



Universidad Central de Venezuela
Facultad de Ciencias
Escuela de Computación
Centro de Computación Distribuida y Paralela

**Herramienta para el análisis de la interacción,
identificación de patrones y clasificación de
usuarios en humanos, ciborgs y bots de la red de
microblogging Twitter**

Trabajo Especial de Grado presentado ante la ilustre

Universidad Central de Venezuela por

Br. Josemy Duarte

Br. Gabriel Rodríguez

Tutores:

Prof. Jesús Lares

Prof. José R.Sosa

Caracas, Octubre del 2016

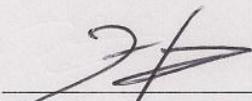
UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE COMPUTACION

ACTA DEL VEREDICTO

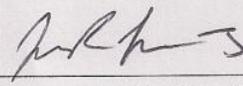
Quienes suscriben, Miembros del Jurado designado por el Consejo de la Escuela de Computación para examinar el Trabajo Especial de Grado, presentado por los bachilleres Josemy Inoel Duarte Fernández C.I.: 21.410.609 y Gabriel Eduardo Rodríguez González C.I.: 22.523.150, con el título "Herramienta Para El Análisis De La Interacción, Identificación De Patrones Y Clasificación De Usuarios En Humanos, Ciborgs Y Bots De La Red De Microblogging Twitter", con el fin de cumplir con el requisito legal para optar al título de Licenciado en Computación, dejan constancia de lo siguiente:

Leído el trabajo por cada uno de los Miembros del Jurado, se fijó el día 17 de Octubre de 2016, a las 10:30 am, para que sus autores lo defendieran en forma pública, en el aula PAIII, lo cual estos realizaron mediante una exposición oral de su contenido, y luego respondieron satisfactoriamente a las preguntas que les fueron formuladas por el Jurado, todo ello conforme a lo dispuesto en la Ley de Universidades y demás normativas vigentes de la Universidad Central de Venezuela. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobarlo con una nota de 20 puntos.

En fe de lo cual se levanta la presente acta, en Caracas el 17 de Octubre de 2016, dejándose también constancia de que actuó como Coordinador del Jurado el Profesor Tutor Jesús Lares.



Prof. Jesús Lares
Tutor



Prof. José R. Sosa
Co-tutor



Prof. Héctor Navarro
Jurado



Prof. Haydymar Nuñez
Jurado

Resumen

Twitter es una red social de microblogging que experimentó un aumento descomunal de popularidad en el año 2009, convirtiéndola en una de las plataformas sociales más influyentes en la actualidad. Este aumento de importancia ocasionó que surgieran distintos tipos de cuentas que perjudican la interacción entre los usuarios esparciendo contenido spam, influyendo opiniones y realizando publicaciones con fines meramente publicitarios dentro de la plataforma.

Este tipo de usuarios son conocidos como usuarios bots, los cuales se caracterizan por tener un comportamiento automático y programado para cumplir sus funciones. Sin embargo, los bots no son el único tipo de usuarios que pueden tener un comportamiento automatizado; existen usuarios humanos que pueden asistir cuentas bot o utilizar herramientas para programar parte de su comportamiento. Este tipo de usuarios es denominado por distintos estudios como usuarios ciborgs.

Durante la siguiente investigación se estudiaron las características inherentes al contenido y el comportamiento de los usuarios venezolanos de la plataforma de Twitter con el fin de detectar patrones que permitan clasificar a los usuarios en tres categorías: humanos, ciborgs y bots.

Además, se desarrolló una herramienta que permite realizar el proceso de clasificación de forma automática basándose en un modelo entrenado a partir de conjuntos de usuarios humanos, ciborgs y bots categorizados de forma manual por un componente humano, permitiendo comparar los resultados de las características evaluadas de los usuarios clasificados en contraposición con los usuarios utilizados para el entrenamiento del modelo.

Palabras clave: Venezuela; Twitter; humanos; ciborgs; bots; modelo; clasificación; comportamiento; patrones;

Agradecimientos

A nuestras madres (Mireya y Mireya) y padres (José y Waldir) gracias por su apoyo incondicional, por la paciencia, por el inmenso esfuerzo que hicieron para darnos la oportunidad de formarnos en nuestra grandiosa Universidad Central de Venezuela y por nunca dudar de nosotros.

A nuestros tutores Jesús Lares y José R. Sosa, gracias por su apoyo y paciencia durante todo el proyecto.

A la profesora Izaskun Petralanda, por transmitirnos su sentido de amor por la ciencia y la ética profesional.

A la Universidad Central de Venezuela, a la Escuela de Computación y a todos y cada uno de los profesores que contribuyeron en nuestra formación académica.

A la Fuerza Original, que permitió la culminación de esta fase de nuestras vidas.

Por parte de Josemy Duarte:

A Gabriel R. por su esfuerzo y dedicación no solo en este proyecto, sino en todos los que enfrentamos durante la carrera.

A mi hermano Krsna, por su ayuda y apoyo en los momentos de necesidad.

A mi pareja y compañera Estefanie G., por su comprensión, motivación y contagiosa alegría.

Por parte de Gabriel Rodríguez:

A Josemy D. por su apoyo, amistad, esfuerzo y visión innovadora en cada una de los retos que emprendimos en el transcurso de la carrera a pesar de las dificultades.

A mi compañera y pareja Nakary M. por su motivación, apoyo moral y asistencia para la representación visual estilizada del contenido.

Tabla de contenidos

| | |
|---|------|
| Resumen | VIII |
| Agradecimientos | IX |
| Introducción | IX |
| Capítulo 1 | 1 |
| El Problema | 1 |
| 1.1. Planteamiento del Problema..... | 1 |
| 1.2. Justificación..... | 2 |
| 1.3. Objetivos de la investigación | 3 |
| 1.3.1. Objetivo General..... | 3 |
| 1.3.2. Objetivos Específicos..... | 3 |
| 1.4. Antecedentes | 3 |
| 1.5. Alcance | 4 |
| Capítulo 2 | 5 |
| Marco Conceptual..... | 5 |
| 2.1. Ciencias de datos..... | 5 |
| 2.1.1. Áreas de aplicación..... | 5 |
| 2.1.2. Grandes Volúmenes de Datos (Big Data) | 5 |
| 2.1.3. Bases de datos | 7 |
| 2.1.4. Organización y estructura de los datos | 10 |
| 2.2. Lenguajes de programación | 12 |
| 2.2.1. Python | 13 |
| 2.2.2. Scala..... | 13 |
| 2.2.3. JavaScript..... | 14 |
| 2.3. Herramientas para la Ciencia de Datos | 15 |
| 2.3.1. Acceso y procesamiento de datos | 15 |
| 2.3.2. Virtualización | 19 |
| 2.3.3. Visualización e interfaz de usuario..... | 22 |
| 2.4. Minería de datos..... | 24 |
| 2.4.1. Proceso de la minería de datos..... | 25 |

| | |
|--|----|
| 2.4.2. Métodos, técnicas y algoritmos de la minería de datos | 28 |
| 2.5. Redes sociales | 33 |
| 2.5.1. Técnicas para el análisis de redes sociales | 34 |
| 2.5.3. Usuarios | 36 |
| Capítulo 3 | 38 |
| Método de Desarrollo..... | 38 |
| 3.1 Metodología fundacional para la ciencia de datos | 38 |
| Capítulo 4 | 41 |
| Desarrollo de la Solución | 41 |
| 4.1 Entendimiento del negocio | 41 |
| 4.2 Enfoque analítico..... | 41 |
| 4.2.1 Componente basado en el perfil de usuario | 41 |
| 4.2.2 Componente basado en el contenido publicado..... | 42 |
| 4.3 Requerimientos de los datos | 43 |
| 4.4 Recolección de datos | 43 |
| 4.4.1 Construcción del árbol de cuentas | 43 |
| 4.4.2 Obtención de timelines de usuarios | 44 |
| 4.5. Entendimiento de los datos | 44 |
| 4.5.1. Volumen de tweets | 44 |
| 4.5.2. Longitud de Tweets..... | 45 |
| 4.5.3. Uso de enlaces | 46 |
| 4.5.4. Contenido spam..... | 46 |
| 4.5.5. Dispositivos de publicación | 47 |
| 4.5.6. Menciones a otros usuarios | 48 |
| 4.5.7. Respuestas a publicaciones | 49 |
| 4.5.8. Relación entre seguidores y amigos | 49 |
| 4.5.9. Entropía de los grupos | 50 |
| 4.5.10. Horarios de alto tráfico | 51 |
| 4.6. Preparación de los datos..... | 52 |
| 4.6.1. Calculo de la muestra de datos..... | 52 |
| 4.6.2. Clasificación manual de usuarios..... | 54 |

