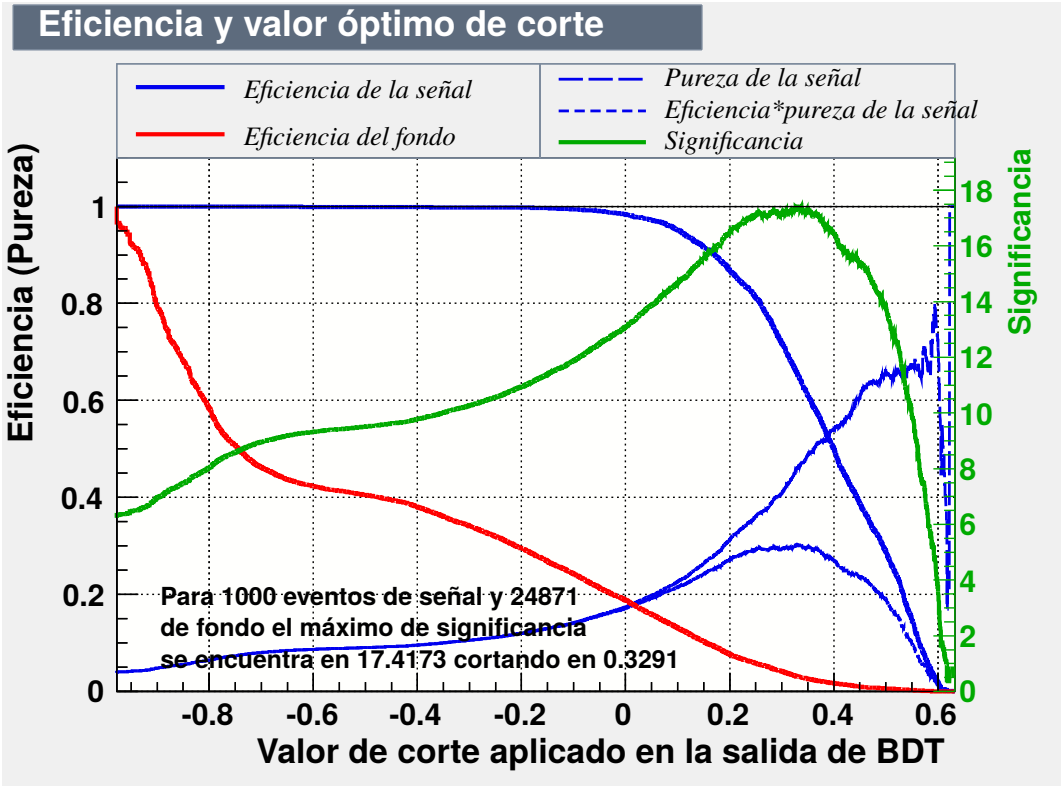


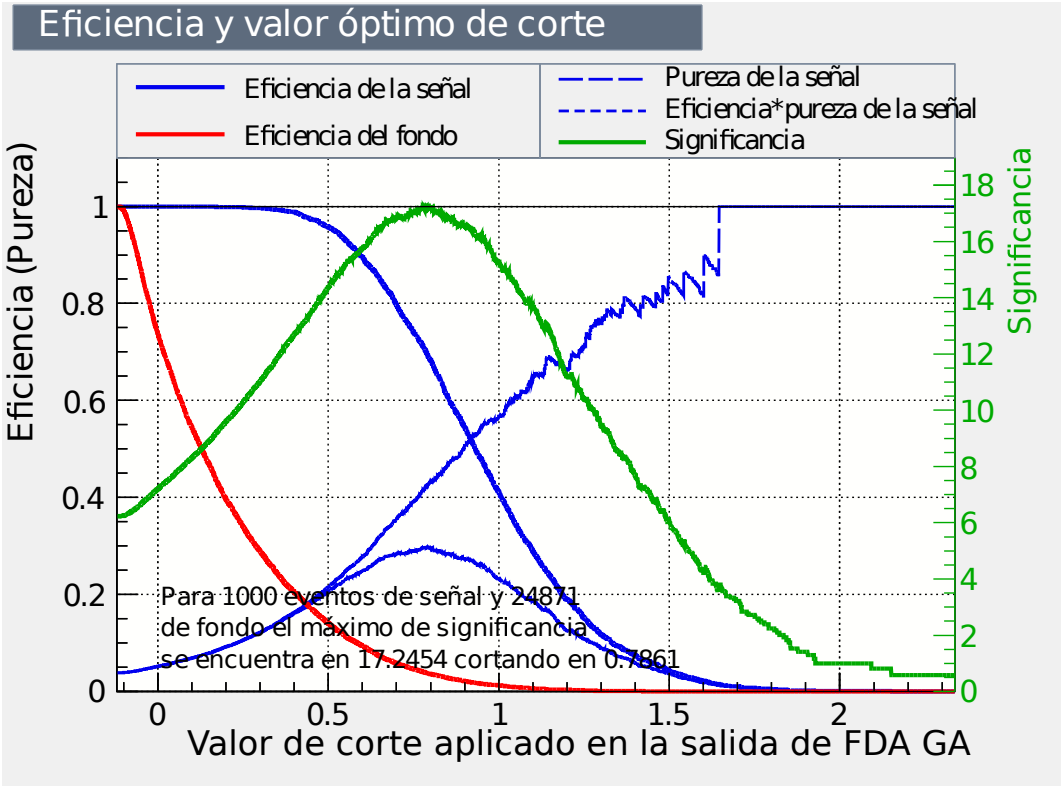
Figura 20: Distribución correspondiente al método BDT con corte en máxima significancia. La forma de la distribución de N_{MPI} es similar para otros cortes en ξ, el cambio principal al cambiar de corte es en el promedio y la desviación estándar de la distribución.

Se puede observar que el valor de la pureza en las distribuciones está relacionado principalmente a la diferencia entre el valor promedio de la distribución y el valor de corte arbitrario que impusimos $N_{MPI} > 15$. Si bien es posible obtener distribuciones con N_{MPI} promedio significativamente mayor a 15 al pedir alta pureza, es preferible aumentar el valor de corte en N_{MPI} de entrenamiento y repetir el entrenamiento para garantizar un desempeño óptimo de los métodos en ese valor de N_{MPI} deseado.

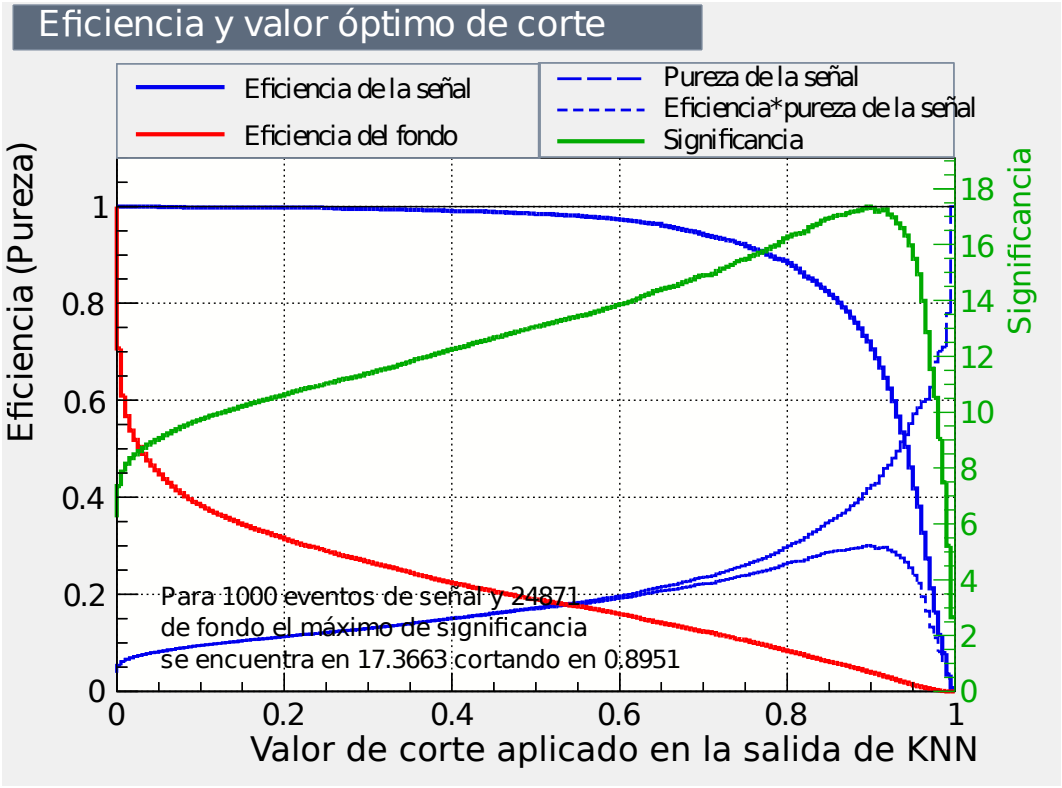
En la figura 21 se muestran las curvas de pureza, eficiencia y significancia de cada uno de los métodos ya optimizados, así como el punto de corte sugerido en base el criterio de máxima significancia.


Eficiencia y valor óptimo de corte
Eficiencia de la señal
Eficiencia del fondo
Pureza de la señal
Eficiencia*pureza de la señal
Significancia
Eficiencia (Pureza)
Significancia
Para 1000 eventos de señal y 24871
de fondo el máximo de significancia
se encuentra en 17.4173 cortando en 0.3291
Valor de corte aplicado en la salida de BDT

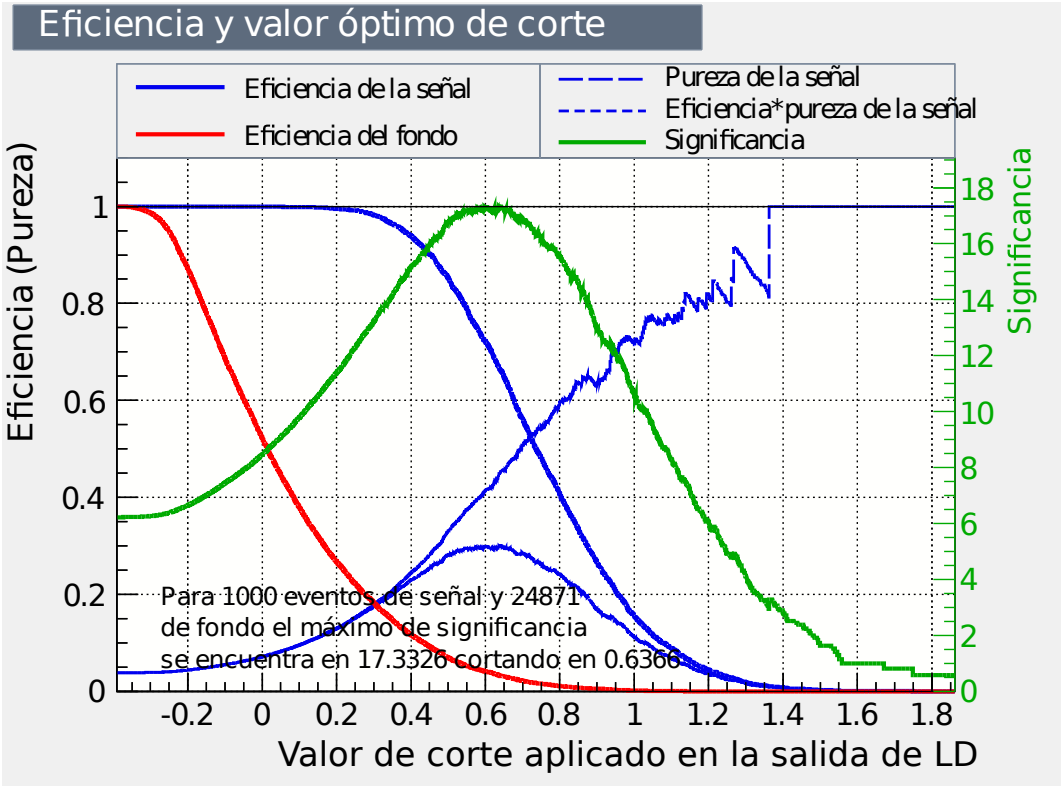
(a) BDT


Eficiencia y valor óptimo de corte
Eficiencia de la señal
Eficiencia del fondo
Pureza de la señal
Eficiencia*pureza de la señal
Significancia
Eficiencia (Pureza)
Significancia
Para 1000 eventos de señal y 24871
de fondo el máximo de significancia
se encuentra en 17.2454 cortando en 0.7861
Valor de corte aplicado en la salida de FDA GA

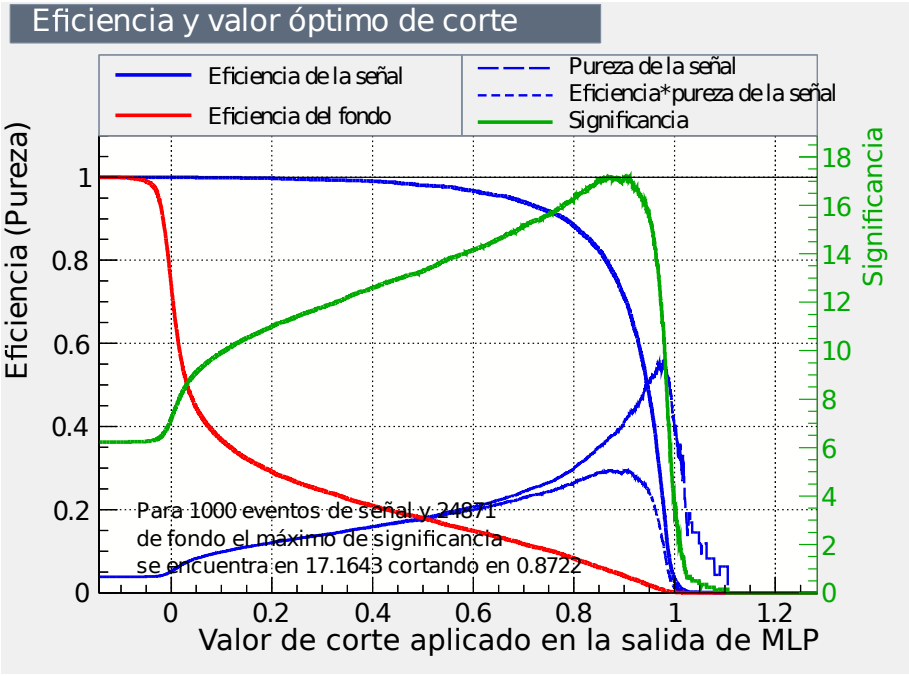
(b) FDA


Eficiencia y valor óptimo de corte
Eficiencia de la señal
Eficiencia del fondo
Pureza de la señal
Eficiencia*pureza de la señal
Significancia
Eficiencia (Pureza)
Significancia
Para 1000 eventos de señal y 24871
de fondo el máximo de significancia
se encuentra en 17.3663 cortando en 0.8951
Valor de corte aplicado en la salida de KNN

(c) KNN


Eficiencia y valor óptimo de corte
Eficiencia de la señal
Eficiencia del fondo
Pureza de la señal
Eficiencia*pureza de la señal
Significancia
Eficiencia (Pureza)
Significancia
Para 1000 eventos de señal y 24871
de fondo el máximo de significancia
se encuentra en 17.3326 cortando en 0.6366
Valor de corte aplicado en la salida de LD

(d) LD



(e) MLPBNN

Figura 21: Curvas de eficiencia, pureza y significancia en función del punto de corte para los 5 métodos entrenados

Los valores de corte usados se encuentran en el Cuadro 3 mostrado a continuación:

Método	**Valor**
FDA_{GA}	0.7861
MLPBNN	0.8722
BDT	0.3291
LD	0.6366
KNN	0.8951

Cuadro 3: Valores de corte aplicados a cada uno de los métodos. El valor de corte ξ_{corte} de cada método es tal que maximice la significancia

5.8. Validación de los algoritmos

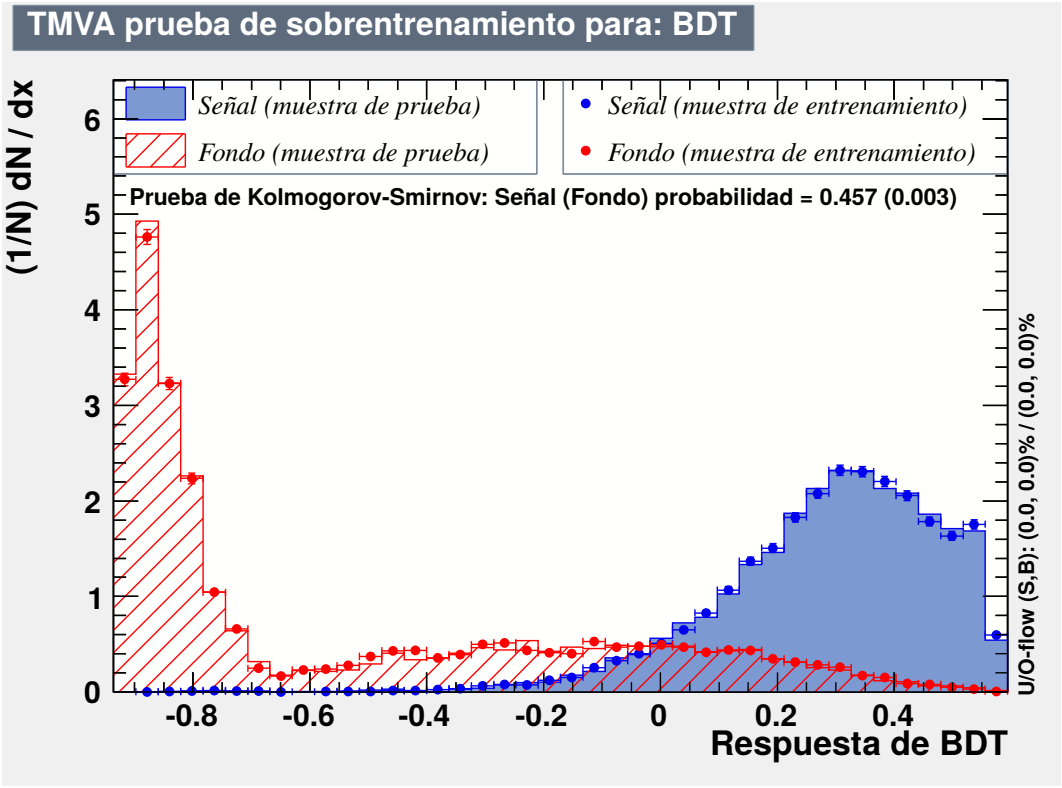
En la sección anterior entrenamos los algoritmos de aprendizaje de máquina, los optimizamos y determinamos un valor de corte sobre la respuesta de cada método. Para este momento ya contamos con un algoritmo aplicable el cual podríamos correr sobre los datos y obtener una clase predicha como señal y otra como fondo, sin embargo, aún no tenemos idea si nuestros algoritmos presentan algún problema serio como sobre entrenamiento o si el modelo generado es inútil fuera del MC con el que fue entrenado. Para descartar estos problemas se deben realizar la prueba de Kolmogórov-Smirnov y la evaluación en generadores distintos al de entrenamiento.

5.8.1. Prueba de sobre entrenamiento

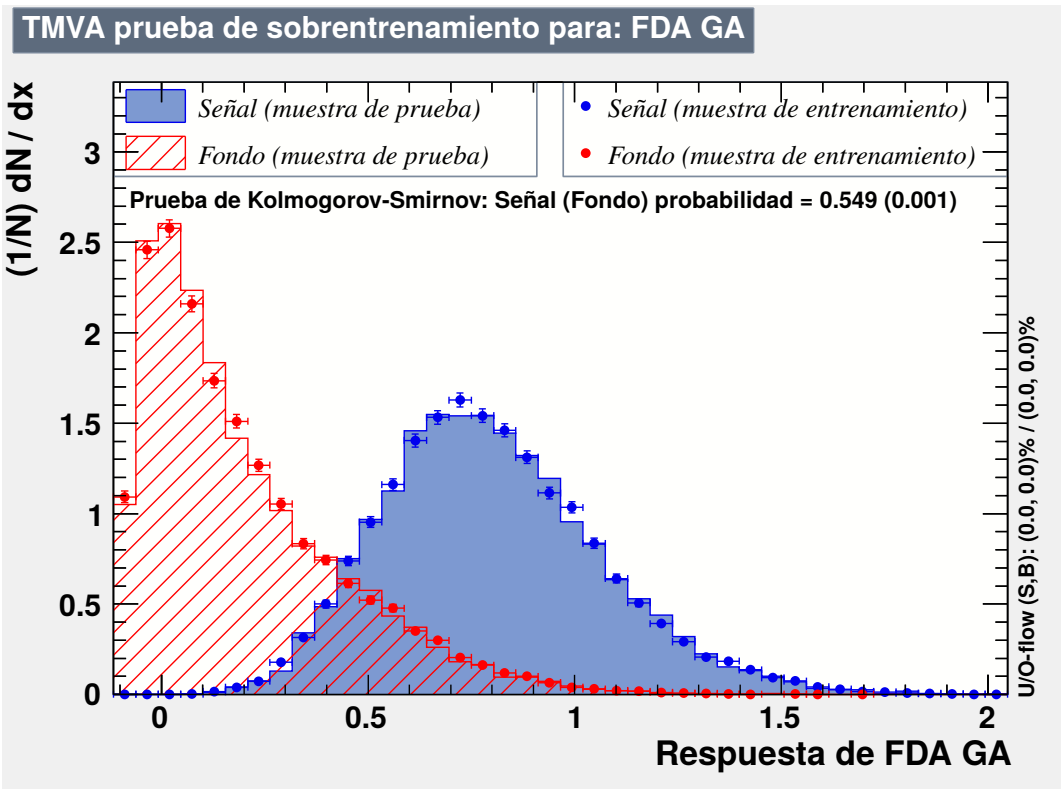
La prueba de sobre entrenamiento consiste en aplicar la prueba Kolmogórov-Smirnov de bondad de ajuste la cual se encuentra descrita en la sección 3.1.1.3.4, la muestra usada para generar las curvas de señal y de fondo

corresponde a 1000 eventos de señal y 1000 de fondo tanto para las muestras de entrenamiento como para la muestra de prueba.

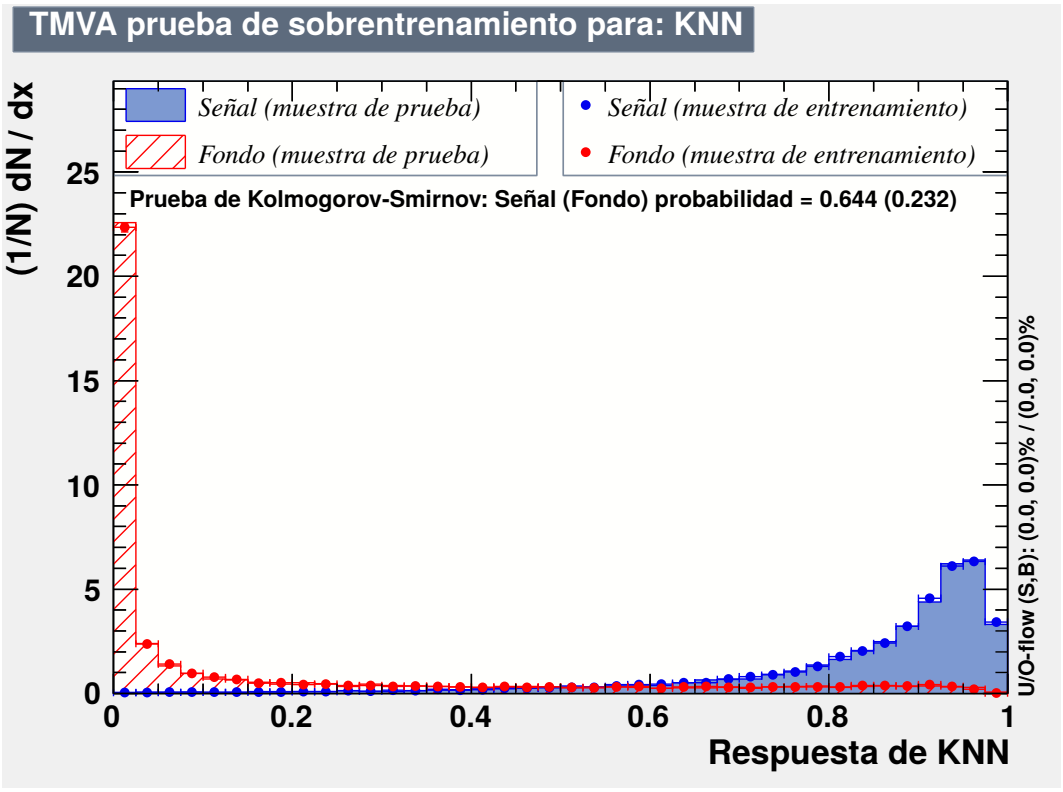
El valor de corte que se está utilizando para definir sobreentrenamiento será 0.001 valores menores se considerará que el método está sobreentrenado, valores entre 0.01 y 0.001 se considerarán parcialmente sobreentrenado y mayores a 0.01 no presentan sobreentrenamiento.



