Universidad Nacional de Luján

Clasificación de flujos de datos continuos y multi etiquetados

Tesina de grado presentada para optar al título de
Licenciado en Sistemas de Información

Juan Cruz Cardona

Director: Santiago Banchero

2021

CLASIFICACIÓN DE FLUJOS DE DATOS CONTINUOS Y MULTI ETIQUETADOS

La clasificación multi-etiquetas es un paradigma de aprendizaje supervisado que generaliza las técnicas clásicas de clasificación para abordar problemas en donde cada instancia de una colección se encuentra asociada a múltiples etiquetas. La mayor parte de los trabajos de investigación han sido realizados en contextos de aprendizaje por *batch*. Los ambientes de flujo continuo de datos (o *streaming*) presentan nuevos desafíos a esta área debido a las limitaciones de tiempo de respuesta y almacenamiento que acarrean. A esto se agrega la naturaleza evolutiva de este tipo de escenarios, que obligan a los algoritmos a adaptarse a cambios de concepto. En la presente investigación se aplican algoritmos de clasificación multi-etiquetas a colecciones estructuradas y no estructuradas. Los experimentos se llevarán a cabo en ambientes simulados de *streaming* de datos para conocer el impacto que produce este contexto sobre los resultados de la clasificación y acoplar el modelo a escenarios del mundo real. A su vez, se partirá de estas colecciones de datos para generar instancias sintéticas y así producir flujos potencialmente infinitos. Por último, se abordarán estrategias de ensambles de algoritmos en búsqueda de una mejora en la calidad de la tarea de predicción de objetos no observados por el modelo. De esta manera, se proveerá a la comunidad de nuevos estudios experimentales sobre algoritmos y colecciones ya conocidos del área de clasificación multi-etiquetas, de manera tal de extender el conocimiento sobre su rendimiento bajo escenarios evolutivos y de naturaleza variable.

Palabras claves: clasificación, multi-etiquetas, *streaming*, algoritmos, flujos.

Índice general

1. INTRODUCCIÓN

1.1. Fundamentos

En los últimos años ha habido un aumento considerable de datos de diversa índole y generados por fuentes heterogéneas. Según los autores Gantz y Reinsel, el volumen total de datos creados y replicados en el mundo durante el año 2011 supera los 1.8 ZB (zettabytes) y se ha estimado que duplica cada dos años [7]. Los avances en el área de Tecnología de la Información (TI) han contribuido a una continua producción de datos y expansión del campo digital, tal es el caso para la red social *Facebook*, la cual recibe cada hora un flujo de 10 millones de fotos que publican sus usuarios [12]. A estas grandes colecciones de datos se las conoce como *big data* y acarrean nuevas oportunidades y desafíos al campo de las ciencias de la computación. En cuestiones económicas, un análisis a gran escala en búsqueda de tendencias en el comportamiento de los usuarios o clientes de un sistema puede dar una ventaja competitiva en el mercado y, en adición, proveer de un servicio valioso a la comunidad. Potencialmente, la *big data* puede ser una fuente que proporcione a la comunidad de conocimiento nuevo sobre el mundo en el que habita, o como ha mencionado Fayyad, Piatetsky-Shapiro y Smyth en su escrito sobre el descubrimiento de conocimiento, "Los datos que percibimos de nuestro ambiente son la evidencia básica que usamos para construir teorías y modelos sobre el universo en el que vivimos"[1].

Sin embargo, volúmenes masivos de datos tornan obsoletos los tradicionales métodos manuales de análisis de datos y surge la necesidad de desarrollar técnicas automatizadas para extraer patrones en los datos y obtener conocimiento. Con este fin, se han desarrollado técnicas en las áreas de minería de datos y aprendizaje de máquinas que abordan estas colecciones en búsqueda de conocimiento válido y útil. Dichas técnicas se han enfocado en el aprendizaje por *batch* [6], lo que significa que el algoritmo dispone de la colección completa, almacenada en disco, y con la cual genera un modelo a partir de una o múltiples iteraciones sobre todos los datos. No obstante, el aprendizaje por *batch* trae aparejada una dificultad en su misma definición: requiere de todos los datos de la colección presentes y accesibles en todo momento, lo cual no siempre es posible. Además se suma una limitante que es clave en el contexto actual de alta disponibilidad de datos: hoy en día una buena parte de los datos generados proviene de flujos continuos o '*streamings*' de datos [2]. Estos flujos son potencialmente ilimitados, arriban de a una instancia por vez, y son analizados con restricciones altas de tiempo de procesamiento y de memoria. Tal es el caso para aplicaciones de sensores, monitoreo de redes y administración de tráfico, flujo de clics de un usuario en la web, redes sociales, entre otros. Los algoritmos de aprendizaje que actúen en este entorno dinámico deben contar con mecanismos que permitan manejar cambios en la naturaleza o distribución de los datos, tanto para incorporar datos nuevos, como para descartar los datos antiguos. Por estas razones, se torna necesario que las aplicaciones basadas en clasificación en tiempo real adapten sus operaciones de entrenamiento y predicción para lograr mejores resultados [17].

Dentro del área de minería de datos, una de las principales tareas es la de clasificación, la cual consiste en entrenar un modelo que sea capaz de asignar una única etiqueta a una instancia desconocida. No obstante, existen problemas de clasificación en donde múltiples etiquetas son necesarias para caracterizar una instancia. Por ejemplo, una noticia de diario

[1] "Data we capture about our environment are the basic evidence we use to build theories and models of the universe we live in" [5, p. 2]. Traducción propia.

referida al accidente aéreo que sufrió el plantel de fútbol del club Chapecoense puede ser clasificado en la categoría de "Fútbol" tanto como en la de "Tragedias". Del mismo modo, un video documental sobre la vida de Borges puede anotarse como "Biografía", "Literatura" o incluso "Buenos Aires" si se mostraran imágenes de la ciudad. Este tipo de problemas es llamado Clasificación multi-etiquetas (MLL) [2] y representa un nuevo paradigma de aprendizaje automático, con sus propios retos por afrontar y que aún no ha sido suficientemente explorado en proyectos de investigación.

Una clasificación multi-etiqueta permite conocer el grado de correlación entre una instancia de la colección y una o más etiquetas. Esta cualidad significa un mayor poder de generalización con respecto a la clasificación tradicional de única etiqueta, ya que puede abarcar esos mismos problemas y otros de mayor número de etiquetas. Además existen algoritmos que aprovechan la correlación entre etiquetas para mejorar la eficiencia de la clasificación y la calidad de la predicción.