| | |
|--|----|
| 4.6.3. Creación de conjuntos de Tweets spam y no spam | 54 |
| 4.6.4. Consideraciones | 55 |
| 4.7. Modelado | 55 |
| 4.8. Evaluación..... | 57 |
| 4.9. Despliegue | 58 |
| 4.9.1. Configuración de Credenciales | 59 |
| 4.9.2. Inicio de sesión por medio de Twitter..... | 59 |
| 4.9.3. Entrenamiento del modelo | 59 |
| 4.9.4. Clasificar usuarios..... | 60 |
| 4.9.5. Visualizar resultados | 61 |
| 4.10. Retroalimentación | 63 |
| Capítulo 5 | 64 |
| Conclusiones | 64 |
| 5.1. Contribución | 65 |
| 5.2. Recomendaciones..... | 65 |
| 5.3. Trabajos futuros | 66 |
| Anexos..... | 67 |
| 1. Instrucciones para la creación del contenedor..... | 68 |
| Instrucciones del Dockerfile | 69 |
| 2. Instrucciones de despliegue del API de clasificación y la aplicación web 71 | |
| Despliegue de contenedor con Mongo..... | 72 |
| Despliegue de contenedor con API..... | 72 |
| Despliegue de la aplicación web | 72 |
| 3. Peticiones al API de clasificación | 73 |
| Archivo de configuración del API de clasificación | 74 |
| Entrenar clasificador spam | 75 |
| Entrenar juez | 75 |
| Evaluar directorio de usuarios | 76 |
| Evaluar usuario..... | 76 |
| Referencias..... | 77 |

Índice de figuras

| | |
|--|----|
| Figura 1. Ejemplo de tabla | 8 |
| Figura 2. Arquitectura Apache Hadoop. | 16 |
| Figura 3. MapReduce..... | 17 |
| Figura 4. Arquitectura de Spark..... | 18 |
| Figura 5. Etapas de un proceso de minería de datos | 27 |
| Figura 6. Ejemplo de grupos en conjuntos de datos analizados..... | 30 |
| Figura 7. Etapas de la Foundational Methodology for Data Science | 38 |
| Figura 8. FDA del número de tweets..... | 45 |
| Figura 9. FDA del promedio de longitud de tweets..... | 45 |
| Figura 10. FDA de la proporción de enlaces a fuentes externas | 46 |
| Figura 11. FDA de la proporción de contenido spam..... | 47 |
| Figura 12. Proporción de uso de medios de publicación | 48 |
| Figura 13. FDA de la proporción de menciones | 48 |
| Figura 14. FDA de la proporción de respuestas | 49 |
| Figura 15. FDA de la reputación de las cuentas..... | 50 |
| Figura 16. FDA de la entropía relativa..... | 50 |
| Figura 17. Publicaciones de humanos por hora en la semana (GMT-0)..... | 51 |
| Figura 18. Publicaciones de ciborgs por hora en la semana (GMT-0)..... | 51 |
| Figura 19. Publicaciones de bots por hora en la semana (GMT-0)..... | 52 |
| Figura 20. Número de usuarios activos de Twitter por Statista..... | 53 |
| Figura 21. API de Clasificación desplegada | 58 |
| Figura 22. Pantalla de configuración y entrenamiento..... | 60 |
| Figura 23. Pantalla de clasificación individual de usuario..... | 61 |
| Figura 24. Pantalla de resultados de clasificación | 62 |
| Figura 25. Pantalla con matriz de confusión del juez entrenado..... | 62 |

Índice de tablas

| | |
|---|----|
| Tabla 1. Hadoop vs Spark..... | 19 |
| Tabla 2. Máquinas Virtuales vs Contenedores | 22 |
| Tabla 3. Meteor y Mean | 24 |
| Tabla 4. Comparación entre la exactitud del algoritmo Bayesiano y el Random Forest..... | 55 |
| Tabla 5. Tabla de pesos de las características de los usuarios | 56 |
| Tabla 6. Matriz de confusión sobre los resultados del Juez | 57 |

Introducción

Las redes sociales en línea han logrado adquirir gran importancia en cuanto a la interacción social refiere. Son medios de comunicación e intercambio de información. La informalidad de algunas redes sociales alienta a los usuarios a expresar sus opiniones o detalles personales de sus vidas. Dentro de la gran cantidad de redes sociales disponibles en línea, Twitter se posiciona como una de las principales, permitiendo a sus usuarios realizar publicaciones de una longitud máxima de 140 caracteres. Además de permitir obtener dichas publicaciones directamente de la fuente, debido a su sistema de seguidores y amigos.

Twitter se presenta como una red de interacciones públicas que contienen información sobre millones de usuarios. El uso que la mayoría de los usuarios le dan a esta red incluye desde simples conversaciones, publicaciones sobre sus actividades diarias o pensamientos espontáneos, hasta la obtención de noticias o la publicación de fotos, videos o links a otras páginas... En general, esta variedad de usos presentan a Twitter como un buen candidato para el estudio del comportamiento humano.

Entre las diferentes ventajas que pueden encontrarse respecto al uso de Twitter para investigaciones científicas, se puede destacar la sencillez con la que se logran obtener los datos, además de toda la metadata disponible. Esta característica permite relacionar una simple publicación, con su autor, el momento en la que fue generada, el dispositivo desde el que fue publicado y hasta el lugar en el que se encontraba el usuario. Ofreciendo una variedad de datos con gran cantidad de información para extraer y analizar.

Distintos estudios han realizado análisis a partir de esta información, permitiéndoles clasificar a los usuarios según su comportamiento y el contenido que generan. Una clasificación generalmente aceptada es la división de usuarios entre los humanos, cuentas regulares con comportamiento común y generación de contenido original; y los bots, cuentas con comportamiento automatizado que usualmente tienen el propósito de esparcir contenido spam por la red.

Esta investigación se dividió en cinco grandes partes. En la primera parte se plantea el problema que se busca solucionar, su justificación, sus objetivos y el alcance de la misma. En la segunda parte se describen todos los fundamentos teóricos en los cuales se basó la investigación para llegar a la solución desarrollada. En la tercera parte se explica la metodología de trabajo usada, la cual fue propuesta por IBM para proyectos de ciencia de datos. En la cuarta parte se muestra detalladamente los procesos y actividades desarrolladas en cada paso de la metodología. Y finalmente en la quinta parte, se concluirá la investigación y

además se harán recomendaciones y especificaciones para trabajos futuros sobre la herramienta desarrollada y la investigación realizada.

Capítulo 1

El Problema

1.1. Planteamiento del Problema

En la actualidad, los medios y tecnologías de comunicación que permiten la integración y comunicación inmediata entre individuos a lo largo y ancho del globo alcanzaron un nivel de influencia y penetración social como pocos fenómenos lo han hecho en los últimos 20 años. Uno de los mayores exponentes de estos “nuevos” medios de comunicación es la red de microblogging Twitter, una de las tecnologías de información con mayor crecimiento desde su creación en el 2006, pasando de 475.000 usuarios en febrero de 2008 a más de 7.038.000 usuarios para febrero de 2009, experimentando un crecimiento anual de 1372%. [1]

Este aumento descomunal de popularidad posicionó a Twitter dentro del espectro mundial de comunicación y hasta la fecha contabiliza alrededor de 320 millones de usuarios activos por mes. [2]

Sin embargo, a medida que este tipo de redes/medios sociales crecen se desarrollan ciertos elementos o fenómenos que suelen considerarse típicos o comunes, tales como el contenido spam, las publicaciones o entradas con fines meramente publicitarios y, en especial, los usuarios bots.