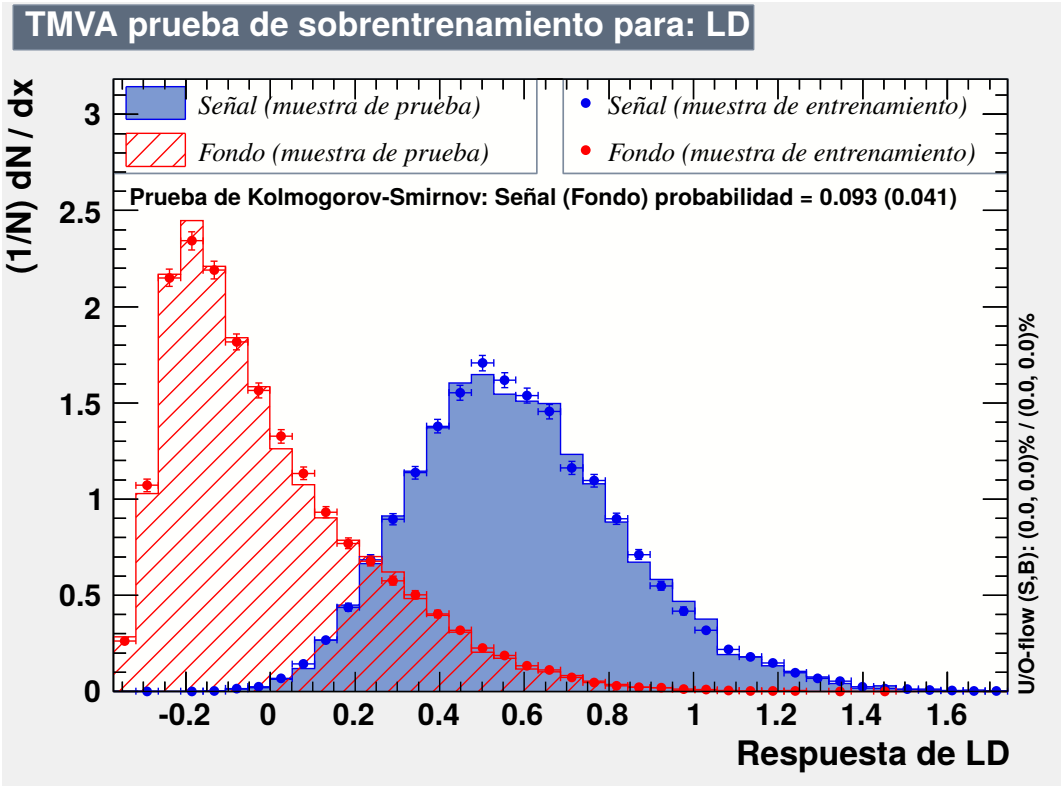
(a) BDT


TMVA prueba de sobrentrenamiento para: FDA GA
Señal (muestra de prueba)
Fondo (muestra de prueba)
Señal (muestra de entrenamiento)
Fondo (muestra de entrenamiento)
Prueba de Kolmogorov-Smirnov: Señal (Fondo) probabilidad = 0.549 (0.001)
(1/N) dN / dx
U/O-flow (S,B): (0.0, 0.0)% / (0.0, 0.0)%
Respuesta de FDA GA

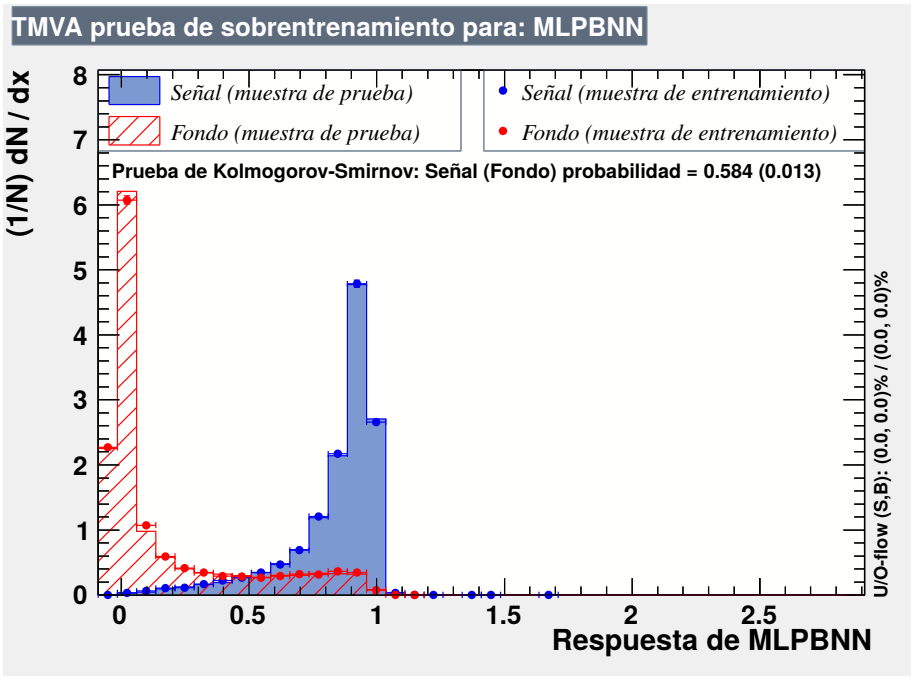
(b) FDA


TMVA prueba de sobrentrenamiento para: KNN
Señal (muestra de prueba)
Fondo (muestra de prueba)
Señal (muestra de entrenamiento)
Fondo (muestra de entrenamiento)
Prueba de Kolmogorov-Smirnov: Señal (Fondo) probabilidad = 0.644 (0.232)
(1/N) dN / dx
U/O-flow (S,B): (0.0, 0.0)% / (0.0, 0.0)%
Respuesta de KNN

(c) KNN



(d) LD



(e) MLPBNN

Figura 22: Distribuciones de las respuestas del método para la muestra de entrenamiento y de prueba, se aplicó la prueba de Kolmogórov-Smirnov para determinar la consistencia entre las dos curvas para los 5 métodos entrenados

5.8.2. Dependencia del modelo Monte Carlo

Las diferencias entre los dos modelos de N_{MPI} correspondientes a Pythia 6 y Pythia 8 son significativas y se puede observar en las curvas de N_{MPI} de la figura 23

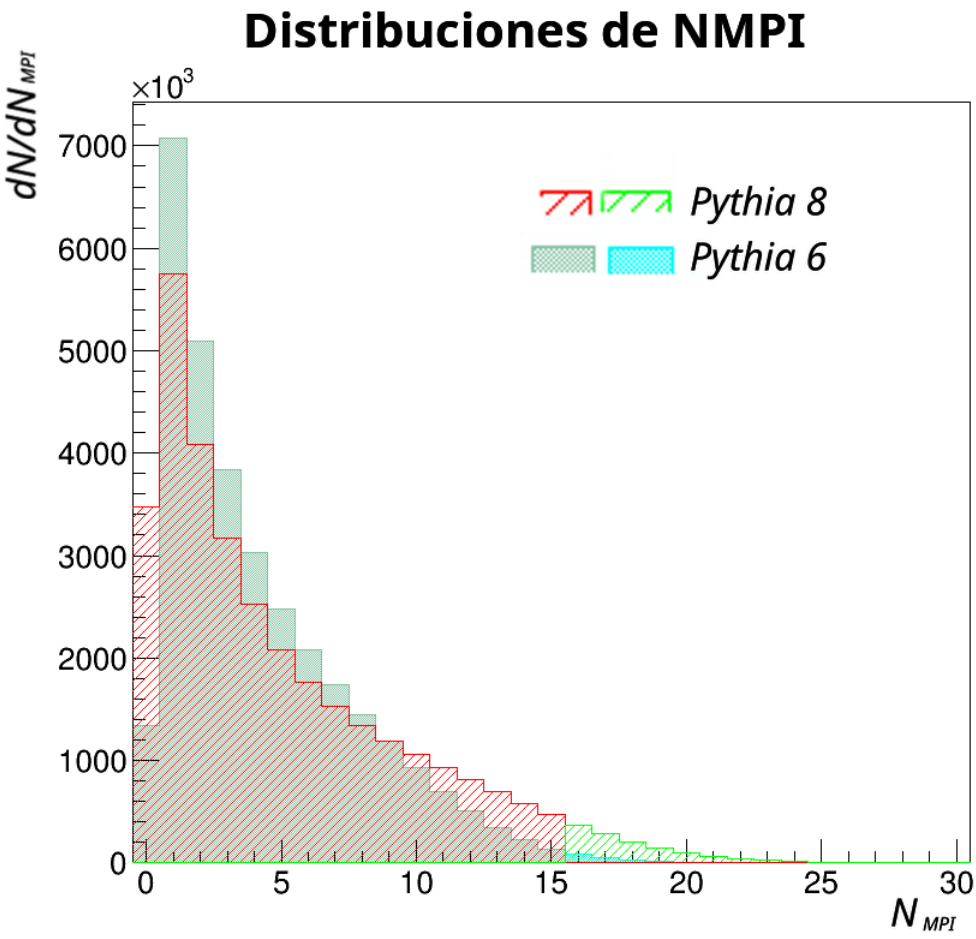
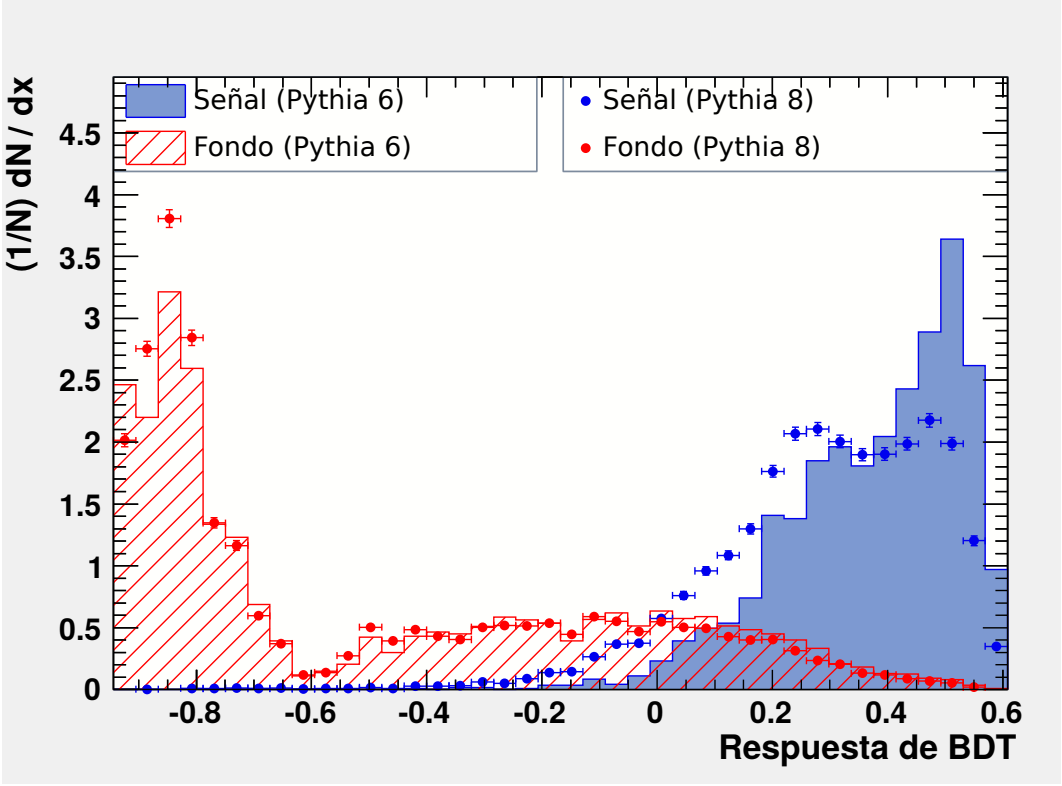
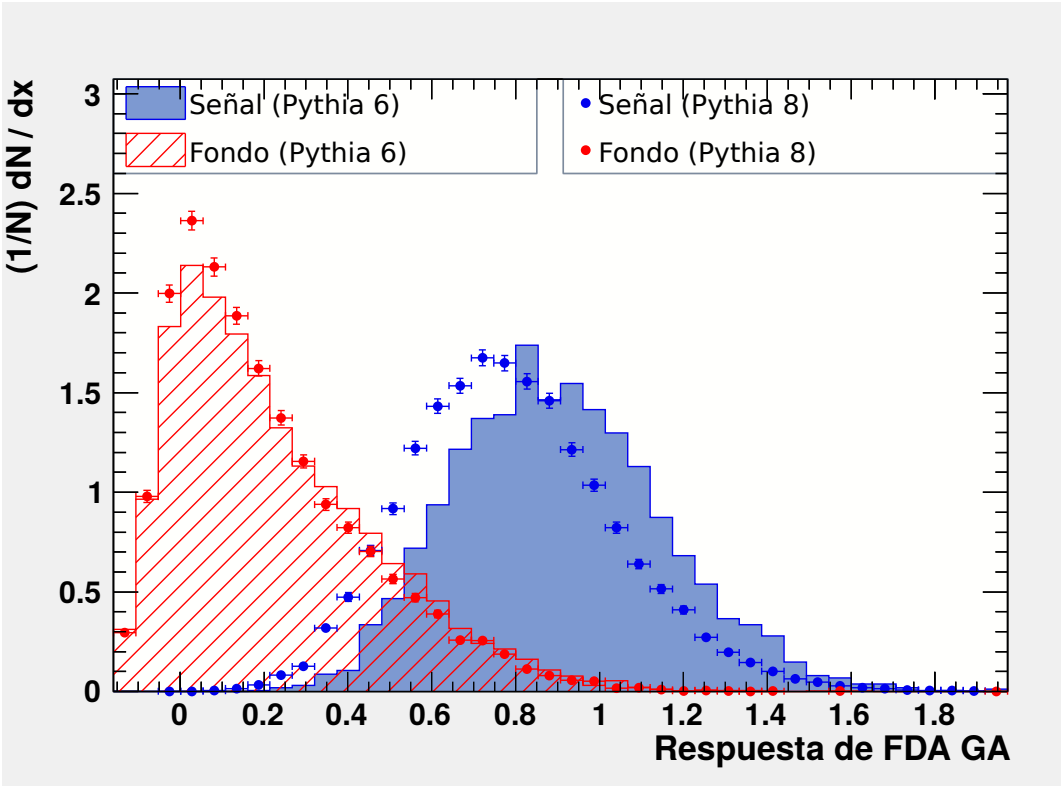


Figura 23: Distribuciones de N_{MPI} en los modelos Pythia 6 y Pythia 8.

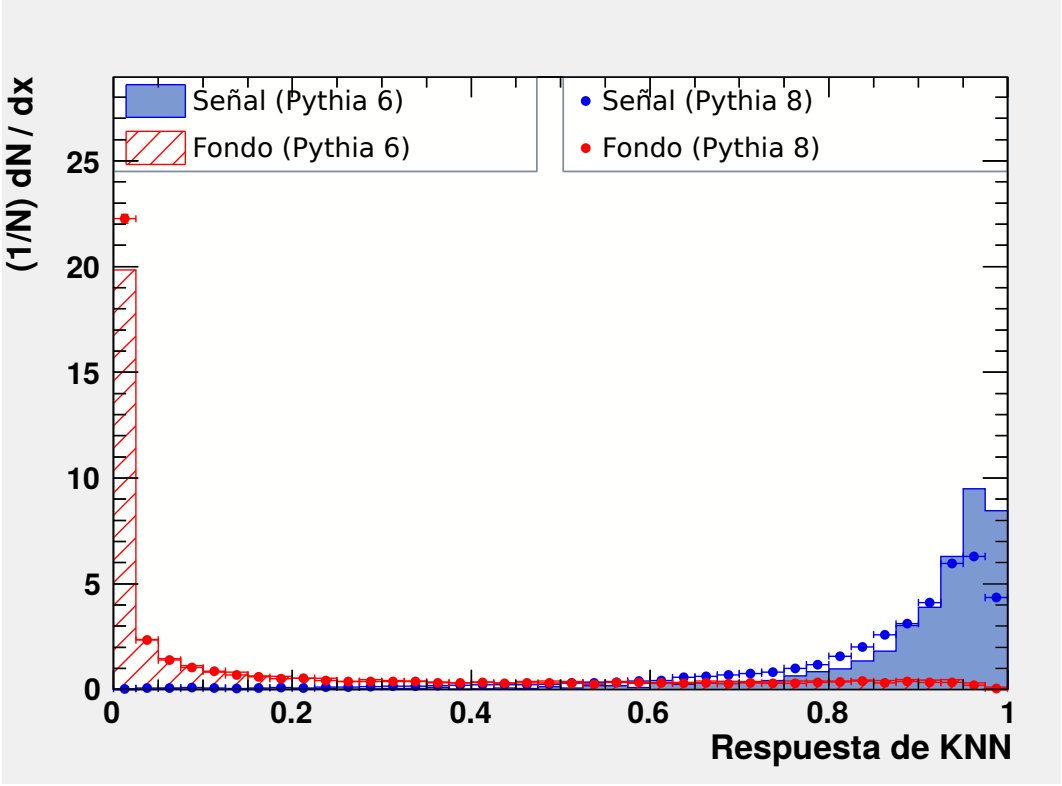
Una de las diferencias principales entre ambos modelos es que Pythia 6 tiende a favorecer más la creación de eventos con menor N_{MPI} que Pythia 8 y por lo tanto el N_{MPI} promedio es menor en Pythia 6. Adicionalmente en Pythia 8 la distribución de N_{MPI} es más ancha lo cual permite la existencia de eventos con $N_{MPI} > 20$ los cuales en Pythia 6 son prácticamente inexistentes.


Señal (Pythia 6)
Fondo (Pythia 6)
Señal (Pythia 8)
Fondo (Pythia 8)
(1/N) dN / dx
Respuesta de BDT

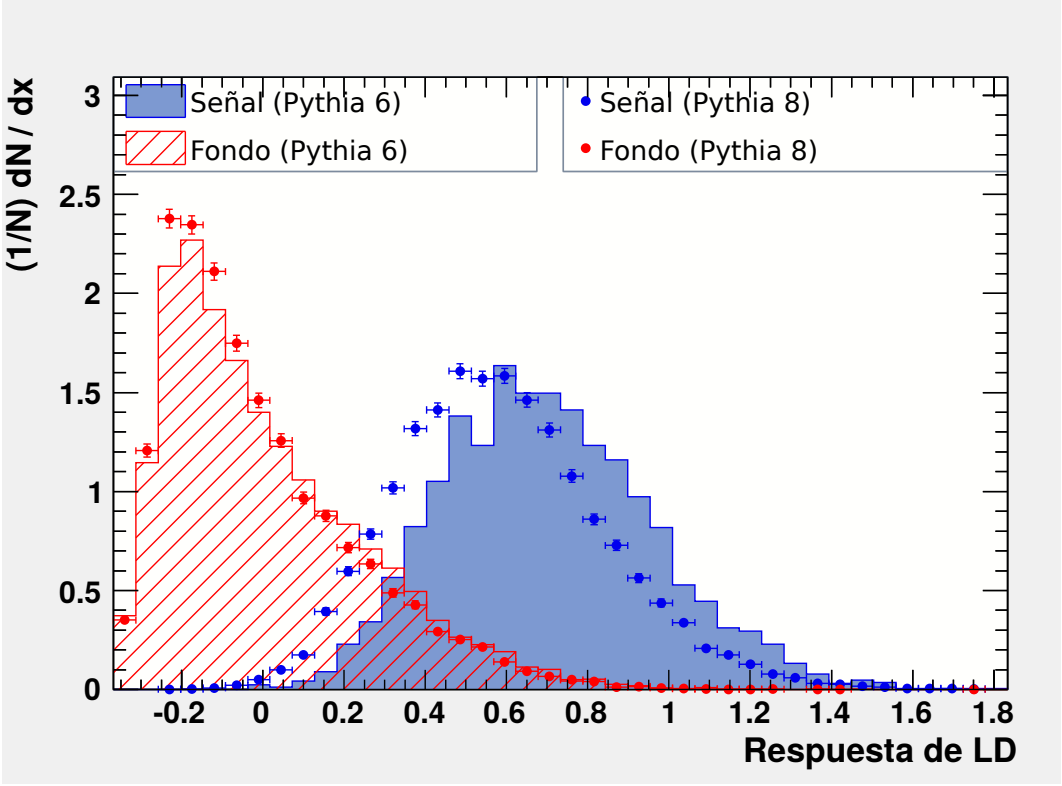
(a) BDT


Señal (Pythia 6)
Fondo (Pythia 6)
Señal (Pythia 8)
Fondo (Pythia 8)
(1/N) dN / dx
Respuesta de FDA GA

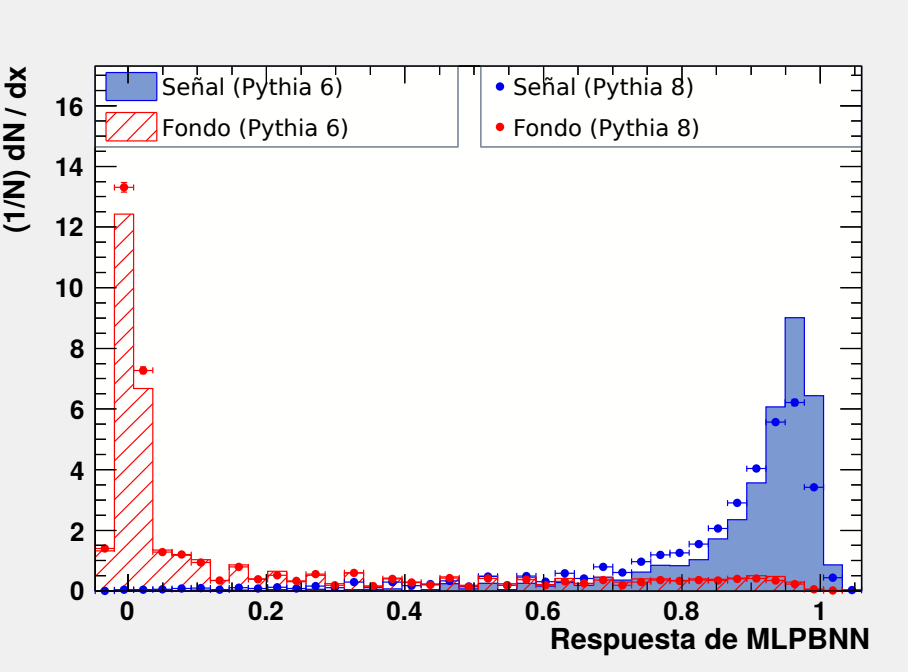
(b) FDA


Señal (Pythia 6)
Fondo (Pythia 6)
Señal (Pythia 8)
Fondo (Pythia 8)
(1/N) dN / dx
Respuesta de KNN

(c) KNN


Señal (Pythia 6)
Fondo (Pythia 6)
Señal (Pythia 8)
Fondo (Pythia 8)
(1/N) dN / dx
Respuesta de LD

(d) LD

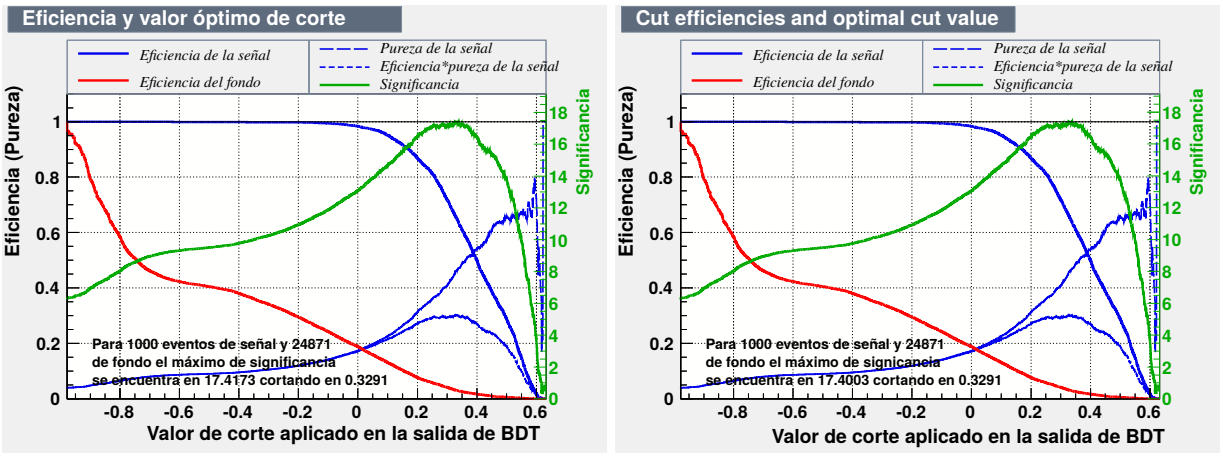


(e) MLPBNN

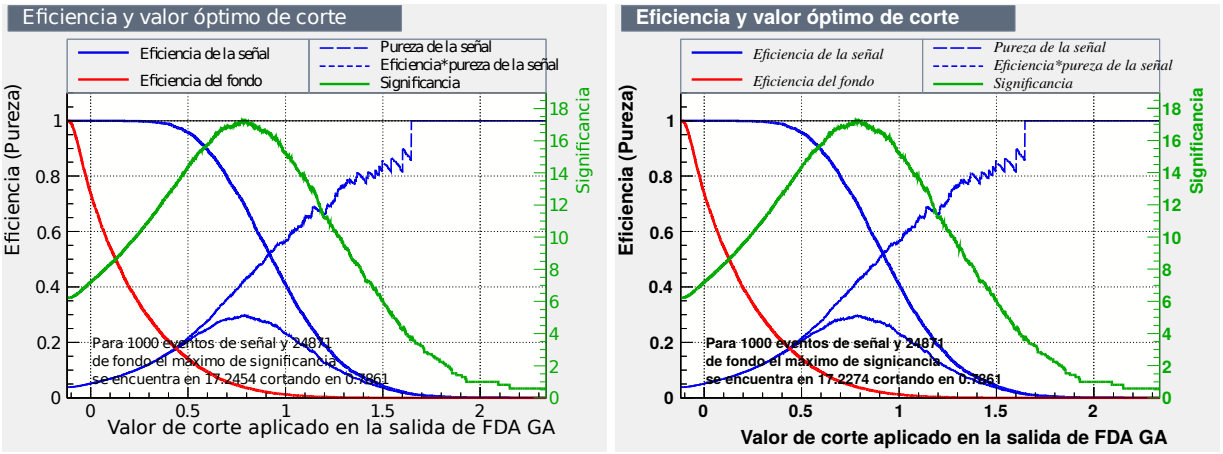
Figura 24: Distribuciones de las respuestas del método para la muestra de entrenamiento en Pythia 8 y para la muestra de prueba en Pythia 6. Las curvas no coinciden pues provienen de generadores distintos.