El campo de MLL se ha desarrollado considerablemente en los últimos años pero hasta el momento muchos de estos trabajos se han llevado a cabo en ambientes estáticos de aprendizaje por *batch* [15], en consecuencia, se hace necesario encarar nuevos proyectos que aborden clasificaciones MLL en contextos de *streaming* de datos. El desafío entonces consiste en crear clasificadores que sean capaces de manejar un inmenso número de instancias y adaptarse al cambio, a la vez que estar preparados para hacer tareas de predicción en cualquier momento, y todo esto en un contexto de altas restricciones de tiempo de respuesta y memoria.

1.2. Descripción del tema de estudio

1.2.1. Clasificación Multi-etiquetas

Tradicionalmente, el aprendizaje supervisado ha consistido en asociar una instancia o ejemplo a una única etiqueta. Dicho ejemplo es una representación de un objeto del mundo real, y por lo tanto, consta de características o *features* particulares. La etiqueta corresponde a un significado semántico o concepto que lo caracteriza. La tarea de clasificación entonces, reside en aprender una función que permita enlazar ejemplos no observados con una etiqueta. Es preciso notar aquí que dicha definición encubre la restricción de que cada instancia pertenece a una única etiqueta, o dicho de otra manera, cada objeto del mundo real se asocia a un único concepto y ningún otro. Sin embargo, existen problemas de clasificación donde más de una etiqueta puede ser asignada a un ejemplo. La anterior presunción no se amolda a problemas complejos donde un objeto pueda tener más de un significado simultáneamente.

Tareas de este tipo pueden surgir en áreas como las de categorización de texto, recuperación de información musical, clasificación semántica de escenas, anotación automática de videos o clasificación de genes y funciones proteicas. A modo de ejemplo, en el campo mencionado de clasificación semántica de escenas, la foto de un paisaje que ilustra una montaña y una playa puede asociarse a las categorías de 'playa' y 'montaña', simultáneamente [8]; en bioinformática, cada gen puede ser asociado a clases según su función, tales como 'metabolismo', 'transcripción' o 'síntesis proteica' [21]; por último, en recuperación de información musical una pieza sinfónica puede tener *tags* como 'Mozart', 'piano' o 'clásica'.

Este nuevo paradigma es llamado 'Clasificación multi-etiquetas' y ataca problemas con las siguientes características [8]:

[2] Siglas provenientes de su abreviación en inglés, Multi-label learning

- El conjunto de etiquetas es previamente definido y tiene un significado interpretable por un humano.
- El número de etiquetas es limitado y no mayor que el número de atributos.
- En caso que el número de atributos sea grande, se debe poder aplicar estrategias de reducción de atributos.
- El número de ejemplos puede ser grande.
- Las etiquetas pueden estar correlacionadas. Esto significa que se pueden aplicar técnicas que exploten estas relaciones con el objetivo de reducir los tiempo de procesamiento de los algoritmos.
- La distribución de los datos puede estar desbalanceada, es decir, que una etiqueta puede tener un mayor número de ejemplos que otras.

Asimismo, surge un desafío a superar: el conjunto de etiquetas posible crece exponencialmente ante cada nueva adición de una etiqueta. Por ejemplo, si se tuvieran 20 etiquetas, la cantidad posible de conjuntos de etiquetas distintos excedería el millón (2^{20}). Esto implica un tamaño exorbitante del espacio de salida y, en consecuencia, costos computacionales altos. En ese sentido, se ha buscado desarrollar algoritmos que aprovechan las correlaciones o dependencias entre etiquetas. Por ejemplo, la probabilidad de que una noticia que contiene los términos 'pelota' y 'gol' sea anotada con la etiqueta 'fútbol' sería mayor que si se etiquetara con la etiqueta 'tenis'. Zhang y Zhang clasifican estos algoritmos en tres grupos según la estrategia de correlación aplicada [21]:

Estrategia de primer orden La tarea de MLL es dividida en 'q' tareas de clasificación binarias, siendo 'q' el número de etiquetas de la colección.

Estrategia de segundo orden La tarea de MLL se basa en la generación de relaciones de a pares de etiquetas ya sea por *rankings* entre clases relevantes y no relevantes o por interacción entre pares de etiquetas.

Estrategia de alto orden La tarea de MLL considera relaciones de alto orden entre etiquetas.

Las estrategias de primer orden son conceptualmente simples y eficientes pero logran resultados de menor calidad ya que no consideran correlaciones. Las estrategias de segundo orden tienen un mayor poder de generalización pero no todos los problemas de MLL pueden ser abarcados. El último grupo, por su parte, modela las correlaciones más potentes pero conlleva un costo computacional alto.

En el último tiempo, muchos son los algoritmos que han sido desarrollados para atacar el problema de la clasificación MLL. La comunidad de investigación ha aceptado la taxonomía definida por Tsoumakas y Katakis, para estudiar y clasificar los distintos algoritmos de la literatura [8]. La misma propone dos grandes grupos:

Métodos de transformación del problema Este tipo de algoritmos transforman el problema de clasificación MLL en un problema de clasificación tradicional. Ejemplos típicos de este grupo son Binary Relevance (BR) [18] y Classifier Chains (CC) [15]. Ambos convierten la tarea en una de clasificación binaria.