La situación expuesta anteriormente conlleva a plantearse interrogantes vitales para la gran mayoría de estudios, análisis y proyectos basados en esta plataforma: ¿Cuántos de los usuarios activos dentro de Twitter son realmente usuarios humanos? ¿Es posible detectar de forma automática si un usuario es bot?

El proyecto a desarrollar pretende presentar una herramienta que permita ayudar a la resolución de las interrogantes previamente planteadas. El estudio se realizará a partir de una muestra de datos extraídos de la red en cuestión con la intención de crear un modelo que, en base a la exploración del contenido generado por los usuarios, su comportamiento y relaciones con el resto de individuos en el medio, permita clasificar de forma automática si un usuario es humano, bot o ciborg.

1.2. Justificación

Los bots son uno de los elementos más perjudiciales dentro de las tecnologías de información, capaces de generar buena parte del tráfico de datos e información dentro de las mismas. Aunado a esto, el uso de bots está altamente relacionado con problemáticas que van desde la generación masiva de contenido spam hasta la suplantación a gran escala de identidades.

El impacto de los bots varía dependiendo de la plataforma en la que se encuentre interactuando. En el caso de Twitter, uno de los escenarios más delicados relacionado a los bots es la generación de falsas tendencias que, con la fuerza y cantidad de contenido suficiente, son capaces de influir considerablemente en la opinión pública sobre un individuo, ente o situación particular. Otro escenario bastante común es el uso de bots para la creación de campañas comerciales de promoción o desacreditación de compañías, marcas, o productos de forma maliciosa.

En Venezuela son pocas las investigaciones que se concentran en evaluar la magnitud de las cuentas bots dentro de las comunidades de Twitter y, consecuentemente, su posible influencia sobre la opinión pública sobre temas o tendencias importantes. El presente trabajo de investigación pretende marcar un punto y aparte sobre esta situación, proporcionando una herramienta que permita a los individuos e instituciones interesados en la materia sobrepasar el obstáculo que los bots representan sobre el desarrollo de sus estudios e investigaciones.

1.3. Objetivos de la investigación

1.3.1. Objetivo General

Estudiar la interacción de los usuarios de la red de microblogging Twitter para la identificación de patrones y clasificación de los usuarios en humanos, bots y ciborgs.

1.3.2. Objetivos Específicos

- Desarrollar un proceso para recolectar datos de la plataforma de Twitter a través de su API.
- Examinar el contenido generado por los usuarios de Twitter para reconocer las diferencias entre los usuarios humanos, ciborgs y bots.
- Analizar el comportamiento de los usuarios dentro de Twitter para descubrir patrones de conducta.
- Definir los indicadores que permitan clasificar a los usuarios de Twitter en humanos, bots o ciborgs.
- Desarrollar una aplicación para la ejecución de pruebas sobre el modelo y visualización de resultados.
- Verificar la confianza de los resultados obtenidos por la aplicación.

1.4. Antecedentes

Desde su creación en el 2006 y gracias a su impacto social, Twitter ha sido objeto de estudio en múltiples áreas. La evolución de esta y otras plataforma de social media fue estudiada de forma reciente por Ferrara et col. [3]. Su trabajo también abordó el efecto que tienen los bots sobre estas plataformas, planteando un conjunto de consecuencias negativas de gran relevancia y la composición de distintos sistemas de detección de usuarios bots.

El efecto de los bots sobre las redes sociales es también estudiado por Boshmaf et col. [4], cuyo trabajo se concentró en evaluar y determinar cuan vulnerables son las redes sociales con respecto a la infiltración a gran escala por parte de los “socialbots”: programas de computadora que controlan cuentas de redes sociales e imitan usuarios reales.

Por otra parte, en cuanto a la interacción entre los usuarios humanos y usuarios bots, Murgia et col. [5] realiza una serie de experimentos preliminares, tomando como caso de uso al sitio web Stack Overflow, con el fin de determinar en qué medida un bot puede simular el comportamiento de un humano y cuál es la retroalimentación que recibe.

El propósito y la influencia de los usuarios bot no necesariamente deben ser siempre considerados como un elemento perjudicial para una plataforma o red social, por lo tanto, la capacidad de discernir cuales cuentas automatizadas son benignas y cuales son malignas es fundamentalmente importante para modelos de clasificación de usuarios y sistemas de detección de bots. Dentro de esta perspectiva, Penna [6] realizó un estudio sobre el comportamiento en línea de tres tipos de cuentas de usuarios en Twitter: 1) personales, que corresponden a usuarios humanos; 2) asistidas, referente a las cuentas de corporaciones, y 3) bots. Aparte, realizaron un análisis estadístico sobre los perfiles de usuarios y crearon dos algoritmos de Machine Learning basados en el comportamiento de las publicaciones: 1) un clasificador Bayesiano ingenuo y 2) un modelo de predicción probabilístico.

En adición a estudios concentrados en la categorización de usuarios en Twitter, Chu et col. [7] determinaron las principales diferencias entre usuarios humanos, bots y ciborgs con respecto a los patrones de publicación, el contenido de los tweets y las características asociadas a los perfiles. Fundamentándose en los resultados obtenidos, propusieron un sistema de clasificación compuesto por 4 componentes: 1) un componente basado en la entropía, 2) un componente de detección de spam, 3) un componente de propiedades de las cuentas, y 4) un componente para la toma de decisiones.

1.5. Alcance

El presente estudio se realizará sobre una muestra de datos estática e invariable, que serán obtenidas a través de la Interfaz de Programación de Aplicaciones de Twitter (API), de la cual se asumirán como venezolanos o con una estrecha relación con Venezuela a la gran mayoría de usuarios evaluados.

Se establecerá un conjunto finito de indicadores clave para la clasificación de los usuarios en humanos, bots o ciborgs; a su vez se definirán los procesos, actividades y algoritmos que permitan extraer la información necesaria de la muestra de datos para evaluar dichos indicadores.

Capítulo 2

Marco Conceptual

2.1. Ciencias de datos

La ciencia de datos es un campo interdisciplinario que busca la extracción de conocimiento a partir de datos en sus distintas formas (estructurados, no estructurados o semi-estructurados), aplicando métodos y técnicas provenientes de campos como la estadística, la minería de datos, el análisis predictivo, el procesamiento distribuido y paralelo, entre otros. [8][9]

A partir de los resultados obtenidos del proceso de extracción de conocimiento se pueden construir herramientas que permitan analizar los resultados y realizar procesos de toma de decisiones.

2.1.1. Áreas de aplicación

La ciencia de datos incorpora metodologías, técnicas y herramientas de diversos campos (Matemáticas, Computación, Estadística, Ingeniería de datos, Modelos probabilísticos, entre otros) para lograr cumplir sus objetivos, analizar grandes conjuntos de datos, hacer distintos estudios, descubrir patrones, todo orientado a obtener conocimiento. Por lo tanto, la ciencia de datos puede ser aplicada en prácticamente cualquier área que requiera de dichas prácticas, destacando entre muchas la robótica, la inteligencia artificial, la economía, la medicina y hasta la biología.

2.1.2. Grandes Volúmenes de Datos (Big Data)

Desde la prehistoria el hombre ha sentido la necesidad de grabar o almacenar datos, ya sea en una pintura rupestre, en un papiro, una hoja o en un disco duro. Sin embargo, aun cuando la necesidad ha sido la misma (almacenar datos), la cantidad de datos a guardar ha incrementado de forma exponencial, por lo que no se tenía la capacidad para almacenar los datos que se obtenían. Luego, el problema era que la velocidad con la que se obtenían los datos superaba la velocidad con la que se podían procesar. Posteriormente, se descubrió que muchos datos no se estaban estudiando debido a su heterogeneidad.