Debido a la gran diferencia entre los modelos, los eventos con alto N_{MPI} (mayor a 15) son más escasos en Pythia 6 pasando de 24871 eventos de fondo por cada 1000 de señal en Pythia 8 a 185025 eventos de fondo por cada 1000 de señal en Pythia 6 haciendo la labor de los algoritmos de clasificación mucho más difícil en Pythia 6 comparado con Pythia 8.

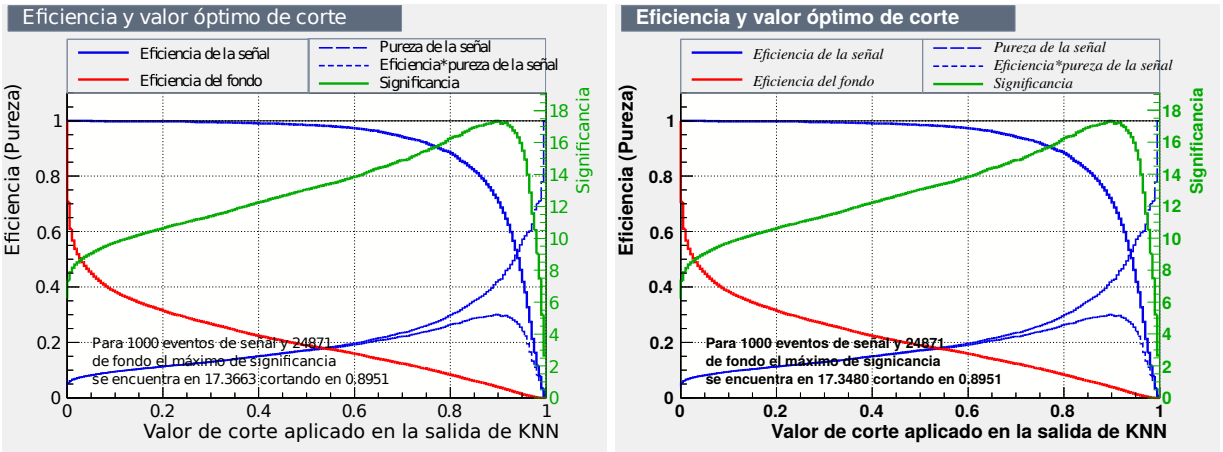
Para poder comparar entre los dos modelos se requiere ocupar un marco común en donde ambos modelos presenten la misma abundancia señal-fondo. Una forma de hacer esto es tomar N eventos de fondo aleatoriamente y M eventos de señal de cada generados donde N/M es el cociente señal/fondo del generador original (del cual ya tenemos las curvas). En nuestro caso en particular se tomaron 1000 eventos de señal y 24871 de fondo de ambos generadores para obtener las curvas de eficiencia, pureza y significancia en ambos modelos y comprobar compatibilidad entre las curvas, las curvas resultantes de eficiencia, pureza y significancia se muestran a continuación en la figura 25.



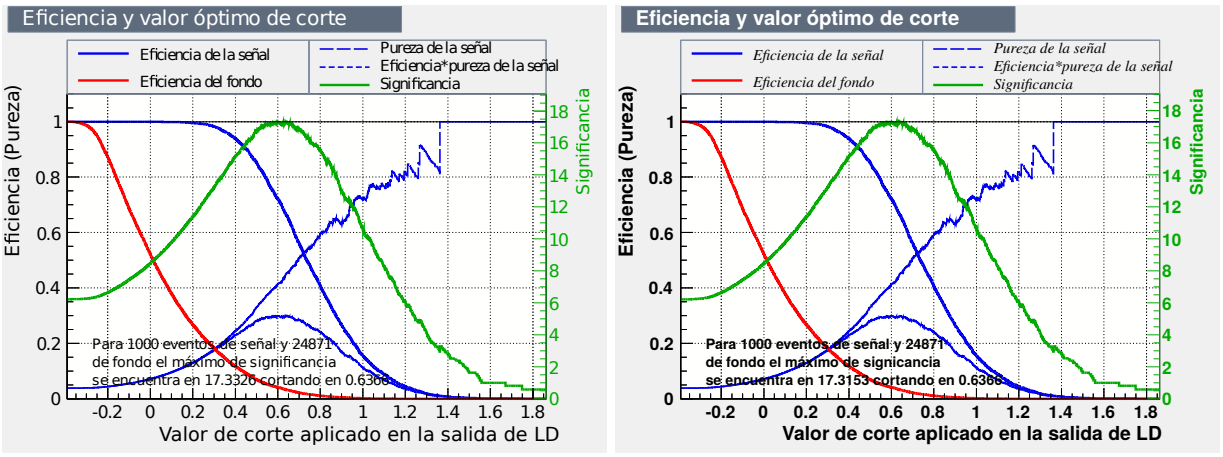
(a) BDT



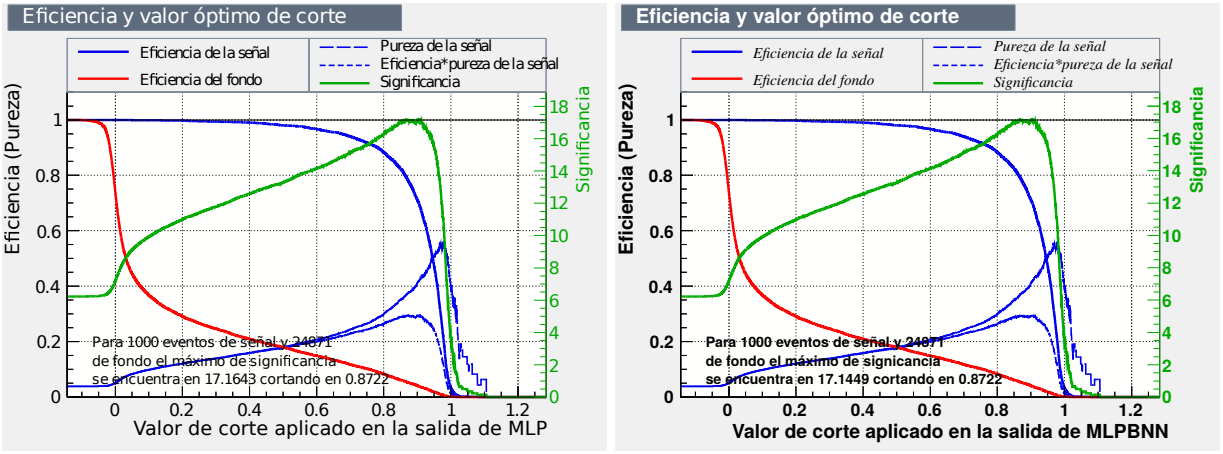
(b) FDA



(c) KNN



(d) LD



(e) MLPBNN

Figura 25: Curvas de eficiencia, pureza y significancia calculadas para una abundancia relativa de 24871 evento des fondo por 1000 de señal en Pythia 8 (izquierda) y Pythia 6 (derecha).

Como se puede observar en las gráficas de la figura 25 las curvas obtenidas en Pythia 8 y Pythia 6 ya corregidas por la diferencia en abundancia entre ambos modelos de N_{MNPI} son prácticamente indistinguibles hasta el punto de tener máxima significancia en el mismo punto de corte en ambas versiones de Pythia. Lo cual indica que el modelo de clasificación generado por el aprendizaje de máquina es independiente del MC utilizado y las reglas aprendidas pueden extrapolarse a otros MC (por lo menos de Pythia 8 a Pythia 6).

5.9. Separaciones obtenidas

5.9.1. Número de múltiples interacciones partónicas

Para la clase de señal predicha por cada uno de los métodos incluyendo el método tradicional se graficó la distribución de N_{MPI}. Las curvas corresponden a los generadores Pythia 8 (mismo que en entrenamiento) a la izquierda y Pythia 6 (otro modelo de N_{MPI} distinto) a la derecha, el número de eventos señal-fondo no fue ajustado y corresponde a la abundancia natural del generador. Note que en Pythia 8 la abundancia es 24:1 mientras que en Pythia 6 es de 185:1 lo cual hace a Pythia 6 un modelo más difícil de trabajar.

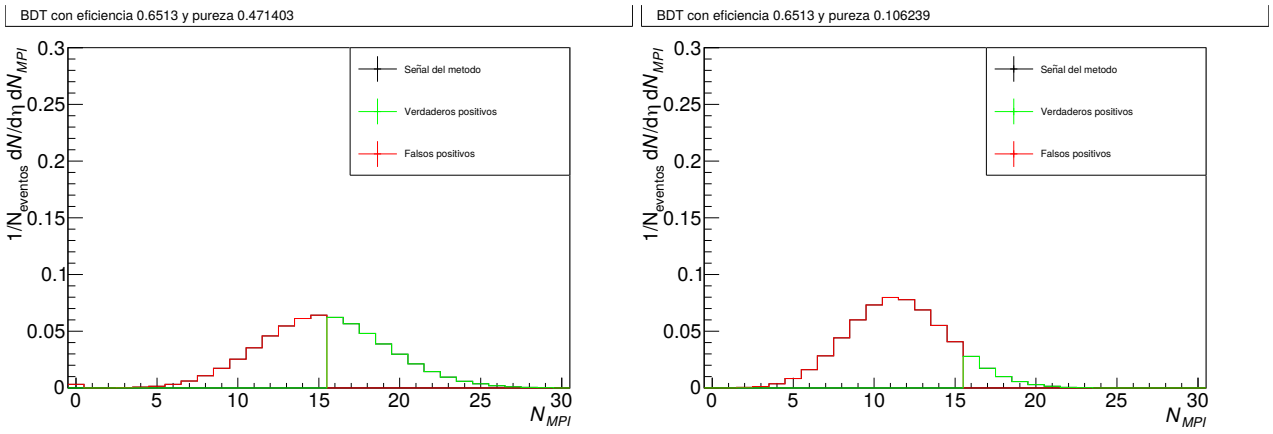
Los valores de N_{MPI} promedio y eficiencia de las siguientes distribuciones generadas por los algoritmos de clasificación son:

	Eficiencia	**Pureza**	**N_{MPI} promedio**
BDT	0.464	0.471	15.25
LD	0.478	0.458	15.11
MLPBNN	0.611	0.396	14.44
FDA_{GA}	0.520	0.425	14.78
KNN	0.532	0.435	14.89
Método tradicional	0.182	0.444	15.01

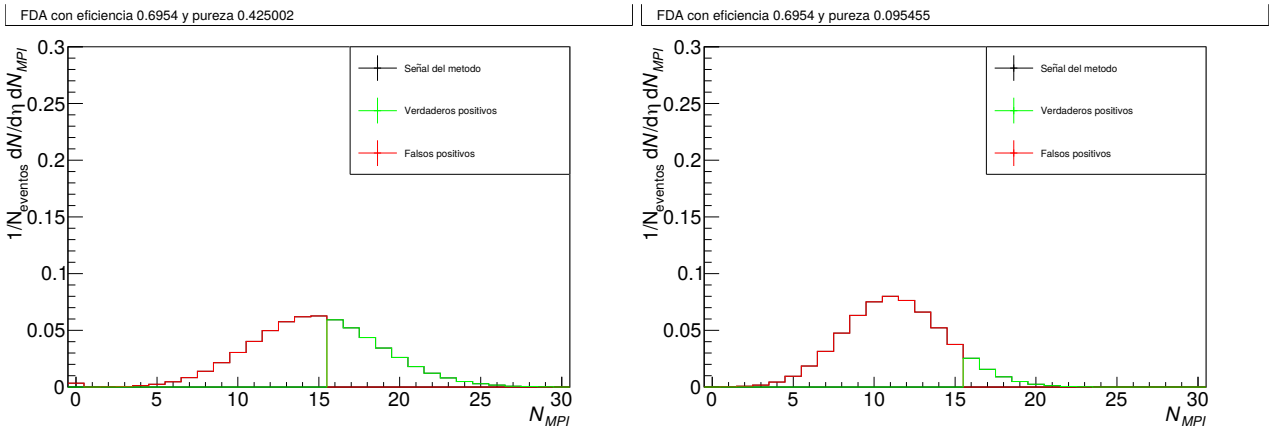
Cuadro 4: Resultados para Pythia 8

	Eficiencia	**Pureza**	**N_{MPI} promedio**
BDT	0.651	0.106	11.55
LD	0.663	0.106	11.55
MLPBNN	0.782	0.078	10.89
FDA_{GA}	0.69	0.095	11.34
KNN	0.716	0.092	11.27
Método tradicional	0.270	0.105	11.55

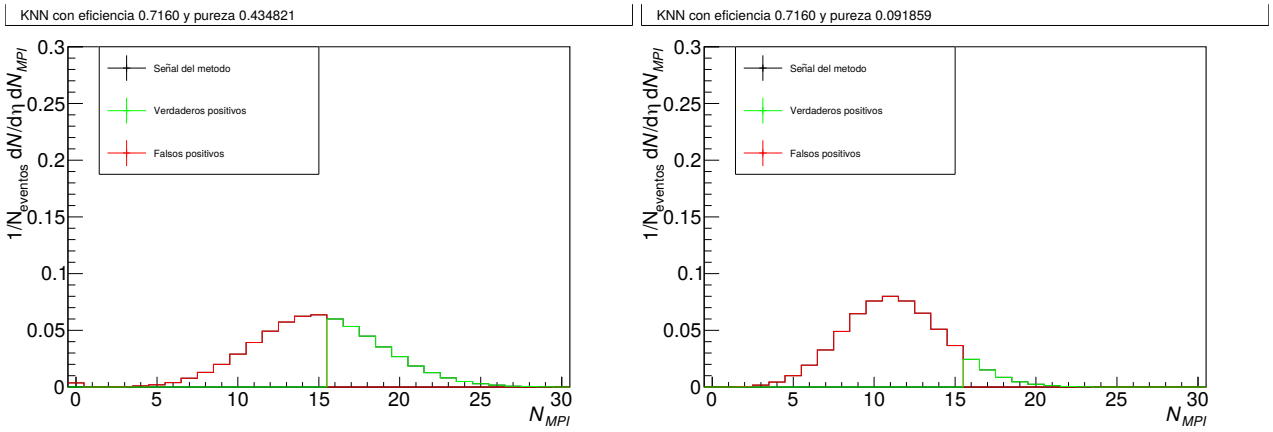
Cuadro 5: Resultados para Pythia 6



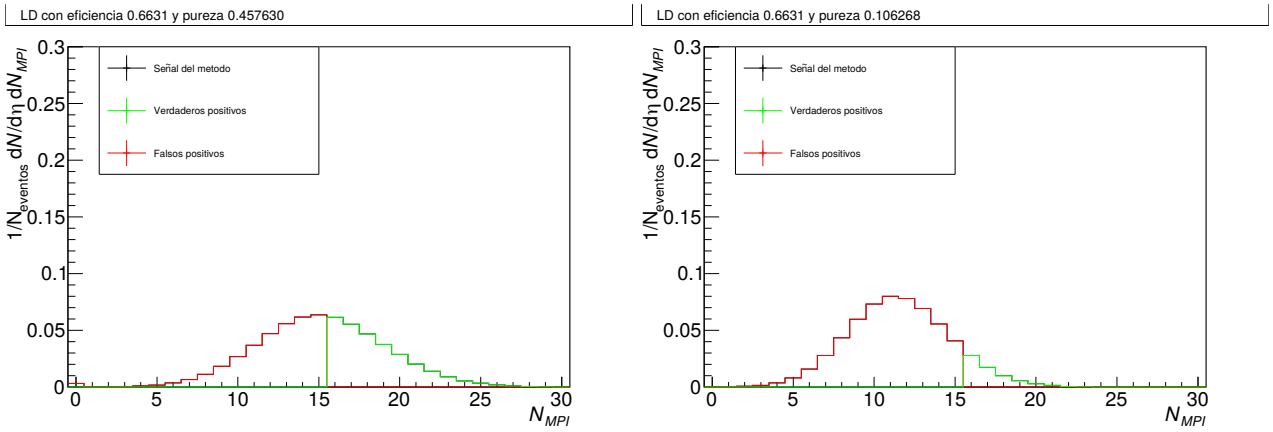
(a) BDT



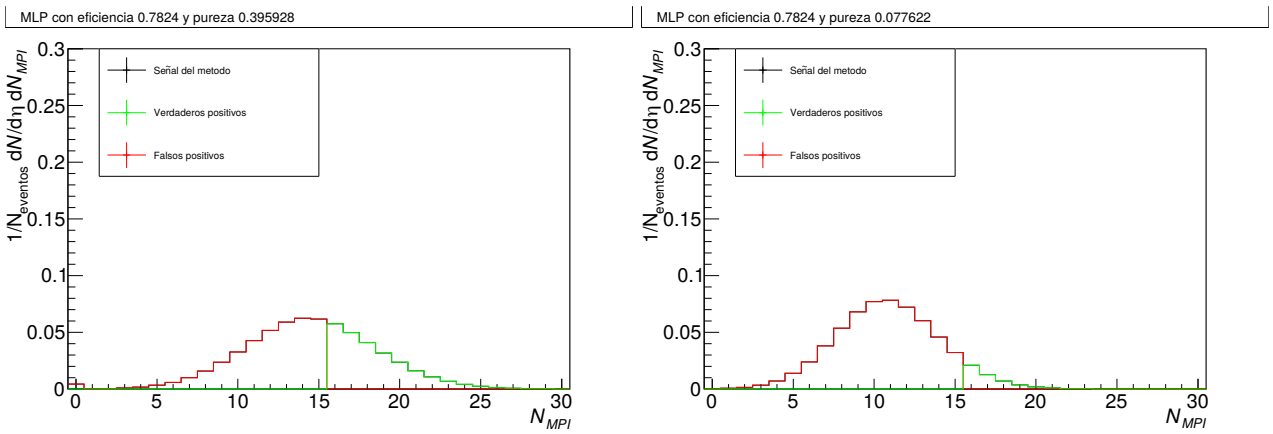
(b) FDA



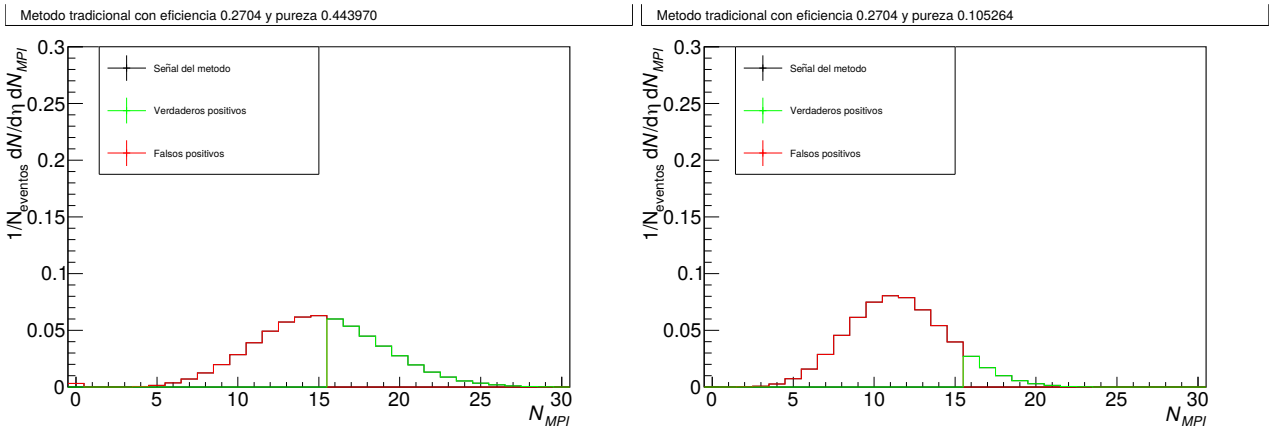
(c) KNN



(d) LD



(e) MLPBNN



(f) Método tradicional

Figura 26: Distribuciones de N_{MPI} de los eventos clasificados como señal por cada uno de los 5 métodos de ML y el método tradicional. Se muestra las distribuciones de Pythia 8 (MC de entrenamiento) a la izquierda y Pythia 6 (MC con modelo de N_{MPI} distinto) a la derecha

5.9.2. Esferocidad verdadera

En el método tradicional se corta sobre la esferocidad reconstruida induciendo un sesgo en esta variable. Podemos observar en la figura 27 la distribución de esferocidad reconstruida en eventos de $N_{MPI} > 15$. Como se puede observar también hay algunos eventos de baja esferocidad pertenecientes a alto N_{MPI} los cuales se ven eliminados al aplicar el corte tradicional.