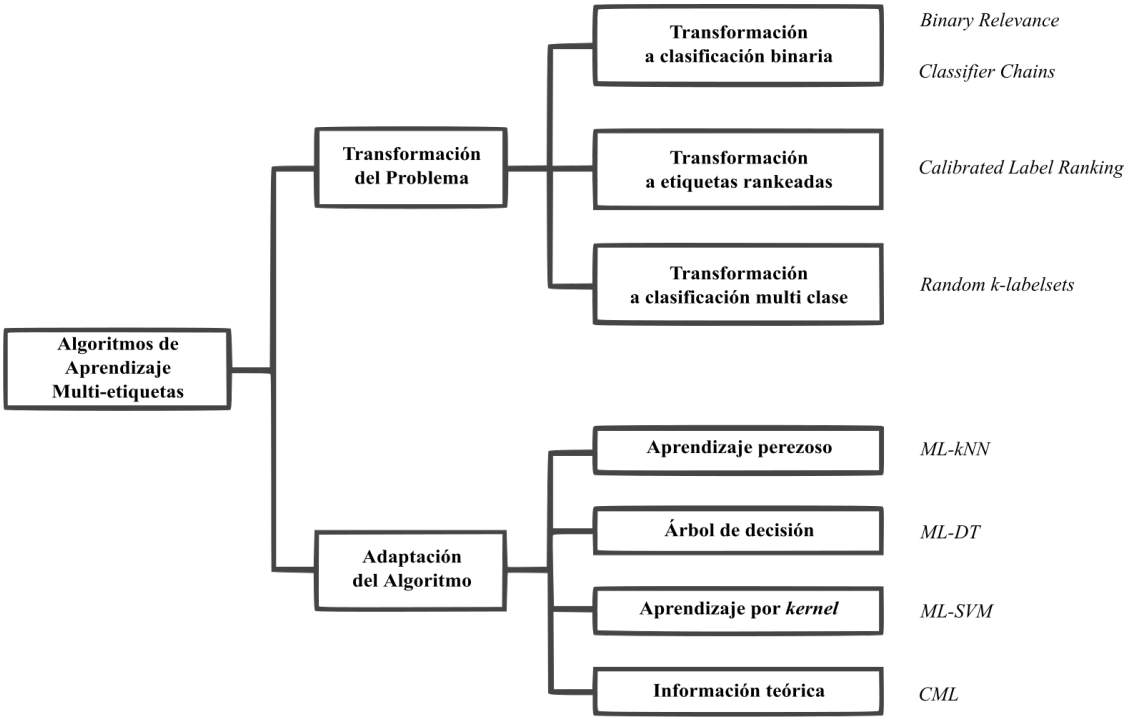

Fig. 1.1: Categorización de los algoritmos de MLL más representativos

Métodos de adaptación del algoritmo Son algoritmos que toman métodos tradicionales de aprendizaje, como árboles de decisión o *naive bayes*, y los adaptan a la tarea de clasificación MLL.

A modo ilustrativo, la figura 1.1 es un diagrama de la taxonomía de algoritmos confeccionado por Zhang y Zhou [22]

1.2.2. Flujos continuos de datos

Hoy en día, los datos pueden ser generados por elementos de continuo monitoreo del medio, tales como sensores o archivos de registro. La clasificación de *streaming* de datos se enfoca en problemas de este tipo, en donde objetos del mundo real son analizados en tiempo real. En este tipo de escenarios, los datos deben cumplir con las siguientes características [6]:

- Los datos están disponibles a través de flujos continuos e ilimitados en el tiempo.
- Las regularidades subyacentes de los datos no son estacionarias sino que pueden evolucionar.
- La data ya no es considerada independiente e idénticamente distribuida.
- La data está situada tanto en el espacio como en el tiempo.

Para hacer frente a estas características de los datos, los sistemas o algoritmos deben contar con los siguientes requerimientos [10]:

- El tiempo para procesar cada registro debe ser constante y pequeño.

- Debe usar un tamaño fijo de memoria principal y no dependiente de la cantidad de registros ya procesados.
- El modelo es generado a partir de una única pasada sobre los datos.
- Debe contar con un modelo listo para realizar predicciones en cualquier momento.
- Idealmente deberá producir un modelo equivalente o casi idéntico al que hubiera sido producido en un ambiente de *batch*.
- Debe mantenerse actualizado ante evoluciones o derivas de concepto en los datos.

Los algoritmos que cumplen con estas cualidades son llamados algoritmos de aprendizaje adaptativo y, aplicados a tareas de clasificación como el de multi-etiquetas, pueden aprovecharse para entrenar un mayor número de objetos y predecir en cualquier instante.

1.3. Motivación

Ante la necesidad de hacer frente a un contexto global de generación masiva de datos y a ritmo acelerado, se hace preciso fortalecer las técnicas de aprendizaje automático actualmente presentes en el campo. En este escenario ya no es posible contar con todos los datos almacenados físicamente y la idea de generar un modelo completo para luego evaluarlo en una fase posterior debe ser reemplazada por una en donde el modelo esté siempre listo para realizar predicciones y al mismo tiempo ser capaz de re-entrenarse y recalcular las métricas de evaluación ante cada nueva instancia abordada. Todo esto en un contexto cambiante, de alta disponibilidad y de limitación en el espacio de almacenamiento. Si bien existen métodos de clasificación para flujos continuos que han dado resultados satisfactorios, aún es un campo conveniente de ser abordado. Asímismo, reproducir los experimentos realizados y fortalecer las técnicas y herramientas actuales puede ser beneficioso para lograr estudios precisos y pormenorizados que sean valiosos tanto para la comunidad científica en sí misma, como también para la sociedad en general.

Por otro lado, si bien existen en el mundo real infinidad de datos multi etiquetados aún no es posible hallar colecciones disponibles al público que cuenten con todas las características de un flujo continuo de datos. Uno de los enfoques abordados es convertir las colecciones existentes en flujos tales que arriben en conjuntos predefinidos y a lo largo del tiempo. De esta manera los algoritmos pueden ser utilizados para realizar clasificaciones en un ambiente similar al de un escenario de *streaming*. Sin embargo, estas colecciones tienen un número limitado de instancias y por lo tanto no cumplen con la condición de ser teóricamente infinitos. Es entonces aquí donde surgen las técnicas de generación sintética de instancias, que buscan reproducir la distribución subyacente de los datos para simular colecciones de datos del mundo real. La contracara de este enfoque es que, si bien existen técnicas y herramientas para generar datos etiquetados, buena parte de ellos son solo aplicables para instancias de una única etiqueta y los que logran generar datos multi etiquetados no han sido lo suficientemente explorados en el área. Hasta el momento, los generadores de instancias multi etiquetadas son capaces de generar datos cercanos a los de colecciones del mundo real [14] y brindan la posibilidad de realizar estudios relativamente certeros de algoritmos de clasificación [16]. No obstante, debe notarse también que si bien se han obtenido colecciones sintéticas en sí mismas aún no han logrado generar instancias para una colección en concreto, respetando sus cualidades particulares y que las distinguen de otras, tales como la co-ocurrencia de etiquetas, la densidad y cardinalidad de las etiquetas y la relación entre las etiquetas y sus *features*, por mencionar algunas. De

lograr esta aproximación se podrán realizar estudios sobre el impacto de los algoritmos sobre flujos de datos de naturaleza distintiva, o en otras palabras, entender en qué medida un algoritmo es más apropiado que otro para un conjunto de datos en un determinado contexto.

Esta última idea mencionada, es decir, el ser capaz de hallar las fortalezas y debilidades de un algoritmo de MLL en un contexto determinado es clave para evaluar la clasificación y entender los resultados obtenidos. Estudios como el de Sousa y Gama [17] o el de Read y col. [16] se han topado con que algoritmos de menor complejidad pueden ser competitivos o incluso superar las métricas de otros algoritmos más complejos. Esta variabilidad en los resultados no solo contrae la necesidad antes mencionada de realizar más estudios al respecto, sino que también abre las puertas a incursionar en soluciones de ensambles de algoritmos. Estos ensambles han dado probada muestra de potenciarse ante la diversidad de resultados obtenidos por sus estimadores base [13], ya que son capaces de disminuir el error total mediante estrategias combinativas. De cualquier manera, las estrategias de ensamble existentes para flujos continuos no han recibido la misma atención que aquellas aplicadas sobre ambientes de *batch* y queda mucho camino por recorrer.