Estos tres problemas han incrementado su influencia en el desarrollo y funcionamiento de la tecnología en los últimos años, ocasionando el nacimiento de nuevos paradigmas que buscan solventar la problemática mencionada

anteriormente. De estos problemas o necesidades, surge el término Big Data, comúnmente utilizado para referirse a conjuntos de datos que superan la capacidad del software y/o hardware habitual para ser capturados, gestionados y procesados en un tiempo razonable.

Bajo este mismo orden de ideas, el analista Doug Laney (2001), empleado de META Group (ahora parte de Gartner Inc.), publicó un artículo titulado "*3-D Data Management: Controlling Data Volume, Velocity and Variety*" en el cual aborda estos tres aspectos relacionados al manejo de grandes volúmenes de datos:

Mientras las empresas luchan para consolidar sus sistemas y agrupar sus bases de datos redundantes para ser capaces de permitir consistencia a nivel operacional, analítico y operativo, las dinámicas condiciones económicas han hecho a este trabajo más difícil. E-commerce en particular, ha abordado los desafíos del manejo de datos basado en tres dimensiones: volumen, velocidad y variedad. En el 2001/02, las organizaciones deben considerar un conjunto variado de soluciones para lidiar con cada dimensión. (Doug Laney, 2001) [10]

Para una mejor comprensión de la problemática comentada anteriormente, se puede profundizar un poco en lo popularmente conocido como las 3 Vs de Big Data:

- Volumen: Característica relacionada a la cantidad de datos generados. Dependiendo de las capacidades del sistema, la cantidad de datos que define dicha característica puede ser variante (terabytes, exabytes, zettabytes...). Para el procesamiento de estas cantidades de datos no es recomendable utilizar bases de datos tradicionales debido a que su rendimiento es deficiente y no proveen técnicas de particionamiento, ocasionando que además sea costoso.[11]
- Velocidad: Característica relacionada con la velocidad a la cual son generados y procesados los datos. En este sentido, lo primordial es lograr que el análisis sea oportuno. Existen herramientas que permiten el análisis de estos datos sin tener siquiera que almacenarlos. [11]
- Variedad: Característica relacionada a la inherente heterogeneidad de los datos. Los datos serán de diferentes tipos, provenientes de diferentes fuentes y además podrán ser estructurados, semi-estructurados o no estructurados. [11]

Según distintos autores, en la actualidad el paradigma de BigData se debe abordar desde una mayor cantidad de dimensiones. Partiendo de lo anterior, se han estudiado distintos modelos que proponen nuevas dimensiones o Vs que complementan a las anteriormente mencionadas y permiten una evaluación más

amplia del espectro de los problemas que se tratan y las estrategias que se elaboran dentro de este paradigma.

En el año 2013, Mark van Rijmenan propone nuevas Vs para mejorar el entendimiento de la increíblemente compleja naturaleza del BigData, dichas dimensiones son las siguientes:

- Variabilidad: Se refiere a la capacidad que tienen los datos de variar su significado de forma rápida. Un caso ejemplo de esta propiedad es cuando se necesita ejecutar técnicas de procesamiento de lenguaje para la recolección de datos.^{[12][13]}
- Veracidad: Es una propiedad relacionada a la correctitud de los datos. Recibir una gran cantidad de datos en distintos volúmenes a una gran velocidad es irrelevante si la data no es correcta. Las estrategias de BigData necesitan asegurar que la data es correcta así como debe serlo el análisis que se realiza sobre la misma. ^[12]
- Visualización: Se refiere al proceso que consiste en hacer que las vastas cantidades de datos sean comprensibles, es decir, que sean fáciles de leer y entender. La visualización no se compone de gráficos ordinarios y simples, sino de gráficos complejos que incluyen muchas variables sobre los datos mientras se mantienen comprensibles para permitir la identificación sencilla de información de interés y recomendaciones. ^[12]
- Valor: Se refiere al valor oculto en los datos. Así se tengan grandes volúmenes de datos, si estos no poseen valor alguno, no se podría finalmente obtener información valiosa de estos.

2.1.3. Bases de datos

De acuerdo a Merriam-Webster, una base de datos es una (usualmente) gran colección de datos, organizada para una rápida búsqueda y recuperación.

Otra definición conocida es la del Dr. Naphtali Rishe, hecha en el año 1992 en su publicación *Database Design: The Semantic Modelling Approach*, en la cual define las bases de datos como un almacén de información actualizable de una aplicación, que oculta del usuario los aspectos físicos del almacenamiento y la representación de la información. La información almacenada en una base de datos es accesible a un nivel lógico sin necesidad de involucrar los conceptos físicos de su implementación. (Rishe, 1992) ^[14]

2.1.3.1. Modelos de bases de datos

Los modelos de bases de datos se pueden clasificar según su estructura lógica y el modo de almacenar, organizar y manipular los datos. Principalmente se

manejan dos tipos, las bases de datos relacionales y las bases de datos no relacionales.

2.1.3.1.1. Bases de datos relacionales

En el año 1970, Edgar Frank Codd en su obra *A Relational Model of Data for Large Shared Data Banks*, asentó las bases del famoso paradigma relacional. [15]

Una base de datos relacional es una colección de datos organizados en un grupo de tablas formalmente descritas, las cuales pueden ser accedidas o re-ensambladas de diferentes formas. [16]

La manera estándar para acceder a una base de datos y que suele servir de interfaz para los usuarios administradores de bases de datos es el Structured Query Language (SQL). Las sentencias SQL suelen usarse para obtener información de la bases de datos, agregar información, borrar o modificarla, entre otros.

En terminología de bases de datos relacionales, la conexión entre distintas tablas se le denomina "relación", a las columnas se les conoce como "atributos" y a las filas como "registros" o "tuplas".

El diagrama muestra una tabla con cuatro columnas: ID, Título, Año y Director. Hay tres filas de datos. Una flecha roja apunta desde la etiqueta 'columna' hacia la columna 'Título'. Otra flecha roja apunta desde 'registro' hacia la primera fila. Una tercera flecha roja apunta desde 'tabla' hacia el borde inferior de la tabla.

| ID | Título | Año | Director |
|----|----------------------------------|------|---------------|
| 1 | The player | 1992 | Robert Altman |
| 2 | Cookie's fortune | 1999 | Robert Altman |
| 3 | The man who shot Liberty balance | 1992 | John Ford |

Figura 1. Ejemplo de tabla

Fuente: Propia

Algunas bases de datos relacionales conocidas son: MySQL, Oracle, PostgreSQL, MariaDB.