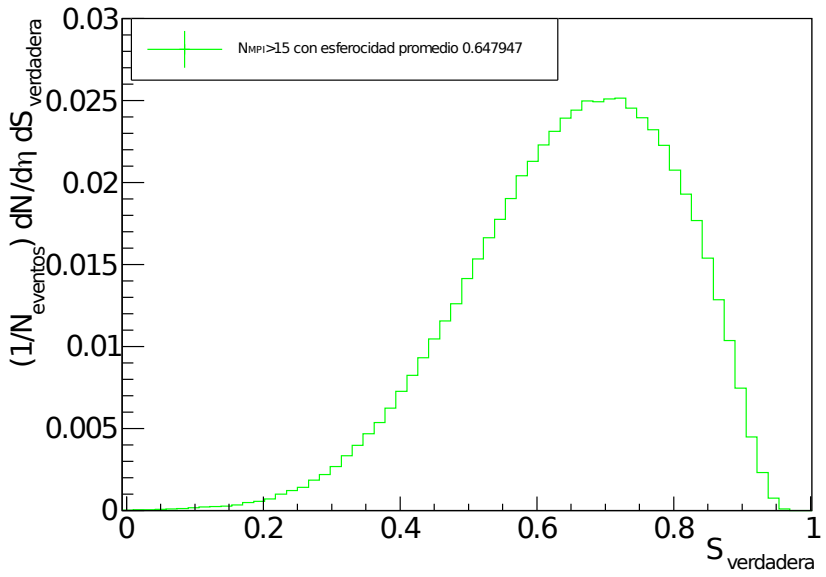
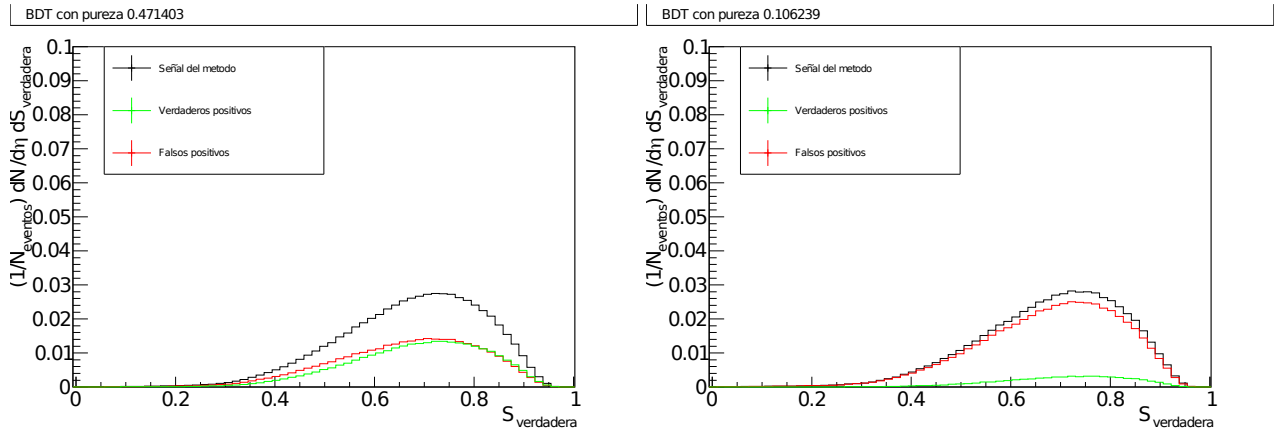
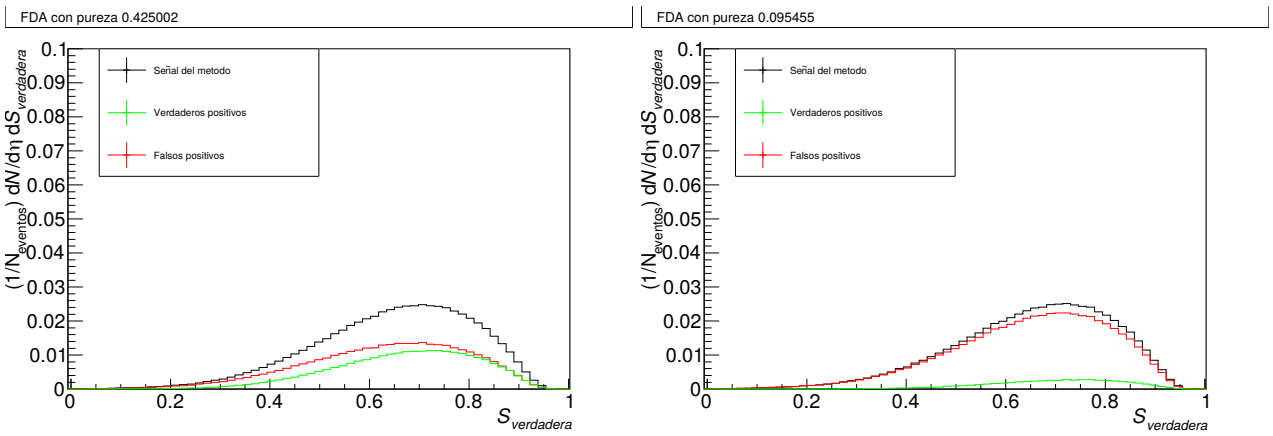


Figura 27: Distribución de esferocidad verdadera en los eventos con $N_{MPI} > 15$ a nivel generador

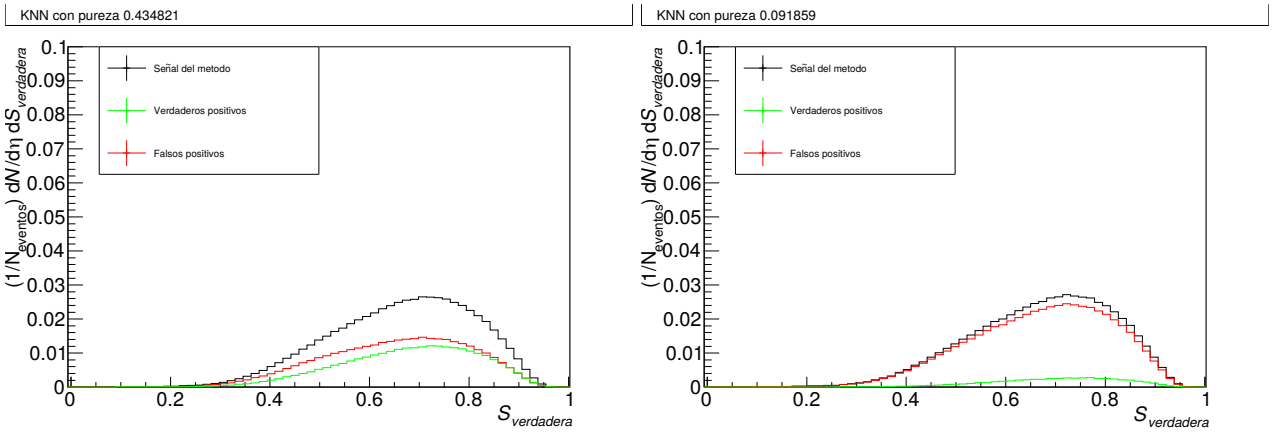
Se puede observar en la figura 26 las distribuciones de esferocidad para los métodos entrenados y para el método tradicional y comparar con las distribuciones de esferocidad para eventos con $N_{MPI} > 15$. A diferencia del método tradicional se puede observar que los métodos de aprendizaje de máquina permiten eventos de baja esferocidad con alto N_{MPI}, esta región de baja esferocidad es la principal causante del aumento de eficiencia observado en la clasificación por aprendizaje de máquina.



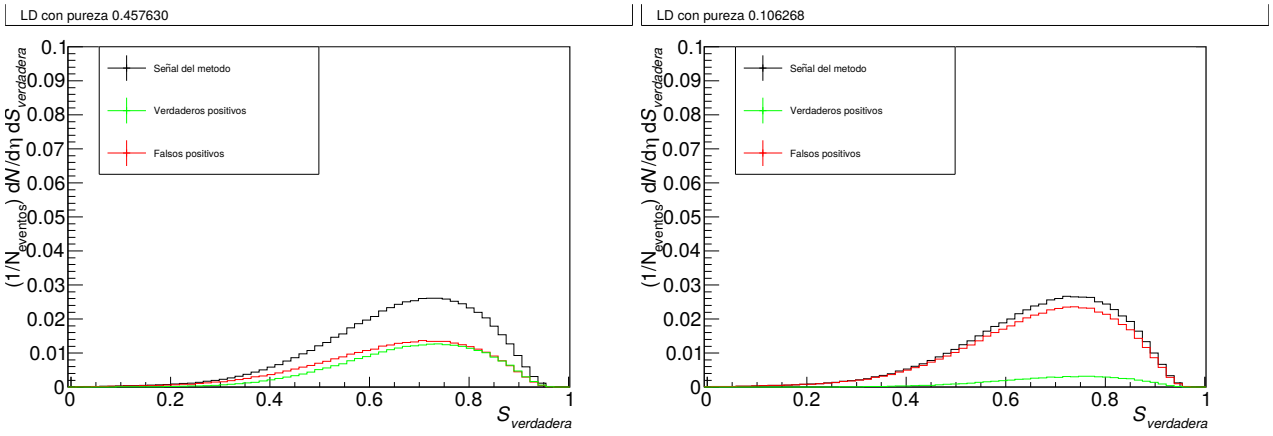
(a) BDT



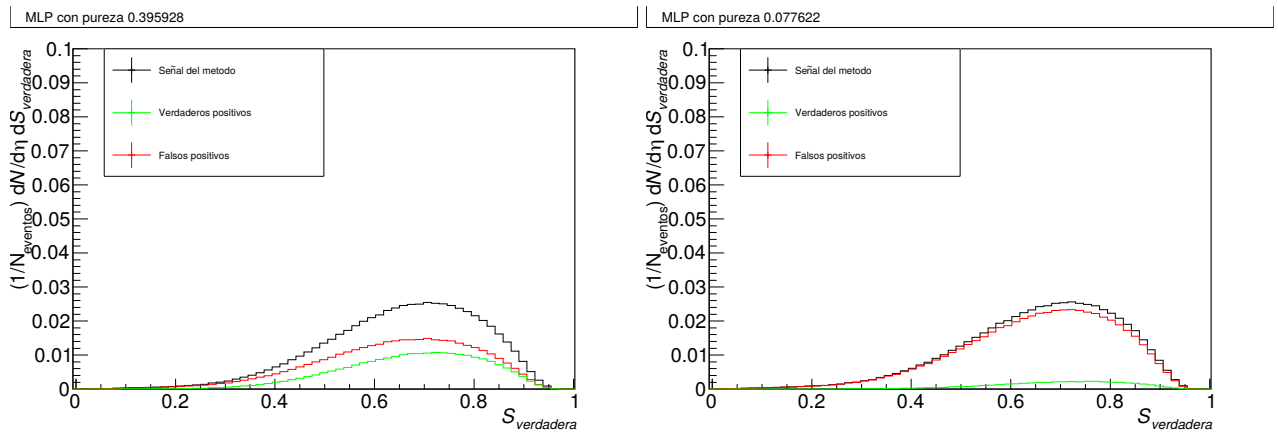
(b) FDA



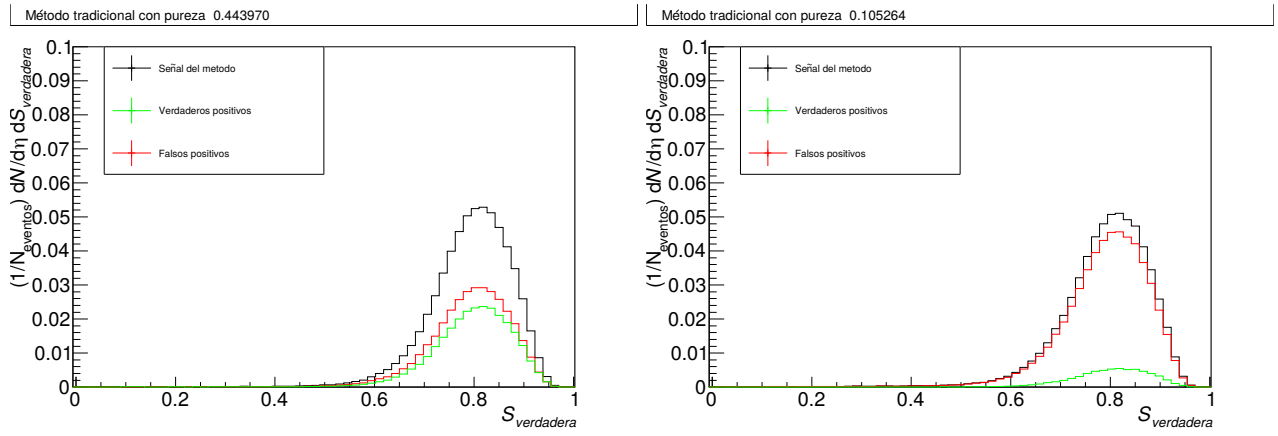
(c) KNN



(d) LD



(e) MLPBNN



(f) Método tradicional

Figura 28: Distribuciones de esferocidad de los eventos clasificados como señal por cada uno de los 5 métodos de ML y el método tradicional. Se muestra las distribuciones de Pythia 8 (MC de entrenamiento) a la izquierda y Pythia 6 (MC con modelo de N_{MPI} distinto) a la derecha

5.10. Comprobación de resultados mediante cociente protón-pion

Una forma de comprobar nuestra selección de eventos es recurrir a los cocientes $\frac{p}{\pi}$ ya que a diferencia del cociente en los jets donde $\frac{p}{\pi}$ escala en función del p_{T} (figura 29), en el evento subyacente el cociente se satura y se vuelve plano en función del p_{T} para p_{T}s altos (figura 30),.

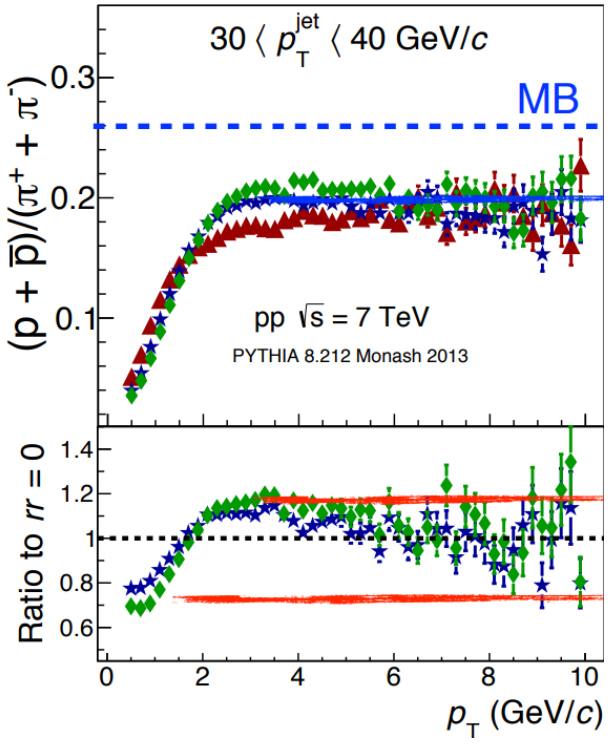


Figura 29: Cociente p/π en el UE (Pythia 8). Obtenido de [39]

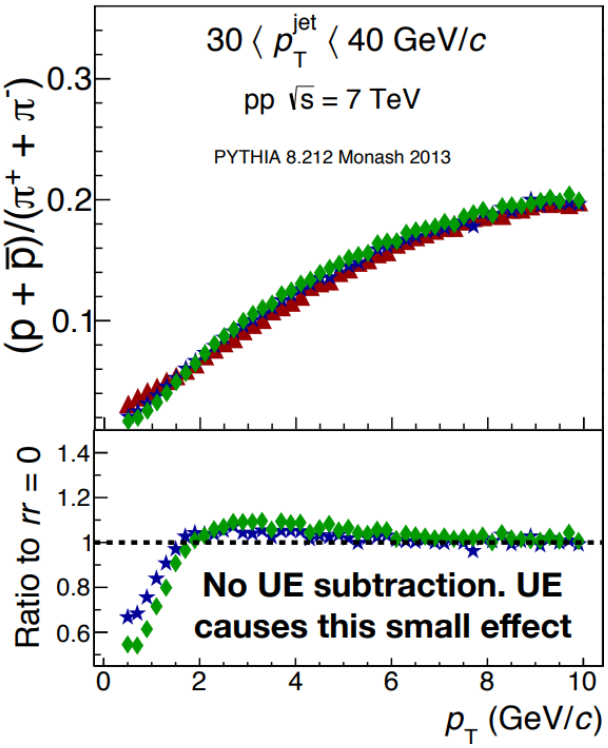
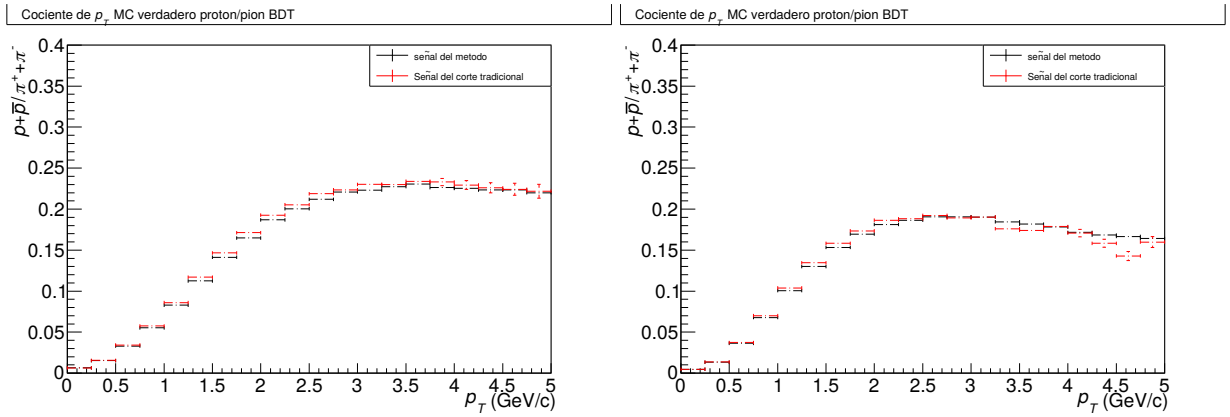
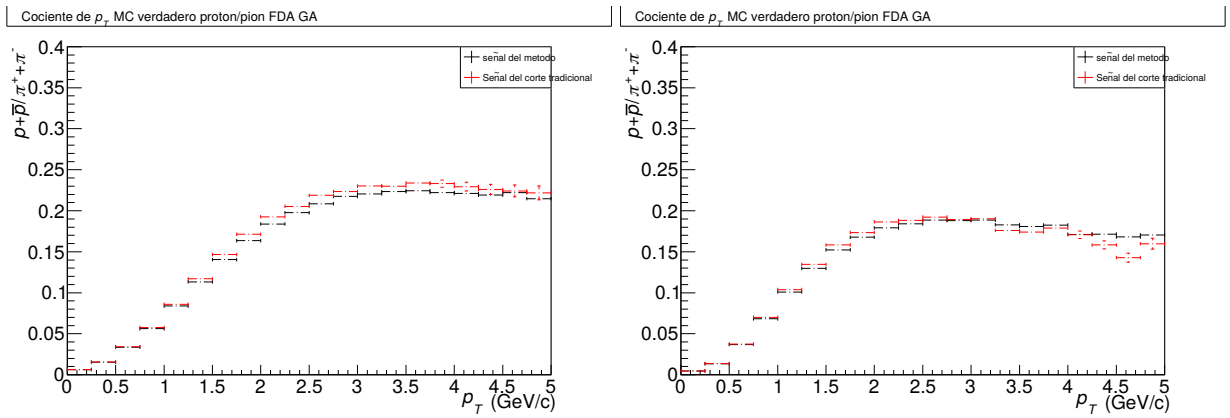


Figura 30: Cociente p/π en la región del jet (Pythia 8). Obtenido de [39]

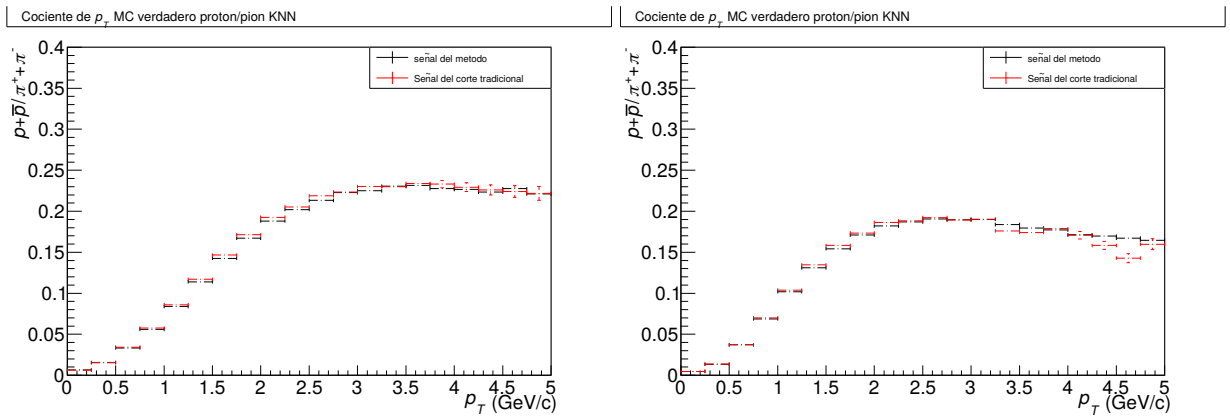
Dado que estamos aislando eventos de alto N_{MPI} se espera que correspondan a una fuerte presencia de efectos provenientes del UE y baja contribución de jets, por lo que la forma de la distribución debe ser similar a la figura 30.



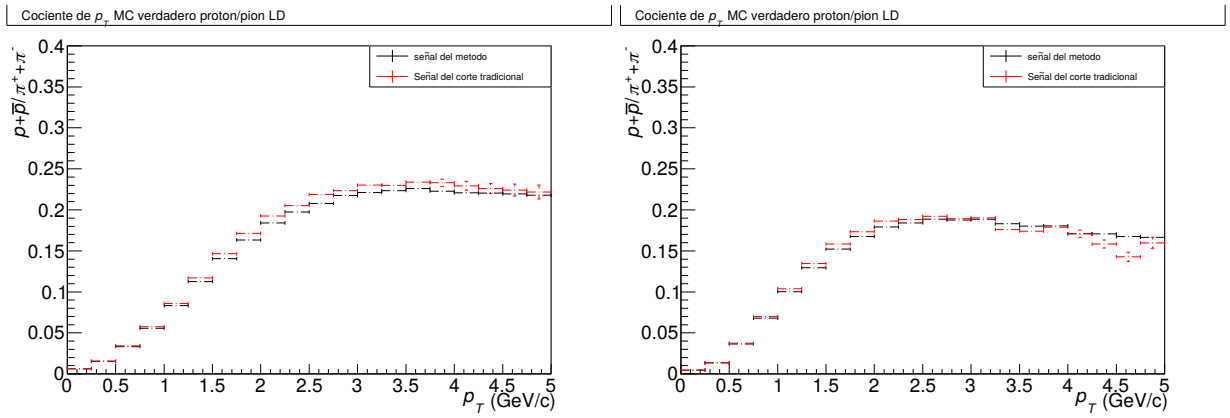
(a) BDT



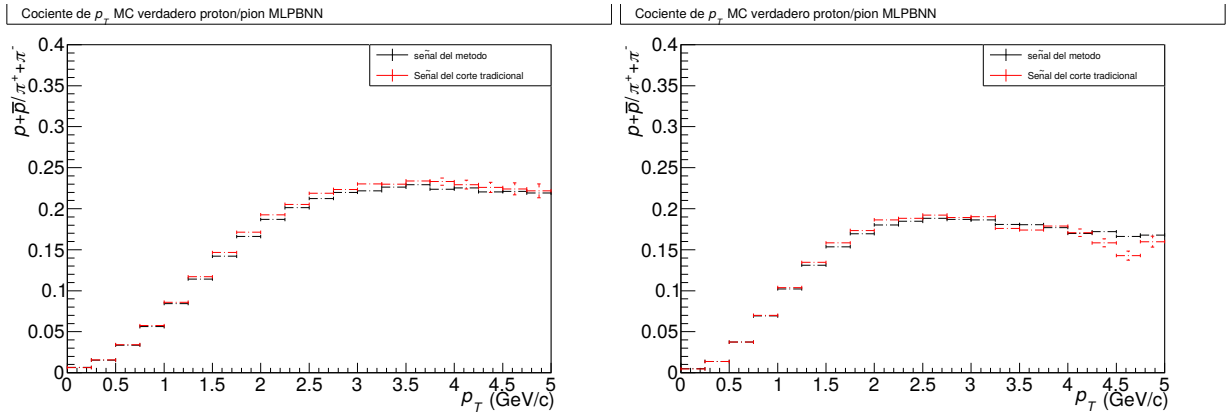
(b) FDA



(c) KNN



(d) LD



(e) MLPBNN

Figura 31: Cocientes p/π para los 5 métodos de ML comparados con el cociente p/π del método tradicional. A la izquierda Pythia 8 (generador de entrenamiento) y a la derecha la comparación con Pythia 6 (generador con modelo distinto)

6. Conclusiones

En lugar de restringirnos únicamente a la esferocidad y la multiplicidad las cuales se han venido ocupando en el método tradicional de selección de alto N_{MPI}, también se exploraron otras 5 variables de las cuales mediante un análisis estadístico basado en los coeficientes de correlación lineales y separación $< S^2 >$ se pudieron identificar 4 variables principales que mostraban un alto poder de clasificación para N_{MPI}. En orden de importancia: Multiplicidad, p_{T} promedio, retroceso y esferocidad.