1.4. Objetivos

A partir del marco planteado, el presente trabajo tiene por objetivo principal realizar estudios sobre el impacto de distintos algoritmos de clasificación sobre datos multietiquetados en ambientes de *streaming* provenientes de distintas fuentes de origen y de distinta naturaleza, en particular se seleccionan colecciones de datos que son puntos de referencia en la literatura y que poseen características distintivas entre sí, tales como el número de etiquetas, el número de *features*, o la cantidad de instancias. A su vez, es necesario convertir estos datos a flujos continuos y a este fin se generan instancias sintéticas que sean fieles a estas características mencionadas, aplicando técnicas existentes pero también extendiéndolas para detectar co-ocurrencias entre etiquetas. De esta manera, se buscan obtener representaciones óptimas de las colecciones. Finalmente, se llevarán a cabo clasificaciones con algoritmos clásicos y con soluciones de ensambles, en búsqueda de maximizar los valores de las métricas de evaluación en cada escenario. Adicionalmente, se diseñan distintas configuraciones de ensambles, variando los estimadores base y probando distintas implementaciones. Con esto último en mente, se desarrolla una versión del algoritmo de mayoría de voto para el lenguaje Python y se compara su rendimiento contra el de las implementaciones de Java .

describir qué es este algoritmo de mayoría de voto

Con esto en mente, se listan a continuación los objetivos particulares del trabajo:

- Obtener colecciones de datos que cumplan con las propiedades requeridas para considerarse un flujo continuo. Las características deben variar entre colecciones. Cada colección debe tener las instancias propias del juego de datos e instancias sintéticas potencialmente ilimitadas.
- Generar flujos continuos de datos a partir de la colección proporcionada replicando su número de etiquetas, atributos e instancias, su cardinalidad y densidad de etiquetas y la co-ocurrencia entre dos etiquetas.

referencia bibliografica

- Ejecutar algoritmos de clasificación de MLL, para obtener modelos y realizar evaluaciones sobre los resultados. Todo esto sobre distintos escenarios de flujos continuos.
- Proponer una solución de ensambles a partir de la combinación de algoritmos seleccionados de la literatura.

no estan los experimentos todavía, este punto podría cambiar

1.5. Aportes

El presente trabajo de investigación aborda el campo de aprendizaje por multi-etiquetas a partir de la experimentación y evaluación de técnicas y algoritmos de la literatura sobre colecciones de naturaleza cambiante. MLL es un paradigma emergente de aprendizaje supervisado cuyas características implícitas abren paso a nuevos desafíos que derivan del crecimiento exponencial de etiquetas y sus combinaciones, y del costo computacional de entrenar y consultar el modelo. También suelen presentarse otras propiedades como la alta dimensionalidad, data evolutiva y desbalanceada o dependencia entre etiquetas, las cuales implican una re-significación de las técnicas y métodos tradicionales del área de minería de datos.

El paradigma de MLL ha dado muestras de su eficiencia en términos de tiempos de configuración y ejecución de las tareas, bajo diversos campos de aplicación tales como los de categorización de texto, diagnósticos médicos, minería de redes sociales o análisis de datos químicos, y se mantiene en constante expansión hacia nuevos dominios de aplicación. Asimismo, su continua integración a problemas de diversa naturaleza ha contribuido a alimentar esta tendencia.

La tarea de clasificación de *streaming* de datos se enfoca en problemas donde objetos del mundo real son generados y procesados en tiempo real. Datos de este tipo, y que además poseen múltiples etiquetas, son frecuentes en escenarios del mundo real tal como sitios de publicación de imágenes, correos electrónicos o portales de noticias. Abordar este tipo de problemas implica que los algoritmos sean capaces de identificar cambios de concepto en los datos y adaptarse al nuevo contexto. De lograr esto, se podrá generar modelos más sólidos, ya que se cuenta con un mayor número de objetos, y que se encontrarán aptos para predecir en cualquier momento. Surge entonces el reto de crear clasificadores que actúen en ambientes de altas restricciones computacionales y sean capaces de manejar un inmenso número de instancias, lidiar con evoluciones en los datos y estar listos para resolver tareas de predicción en tiempo real.

A diferencia de otros trabajos de investigación recientes, este proyecto lleva a cabo estudios experimentales sobre el tema de clasificaciones multi-etiquetas, para hallar las fortalezas y debilidades de distintos algoritmos de aprendizaje sobre distintos tipos de colecciones, con miras a aportar de un mayor conocimiento empírico sobre el tema a la comunidad científica especializada en tareas de clasificación de flujos de datos multi-etiquetados. El presente trabajo espera contribuir al estudio de métodos y técnicas asentadas, pero también examinar algoritmos exitosos del campo de aprendizaje por *batch*, particularmente los de ensambles de estimadores, a fin de extender su funcionalidad a ambientes de flujos continuos, analizar su desempeño y determinar en qué medida son aptos o no para este tipo de ambientes.

1.6. Organización del Trabajo

Describir las distintas secciones del trabajo

2. PRELIMINARES

En esta sección se presenta el marco teórico de este trabajo, dando un panorama general de cada una de las disciplinas abordadas e introduciendo los conceptos básicos y fundamentales para entender el proyecto. Se comienza con la definición de la taxonomía del campo de estudio, luego

describir las siguientes secciones

2.1. Taxonomía del Campo de Estudio

En pocas palabras, el presente trabajo de investigación se enmarca en las áreas de *big data* y minería de datos, con aplicación en escenarios de *streaming* o flujos continuos de datos y abordando clasificaciones multi-etiquetas. También se aprovechan técnicas del área de procesamiento de lenguaje natural para tratar corpus de texto libre y extraer *features* o características representativas de los datos.

La figura 2.1 es un esquema que ilustra la taxonomía del campo de estudio y la interrelación entre las áreas de investigación involucradas.

2.2. Aprendizaje Automático

El aprendizaje automático, también conocido por su término en ingles "*Machine Learning*", se enmarca dentro del área de la Inteligencia Artificial (IA) y estudia cómo las computadoras pueden "aprender" o mejorar su rendimiento meramente a partir de datos y sin la intervención de un ser humano. La idea detrás de esta disciplina es lograr reconocer patrones subyacentes en los datos y tomar decisiones en base a estos. Por ejemplo, un problema de aprendizaje automático es el de reconocer dígitos escritos a mano a partir de un conjunto de ejemplos (ver figura 2.2). Aquí se tienen un conjunto de imágenes, cada una representando un dígito del 0 al 9, y el objetivo es construir un modelo que sea capaz de detectar de qué dígito se trata. Otro ejemplo es el de hallar documentos de texto que son relevantes a una consulta del usuario. En este caso el modelo recibe un conjunto acotado de términos, los cuales describen una necesidad de información del usuario, y el modelo debe ser capaz de retornar los documentos que satisfacen la consulta.