2.1.3.1.2. Bases de datos no relacionales

Hace algunos años, los programadores web comenzaron a utilizar distintas técnicas para almacenar los datos usados con mayor frecuencia en la memoria RAM, dejando de hacer uso del camino más lento que implica el acceso a la base de datos completa desde el disco. El patrón de codificación utilizado para lograrlo, requería que todos los accesos a los datos se escribieran utilizando primitivas clave/valor, además de las tradicionales consultas SQL en la base de datos principal. Como los desarrolladores consiguieron más cómodo este enfoque, comenzaron a experimentar con bases de datos que utilizaban una interfaz de

clave/valor para el almacenamiento persistente, así como para la memoria caché, ya que igualmente la mayor parte de sus consultas se expresaban de esa forma. (Warden, 2011) [17]

Este es un raro ejemplo de la eliminación de una capa de abstracción, ya que la interfaz de clave/valor es menos expresiva y de un nivel más bajo que un lenguaje como el SQL. Estos sistemas requieren más trabajo de parte del desarrollador, pero ofrecen mucha más flexibilidad y control sobre el funcionamiento de la base de datos que se está manejando. También facilita a los desarrolladores de base de datos crear sistemas nuevos y experimentales para probar nuevas soluciones a requerimientos más complicados, como la escalabilidad, conjuntos de datos ampliamente distribuidos o aplicaciones de alto rendimiento. (Warden, 2011) [18]

2.1.3.1.2.1. Principales diferencias con las bases de datos SQL

- No utilizan SQL como lenguaje de consultas: La mayoría de las bases de datos NoSQL evitan utilizar este tipo de lenguaje o lo utilizan como un lenguaje de apoyo.[19]
- No utilizan estructuras fijas como tablas para el almacenamiento de los datos.[19]
- No suelen permitir operaciones JOIN: Al disponer de un volumen de datos tan extremadamente grande suele resultar deseable evitar los JOIN. Esto se debe a que, cuando la operación no es la búsqueda de una clave, la sobrecarga puede llegar a ser muy costosa.[19]
- Arquitectura distribuida: Las bases de datos relacionales suelen estar centralizadas en una única máquina o bien en una estructura maestro-esclavo, sin embargo en los casos NoSQL la información puede estar compartida en varias máquinas.[19]

2.1.3.1.2.2. Tipos de bases de datos NoSQL

Dependiendo de la forma en la que almacenen la información, se pueden encontrar varios tipos distintos de bases de datos NoSQL. Los tipos más utilizados son:

Bases de datos clave/valor. Son el modelo de base de datos NoSQL más popular, además de ser la más sencilla en cuanto a funcionalidad. En este tipo de sistema, cada elemento está identificado por una llave única, lo que permite la recuperación de la información de forma muy rápida. Se caracterizan por ser muy eficientes tanto para las lecturas como para las escrituras.[19]

Bases de datos orientadas a documentos: Este tipo almacena la información como un documento, generalmente utilizando para ello una estructura simple como JSON o XML y donde se utiliza una clave única para cada registro. Este tipo de implementación permite, además de realizar búsquedas por clave/valor, realizar consultas más avanzadas sobre el contenido del documento. Son las bases de datos NoSQL más versátiles. Se pueden utilizar en gran cantidad de proyectos, incluyendo muchos que tradicionalmente funcionarían sobre bases de datos relacionales.^[19] Una de las bases de datos orientadas a documentos más usadas en MongoDB.

MongoDB se una base de datos ágil que permite a los esquemas cambiar rápidamente cuando las aplicaciones evolucionan, proporcionando siempre la funcionalidad que los desarrolladores esperan de las bases de datos tradicionales, tales como índices secundarios, un lenguaje completo de búsquedas y consistencia estricta. Fue creada para brindar escalabilidad, rendimiento y gran disponibilidad, escalando de una implantación de servidor único a grandes arquitecturas complejas de centros multidados. MongoDB brinda un elevado rendimiento, tanto para lectura como para escritura, potenciando la computación en memoria (in-memory). La replicación nativa de MongoDB y la tolerancia a fallos automática ofrece fiabilidad a nivel empresarial y flexibilidad operativa. ^[77]

Bases de datos en Grafo: En este tipo de bases de datos, la información se representa como nodos de un grafo y sus relaciones con las aristas del mismo, de manera que se puede hacer uso de la teoría de grafos para recorrerla. Para sacar el máximo rendimiento a este tipo de bases de datos, su estructura debe estar totalmente normalizada, de forma que cada tabla tenga una sola columna y cada relación dos. ^[19]

Bases de datos columnares: Estas bases de datos cambian el enfoque clásico de organización por filas a una organización por columnas. En ellas, las unidades de datos ya no son insertadas como filas, sino que son convertidas en pares de claves y valores que constituyen la definición de cada una de las columnas. Con este cambio se mejora la velocidad en lecturas, ya que es muy rápido consultar un número reducido de columnas, sin embargo, este enfoque no es eficiente para realizar escrituras.

2.1.4. Organización y estructura de los datos

Acorde a la forma en la que se encuentren definidos los datos, se pueden clasificar según su estructura y organización.

2.1.4.1. Datos estructurados

La data estructurada hace referencia a cualquier dato que resida en un campo dentro de un registro o archivo. Esto incluye datos contenidos en bases de datos relacionales y hojas de cálculo. [20]

La data estructurada tiene la ventaja de ser fácilmente registrada, almacenada, consultada y analizada. En algún momento, debido al alto costo y limitaciones del almacenamiento, la memoria y el procesamiento, la única forma efectiva de manejar los datos era a través de las bases de datos relacionales y las hojas de cálculo. Cualquier cosa que no pudiera registrarse de forma estructurada debía ser almacenada en una hoja. [20]

2.1.4.2. Datos no estructurados

Datos no estructurados se refiere a información que no sigue un modelo de datos predefinido o que no está organizada de una manera predefinida. También pueden suceder cosas como que la data es estructurada, pero no está formalmente definida en un modelo de datos.

Los datos no estructurados suelen ser pesados en texto, aunque pueden contener información como fechas, números y hechos. Esto resulta en ambigüedades que hacen que sea difícil el procesamiento usando algoritmos tradicionales y suelen implicar un almacenamiento más complejo en bases de datos relacionales.

La data no estructurada incluye social media (tweets, blogs, posts, entre otros), call centers, emails, imágenes, entrevistas...

La distinción entre los datos estructurados y los datos no estructurados es importante, debido a la búsqueda del razonamiento automatizado, en la actualidad se requiere un análisis de ambos tipos de datos. El análisis predictivo requiere de ambos, y sin la integración y análisis de los datos no estructurados con los estructurados es imposible describir exhaustivamente, explicar, predecir o prescribir algún comportamiento. (Androile, 2015) [21]

2.1.4.3. Datos semi-estructurados

En términos generales, datos semiestructurados son aquellos que ni son datos en bruto, ni están estrictamente definidos en un sistema convencional de base de datos. Claramente, esta definición no es muy precisa. De hecho, una misma pieza de información puede ser vista como data no estructurada en una etapa temprana de procesamiento, pero luego convertirse en datos estructurados al haberle realizado algún tipo de análisis. (Abiteboul, 2006) [22]

En los datos semiestructurados, la información que está normalmente asociada con un esquema, está contenida dentro de los propios datos, que es la razón por la cual algunas veces se les llama “auto descriptivos”.

Los datos semiestructurados tienen una amplia variedad de formas para un vasto rango de aplicaciones, tales como bases de datos genómicas, bases de datos científicas, bibliotecas de programas y más generalmente, bibliotecas digitales, documentación online, comercio online. (Abiteboul, 2006) [18] Algunos de los formatos más utilizados definidos en este tipo de datos, son los archivos XML y JSON, entre muchos otros.

2.2. Lenguajes de programación

Una definición utilizada con frecuencia para lenguajes de programación es “Una notación para comunicar a una computadora lo que queremos hacer.”, pero ésta definición es inadecuada. Antes del 1940, las computadoras eran programadas vía “cables”: switches eran configurados por el programador para definir el cableado interno de la computadora para ejecutar la tarea requerida. Esto, efectivamente le comunicaba a la computadora el procesamiento que era deseado, sin embargo, la programación, si es que se le puede llamar así, consistía en la costosa y propensa a errores actividad de tumbar y reestructurar el hardware. (Louden, 2011) [23]

Un lenguaje de programación es un sistema de notaciones para describir la computación, en una forma leíble tanto para el humano, como para la máquina. (Louden, 2011) [23]

Un lenguaje de programación es considerado un conjunto de caracteres y reglas para su combinación, con las siguientes características:

1. El conocimiento del lenguaje de maquina es innecesario. [24]
2. Hay potencial para ejecutarlo en otras computadoras. [24]
3. Hay una variedad de instrucciones (de una a muchas). [24]
4. Hay una notación que es más cercana al problema de lo que sería el lenguaje ensamblador. [24]

La descripción de un lenguaje de programación usualmente se divide en dos componentes: la sintaxis (la forma en que se escribe) y la semántica (lo que significa).