De forma similar de 11 métodos de aprendizaje de máquina propuestos en un inicio se seleccionaron 5 (LD, FDA_{GA}, MLPBNN, BDT, KNN) en base a la integral ROC y su capacidad para tratar problemas lineales. Finalmente se optimizaron estos métodos para generar modelos ligeros e interpretables sin comprometer su poder de clasificación original.

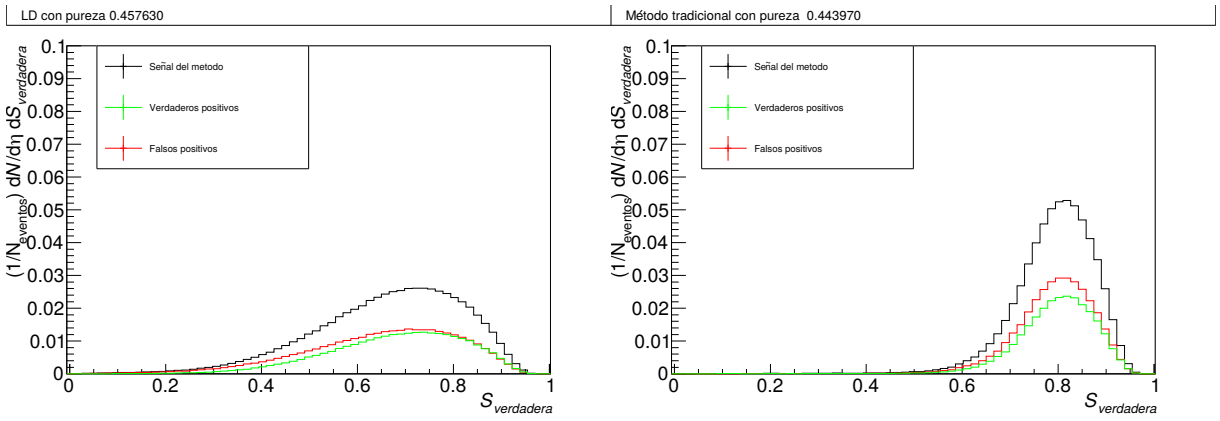
Mediante la prueba de Kolmogorv-Smirnov se determino el grado de sobre entrenamiento de los modelos generados, el cual en FDA_{GA} y BDT es apreciable, sin embargo, aún está dentro del límite aceptado. También se probó la dependencia de los modelos de clasificación generados respecto al MC y se determinó que era despreciable.

Para una distribución de N_{MPI} similar a la obtenida por el método tradicional, se obtuvo un aumento significativo de la eficiencia de clasificación al usar aprendizaje de máquina.

	Eficiencia	**$\mathbf{N_{MPI}}$ promedio**
BDT	0.464 (+154.9 %)	15.25 (+0.24)
LD	0.478 (+162.6 %)	15.11 (+0.1)
MLPBNN	0.611 (+235.7 %)	14.44 (-0.57)
$\mathbf{FDA_{GA}}$	0.520 (+185.7 %)	14.78 (-0.23)
KNN	0.532 (+192.3 %)	14.89 (-0.12)
Método tradicional	0.182 (0.0 %)	15.01 (0.0)

Cuadro 6: Eficiencia y $\mathbf{N_{MPI}}$ promedio por método. Resultados para Pythia 8

Una parte importante de esta mejora proviene de eventos de baja esferocidad que resultaban cortados por el método tradicional al requerir alta esferocidad reconstruida. Esto se puede ver claramente en la figura 32.



(a) LD **(b)** Método tradicional

Figura 32: Comparación entre las distribuciones de esferocidad verdadera para LD y el método tradicional en Pythia 8. Se observa como el método tradicional rechaza eventos de baja esferocidad, mientras que LD permite seleccionar alto N_{MPI} aún a con baja esferocidad.

La forma del cociente $\frac{p}{\pi}$ revela una saturación de este para valores altos de p_{T} el cual está relacionado con una fuerte presencia del UE incluyendo un alto N_{MPI}. Adicionalmente este cociente también revela que se está rechazando correctamente los eventos de tipo jet.

Mediante el análisis de los modelos generados por el aprendizaje de máquina es posible extraer nuevo conocimiento, el cual puede ser interpretado y apoyar en el entendimiento de la física estudiada. Por ejemplo, en la figura 33 correspondiente a el modelo de red neuronal generado por MLPBNN para la clasificación $N_{MPI} > 15$, se puede observar como el 4° nodo de la capa 1 (el de mayor peso) esta formado por una relación lineal con el p_{T} promedio e inversa con la multiplicidad. Es decir a primera aproximación el corte debiese darse en p_{T} promedio bajo y multiplicidad alta en vez de esferocidad y multiplicidad altas, no solo eso, sino que también revela que la esferocidad (la cual se ha venido usando hasta el momento) resulta ser la menos importante de las 4 variables usadas.

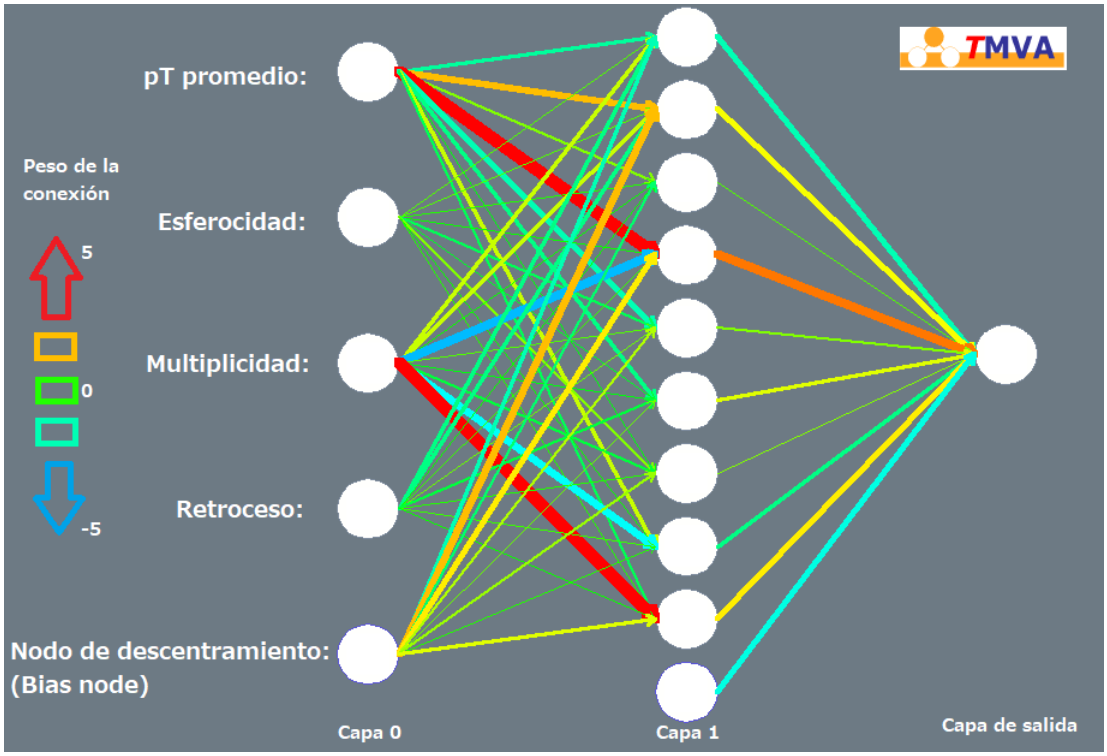


Figura 33: Modelo de neuronas generado por el algoritmo MLPBNN.

En esté trabajo se ocupó el aprendizaje de máquina en su faceta de clasificación ya que permite una comparación directa con el método tradicional, sin embargo, se puede entrenar a los algoritmos para realizar una regresión sobre N_{MPI} lo cual permitiría obtener un valor de N_{MPI} de regresión el cual podría usarse de forma similar a una medición generada por un detector, pero con la gran diferencia que el N_{MPI} no tiene parte reconstruida en los detectores (actualmente). El error de la regresión sería posible estimarlo por medio de un equivalente a las matrices de detector, que relacionen la dispersión de N_{MPI} verdadero con el obtenido por regresión.

El uso del aprendizaje de máquina como clasificación demostró lograr mejoras respecto al método tradicional de corte y su extensión a regresión promete abrir una nueva área de oportunidades al permitir reconstruir cantidades como el N_{MPI} las cuales actualmente solo existen a nivel generador y no poseen parte reconstruida.

7. Bibliografía

[1] Alec Radford. *Optimizers animation incluida en CS231n Convolutional Neural Networks for Visual Recognition.* http://cs231n.github.io/neural-networks-3/

[2] Julia Woithe et al. (2017) *Let's have a coffee with the Standard Model of particle physics!.* Phys. Education Volume 52 Number 3. DOI: 10.1088/1361-6552/aa5b25

[3] Han Xiao and Kashif Rasul and Roland Vollgraf (2017) *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms.* arXiv:1708.07747

[4] Acevedo Nuñez, Melisa Andrea. (2017). *Machine learning : algoritmos de clasificación y sus aplicaciones en el análisis de datos. (tesis de pregrado).* UNAM. FES Acatlan

[5] M. E. Peskin and D. V. Schroeder. (1995). *An Introduction To Quantum Field Theory.* Reading. USA: Addison-Wesley

[6] Nagashima, Y. (2010). *Elementary Particle Physics. Volume 1: Quantum Field Theory and Particles* Osaka University. Japan. ISBN: 978-3-527-40962-4. Brock, I.

[7] K.A.Olive et al. (Particle Data Group) (2014). *Chinise Physics C. Volume 38 Number 9* DOI: 10.1088/1674-1137/38/9/090001

[8] Education, Communication and Outreach Group CERN (2017). *LHC faq the guide* https://home.cern/sites/home.web.cern.ch/files/2018-07/CERN-Brochure-2017-002-Eng_0.pdf

[9] ALICE Collaboration (2018) *Multiplicity dependence of light-flavor hadron production in pp collisions at* $\sqrt{s} = 7TeV$. arXiv:1807.11321v1

[10] P. Bartalin et al (2011) *Multiparton interactions at the LHC.* arXiv:1111.0469v2

[11] Grupen, Claus and Boris, Schwartz (2008). *Particle Detectors* Cambridge university press. ISBN: 978-0-521-18795-4

[12] Alice collaboration (2005). *Physics performance report Volume II* CERN. ISBN: 92-9083-258-4

[13] W. Blum and L. Rolandi. (s.f). *Particle Detection with Drift Chambers.* Springer

[14] D. Griffiths. (2008). *Introduction to Elementary Particles.* Wiley-VCH, 2nd edition,

[15] T. Martin, P. Skands, S. Farrington (2016) *Probing Collective Effects in Hadronisation with the Extremes of the Underlying Event.* arXiv:1603.05298v2

[16] Ortiz Antonio,Valencia Palomo Lizardo (2019) *Probing color reconnection with underlying event observables at the LHC energies.* DOI: 10.1103/PhysRevD.99.034027

[17] A. Hoecker, P. Speckmayer, J. Stelzer, J. Therhaag, E. von Toerne, and H. Voss, "TMVA: Toolkit for Multivariate Data Analysis," PoS A CAT 040 (2007) [physics/0703039]

[18] A. Hoecker, P. Speckmayer, J. Stelzer, J. Therhaag, E. von Toerne, H. Voss. (2013). *TMVA 4 Toolkit for Multivariable Data Analysis with ROOT users guide.* http://tmva.sourceforge.net/docu/TMVAUsersGuide.pdf

[19] Matteo Cacciari, Gavin P. Salam, Gregory Soyez. *TFastJet user manual.* arXiv:1111.6097

[20] Matteo Cacciari, Gavin P. Salam. *Dispelling the* N^3 *myth for the Kt jet-finder.* hep-ph/0512210

[21] Torbjörn Sjöstrand, Stephen Mrenna y Peter Skands. (2006). *Pythia 6.4 Physics and Manual* .hep-ph/0603175

[22] Torbjörn Sjöstrand et al. (2014). *An Introduction to PYTHIA 8.2* arXiv:1410.3012

[23] Nicolas Chanon (2012) *Multivariate analysis in high energy physics (Notas de clase).* ETH Institute for particle physics, Zürich.

[24] A. Romanino (2009) *The Standard model of particle physics (Notas de clase).* International Baikal Summer School, Bolshie Koty, Russia

[25] Joniale (2017) *Generaciones$_d$elamateria.png*

[26] P.Skands,S.Carrazza,J.Rojo (2014). *Tuning PYTHIA 8.1: the Monash 2013 Tune.* arXiv:1404.5630v1

[27] P.Z.Skands (2014). *Tuning Monte Carlo Generators: The Perugia tunes.* arXiv:1005.3457

[28] R. Brun, F.Bruyant, M.Maire, A.C.MCPherson, P.Zanarini (1987). *GEANT3 USER'S GUIDE.* https://cds.cern.ch/record/1119728/files/CERN-DD-EE-84-1.pdf

[29] Alice Collaboration *ALICE Offline* http://alice-offline.web.cern.ch/Activities/Simulation/EventGeneration.html

[30] Balch Tucker and Chakraborty Arpan *Machine Learning for tradiong (Curso)* Ofrecido por Geogia Tech como CS7646 y disponible en Udacity https://www.udacity.com/course/machine-learning-for-trading--ud501

[31] Karsch, Frithjof (2001) *Lattice QCD at High Temperature And Density.* arXiv:hep-lat/0106019v2

[32] Keras *Keras: The Python Deep Learning library- Documentation* https://keras.io/

[33] Hyttsten Magnus, Delgado Juan y Bailey Paige *Intro to TensorFlow for Deep Learning (Curso)* Ofrecido por TensorFlow y disponible en Udacity https://www.udacity.com/course/intro-to-tensorflow-for-deep-learning--ud187

[34] Helge Voss (2015) *TMVA Tutorial (Diapositivas de conferencia).* https://indico.cern.ch/event/402660/sessions/162371/attachments/1193153/1732419/TMVA.pdf

[35] Adrian Bevan (2013). *Statistical data analysis for the physical sciences* Cambridge University Press, Cambridge, Reino Unido ISBN: 978-1-107-67034-1

[36] Nicolas Chanon (2012) *Multivariable analysis in high energy physics (Notas de clase).* ETH Institute for particle physics. https://people.phys.ethz.ch/~pheno/Lectures2012_StatisticalTools/slides/Chanon3.pdf

[37] ALICE Collaboration (2019). *Charged-particle production as a function of multiplicity and transverse spherocity in pp collisions at* $\sqrt{s}$ *=5.02 and 13 TeV* . arXiv:1905.07208

[38] ALICE Collaboration (2008) *Imágen encontrada en ALICE- A Large Ion Collider Experiment* http://aliceinfo.cern.ch/Public/en/Chapter2/Chap2Experiment-en.html

[39] Ortiz Antonio (2018) *Physics of small systems using UE observables (Diapositivas).* UNAM utilizadas en el "10^{th} International Workshop on Multiple Partonic Interactions at the LHC" https://indico.cern.ch/event/736470/contributions/3219266/attachments/1769760/2875045/aortiz_MPI2018.pdf

[40] Kevin Dusling, Wei Li, Björn Schenke (2016). *Novel Collective Phenomena in High-Energy Proton-Proton and Proton-Nucleus Collisions* International Journal of Modern Physics E. arXiv:1509.07939v2

8. Anexos

8.1. Tipos de aprendizaje de máquina

Los algoritmos de aprendizaje de máquina se pueden clasificar según su aplicación, según el tipo de problema que pueden resolver o según sus características técnicas. Algunas de las categorías más comunes son[30]:

8.1.1. Supervisado y no supervisado

Los algoritmos de aprendizaje supervisado requieren de un agente externo como una persona u otro programa el cual actué como profesor y les indiqué mediante ejemplos que es lo que espera de ellos. La principal característica de estos algoritmos es que la información de entrenamiento se compone de pares (X, Y) donde X son las variables de entrada del algoritmo y Y es la salida esperada, la cual puede ser una clase (clasificación), una cantidad (regresión). Algunos de los principales ejemplos de aprendizaje supervisado son la clasificación de imágenes por etiquetas, la detección de patrones de fraude, predicción de valores de acciones en la bolsa, etc. La mayoría de los algoritmos de aprendizaje de máquina se encuentran en esta categoría y es la que vamos a utilizar utilizando a lo largo de esta tesis.

Los algoritmos de aprendizaje no supervisado en contraste con los supervisados no cuentan con un valor Y esperado para cada entrada X. Sino que solo cuentan con un montón de información la cual deben separar en n clases o grupos que tengan características similares entre sí. En este caso no se tiene idea de que puedan ser estas características o en ocasiones ni siquiera se sabe exactamente cuántos grupos puede haber en una muestra.

Un ejemplo que espero me permita explicar cómo funcionan estos algoritmos es un sistema de clasificación de imágenes. Las imágenes que estaré usando pertenecen a la base de datos MNIST Fashion dataset la cual contiene imágenes de 10 clases de ropa distintas con sus correspondientes etiquetas y está diseñando para probar sistemas de inteligencia artificial (Ver Figura 34).

Figura 34: Imágenes explicativas de prueba pertenecientes a la base *MNIST Fashion dataset*. Cada una de esta imágenes contienen etiquetas indicando la clase de ropa a la que pertenece la imagen.

Un programa de aprendizaje supervisado siempre va a regresar como respuesta o bien una clase (la de más alta probabilidad) o bien puede regresar una distribución de probabilidad indicando que tan probable es que el objeto pertenezca a cada una de las clases. Es necesario recalcar que las clases fueron indicadas durante el entrenamiento por medio de etiquetas en pares (X, Y) con X la imagen y Y la clase a la que pertenece dicha imagen (Ver Figura 35).

Figura 35: Respuesta obtenida tras el entrenamiento de una red neuronal secuencial compuesta por 100+10 neuronas distribuidas en 2 capas. Bajo cada imagen se encuentra la clase predicha y entre paréntesis la clase correcta.

Las distribuciones de probabilidad indicadas en las gráficas a la derecha de cada imagen se encuentran en el siguiente formato.

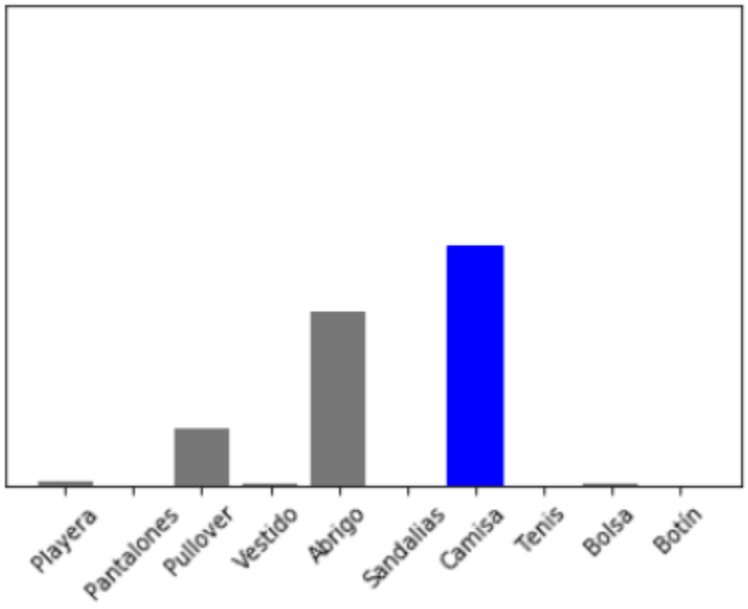


Figura 36: Detalle de la gráfica de distribución de probabilidad (respuesta) correspondiente a la primer imagen

Un programa de aprendizaje no supervisado agrupará los mismos objetos por semejanzas de acuerdo a su propio criterio, el cual no tiene por qué coincidir con las clases que esperamos. Por ejemplo, puede clasificar en: Clase 1 'Objetos lagos verticales oscuros', Clase 2 'Objetos verticales oscuros', Clase 3 'Objetos verticales claros', Clase 4 'sandalias de piso', Clase 5 'Objetos tipo zapatilla', Clase 6 'Objetos tipo bolsa'. (Ver Figura 37)

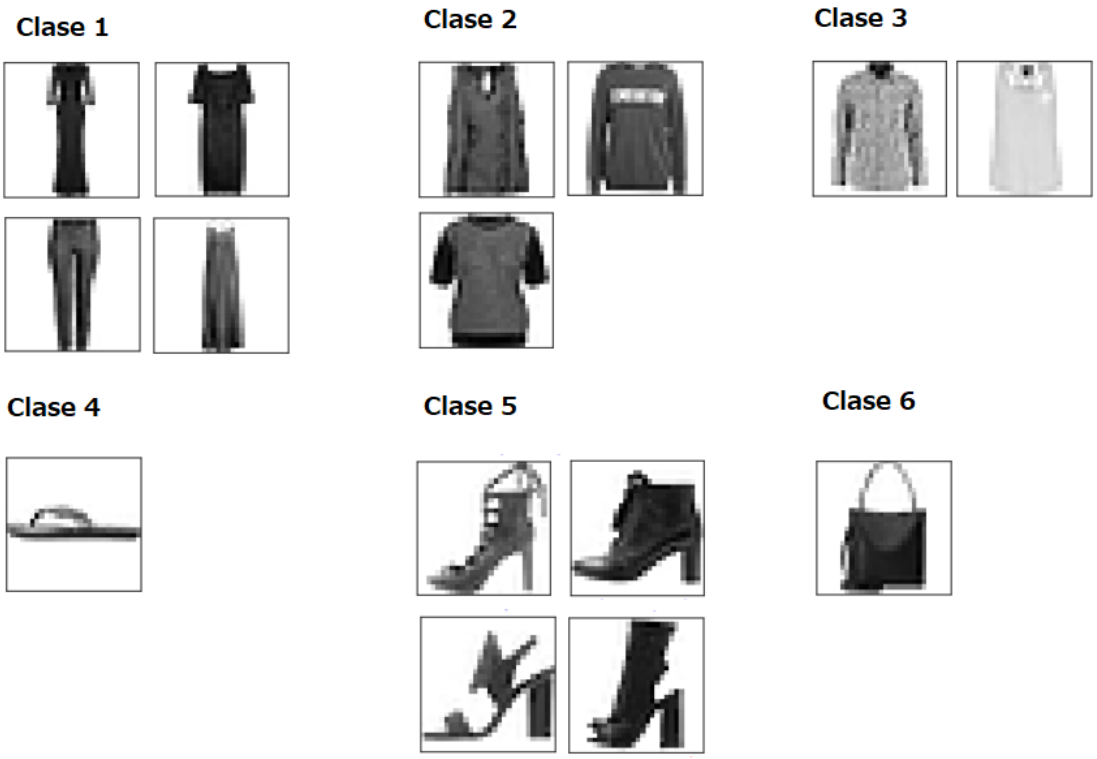


Figura 37: Clasificación no supervisada (por semejanzas). Es importante notar que los grupos coinciden parcialmente, pero exactamente con los grupos de las etiquetas.