Estos problemas se suelen categorizar en aprendizaje supervisado o no supervisado, de acuerdo a si se conoce o no de antemano el concepto o etiqueta que define a los datos. Se desarrollará más sobre este punto en las próximas secciones. De entre los problemas de aprendizaje supervisado se destaca aquí el de clasificación, el cual será descrito a continuación.

2.3. Clasificación

La clasificación es una tarea de minería de datos muy popular que consiste en hallar modelos que describen la o las clases intrínsecas de los datos. La clase corresponde a un concepto que representa al dato y es una etiqueta categórica, es decir, un valor discreto de entre un conjunto de valores previamente conocidos. Estos modelos, también llamados clasificadores, son capaces de predecir la clase a la que corresponden datos previamente desconocidos. Por ejemplo, se puede construir un modelo de clasificación para categorizar nuevos correos electrónicos de acuerdo a si se trata de correo basura (también conocido como "*spam*") o no. Dicho análisis puede ayudar a obtener un mayor entendimiento de

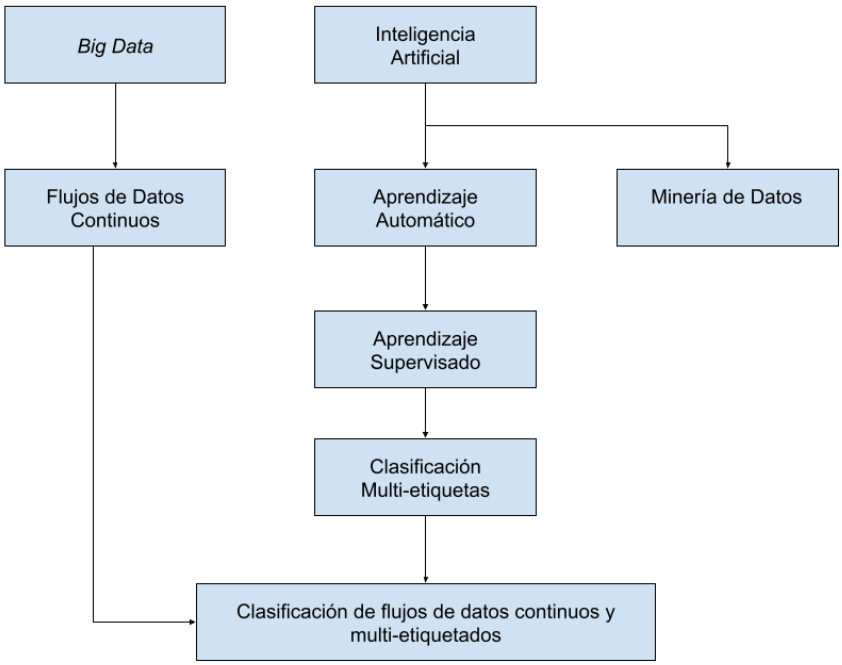

Fig. 2.1: Taxonomía del campo de estudio.

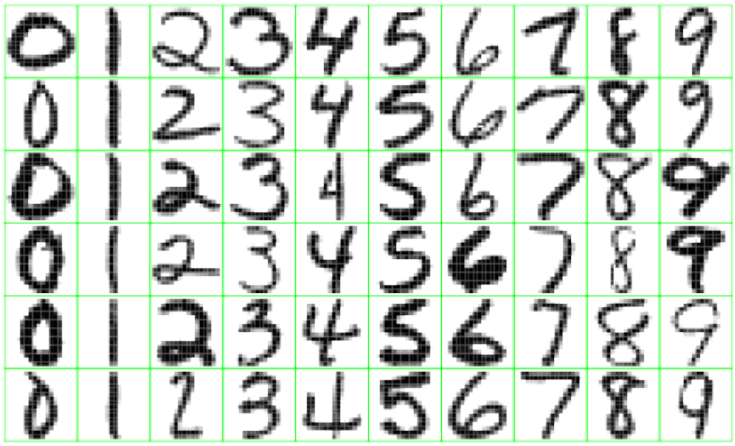

Fig. 2.2: Dígitos escritos a mano. Fuente: *The Elements of Statistical Learning* (2009).

los datos a alto nivel. Las tareas de clasificación han sido aplicadas en áreas tales como las de aprendizaje automático, reconocimiento de patrones o estadística. En un principio, buena parte de los algoritmos se ejecutaban en memoria, con la limitación de espacio de almacenamiento que eso conlleva. Investigaciones más recientes han desarrollado técnicas para escalar los algoritmos de tal manera que puedan manejar datos de mayor tamaño, alojados en memoria, en disco o procesados bajo demanda. Las aplicaciones para este tipo de tareas son numerosas y entre ellas se encuentran las de detectar fraudes o realizar diagnósticos médicos, entre otras.

La clasificación de datos consta de dos etapas, una de aprendizaje y otra de clasificación o predicción. Durante la tarea de aprendizaje se construye el modelo de clasificación el cual describe un determinado número de clases o conceptos. También se conoce esta etapa como la de entrenamiento ya que se selecciona un subconjunto de los datos, llamado conjunto de entrenamiento, que consta de instancias o tuplas seleccionadas aleatoriamente y con una o más etiquetas asociadas. Formalmente, el problema de clasificación puede ser formulado de la siguiente manera. Se recibe un conjunto etiquetado de instancias, tupas o ejemplos de la forma (X, y) donde cada tupla es un vector $X = (x_1, x_2, ..., x_n)$, siendo cada valor una característica distintiva, atributo o feature de la instancia. El vector y por su parte toma un valor de entre n clases diferentes.

Este tipo de tareas se engloban dentro del campo de aprendizaje supervisado ya que para cada instancia la etiqueta es conocida de antemano, y es aprovechada para guiar o, siguiendo la metáfora, "supervisar" el aprendizaje del clasificador. Esta es la diferencia principal contra algoritmos de aprendizaje no supervisado, en los cuales la etiqueta no es conocida y se deben aplicar técnicas para salvar esta restricción.

La primera etapa de una clasificación puede ser vista también como el aprendizaje de una función $y = f(X)$ que pueda predecir la clase y para una tupla X. Por ejemplo, X podría ser un mensaje de correo y la etiqueta y la decisión de si se trata de un correo basura o no. Desde esta perspectiva queremos aprender una función que sea capaz de distinguir las clases subyacentes. Usualmente, esta asociación es llevada a cabo por algoritmos de aprendizaje, los cuales internamente usan funciones matemáticas o reglas de decisión. Algunos ejemplos de este tipo de algoritmos son los árboles de decisión, *naive* bayes, perceptrón, entre otros.