Suelen clasificarse en lenguajes interpretados y compilados. Los lenguajes interpretados, manejan un interpretador, que básicamente es un programa que para ejecutar las rutinas, utiliza una biblioteca contenedora de la implementación de las instrucciones en el lenguaje de programación, con su equivalente en

lenguaje de máquina. A diferencia de los lenguajes compilados, que manejas un compilador específico encargado de traducir las sentencias del lenguaje de programación en código de máquina, también llamado código objeto. El código objeto puede ser ejecutado directamente en la máquina en la que fue compilado.

[25]

2.2.1. Python

Python es un lenguaje de programación de propósito general, interpretado, interactivo y orientado a objetos. Provee estructuras de datos de alto nivel, tales como listas y arreglos asociativos (también llamados diccionarios), escritura dinámica, módulos, clases, excepciones, manejo automático de la memoria, entre otros. Posee una sintaxis simple y elegante, pero poderosa, haciendo énfasis en lograr el fácil entendimiento del código, logrado reducir los costos de mantenimiento. (Sanner. 1999) [26][27]

Python fue diseñado en 1990, por Guido van Rossum. Como muchos otros lenguajes de scripting, es gratuito, incluso para propósitos comerciales, y puede ejecutarse en prácticamente cualquier computador moderno. (Sanner. 1999) [26]

Python es modular por naturaleza. El kernel es ligero y puede ser extendido importando distintos módulos. Python incluye una variedad de extensiones (algunas escritas en Python, otras en C o C++) para operaciones que van desde manipulación de arreglos de caracteres y expresiones regulares al estilo Perl, hasta Interfaces de Usuario Gráficas (GUI), utilidades para el desarrollo web, servicios del sistema operativo, entre muchas más. (Sanner. 1999) [26]

2.2.2. Scala

Scala es un lenguaje de programación, similar a Java, que unifica el enfoque programación orientado a objetos (O.O) con el enfoque funcional. Consiste en un lenguaje puro O.O en el sentido de que cada valor es un objeto. Los tipos y comportamientos de estos objetos son descritos por clases. Las clases a su vez pueden estar compuestas de forma mixta. Scala está diseñado para trabajar sin problemas con dos lenguajes de programación O.O menos puros pero más populares, Java y C#. [28]

Scala ha sido desarrollado a partir del 2001 en el laboratorio de métodos de programación École Polytechnique Fédérale de Lausanne (EPFL). Su primera versión (1.0) fue publicada en Noviembre de 2003. Su versión estable actual es la versión 2.11.7. [28]

Scala debe su nombre al acrónimo de “lenguaje escalable” (Scalable Language, en su idioma original), la cual es una de sus mayores beneficios

además de su colección de librerías que permiten simplificar tareas comunes y regulares como, por ejemplo, convertir automáticamente el tipo de dato de un valor de retorno de acuerdo a cual método se esté usando.

2.2.3. JavaScript

Javascript es un lenguaje de programación interpretado de alto nivel que apareció por primera vez en 1995 bajo la obra de Brendan Eich, empleado en su momento por Netscape Communications. Javascript es utilizado en buena parte de los sitios web y aplicaciones de servidor que existen alrededor del mundo. Se caracteriza por ser un lenguaje funcional multi-paradigma, orientado a objetos (basado en prototipos), imperativo, débilmente tipado y dinámico.

Su sintaxis básica es similar a Java y C++, reduciendo la curva de aprendizaje con respecto a conceptos nuevos necesarios para dominar el lenguaje. Las capacidades dinámicas del lenguaje le permite construir objetos, listas variables de parámetros y variables que pueden contener funciones en tiempo de ejecución; crear scripts dinámicos, realizar introspección de objetos y recuperar el código fuente.

Durante su trayectoria han sido desarrolladas distintas y tecnologías que facilitan el desarrollo de aplicaciones bajo este lenguaje aprovechando sus capacidades e integrándolo con otras herramientas. Entre estas tecnologías se puede destacar a NodeJs.

2.2.3.1. NodeJs

NodeJs es un entorno multiplataforma en tiempo de ejecución de código abierto para el desarrollo de herramientas y aplicaciones, mayormente, del lado del servidor. Consiste en un motor servidor altamente personalizable que realiza su procesamiento en bucles, listo para aceptar y responder a las solicitudes donde cualquiera de dichas solicitudes puede iniciar otras solicitudes a otros componentes del sistema. Este bucle es conocido como el bucle de eventos y es considerado como el “tiempo de ejecución” [29]. NodeJs viene empaquetado con conectores nativos y librerías en relación a, por ejemplo, los protocolos HTTP y SSL, mecanismos de compresión y acceso al sistema de archivos, y los protocolos TCP y UDP; cada uno de ellos cableados con Javascript, permitiendo crear un servidor web simple y dinámico en pocas líneas de código.^[29]

2.3. Herramientas para la Ciencia de Datos

La ciencia de datos es un área relativamente nueva que se encuentra en constante evolución, por lo que no existe un conjunto definitivo de herramientas o soluciones únicas y/o consolidadas. Sin embargo, actualmente muchos investigadores y desarrolladores que aplican por este campo de conocimiento reconocen ciertas prácticas y tecnologías bastante populares que pueden llegar a ser de extrema utilidad en esta materia. A continuación se presentan algunas de estas tecnologías organizadas según su propósito.

2.3.1. Acceso y procesamiento de datos

En el ecosistema de herramientas relacionadas a la ciencia de los datos se encuentran distintas tecnologías que proporcionan acceso a los datos de forma eficaz y la capacidad de interactuar con los mismos de forma útil para explotar las capacidades de la arquitectura en la que se despliegan y así concentrar más esfuerzos en los procesos de análisis. Entre estas tecnologías resaltan dos herramientas consolidadas: Apache Hadoop, con su sistema de archivos e implementación del modelo MapReduce; y Apache Spark.

Apache Hadoop es un proyecto open-source de la comunidad Apache™. Fue creado originalmente por Doug Cutting, quien también fue el cerebro detrás de la librería Lucene y colaborador principal del rastreador web Nutch, ambos proyectos apadrinados por la fundación Apache. [30]

Hadoop consiste en un framework que permite el procesamiento de grandes volúmenes de datos de forma distribuida a través de clústeres usando modelos sencillos de programación. Está diseñado para escalar desde servidores individuales hasta miles de nodos, cada uno de ellos ofreciendo sus propias capacidades de cómputo y almacenamiento. En lugar de depender en el hardware para ofrecer alta disponibilidad, el framework está preparado para detectar y manejar fallas en la capa de aplicación, proveyendo servicios altamente disponibles en la cima del clúster de ordenadores, cada uno de los cuales puede o no estar propenso a fallos. [30]

En un principio se inspiró en las publicaciones de Google referentes a Map Reduce Google File System (GFS). En la actualidad se encuentra en su versión 2.6.2 (28 de Octubre, 2015). [30]

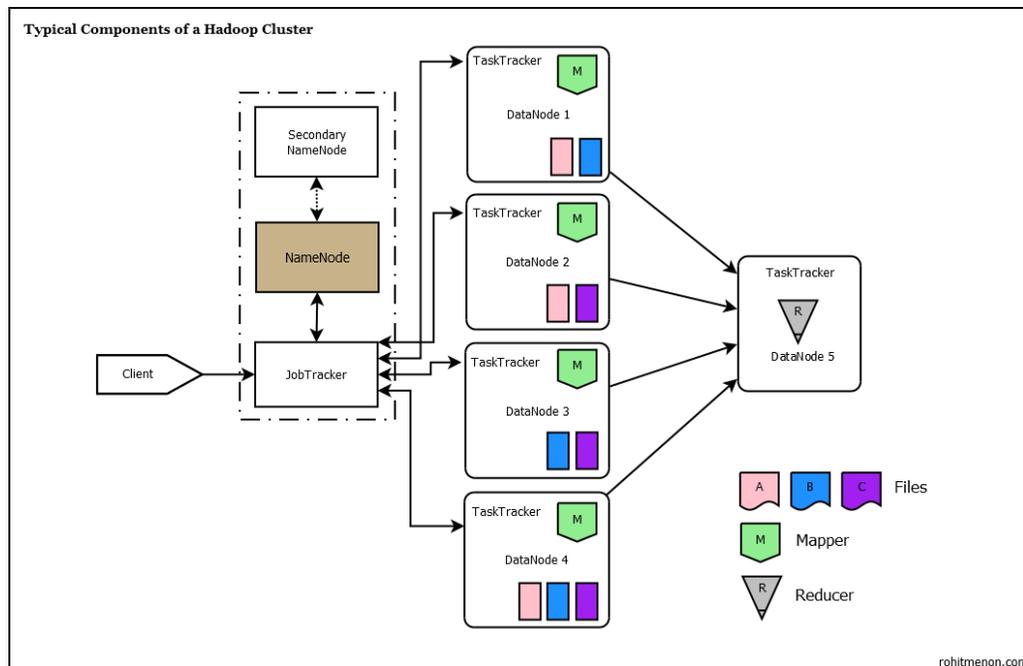


Figura 2. Arquitectura Apache Hadoop.