Es importante recalcar que durante el entrenamiento un método no supervisado nunca ve las etiquetas se intención (Y) mostradas a los métodos supervisados, sino que crea sus propias etiquetas en el trascurso del entrenamiento. En este caso, tanto algunas sandalias como botines tienen forma parecida y entrarían en una sola clase, mientras que sandalias de piso serian su propia clase, también se puede observar que la clase 1 contiene tantos pantalones como vestidos y ocasionalmente también algunas playeras.

Es importante mencionar que esta forma de clasificar información puede llegar a conducir a el descubrimiento de nuevas características comunes que no hubiéramos pensado de otra manera y que resultan importantes para un problema en particular lo cual puede conducir a nuevas ramas de investigación o nuevas formas de ver un problema conocido.

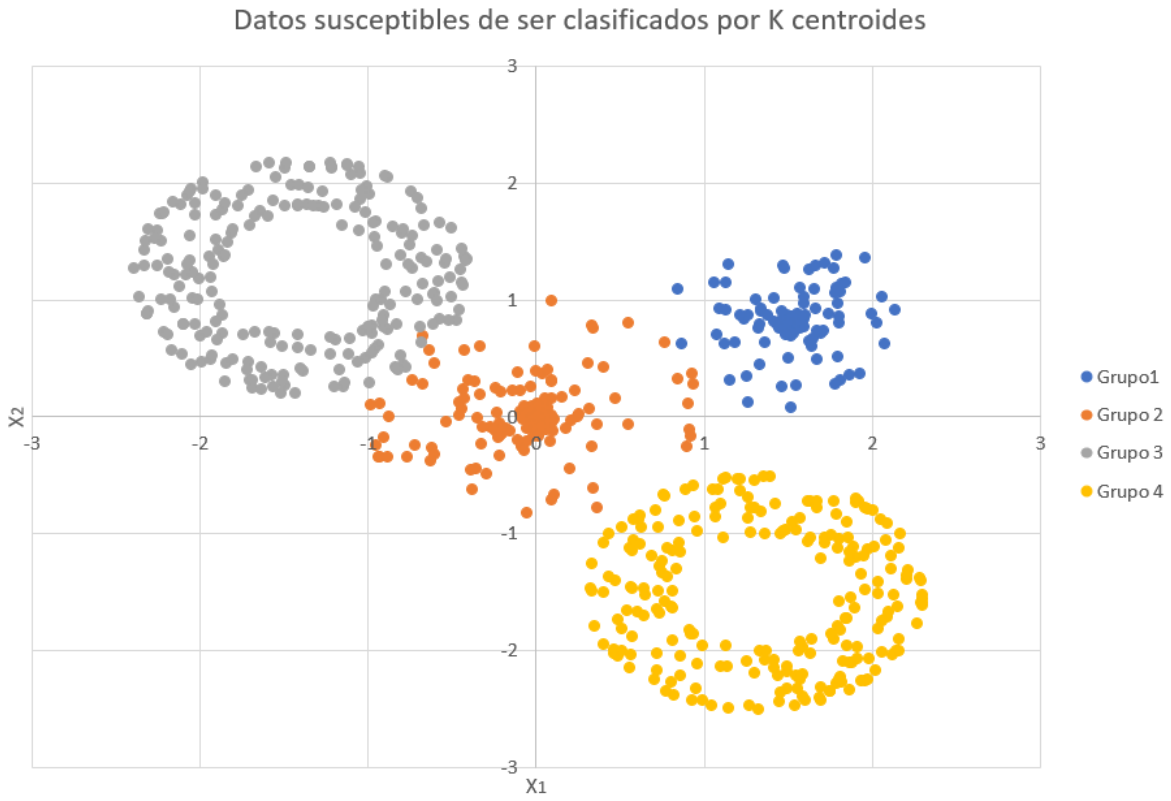


Figura 38: Dispersión de puntos de una distribución susceptible a ser clasificada por K-centroides. Observe que ambos ejes son variables de entrada X1 y X2.

Los principales representantes de esta rama son los algoritmos de clasificación no supervisada k-centroides los cuales agrupan los eventos en k-clústeres distintos. (Ver Figura 38)

8.1.2. Paramétrico y basado en instancias

Esta clasificación indica si se crea o no un modelo para los datos de entrenamiento. En el aprendizaje paramétrico, se crea un modelo como por ejemplo una serie de Taylor el cual está compuesto de parámetros (pesos) los cuales se ajustan sobre los datos proveyendo una función matemática o un algoritmo que permite una vez entrenado el modelo calcular la salida a partir de la entrada y los pesos sin la necesidad de recurrir nuevamente a los datos de entrenamiento (Ver Figura 39).

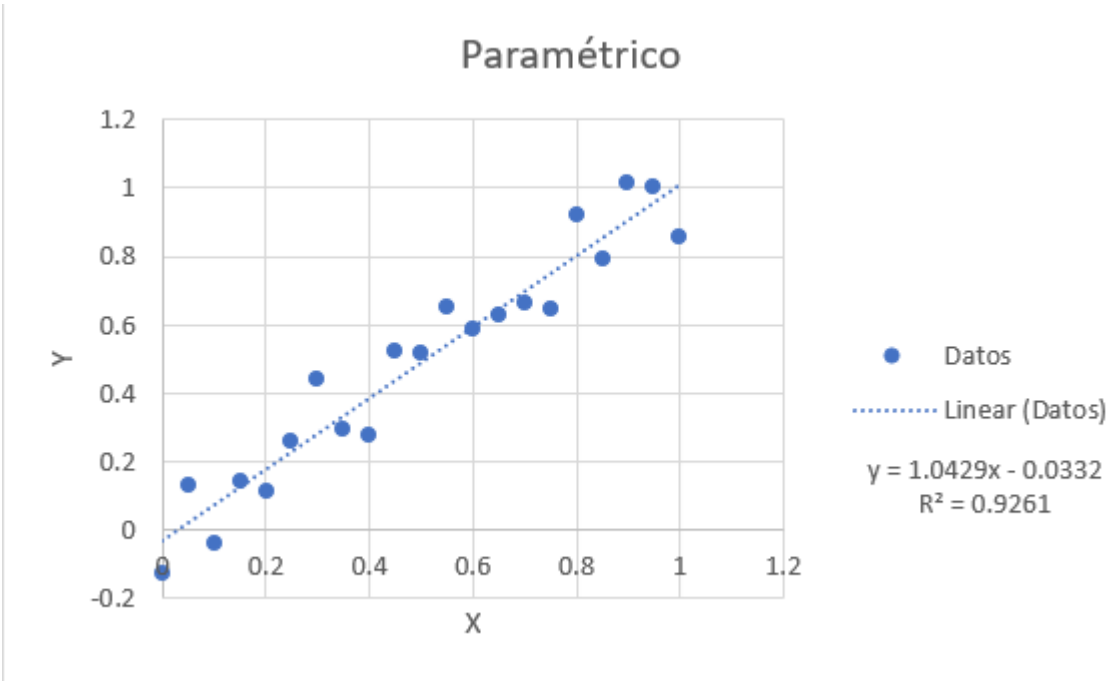


Figura 39: Método paramétrico (Taylor) en una dimensión para la relación lineal Y=X

En contraste el aprendizaje basado en instancias no plantea ningún modelo sobre los datos, sino que en su lugar retiene los datos de entrenamiento y los consulta en ocasión de cada consulta. El ejemplo más sobresaliente de esta familia es el método de K-vecinos, el cual consulta a los K puntos más cercanos en el espacio X al punto que se desea predecir y mediante votación o promedio calcula la respuesta.

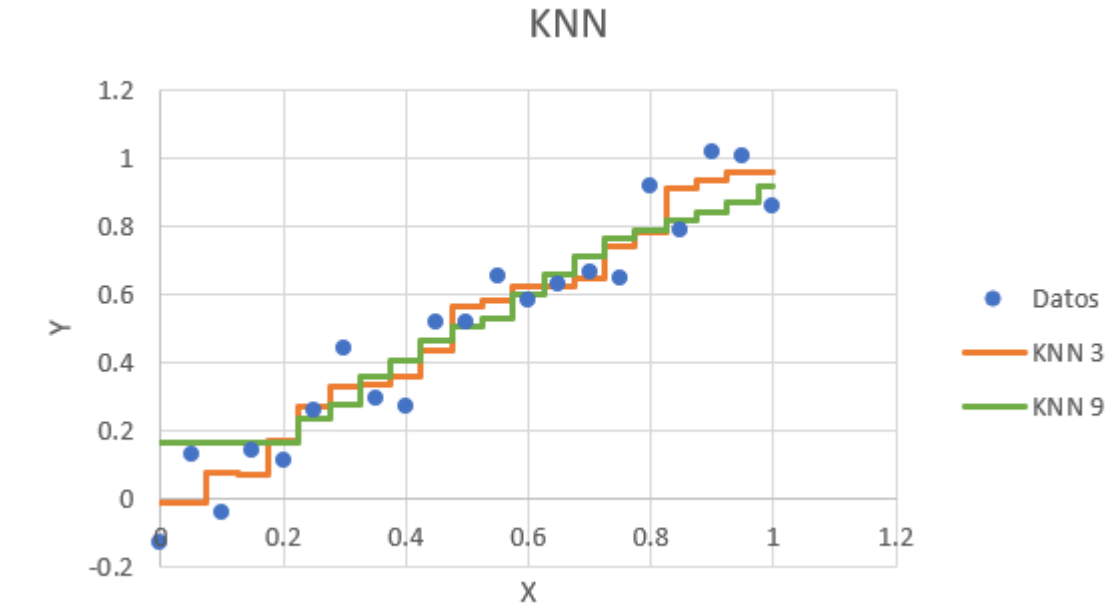


Figura 40: Método basado en instancias (KNN) en una dimensión para la los mismos datos $Y = X$

La ventaja de los métodos basados en instancias es que al no poseer un modelo subyacente estos pueden ajustar a todo tipo de relaciones, ya sean lineales o no lineales además de facilitar la creación de ensambles de clasificadores débiles los cuales pueden mostrar resultados superiores al de un clasificador fuerte solitario.

8.1.3. Regresión y Clasificación

Los métodos de aprendizaje de máquina se pueden dividir según su aplicación en dos clases muy importantes clasificación la cual se enfoca en separar en grupos a los elementos, y en regresión la cual se enfoca en regresar un número correspondiente a una variable que se desea calcular.

Los trabajos de regresión son los más antiguos en el área de aprendizaje de máquina y se pueden trazar sus inicios hasta la época de Gauss. Los métodos correspondientes a esta área buscan ajustar una función $Y(x)$ a los datos, reducir el error entre el valor Y predicho y el valor Y esperado, para finalmente hacer predicciones usando la función $Y(x)$ encontrada.

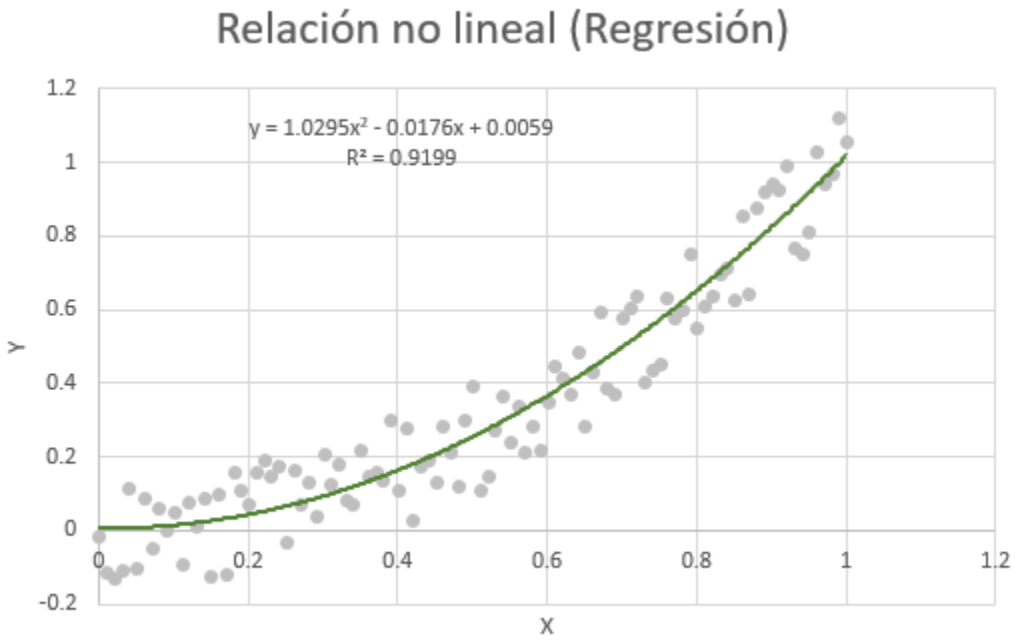


Figura 41: Distribución de puntos y ajuste en un problema de regresión (1-dimensional)

Como se puede ver en la figura 41 los puntos en las tareas de regresión se encuentran localizados a lo largo de la función ajustada, en este caso un polinomio.

Los trabajos de clasificación es posible tratarlos de forma similar a los de regresión. Si planteamos el problema como un problema de parametrización de la frontera, donde la función que deseamos ajustar es aquella que representa a la frontera entre la señal y el fondo. Este enfoque es el usado por el método SVM.

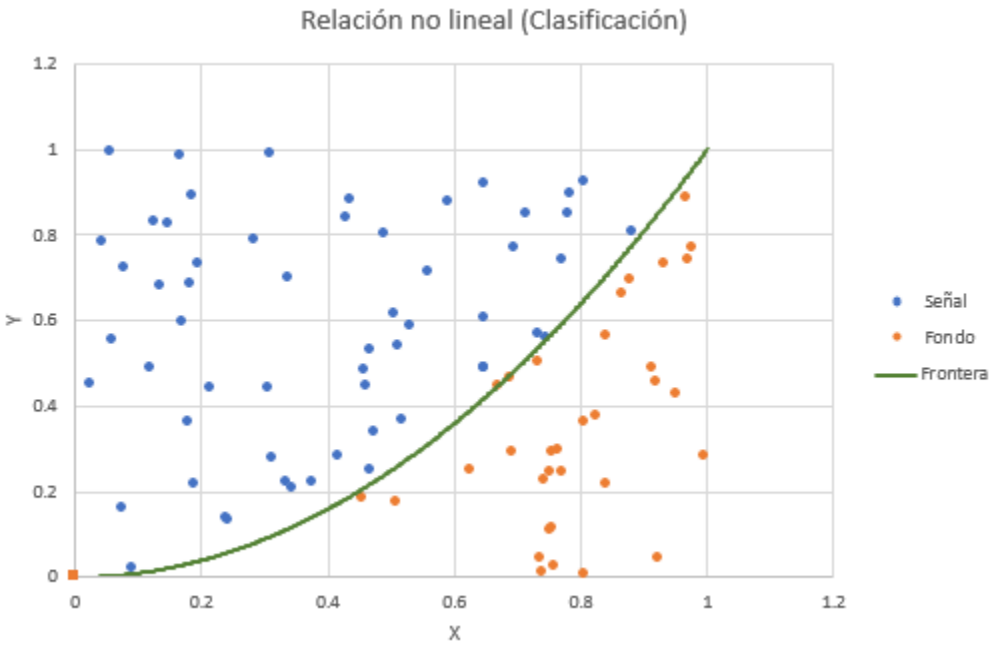
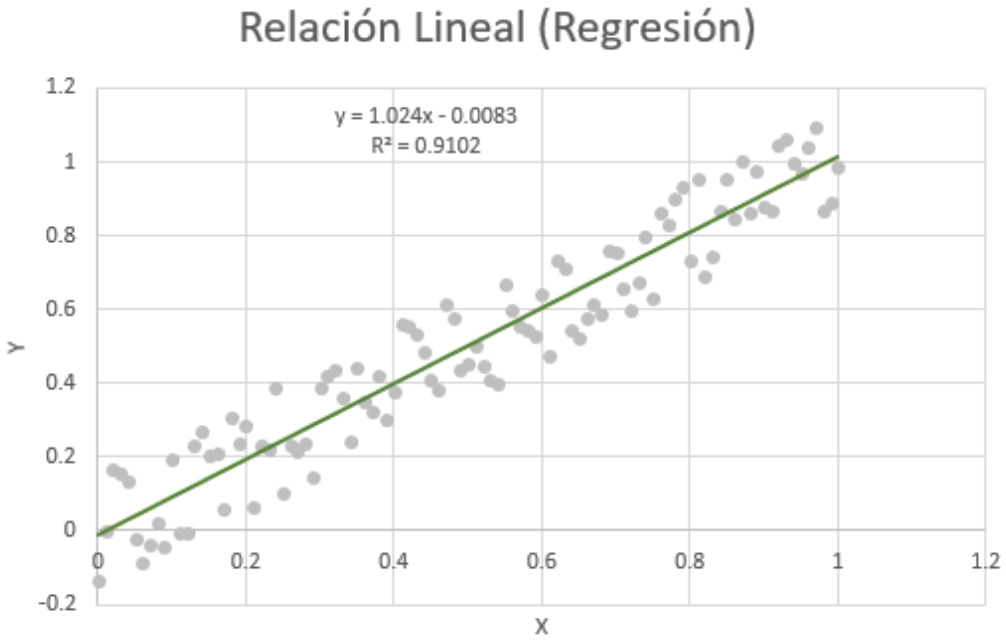


Figura 42: Distribución de puntos y ajuste en un problema de clasificación (1-dimensión)

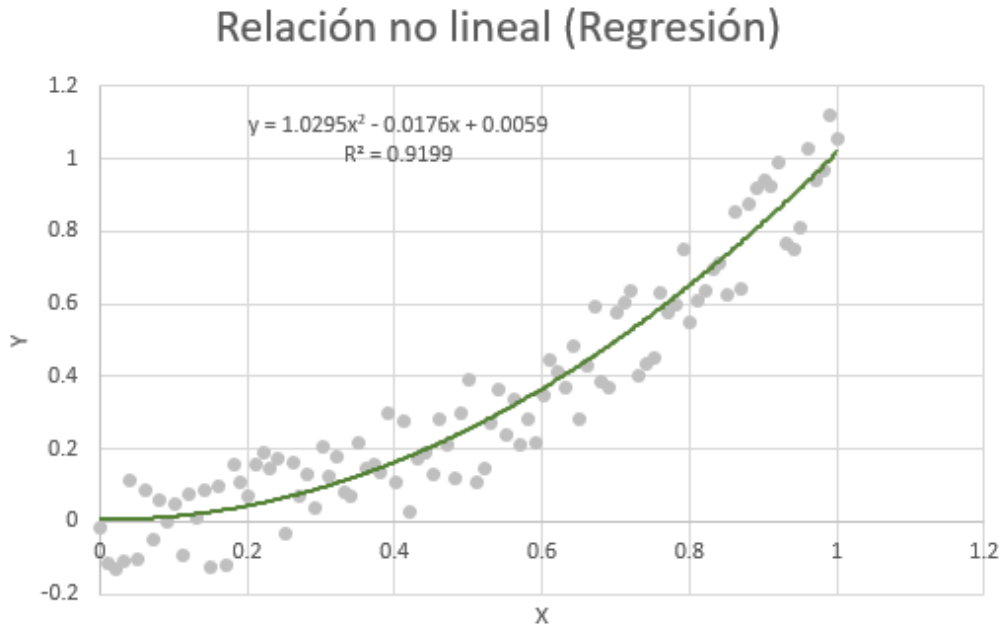
Como se puede ver en la figura 42 los puntos arriba de la función $Y(x)$ determinada por el método corresponden a la señal, mientras que aquellos abajo de la función pertenecen al fondo. En un caso más realista siempre existirá una pequeña región de traslape donde habrá puntos de señal y de fondo combinados. La forma de tratar esta región es la que determinará la pureza y eficiencia de nuestros métodos.

8.1.4. Lineal o no lineal

Esta forma de clasificación corresponde a la dificultad del problema a tratar. Un problema lineal es aquel donde la relación $Y = AX + b$ es suficiente para determinar la solución del problema (Ver Figura 43), este tipo de problema es el más sencillo que tiene sentido resolver por aprendizaje de máquina y prácticamente todos los algoritmos permiten tratarlo.



(a) Lineal

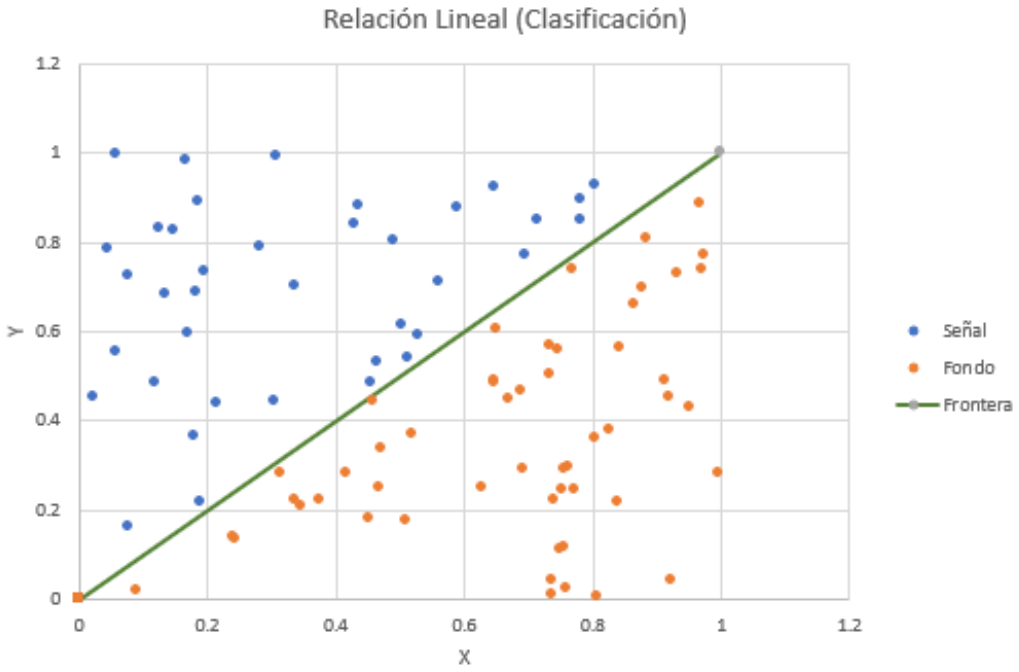


(b) No lineal

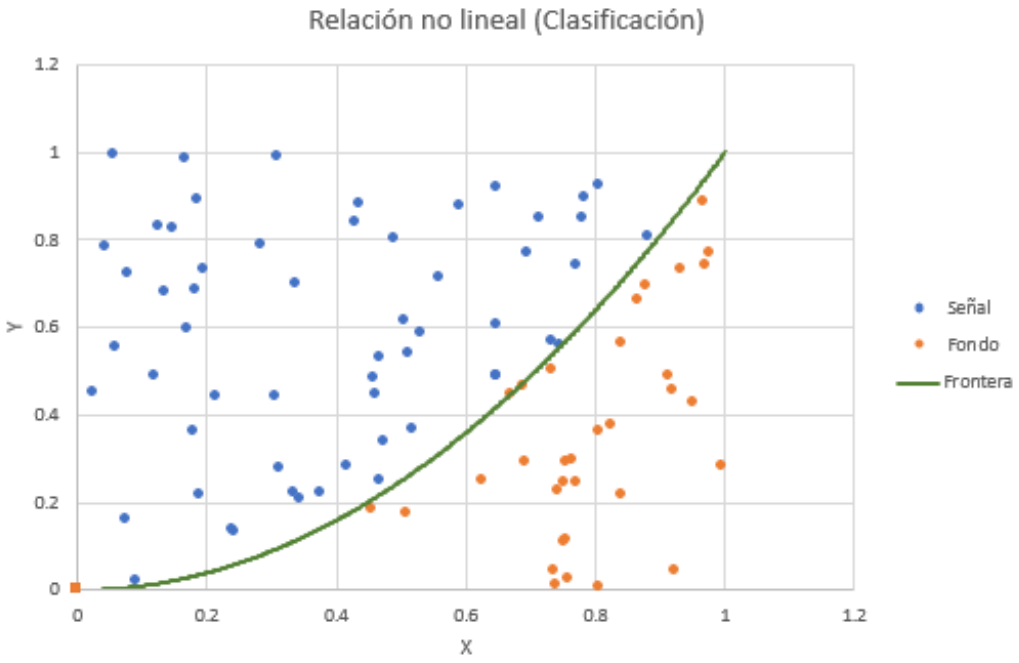
Figura 43: Problema de regresión (1-dimensional).