En la segunda etapa el modelo es usado para clasificar. En primer lugar, se calcula una métrica de evaluación, tal como la exactitud o *accuracy*. Durante la etapa de entrenamiento esta estimación puede ser imprecisa, tomando un valor que tiende a ser "optimista" o que da un valor de exactitud mayor al rendimiento real. Esto sucede porque el clasificador puede llegar a incorporar anomalías particulares en el conjunto de datos de entrenamiento. Este fenómeno es llamado sobreajuste u "*overfit*" y una técnica para reducirlo es separar de entre los datos un subconjunto de prueba o de *testing* que no se usa durante el entrenamiento y a partir del cual se realizan predicciones y se calculan las métricas de evaluación. En este contexto, la tarea de evaluación es fundamental ya que es la vía a partir de la cual se determina qué algoritmos o técnicas son más apropiados que otros para un problema en particular. Además provee la información necesaria para corregir o ajustar los parámetros de los algoritmos y así obtener modelos más robustos.

En definitiva, ambos pasos se aplican consecutivamente con el objetivo de lograr hallar un modelo capaz de predecir etiquetas en instancias nuevas y desconocidas.

Retomando la primera etapa de la clasificación, se describen a continuación algunos algoritmos para generar modelos y que son particularmente relevantes para este trabajo de investigación.

2.3.1. *Naive* Bayes

Naive Bayes es un modelo de clasificación computacionalmente simple pero cuyo rendimiento es competitivo contra otros modelos más complejos. Se dice que es un clasificador estadístico ya que se basa en el teorema de Bayes. La idea es computar una probabilidad para cada una de las clases, basada en los atributos de la instancia y seleccionar aquella de mayor probabilidad. El término "*naive*" es el inglés para el término "*ingenuo*" y nace de la presunción que hace el algoritmo de que los atributos son independientes entre sí, o condicionalmente independientes. Esta presunción raramente se cumple en los escenarios donde se aplica pero contribuye a su simplicidad computacional y a su velocidad durante el entrenamiento.

El teorema de Bayes se define formalmente de la siguiente manera:

$$P(H \mid X) = \frac{P(X \mid H)P(H)}{P(X)} \tag{2.1}$$

En esta ecuación, el vector X es una tupla definida tal como en la sección anterior y en términos bayesianos representa la "evidencia". $P(X)$, por lo tanto, es la probabilidad de que la tupla contenga los atributos que posee. Por su parte, H es la hipótesis de que la tupla pertenece a una determinada clase y $P(H)$ su probabilidad. Esta es conocida como probabilidad "a priori". De la misma manera, $P(H|X)$ es la probabilidad de que la hipótesis H sea cierta bajo la evidencia X. A esta se la llama probabilidad "a posteriori" con H condicionada por X y es el valor que se quiere determinar en una tarea de clasificación. Finalmente, $P(X|H)$ indica la probabilidad de que la tupla tenga unos atributos determinados dado que se satisface la hipótesis.

Por su parte, la fórmula de *Naive* Bayes es similar:

$$P(C_i \mid X) = P(X \mid C_i)P(C_i) \tag{2.2}$$

Aquí el término $P(X)$ es descartado ya que se asume constante para todas las clases. La hipótesis H es representada como C_i que es un valor de la tupla $C = (C_1, C_2, ..., C_m)$, donde m es el número de clases. La presunción "ingenua" es aplicada para el cálculo del término $P(X \mid C_i)$ gracias a lo cual se puede definir de la siguiente manera:

$$P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i) = P(x_1 \mid C_i) \times P(x_2 \mid C_i) \times ... \times P(x_n \mid C_i) \tag{2.3}$$

Finalmente, el modelo seleccionará la clase que maximice el valor de probabilidad.

Como se ha dicho anteriormente, la simplicidad, velocidad computacional y su competitividad en métricas de exactitud hacen de *Naive* Bayes un algoritmo destacado en el campo de aprendizaje automático [19] y ha sido aplicado para problemas diversos, tales como el de hallar errores en programas de computación [1], predecir enfermedades del corazón [4] o detectar ataques en una red de computadoras [11].

2.3.2. Árboles de Decisión

Los árboles de decisión son un modelo de clasificación que se destaca por ser de fácil interpretación e intuitivo para el ser humano. De hecho, se puede generar una representación gráfica del árbol generado para asistir a la comprensión del del modelo y de cómo se comporta durante una predicción. En cuanto a su estructura, un árbol de decisión contiene nodos, cada uno representando un atributo de la colección. Estos nodos se conectan

con otros nodos a partir de enlaces o "ramas" que representan un valor o un rango de valores de ese atributo. Los nodos de menor jerarquía son llamados "hojas" y contienen la clase de la predicción, y el nodo de mayor jerarquía es llamado "raíz". Al momento de predecir una instancia nueva, la clasificación se realiza de la siguiente manera: se toma la instancia nueva, la cual no tiene una etiqueta asociada, y los valores de sus atributos son comparados contra los del árbol, luego se traza un camino desde el nodo raíz hasta la hoja que contiene una clase y dicha clase es la predicción resultante.

Los árboles de decisión se generan a partir de un algoritmo de inducción. Existen varios de estos algoritmos pero todos son variantes que han sido diseñadas bajo un misma principio: construir el árbol de una manera "voraz"[1], comenzando desde el nodo raíz (conocido como enfoque *top-down*) y eligiendo en cada paso el atributo más informativo o que maximice alguna medida de ganancia de información.

Algunos de estos algoritmos son:

ID3 Son las siglas de *Iterative Dichotomiser 3* y fue desarrollado en 1986 por Ross Quinlan. Consiste en crear un árbol de múltiples vías, buscando para cada nodo el atributo categórico que lance la mayor ganancia de información para las clases categóricas. Los árboles crecen en un tamaño máximo y luego se realiza el paso de poda para mejorar el poder de generalización del modelo sobre datos desconocidos.

C4.5 Es la evolución del algoritmo ID3. La principal mejora con respecto a su predecesor es que elimina la restricción de que los atributos deban ser categóricos. Esto lo consigue particionando el valor continuo en rangos o en un conjunto de intervalos discretos. C4.5 convierte el árbol entrenado en conjuntos de reglas de decisión.

CART Son las siglas de Classification and Regression Tree y es un algoritmo muy similar al C4.5 pero que soporta clases numéricas, lo cual permite resolver problemas de regresión.