Fuente: <http://www.rohitmenon.com/index.php/introducing-hadoop-part-ii/>

Hadoop posee un módulo que contiene un conjunto de utilidades que permiten el soporte a otros proyectos de Hadoop denominado Hadoop Common ^[30]. Estas utilidades son consideradas como el núcleo del framework, el cual provee servicios esenciales para otras aplicaciones basadas en Hadoop. Aunado a lo anterior, Hadoop posee distintas implementaciones de ciertos componentes de forma nativa por razones de rendimiento y por falta de disponibilidad de implementaciones de Java. Esos componentes están disponibles en una librería única y dinámicamente enlazada llamada native hadoop library. ^[31]

Uno de los dos mayores atributos de Hadoop es su sistema manejador de archivos llamado Hadoop Distributed File System (o HDFS, por sus siglas en inglés). El mismo está diseñado para soportar aplicaciones, muchas de las cuales hacen uso de procesos Map Reduce, que leen y escriben grandes cantidades de datos en lotes en lugar de generar muchos accesos aleatorios a un montón de archivos pequeños. HDFS busca solucionar los problemas relacionados al almacenamiento de grandes conjuntos de datos mediante la distribución de los mismos. Dicha distribución se logra replicando los datos en distintos nodos y manteniendo fuertes protocolos de organización y disponibilidad.

El otro gran atributo de Hadoop es su implementación del modelo de programación MapReduce, asociado al procesamiento y generación de grandes cantidades de datos. En este modelo, los usuarios especifican dos funciones elementales: una función de “map”, que procesa pares clave/valor para generar un

grupo intermedio de pares clave/valor, y una función “reduce” que combina todos los valores intermedios que posean la misma clave. [32]

Los programas escritos mediante este modelo son paralelizados automáticamente y ejecutados en amplios clústeres de ordenadores convencionales. El sistema se encarga, en tiempo de ejecución, de los detalles referentes al particionamiento de los datos de entrada, la planificación de la ejecución de los programas a través de un conjunto de ordenadores, el manejo de fallos de máquinas y de la administración de la comunicación requerida entre los nodos. Esto permite a los programadores sin experiencia en sistemas paralelos y distribuidos utilizar fácilmente los recursos de un sistema distribuido de gran escala. [32]

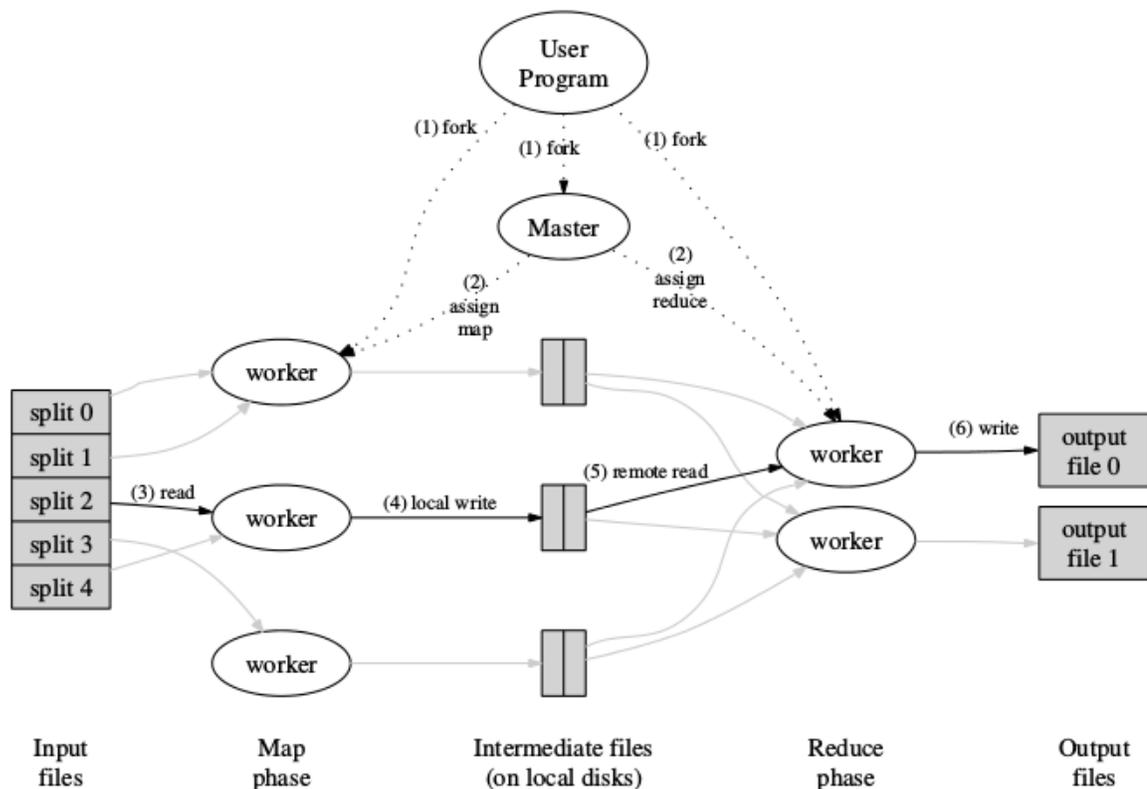


Figura 3. MapReduce

Fuente: <http://static.googleusercontent.com/media/research.google.com/es//archive/mapreduce-osdi04.pdf>

Por otro lado se encuentra Apache Spark, el cual es un motor de procesamiento de datos caracterizado por su velocidad, ejecución en memoria y sus API's elegantes y expresivas que permiten a los desarrolladores realizar con eficiencia transmisiones de datos, machine learning o cargas SQL que requieran accesos rápidos y repetitivos sobre conjuntos de datos. [33]

Spark está diseñado para la ciencia de datos y su abstracción facilita este tipo de tareas. Los científicos de datos suelen usar algoritmos de machine learning, los cuales se caracterizan por ser repetitivos. La habilidad de Spark para mantener en memoria un conjunto de datos incrementa considerablemente la velocidad de procesamiento, haciendo a Spark el motor ideal para la implementación de este tipo de algoritmos. [33]

Apache Spark consiste en un núcleo y un conjunto de librerías. El core es el motor de ejecución distribuida y las API's de Java, Scala y Python ofrecen una plataforma para el desarrollo de aplicaciones distribuidas de extracción, transformación y carga. También existen librerías, desarrolladas sobre el motor, que ofrecen diversas cargas de trabajo para el streaming, SQL y machine learning.