Los problemas no lineales son aquellos donde la relación entra las salidas y entradas o la frontera de señal fondo es más complicada que $Y = AX + b$ y por lo tanto es necesario plantear modelos de mayor complejidad para determinar su forma (Ver Figura 44). Este tipo de problemas usualmente requieren muchos más ejemplos para

poder determinar la relación entre las variables que los problemas lineales. Adicionalmente, se requiere incorporar una supervisión para evitar problemas de sobreajuste (*overfitting*).



(a) Lineal



(b) No lineal

Figura 44: Problema de clasificiación (1-dimensional).

8.2. Métodos utilizados

A continuación, se describen brevemente los algoritmos utilizados durante este trabajo

8.2.0.1. MLPBNN Las redes neuronales son uno de los algoritmos más flexibles y de amplio uso en el área de aprendizaje de máquina. Son capaces de realizan tanto tareas de clasificación como de regresión, así como resolver problemas lineales y no lineales.
Las redes neuronales artificiales están inspiradas en su contraparte biológica, y al igual que estas se basan en una conexión compleja de elementos llamados perceptrones (neuronas en biología). Cada perceptrón cuenta con múltiples entradas y una única salida la cual está relacionada a las entradas mediante una relación lineal más una función de activación. (Ver Figura 45)

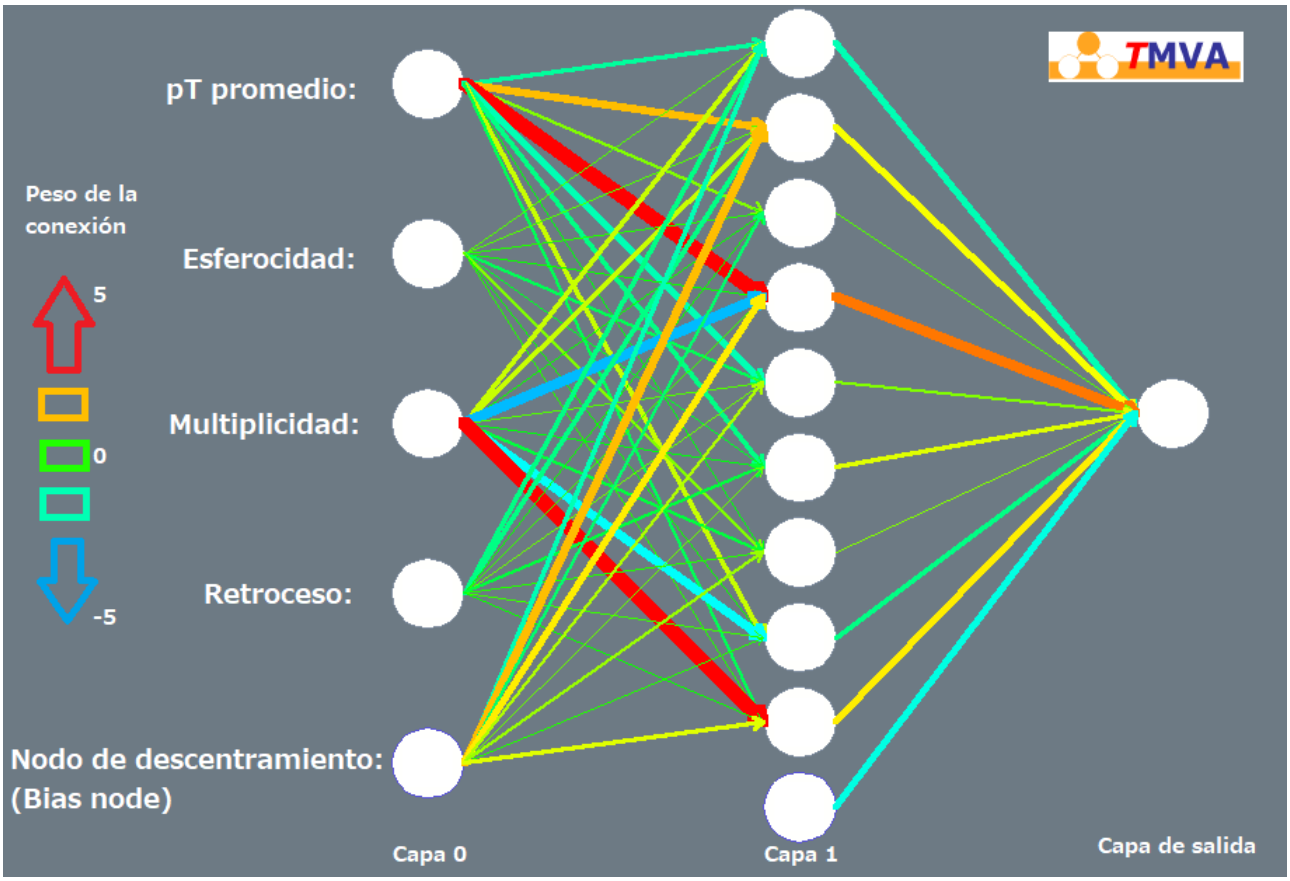


Figura 45: Diagrama mostrando los pesos y distribución de neuronas en el modelo MLPBNN

La función de activación tiene como objetivo mapear los valores de salida de la función lineal a un intervalo, rango y forma adecuados para ya sea aplicarlos a otras neuronas o bien generar una salida con ellos. Algunas de las funciones de activación más populares son: sigmoidal, seno hiperbólico, relu, *softmax* y la identidad (Ver Figura 46).

Las funciones relu, identidad, sigmoidal, tangente hiperbólica, binaria entre otras son populares como funciones de activación para las neuronas pertenecientes a capas ocultas de la red y consisten en aplicar a cada una de las neuronas una de las funciones siguientes:

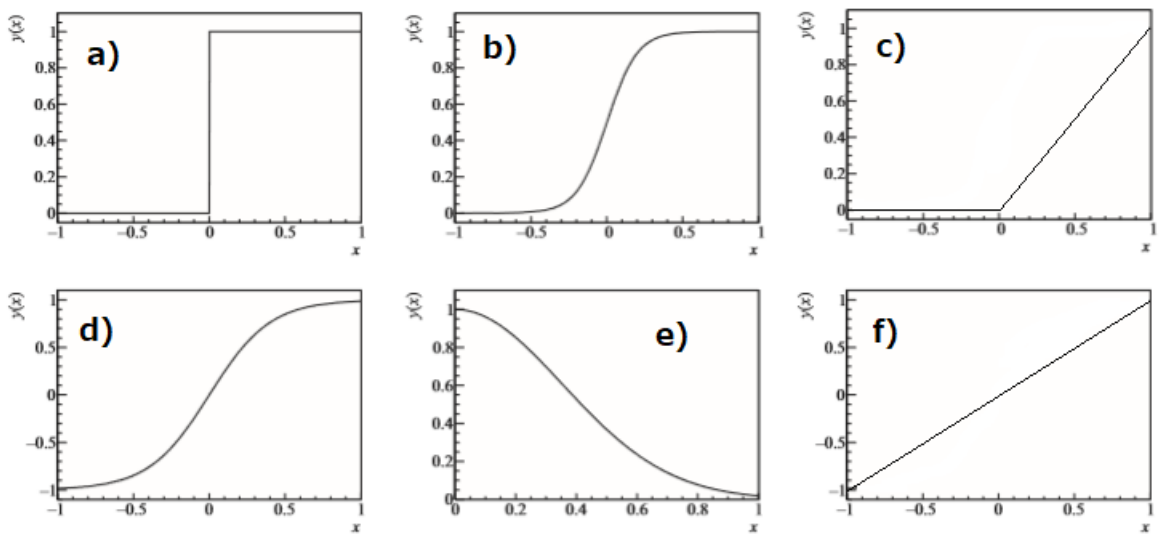


Figura 46: Funciones de activación a)binaria, b)sigmoidal, c) RELU, d)Tangente hiperbólica e) radial, f) identidad. Las gráficas a),b), d) y e) provienen de [34]

La función *softmax* sin embargo es una función de normalización a 1 sobre salida y se ocupa en la última capa en una red neuronal de clasificación, de forma que la salida de esta capa represente una distribución de probabilidades de pertenencia a cada una de las distintas clases posibles. En esta función cada neurona representa una clase de salida y su respuesta normalizada representa la probabilidad de que el elemento pertenezca a dicha clase.

La gran flexibilidad de las redes neuronales se encuentra principalmente en la posibilidad de generar redes tan complejas como nuestro problema lo requiera. Aunque existen redes con retroalimentación las cuales permiten aplicaciones que requieran memoria de eventos, en nuestro caso de clasificación nos combinen utilizar redes sin retroalimentación también conocidas en inglés como (*feed forward*). Este tipo de redes se pueden desarrollar mono capa, sin embargo, lo común es hacerlas multicapa para aprovechar su desempeño en problemas no lineales.

El modelo que se ocupa conocido como MLPBNN (*multilayer perceptron bayesian- neural network*) es un modelo sin retroalimentación con capas completamente conectadas entre sí. En este modelo la entrada de cada capa esta conectadas por medio de pesos a todas las salidas de la capa anterior y además a el valor constante 1 esta última conexión recibe el nombre de nodo de descentramiento. El valor de respuesta de una capa respecto a sus entradas se puede expresar de la siguiente forma:

$$Y_i = F(\sum_j (W_{ij} X_j) + A_i) \tag{10}$$

Donde F es la función de activación la cual va a (salvo *softmax* que normaliza la salida), $W_{i,j}$ es la matriz de pesos, X_j el vector de entrada a la capa y A_i es el vector de descentramiento.

El mecanismo de ajuste de pesos de las redes neuronales es un sistema conocido como (*back-propagation*) que traducido es propagación hacia atrás. Este sistema comienza calculando el error cometido partiendo desde la última capa (la más cercana a la salida) y a partir de ahí calcula el error generado por la capa anterior y así sigue hasta llegar a la primera capa. El método de *back propagation* está relacionado con la elección de las funciones de activación, ya que los primeros algoritmos de aprendizaje profundo (*deep learning*) compuestos por muchas capas de neuronas (más de 3) los cuales usaban la función identidad para activación presentaban problemas pues el gradiente usado para mover los pesos se desvanecía al llegar a las primeras capas y por consiguiente el peso de

estas capas nunca se modificaba. En la actualidad el problema se ha resuelto mediante la aplicación de funciones de activación como relu o seno hiperbólico las cuales han demostrado ser efectivas en aplicaciones de aprendizaje profundo.

Es importante mencionar que existen otros tipos de capas y filtros que se pueden incorporar a las redes neuronales, sin embargo no están incluidos en MLPBNN. Este tipo de capas son por ejemplo las de convolución y *average pooling* (agrupamiento promedio) las cuales permiten a las redes neuronales tratar de forma efectiva imágenes.

8.2.0.2. BDT BDT son las siglas en ingles de árboles de decisiones aumentados. La fuerza de este método se basa en tomar votaciones sobre las decisiones, pensado en que las decisiones grupales son mejores que las decisiones tomadas por un único individuo.

El resultado final del método BDT es obtenido por una votación ponderada por medio de pesos de todos los arboles del bosque, donde el peso indica que tan influyente es el árbol en la votación. Aquellos árboles que generen mejores predicciones obtendrán mayor peso en decisiones futuras, de igual forma al realizar un podado de cada árbol, las ramas que generen las peores decisiones se eliminan.

Por si solo cada árbol es un clasificador pobre en comparación con otros métodos como MLPBNN o SVM debido a que es altamente sensible a fluctuaciones estadísticas de la muestra de entrenamiento y para dos muestras de entrenamiento distintas puede converger a cadenas de decisión completamente distintas y por lo tanto a un resultado distinto.

La forma de lidiar con la alta sensibilidad a las fluctuaciones estadísticas en la muestra es realizar un procedimiento conocido como *bagging* el cual consiste en tomar submuestras aleatorias de la muestra de entrenamiento y para cada una de estas submuestras generar un modelo (árbol). Posteriormente mediante pesos se puede generar una decisión ponderada proveniente de todos los arboles del bosque y por lo tanto con una fluctuación estadística mucho menor que la de un solo árbol.

El proceso de generación de un árbol consiste en paso a paso determinar que variable permite lograr la mejor separación entre señal y fondo mediante un corte rectangular, la determinación de los cortes se realiza mediante el criterio de pureza de la señal. Posteriormente a la creación del árbol se poda mediante la eliminación de nodos insignificantes estadísticamente lo cual conduce a una disminución del sobreentrenamiento del árbol.

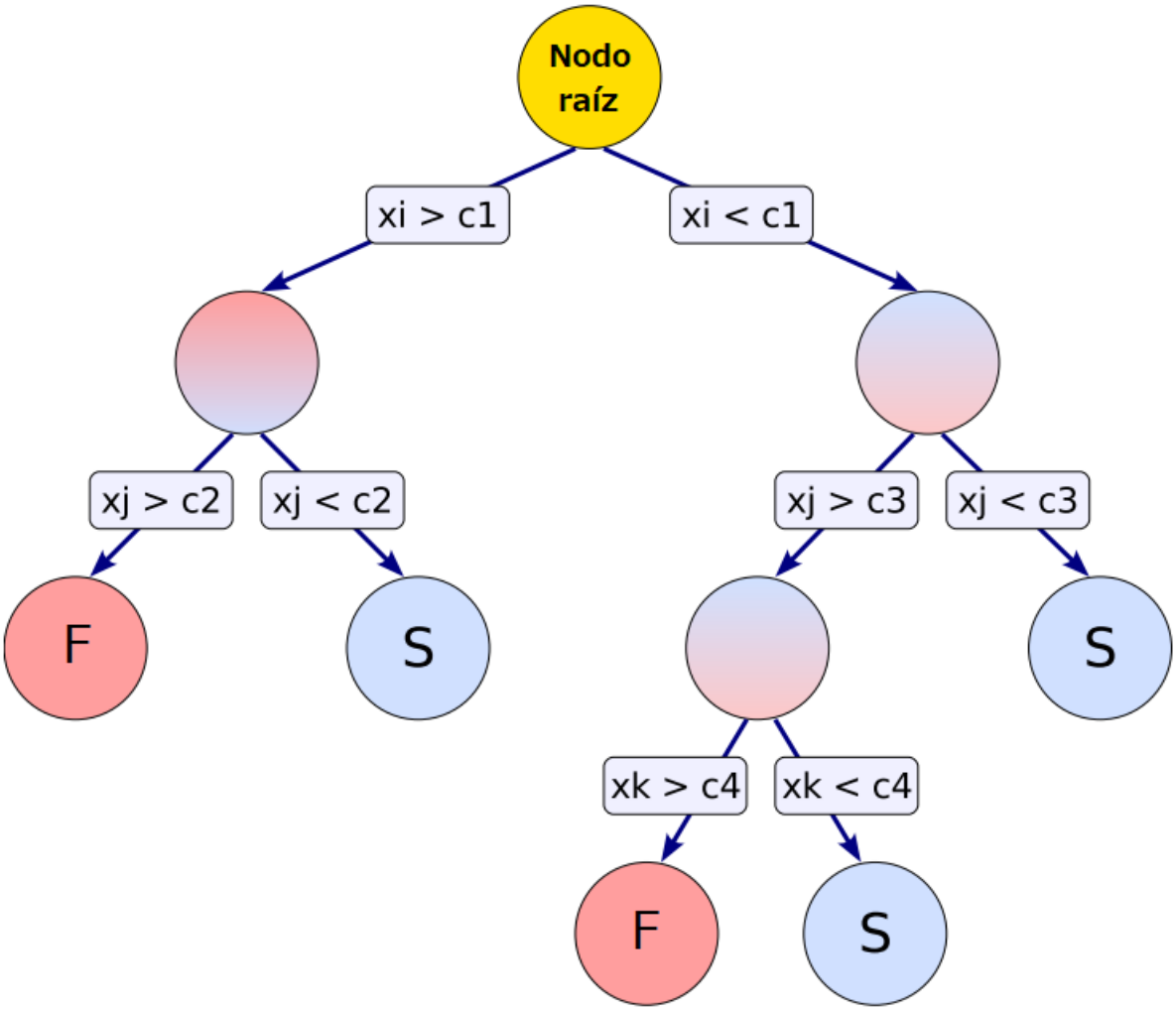


Figura 47: Ejemplo de un árbol de decisiones. Imagen tomada del manual de TMVA[18]

8.2.0.3. BDTD BDTD es la combinación entre BDT y un algoritmo de decorrelación lineal basado en las matrices de correlación. Su ventaja principal es que a diferencia del método BDT solo, la combinación con un algoritmo de decorrelación le permite tratar problemas con variables altamente correlacionadas entre sí y por lo tanto parcialmente redundantes.

En condiciones normales las variables altamente correlacionadas podrían dañar fuertemente al algoritmo BDT ya que cada árbol se compone de decisiones individuales sobre variables y el orden de estas decisiones se calcula a partir de su relevancia individual sobre el resultado final, si se consideran dos decisiones consecutivas en variables no correlacionadas, cada una aporta información nueva a la rama mejorando la decisión. Sin embargo, en el caso de variables correlacionadas ocurren dos cosas, la primera es que ambas variables usualmente aparecerán juntas en la misma rama pues tiene importancias similares y la segunda es que hacer dos cortes no aportará información adicional a la rama generando una rama pobre. Si además el par correlacionado es relevante para el resultado final, esto conducirá a que la gran mayoría de las ramas generadas en los árboles sean ramas pobres.

El algoritmo de decorrelación utilizado se encuentra descrito en el manual de TMVA sección 4.1.2 y consiste en calcular la matriz C'^{β}_{α} tal que la transformación $X'_{\alpha} = C'^{\beta}_{\alpha} X_{\beta}$ lleve las variables correlacionadas X_{β} a las variables independientes X'_{α}.

8.2.0.4. LD El discriminante lineal permite seleccionar eventos al distinguir los valores promedios de las distribuciones de señal y fondo en un espacio de variables transformado donde se las correlaciones entre ellas han sido removidas. En otras palabras, el discriminante lineal trabaja en un espacio decorrelacionado donde busca identificar un eje en el hiper espacio de variables tal que la proyección de las curvas Señal-Fondo en sobre este eje tengan la mayor separación posible (Para detalles sobre el concepto de separación revisé la sección).

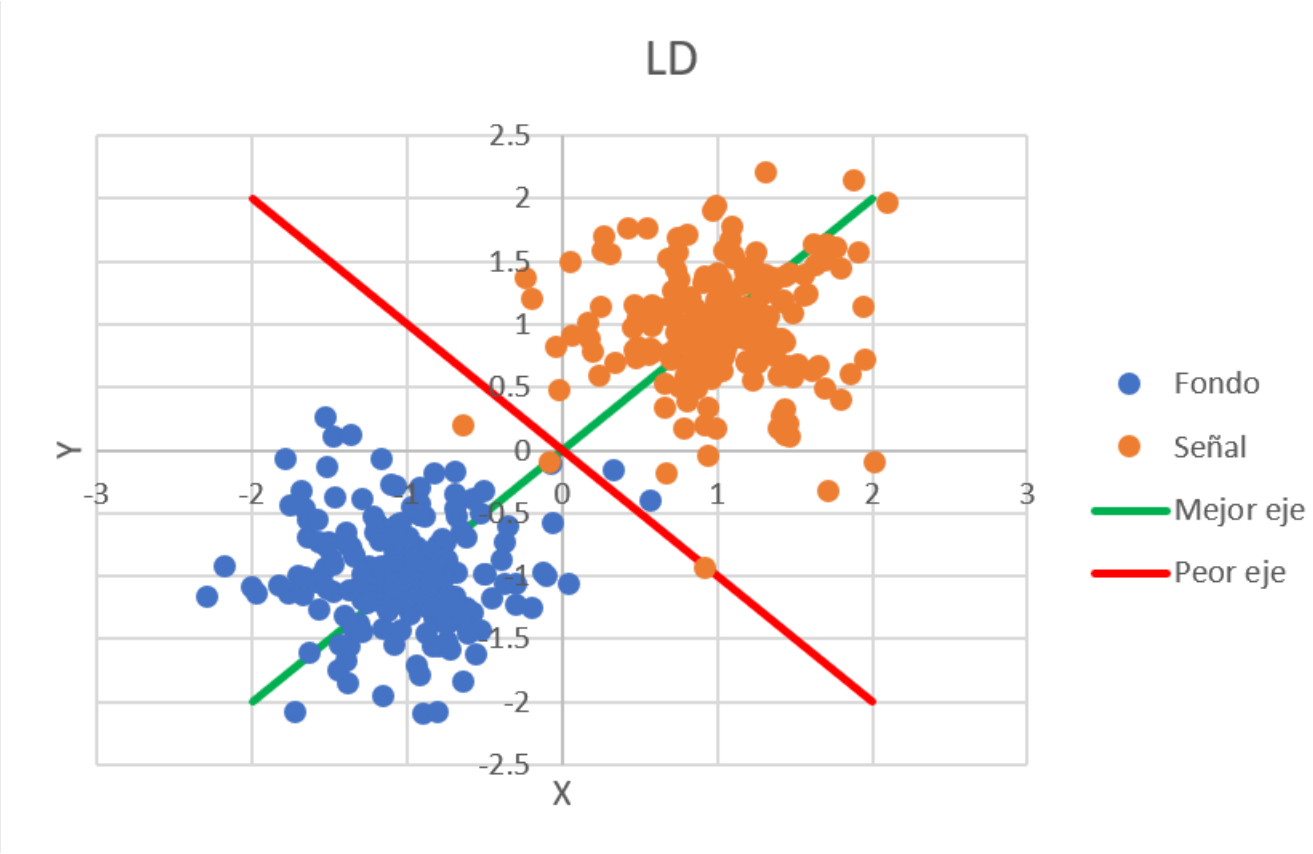


Figura 48: El método LD reduce la dimensión del problema proyectando los datos sobre un solo eje en el espacio multidimensional a analizar. El eje elegido es aquel que logre la mejor separación Señal-Fondo

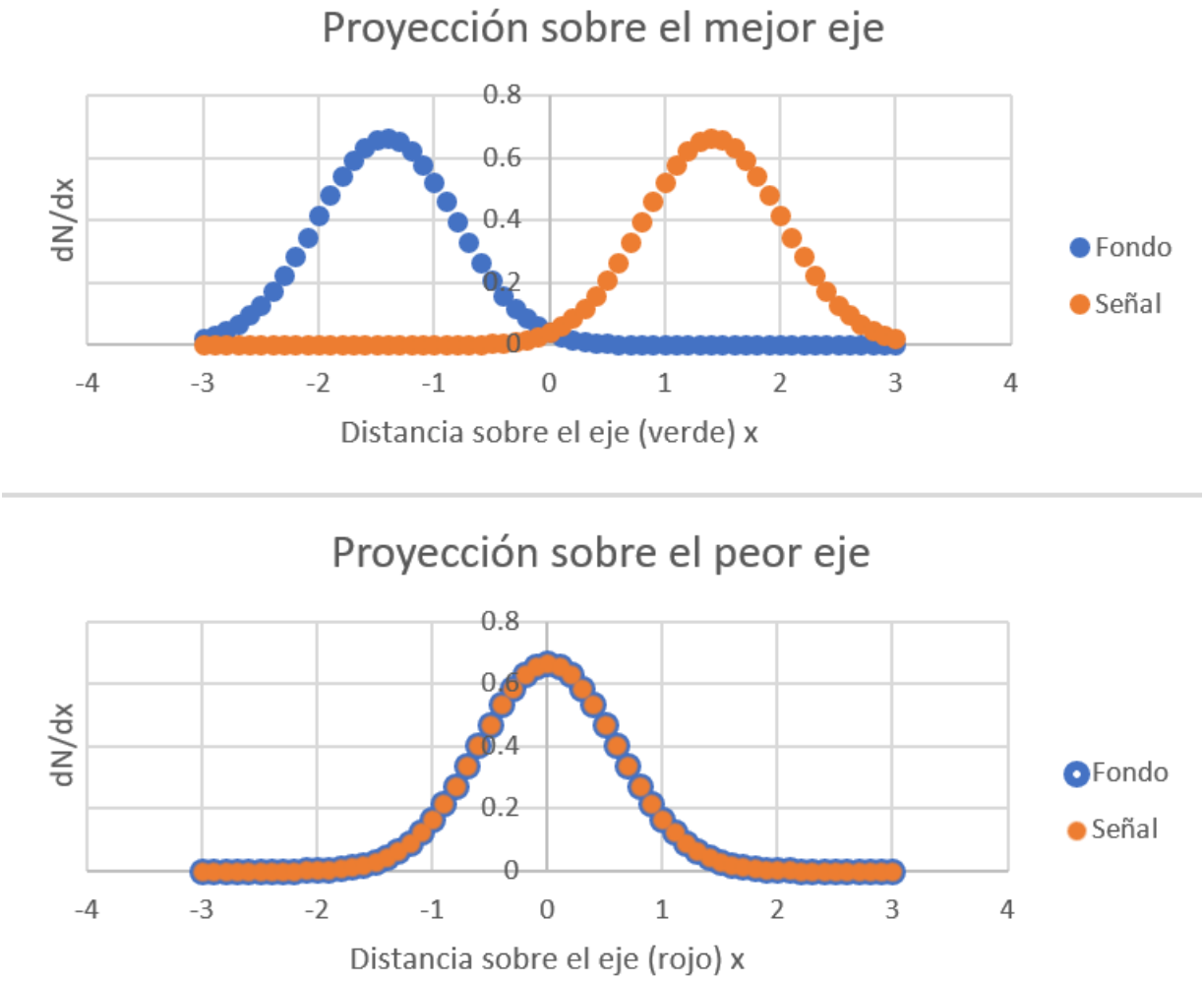


Figura 49: En este ejemplo la separación es ≈ 1 para el mejor eje y 0 para el peor.