Una tarea fundamental en la generación de un árbol es definir un criterio de división para seleccionar el mejor atributo en cada paso. Existen diversas técnicas para abordarla, una de ellas es la de "Ganancia de Información", usada por el algoritmo ID3. La Ganancia de información busca seleccionar el atributo que posee mayor variabilidad o representatividad de los datos y se sustenta en el cálculo de la entropía o medida de desorden. La idea de fondo es hallar el atributo que reduzca la entropía esperada. La entropía en el conjunto de datos D se calcula de la siguiente manera:

$$Entropia(D) = -\sum_{i=1}^{m} p_i \log_2(p_i) \tag{2.4}$$

Aquí p_i corresponde a la probabilidad de que una tupla de D corresponda a la clase C_i.

Luego, la ganancia de información es:

$$Ganancia(A) = Entropia(D) - \sum_{j=1}^{v} \frac{\|D_j\|}{\|D\|} \times Entropia(D_j) \tag{2.5}$$

[1] Se le llama voraz o *greedy* a un algoritmo que busca hallar la opción óptima en cada paso y, de esta manera, alcanzar la solución general óptima para resolver un problema. Esto lo diferencia de algoritmos como los de *backtracking*, los cuales exploran distintas posibilidades y pueden volver al inicio en búsqueda de una mejor solución

Aquí el atributo A divide al conjunto de datos en v particiones, siendo v los valores posibles que toma A. D_j es el subconjunto de los datos cuyas tuplas poseen el valor v del atributo A, siendo $\|D_j\|$ su cardinalidad o número de instancias del subconjunto. Al dividir este término por la cardinalidad del conjunto de datos, se obtiene un valor que representa el peso de la partición y es aplicado sobre la entropía esperada.

Una vez obtenidos los valores de ganancia para cada atributo, se selecciona aquel que maximiza la ganancia y este será el criterio de separación en el nodo.

El algoritmo C4.5 introdujo una mejora en esta técnica llamada "Razón de Ganancia". La misma busca disminuir uno de los efectos adversos que provoca la técnica de ganancia de información, esta es, que tiende a favorecer a atributos con un mayor número de valores posibles. La razón de ganancia, en primer lugar, reemplaza la fórmula $Entropia(D)$ por la siguiente:

$$EntropiaRG_A(D) = -\sum_{j=1}^{v} \frac{\|D_j\|}{\|D\|} \times \log_2(\frac{\|D_j\|}{\|D\|}) \tag{2.6}$$

A su vez, el nuevo cálculo de la ganancia se formula así:

$$RazonGanancia(A) = \frac{Ganancia(A)}{EntropiaRG_A(D)} \tag{2.7}$$

Finalmente, el atributo de mayor razón de ganancia es seleccionado.

Aplicaciones de árboles de decisión en la literatura

Figura de un árbol

Subsección para sgd?, svm?, perceptrones?

2.3.3. Ensambles

Los ensambles son un conjunto de clasificadores que, al ser combinados, pueden realizar mejores predicciones que cualquiera de ellos individualmente. En pocas palabras, el enfoque de ensambles consiste en generar k clasificadores, de un mismo tipo o no, y entrenarlos con subconjuntos de la colección de entrenamiento original. Dada una tupla nueva, cada clasificador devuelve su propia predicción, llamada "voto" y luego el ensamble devuelve la predicción final basada en esos votos.

La aplicación de ensambles en problemas de clasificación nace de la imposibilidad de generar un único modelo capaz de generalizar lo suficiente como para lograr un rendimiento perfecto. Ante la presencia de datos ruidosos, atípicos o erróneos los clasificadores pueden tender a clasificar mejor para un subconjunto de datos y no tan bien para otros. Este escenario es aprovechado por el enfoque de ensambles ya que su éxito tiene correlación con la existencia de diversidad en la clasificación, esto es, que haya variabilidad entre los subconjuntos de datos, modelos o hiper-parámetros, entre otros factores. A mayor esta diversidad, los errores particulares de un ensamble se aíslan y serán filtrados por la clasificación final. Como resultado se espera una disminución del error total de la clasificación así como también una mayor exactitud en la predicción, comparando contra los clasificadores base. Por otro lado, un enfoque de ensambles abre la posibilidad de distribuir y/o paralelizar el cómputo de la predicción, pudiendo así mejorar los tiempos de ejecución durante el entrenamiento.

Existen distintos tipos de ensamble, de acuerdo a su construcción y arquitectura. A continuación se describen 3 de ellos: los ensambles de tipo *bagging*, los de tipo *boosting* y los de tipo *stacked*.

Bagging Esta es una de las primeras técnicas de ensambles conocidas y fue introducida por Breiman[3]. La misma se desarrolla de la siguiente manera: dado un conjunto de entrenamiento D con n tuplas, *bagging* genera un número m de nuevos conjuntos de datos de entrenamiento, cada uno con n tuplas. Para esto se toman tuplas del conjunto original de manera aleatoria y con reemplazo, es decir que puede haber tuplas repetidas y otras que no están incluidas en el nuevo conjunto. Luego a partir de cada conjunto nuevo, se entrena un clasificador M_i. Cada clasificador puede ser del mismos tipo ya que la diversidad está dada por los datos. En la etapa de clasificación, cada modelo M_i genera una predicción que cuenta como un voto. El ensamble cuenta los votos y elige la clase con mayor cantidad de votos, siendo esta la decisión final del ensamble.

Boosting En la técnica de *boosting* se asigna un peso a cada tupla de entrenamiento y se generan un conjunto de clasificadores, uno luego del siguiente. A diferencia del método de *bagging*, *boosting* trabaja siempre sobre el mismo conjunto de datos y la variabilidad está dada por los pesos que son asignados. El proceso es el siguiente: para el primer modelo de clasificación, M_i, los pesos son inicializados en un mismo valor para todas las tuplas. Una vez que se entrena este modelo, los pesos son actualizados de tal manera que el siguiente clasificador $M_i + 1$ trate de manera particular a las tuplas mal clasificadas por Mi, de tal manera de llegar a una clasificación correcta. El clasificador final combina los votos de cada clasificador individual , donde el peso del voto de cada clasificador es una función de su exactitud.

Stacking Stacking es una técnica desarrollada por Wolpert[20] y consiste en entrenar un nuevo clasificador de acuerdo a las predicciones realizadas por otros modelos, tomando la salida de estos modelos como entrada, de tal manera de lograr hallar una combinación que produzca una mejor predicción. Este tipo de ensambles puede ser visto como un conjunto de capas. La primera capa consta de un ensamble de clasificadores que aprenden a partir de los datos de entrenamiento. Esta capa no necesariamente usa clasificadores del mismo tipo, mismos hiper-parámetros o particiones de la colección iguales, quedando estos detalles a cargo de quien diseña esta capa. La siguiente capa es el clasificador individual, o meta-clasificador, que se alimenta de las salidas de los clasificadores de la capa inferior y realiza el aprendizaje a partir de las clases producidas por estas salidas y las clases reales.

Una de las tareas a tener en cuenta durante el entrenamiento de un ensamble es la de combinar las salidas de cada modelo en una salida final. La estrategia más común y simple es la de mayoría de voto, aplicada por los métodos de *bagging*, pero existen múltiples y no necesariamente un ensamble de tipo *bagging* debe aplicar esta estrategia. Por ejemplo, algunos clasificadores pueden decidir producir una salida sólo en el caso de que más de la mitad de ellos coincidan, o incluso ser más restrictivos y obligar a que la coincidencia sea total. El enfoque de *boosting* por su parte, pondera al voto de acuerdo a los pesos que calcula, dando predominio a determinadas instancias. También se suele dar un mayor peso a determinados clasificadores por sobre otros. Este tipo de métodos se los denomina "mayoría de voto ponderada" y pueden llevar a un rendimiento superior.

Aplicaciones de ensambles en la literatura

Figura de ensamble

2.4. Clasificación Multi-etiquetas

2.5. Clasificación de Flujos Continuos de Datos

MLL Clasificación multi-etiquetas. 2

BIBLIOGRAFíA

[1] Ömer Faruk Arar y Kürşat Ayan. «A feature dependent Naive Bayes approach and its application to the software defect prediction problem». En: *Applied Soft Computing* 59 (1 de oct. de 2017), págs. 197-209. ISSN: 1568-4946. DOI: 10.1016/j.asoc.2017.05.043. URL: http://www.sciencedirect.com/science/article/pii/S1568494617303083 (visitado 11-01-2021).

[2] Albert Bifet y Gianmarco De Francisci Morales. «Big Data Stream Learning with SAMOA». En: *2014 IEEE International Conference on Data Mining Workshop*. 2014.

[3] Leo Breiman. «Bagging predictors». En: *Machine Learning* 24.2 (1 de ago. de 1996), págs. 123-140. ISSN: 1573-0565. DOI: 10.1007/BF00058655. URL: https://doi.org/10.1007/BF00058655 (visitado 23-01-2021).

[4] Uma Dulhare. «Prediction system for heart disease using Naive Bayes and particle swarm optimization». En: *Biomedical Research* 29 (1 de ene. de 2018). DOI: 10.4066/biomedicalresearch.29-18-620.

[5] Usama M Fayyad, Gregory Piatetsky-Shapiro y Padhraic Smyth. *Advances in Knowledge Discovery and Data Mining*. Ed. por Fayyad, Usama M. and Piatetsky-Shapiro, Gregory and Smyth, Padhraic and Uthurusamy, Ramasamy. Section: From data mining to knowledge discovery: an overview. Menlo Park, CA, USA: American Association for Artificial Intelligence, 1996. 1–34.

[6] João Gama. *Knowledge Discovery from Data Streams*. 2010.

[7] J Gantz y D Reinsel. «Extracting value from chaos». En: *IDC IView* (2011), págs. 1-12.

[8] Eva Gibaja y Sebastian Ventura. «A Tutorial on Multi-Label Learning». En: *ACM Computing Surveys* 47 (2015).

[9] Trevor Hastie, Robert Tibshirani y Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition*. 2.ª ed. Springer Series in Statistics. New York: Springer-Verlag, 2009. ISBN: 978-0-387-84857-0. DOI: 10.1007/978-0-387-84858-7. URL: https://www.springer.com/gp/book/9780387848570 (visitado 12-01-2021).

[10] Geoff Hulten, Laurie Spencer y Pedro Domingos. «Mining Time-changing Data Streams». En: *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '01. San Francisco, California: ACM, 2001, págs. 97-106.

[11] Harsha K. Kalutarage y col. «Detecting stealthy attacks: Efficient monitoring of suspicious activities on computer networks». En: *Computers & Electrical Engineering* 47 (1 de oct. de 2015), págs. 327-344. ISSN: 0045-7906. DOI: 10.1016/j.compeleceng.2015.07.007. URL: http://www.sciencedirect.com/science/article/pii/S0045790615002384 (visitado 11-01-2021).

[12] V Mayer-Schonberger y K Cukier. *Big Data: A Revolution that Will Transform how We Live, Work, and Think*. An Eamon Dolan book. Houghton Mifflin Harcourt, 2013.

[13] Robi Polikar. «Polikar, R.: Ensemble based systems in decision making. IEEE Circuit Syst. Mag. 6, 21-45». En: *Circuits and Systems Magazine, IEEE* 6 (6 de oct. de 2006), págs. 21-45. DOI: 10.1109/MCAS.2006.1688199.

[14] Jesse Read, Bernhard Pfahringer y Geo Holmes. «Generating Synthetic Multi-label Data Streams». En: (), pág. 16.

[15] Jesse Read y col. «Classifier chains for multi-label classification». En: *Mach. Learn.* 85.3 (2011), págs. 333-359.

[16] Jesse Read y col. «Scalable and efficient multi-label classification for evolving data streams». En: *Machine Learning* 88.1 (1 de jul. de 2012), págs. 243-272. ISSN: 1573-0565. DOI: 10.1007/s10994-012-5279-6. URL: https://doi.org/10.1007/s10994-012-5279-6 (visitado 17-06-2020).

[17] Ricardo Sousa y João Gama. «Multi-label classification from high-speed data streams with adaptive model rules and random rules». En: *Progress in Artificial Intelligence* (2018).

[18] Grigorios Tsoumakas y Ioannis Katakis. «Multi-Label Classification». En: *Int. J. Data Warehouse. Min.* 3.3 (2007), págs. 1-13.

[19] Indika Wickramasinghe y Harsha Kalutarage. «Naive Bayes: applications, variations and vulnerabilities: a review of literature with code snippets for implementation». En: *Soft Computing* (9 de sep. de 2020). ISSN: 1432-7643, 1433-7479. DOI: 10.1007/s00500-020-05297-6. URL: http://link.springer.com/10.1007/s00500-020-05297-6 (visitado 11-01-2021).

[20] David H. Wolpert. «Stacked generalization». En: *Neural Networks* 5.2 (1 de ene. de 1992), págs. 241-259. ISSN: 0893-6080. DOI: 10.1016/S0893-6080(05)80023-1. URL: http://www.sciencedirect.com/science/article/pii/S0893608005800231 (visitado 23-01-2021).

[21] Min-Ling Zhang y Kun Zhang. «Multi-label learning by exploiting label dependency». En: *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '10*. 2010.

[22] Min-Ling Zhang y Zhi-Hua Zhou. «A Review On Multi-Label Learning Algorithms». En: *IEEE Trans. Knowl. Data Eng.* 26 (2014), págs. 1819-1837.