El método LD funciona idealmente para variables linealmente correlacionadas con distribuciones gaussiana, cualquier desviación de esta forma disminuirá el poder de separación alcanzable. Adicionalmente es inútil en variables que presenten promedios iguales en las distribuciones de señal y fondo.

NOTA: Es un método diseñado únicamente para problemas lineales (correlación lineal), aplicarlo en problemas no lineales resultará en un desempeño inferior.

8.2.0.5. Likelihood El modelo Likelihood reproduce las distribuciones señal-fondo en las variables de entrada mediante funciones de densidad de probabilidad (PDF). Esta aproximación ignora todas las correlaciones entre las variables de entrada.

Para lograr un buen desempeño es necesario que todas las variables de entrada estén decorrelacionadas. Adicionalmente es necesario un alto número de eventos de entrenamiento para poblar las colas de las distribuciones.

Este método requiere un preajuste por parte del usuario, el cual debe seleccionar los eventos por bin y opciones de suavizado para cada variable.

8.2.0.6. LikelihoodD El problema con Likelihood es que requiere que las variables estén decorrelacionadas y en la mayoría de los problemas reales siempre existe algún grado de correlación sobre todo lineal entre las variables. LikelihoodD resuelve este problema al agregar un algoritmo de decorrelación lineal previo a la ejecución de Likelihood de forma que se garantice la ausencia de correlaciones lineales entre las variables.

8.2.0.7. PDERSD PDERSD es un algoritmo PDERS combinado con algoritmo de decorrelación. A su vez el algoritmo PDERS es una clase modificada del método PDE.

El algoritmo PDE es la generalización del clasificador proyectivo Likekihood a n dimensiones (n variables de entrada). En el caso donde las funciones de densidad de probabilidad PDF multidimensionales fueran conocidas PDE sería óptimo ya que podría explotar toda la información contenida en las variables de entrada, sin embargo en la práctica la cantidad de ejemplos de entrenamiento necesarios para poblar completamente el espacio fase multidimensional es excesiva y por lo tanto se recurre a métodos de estimación conocidos como Kernel que permiten aproximar la forma de las PDF con sustancialmente menos ejemplos de entrenamiento.

El método PDERS es una versión modificada de PDE donde RS significan (*Range search*) búsqueda en un rango finito alrededor del punto a evaluar. Esta aproximación es similar a la de KNN donde se compara cada evento de prueba o evaluación con los eventos de entrenamiento cercanos en el espacio multidimensional. En el caso de PDERS se define un volumen V finito al rededor del evento i a evaluar y se calcula la respuesta del método mediante la siguiente formula:

$$y_{PDE-RS}(i,V) = \frac{1}{1+(\frac{n_B(i,V)N_S}{n_S(i,V)}N_B)} \tag{11}$$

Con N_B y N_S en número total de eventos de fondo y de señal respectivamente en la muestra de entrenamiento completa y $n_B(i,V)$ y $n_S(i,V)$ el número de eventos de fondo y señal en la vecindad de volumen V al rededor del ejemplo de evaluación i. El valor de $y_{PDE-RS}(i,V)$ Alcanza su máximo con valor 1 en la señal pura y su mínimo con valor 0 en el fondo puro. El método de conteo de $n_B(i,V)$ y $n_S(i,V)$ promedia sobre las PDF en V por lo tanto ignora la información sobre la forma dentro y fuera del volumen V.

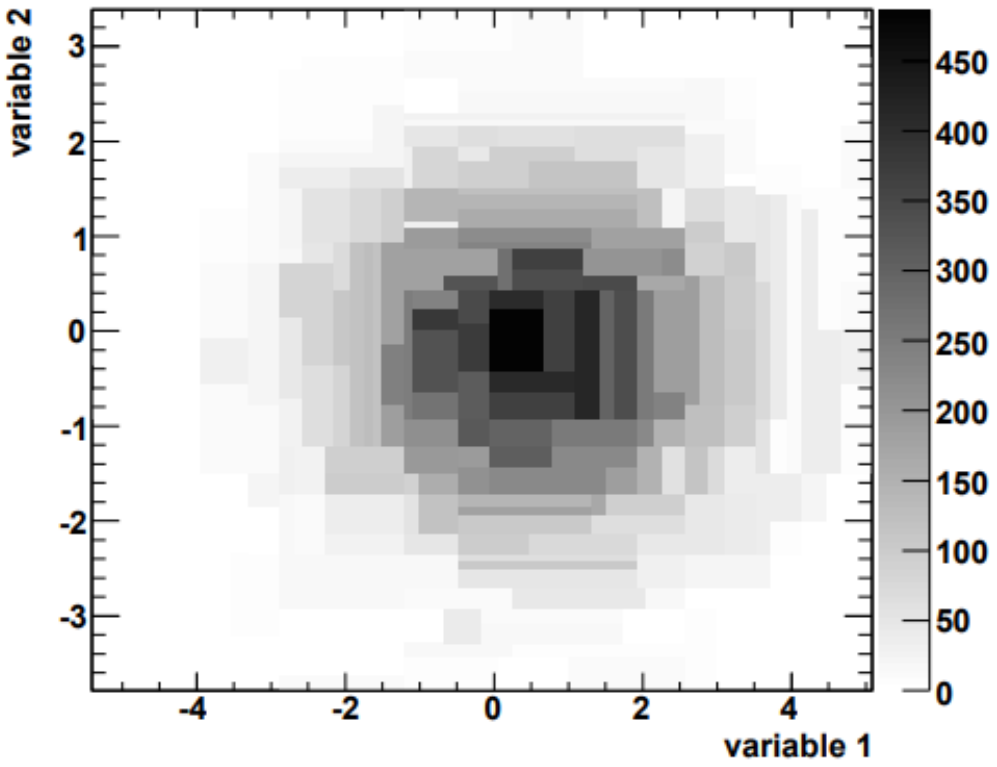
8.2.0.8. PDEFoamBoost PDEFoam se considera una extensión de los métodos PDERS, en PDEFoam se ocupa un histograma multidimensional con bines variables para determinar las densidades de probabilidad en vez de usar un PDF como en PDE o Likelihood, a este histograma se le conoce como *foam*.

El método PDEFoam divide el hiperespacio en K hiperrectángulos (celdas) de densidad constante de eventos, para el caso de clasificación se ocupa o bien densidad similar de señal y fondo o una proporción constante de señal/fondo en cada celda.

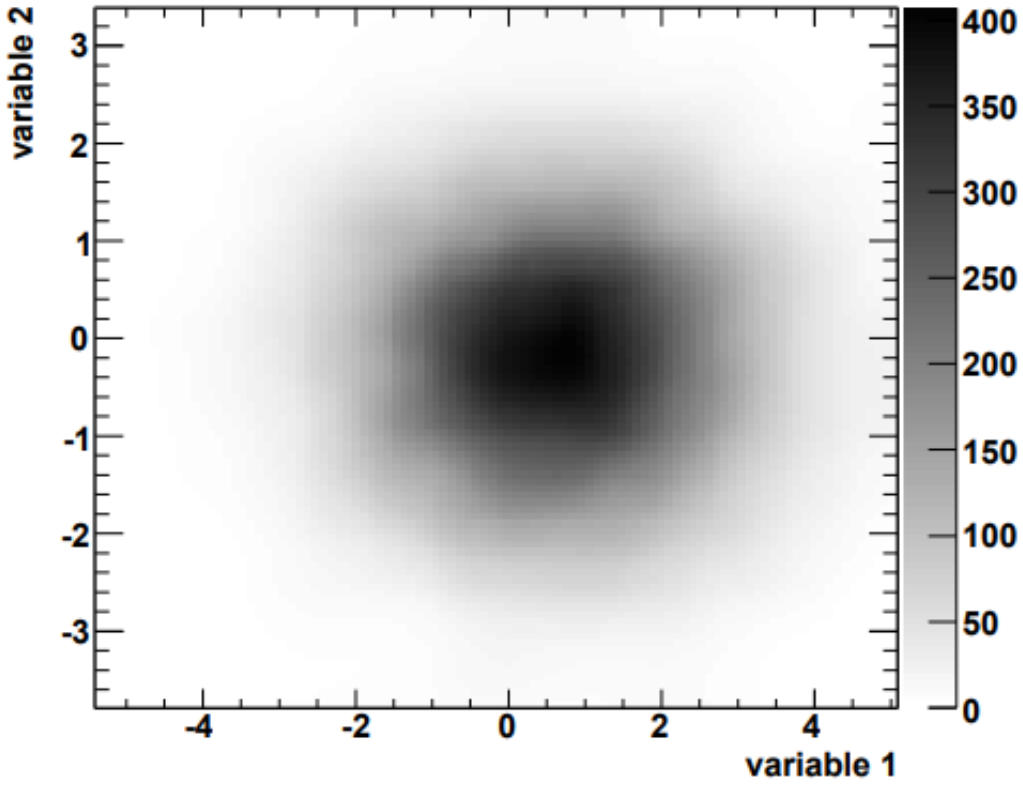
El tamaño de las celdas se ajusta mediante un algoritmo de *binning* donde el tamaño y posición de cada celda se ajustan de forma tal que se minimice la variancia en la densidad de eventos en la celda. La distribución final de densidades se almacena en celdas organizadas en un árbol binario con la finalidad de permitir un almacenamiento y recuperación rápido y eficiente en memoria.

De forma similar a PDERS PDEFoam permite la utilización de Kernels los cuales permiten estimar la distribución de probabilidad usando un número mucho menor de eventos de entrenamiento que los que fuesen necesarios sin el uso del Kernel, la desventaja suele ser un incremento en el tiempo de cómputo.

El siguiente ejemplo (Figura 45) el cual se encuentra en el manual de TMVA pertenece al Kernel Gaussiano el cual tiene como función suavizar la distribución evitando discontinuidades en las fronteras entre celdas.



(a) Sin kernel



(b) Con kernel Gausiano

Figura 50: Comparación entre *foams* generados con y sin Kernel Gaussiano para 5000 eventos y 500 celdas. Imagen tomada del manual de TMVA

8.2.0.9. SVM El modelo SVM para problemas lineales se desarrolló a principio de los 60's sin embargo su contra parte no lineal tardo 30 años más en llegar. El modelo SVM original (lineal) fue diseñado para ser un clasificador en dos clases (señal-fondo) mediante el uso de un hiperplano orientado.

En el modelo de SVM lineal se considera que si los datos sin linealmente separables debe de existir un par vector-escalar $(\vec{w}, b)$ tal que satisfaga:

$$Y_i(\overrightarrow{X_i} \cdot \overrightarrow{w_i} + b) - 1 \geq 0, \forall \tag{12}$$

con $Y_i = \pm 1$ dependiendo si se trata de señal o fondo y la ecuación del hiperplano definida por el par $(\vec{w}, b)$ (Ver figura 51). La forma de determinar los parámetros es tal que se maximicen los márgenes en ambas direcciones, donde el margen se define como $2/|\vec{w}|$ y para el caso general lineal con frontera no ideal (non-separable data) es decir con presencia de una región en la que hay tanto señal como fondo la función de costo es

$$W = \frac{\vec{w}^2}{2} + C \sum_i \zeta_i \tag{13}$$

Donde C es el parámetro de costo el cual escala el costo por clasificación incorrecta permitiendo controlar que la relación entre margen y clasificación incorrecta. En el caso de una frontera perfectamente definida sin mezcla de señal-fondo la suma $\sum_i \zeta_i$ valdría cero y por lo tanto la función de costo no tendría importancia pues la clasificación sería perfecta.

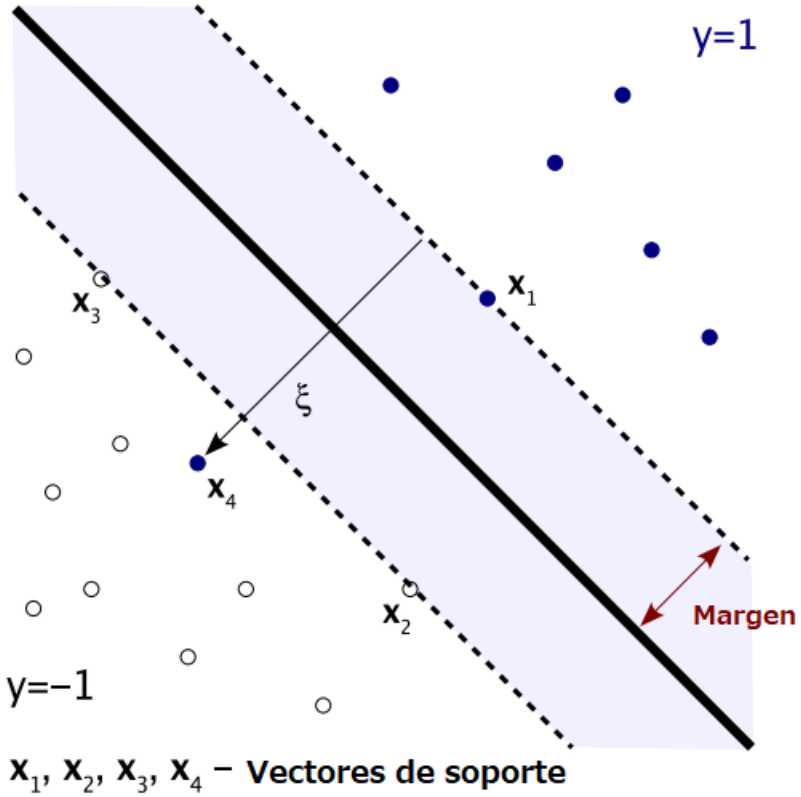


Figura 51: Imagen de los vectores de soporte y el plano de separación en 2 dimensiones para SVM. Tomada del manual de TMVA

El nombre SVM significa máquina de soporte vectorial y toma su nombre de considerar a cada ejemplo de señal o fondo como un vector, en el fondo SVM selecciona un subconjunto de todos los ejemplos de entrenamiento y los ocupa como vectores de soporte a partir de los cuales calcula la función error y ajusta los pesos para generar el hiperplano de separación.

Para resolver casos no lineales es necesario ocupar una función Φ la cual mapea los elementos tal que en el espacio resultante el problema sea linealmente separable y se pueda ocupar el SVM lineal. Sin embargo, determinar la función Φ a partir de los datos de entrenamiento puede resultar imposible o muy complicado por lo que se recurre a un atajo conocido como Kernels.

Durante los cálculos en el lagrangiano de SVM la cantidad utilizada nunca es x_i sola sino $x_i \cdot x_j$ la cual se mapea a $\Phi(x_i) \cdot \phi(x_j)$ por lo que la cantidad transformada de importancia podemos aproximarla mediante el Kernel el cual definimos como $K(\vec{x_i}, \vec{x_i}) \approx \Phi(\vec{x_i}) \cdot \Phi(\vec{x_j})$

Algunos de los Kernels más utilizados para modelar SVM no lineales son el polinomial $K(\vec{x}, \vec{y}) = (\vec{x} \cdot \vec{y} + \Theta)^d$, el gaussiano $K(\vec{x}, \vec{y}) = e^{(\frac{-|\vec{x}-\vec{y}|^2}{2\sigma^2})}$ y el sigmoidal $K(\vec{x}, \vec{y}) = tanh(k(\vec{x} \cdot \vec{y}) + \Theta))$.

8.2.0.10. FDA_{GA} El método FDA permite una aproximación intermedia entre los métodos lineales como LD y los no lineales como SVM o MLPBNN, su objetivo es resolver problemas relativamente simples o problemas parcialmente no lineales.

En FDA el usuario provee una ecuación con parámetros ajustables. FDA permite resolver tanto problemas de clasificación como de regresión, en nuestro caso el interés se encuentra en clasificación. Para el caso de clasificación FDA ajusta los parámetros de la función provista por el usuario de tal forma que el valor de la función corresponda a 1 para señal y 0 para fondo.

La forma de ajuste de parámetros en FDA obedece a la disminución del error definido como:

$$\epsilon = \frac{\sum_{i=1}^{N_S}(F(x_i) - 1)^2 w_i}{W_S} + \frac{\sum_{i=1}^{N_B} F^2(x_i) w_i}{W_B} \tag{14}$$

Para motivos de este trabajo la función utilizada en FDA corresponde a una relación de 1 grado entre las variables $Y = (\sum_{i=1}^{N_{variables}} x_i w_i) + w_0$, lo cual corresponde a ubicar un eje en el hiperespacio tal que la señal y el fondo estén claramente separadas en el. Debido a esto es de esperar que el resultado de FDA con función lineal corresponda cercanamente al resultado de LD, sin embargo, la correspondencia no será exacta ya que ambos algoritmos presentan funciones de error y normalización distintas.

8.2.0.11. KNN KNN es un representante la familia de algoritmos basados en instancias. Esta familia de algoritmos en contraste con los algoritmos paramétricos no plantea ningún modelo para la distribución señal-fondo y por lo tanto puede ajustarse a cualquier tipo de distribución sin importar su linealidad o complejidad.

Los modelos basados en instancias al plantear ningún modelo no requieren de entrenamiento, sin embargo, requieren almacenar una biblioteca con todos los eventos de entrenamiento para consulta posterior. Una de las desventajas de este tipo de métodos es su lentitud en la evaluación y consumo de memoria ambos causados por la necesidad de consultar la biblioteca de eventos de entrenamiento antes de poder hacer una predicción sobre un dato nuevo.

El algoritmo KNN busca los K eventos más cercanos al punto de consulta y mediante votación define el valor de retorno en regresión o la probabilidad en clasificación. La métrica utilizada para definir la cercanía entre dos eventos es la métrica euclídea modificada $R = (\sum_i^{n_{var}} \frac{1}{w_i}|x_i - y_i|^2)^{\frac{1}{2}}$ donde w_i permite asignar escalas a cada dimensión, estos factores de escala entre dimensiones permites comparar variables con distintos rangos como por ejemplo multiplicidad que abarca desde 3 hasta aproximadamente 70 y esferocidad que abarca entre 0 y 1.

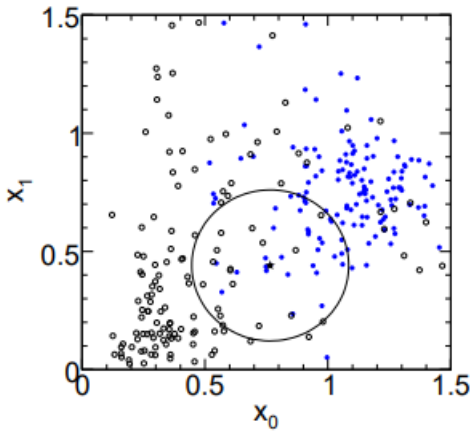


Figura 52: Imagen de KNN en 2 dimensiones tomada del manual de TMVA

En este método se define la probabilidad para un evento particular como $P_S = \frac{k_S}{k}$, donde K es el número fijo de eventos consultados alrededor del punto a evaluar y k_S es el número de estos que corresponden a señal.

En el método KNN el valor de K permite determinar el nivel de sensibilidad del método, un valor bajo de K es adecuado para problemas complejos con fronteras no lineales, sin embargo, un valor bajo de K también implica una mayor tendencia a sobreajustar y menor estadística. Los valores altos de K tiene baja tendencia a realizar sobreajustes y son más estables, sin embargo, tienden a pasar por alto detalles finos de la estructura de la frontera por lo que se les recomienda para problemas más lineales.[30]

8.3. Aplicaciones actuales del aprendizaje de máquina

Los métodos de aprendizaje de máquina tienen múltiples aplicaciones en la actualidad tanto en tareas de regresión como de clasificación. Algunas de las ramas que se ha visto beneficiadas por la aplicación del aprendizaje de máquina son las siguientes:

- Diagnóstico médico de imágenes (tomografías, radiografías, ultrasonido, etc.).
- Finanzas (predicción de precios futuros de productos)
- Banca (Detección de fraude y riesgo crediticio)
- Fondos de inversión. Mediante los algoritmos de determinación de política de inversión los cuales colocan y retiran órdenes de compra y venta de acciones y productos de manera inteligente.
- IA para juegos. Mediante los algoritmos de *Q learning* y derivados ha sido posible crear máquinas capaces de dominar prácticamente cualquier juego de mesa y de vídeo siempre y cuando el espacio de estados y acciones sea finito y las reglas estén bien definidas.
- Asistentes de creación musical. Mediante el aprendizaje de máquina se puede programar a la computadora para crear acompañamientos a piezas musicales basado en ejemplos de un estilo particular de música.
- Generación de voz a partir de texto. Actualmente ya existe un sistema basado en inteligencia artificial el cual es capaz de leer un texto con resultados superiores a los métodos de concatenación o de síntesis los cuales suenan robóticos, el resultado aún es inferior al humano, pero es bastante alentador.
- Sistemas de reconocimiento de voz. Existe una gran variedad de algoritmos en esta rama entre ellos se encuentran los algoritmos de voz a texto, reconocimiento de emociones e identificación del hablante.

- Sistemas de sugerencias. En esta categoría se encuentran sistemas como los usados por empresas como Google, Facebook o Netflix para determinar el contenido a recomendar o anuncios a mostrar de acuerdo a los intereses del usuario y búsquedas recientes.
- Sistemas de traducción. Hasta hace algunos años los asistentes de traducción automática profesionales se encontraban basados en diccionarios de frases y sus traducciones correspondientes, actualmente estos sistemas se han reforzado mediante inteligencia artificial la cual permite traducciones más precisas mediante la consideración del contexto en que se encuentra dicha frase, conduciendo a traducciones más precisas.
- Conducción automática de vehículos. Mediante el aprendizaje de máquina es posible desarrollar elementos específicos de un algoritmo de conducción para uso en un vehículo real como lo son detección de obstáculos, condiciones de terreno, etc.

  Actualmente en ambientes simulados como lo son juegos de carreras o simulación de conducción ha sido posible construir sistemas completamente basados en inteligencia artificial capaces de navegar el mapa sin presentar accidentes.
- Máscaras de selección. Es posible entrenar a un algoritmo de aprendizaje de máquina como las redes neuronales convolucionales para seleccionar regiones de interés en una imagen, usualmente se ocupa para separar una imagen como una cara de una persona o un auto de un fondo el cual presenta un sin fin de variantes.