[33]

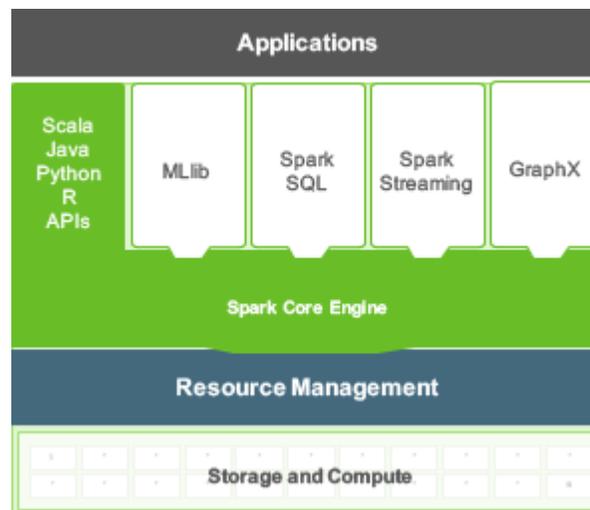


Figura 4. Arquitectura de Spark.

Fuente: <http://hortonworks.com/hadoop/spark/>

- Spark SQL: permite unificar el acceso a data estructurada, la cual puede venir de distintas fuentes. Spark SQL permite la unificación de dicha data en un solo sitio. Tiene compatibilidad con Hive y conexiones mediante JDBC.
- ML: librería escalable de Spark que contiene algoritmos de Machine-Learning.
- GraphX: API facilitado por Spark para la manipulación de grafos.
- Spark Streaming: Provee a los usuarios la capacidad de recuperar trabajos que no pudieron ser finalizados junto con su último estado.

A continuación se presenta una tabla comparativa que refleja bajo ciertos aspectos y a grandes rasgos las mayores diferencias entre Apache Hadoop y Apache Spark.

| Característica | Hadoop | Spark |
|------------------------|---|--|
| Rendimiento | Funciona mejor cuando los datos no caben en memoria y puede correr junto a otros servicios | Funciona mejor cuando toda la data cabe en memoria, especialmente en clusters dedicados. |
| Facilidad de uso | Dificultad para programar considerable. Existen herramientas que facilitan su uso pero conllevan una curva de aprendizaje considerable. | Posee APIs cómodas para Java, Scala y Python. Es fácil escribir funciones definidas por el usuario. Incluye un modo interactivo para correr comandos con feedback inmediato. |
| Costo | Puede ser más barato gracias a la mayor cantidad de personal disponible. | Más efectivos en costos según los benchmarks, aunque la contratación de personal puede costar más. |
| Procesamiento de datos | Hadoop puede ser considerada como la herramienta definitiva para el procesamiento por lotes. | Spark ofrece múltiples funciones y utilidades para muchos escenarios en materia de procesamiento de datos. |
| Seguridad | Hadoop posee más proyectos y soluciones con respecto a la seguridad. | Spark aún tiene aspectos que cubrir en materia de seguridad. |

Tabla 1. Hadoop vs Spark

Fuente: Propia.

2.3.2. Virtualización

La virtualización asistida por hardware, donde una computadora con ciertas capacidades de hardware (disponibles en la arquitectura x86 desde hace ya diez

años y hoy presentes incluso en los CPUs de más baja gama) ejecuta un hipervisor, software específico que se ubica por debajo del sistema operativo y permite que la computadora sea compartida entre distintos sistemas operativos (o distintas instancias del mismo), aparentando ante cada uno de ellos ser una computadora independiente. Como apéndice de esta categoría entrarían los paravirtualizadores, que corren versiones de dichos sistemas operativos que saben que se ejecutarán dentro de una máquina virtual, con lo cual delegan algunos procesos al sistema anfitrión, muchas veces aumentando la estabilidad o reduciendo la penalización de velocidad.

Una modalidad más de virtualización es el uso de contenedores. Esta modalidad puede comprenderse mejor si se contrasta con las anteriores. En vez de proveer una computadora virtual a los sistemas operativos huéspedes, los contenedores buscan proveer un sistema operativo virtual a conjuntos de programas. Quiere decir que a cambio de reducir la flexibilidad, esta modalidad de virtualización provee un mejor rendimiento (las aplicaciones virtualizadas tienen exactamente la misma velocidad que en el hardware nativo) y menor impacto en el resto del sistema (cuando los procesos que forman parte del contenedor no tienen trabajo por realizar, su consumo de recursos es prácticamente cero, a diferencia de un sistema virtualizado, que debe seguir pendiente a posibles eventos).

La reducción de flexibilidad referida consiste en que, dado que el núcleo del sistema operativo en ejecución es uno solo, todos los procesos que sean ejecutados empleando esta técnica deben estar compilados para la misma arquitectura y sistema operativo. Esto significa que por ejemplo, en un sistema anfitrión x86 con Linux, todos los contenedores deberán ser también x86 con Linux.

A principios de 2013, se anunció la liberación de un innovador programa que ha cambiado en gran medida las reglas del juego para el despliegue de aplicaciones: Docker. Docker es un proyecto de código abierto que permite automatizar el despliegue de aplicaciones empaquetándolas junto con sus dependencias dentro de un contenedor virtual que puede ser desplegado, gracias a estar basado en estándares abiertos, en la gran mayoría de las distribuciones de Linux y en Windows. [34]

Cada aplicación dentro de un contener esta encapsulada como una “imagen” y se organizan lógicamente en forma de capas. Los contenedores se encargan de aislar estas imágenes unas de otras y entre la arquitectura subyacente, permitiendo a vez la capacidad de restringir servicios y ofreciendo una visión casi completamente privada a cada aplicación con respecto a la estructura del sistema de archivos, interfaces de red y espacio de proceso. Los contenedores de Docker solamente incluyen los datos de la aplicación con sus dependencias y, gracias a

que pueden ser desplegados en cualquier computadora o infraestructura, no necesitan incluir un sistema operativo para su despliegue, permitiéndole ahorrar bastante espacio y crear contenedores ligeros a diferencia de las máquinas virtuales. [34]

Uniando, los dos temas expuestos: La principal contribución de Docker es que convierte el engorroso despliegue de servidores virtuales en un sencillo despliegue de aplicaciones. Y, claro está, de aplicaciones empaquetadas a la moda, con el mínimo de dependencias. Sin embargo, un uso irresponsable de Docker puede llevar a confiar en software cuyo origen no necesariamente es el esperado, sin saber en qué consisten los cambios y cómo éstos pueden afectar. Los paquetes que dicen ser oficiales de determinadas distribuciones han demostrado haber sido alterados, lo cual denota un punto importante a tomar en cuenta. Además de ser importante tener muy claro las “puertas” que se dejen abiertas en los contenedores que se creen.

En la siguiente tabla se describe a muy alto nivel el tipo de diferencias de características que existen entre máquinas virtuales y contenedores. Es posible que algunas características resulten más o menos deseables dependiendo de las necesidades de su propia aplicación, y que al igual que con todo el software, el trabajo adicional proporciona una mayor compatibilidad de características, especialmente en el área de la seguridad. [35][37]

| Característica | Máquinas virtuales | Contenedores |
|---|--|--|
| Compatibilidad con la seguridad "Predeterminada" | En un mayor grado | En un grado ligeramente menor |
| Memoria en disco necesaria | Sistema operativo más aplicaciones completos | Solo requisitos de aplicación |
| Tiempo que se tarda en iniciar | Significativamente más largo: el arranque del SO más la carga de la aplicación | Significativamente más corto: solamente es necesario que se inicien las aplicaciones porque el kernel ya está en ejecución |
| Portabilidad | Portable con la preparación | Portable en formato de imagen; suelen ser más |

| | adecuada | pequeños |
|----------------------------|--|--|
| Automatización de imágenes | Varía dependiendo del operativo y las aplicaciones | considerablemente del sistema Registro docker; otros |

Tabla 2. Máquinas Virtuales vs Contenedores

Fuente: <https://azure.microsoft.com/es-es/documentation/articles/virtual-machines-windows-containers/>

2.3.3. Visualización e interfaz de usuario

Luego de la aparición de NodeJs en el 2009, distintos grupos de desarrollo lo han integrado con otras tecnologías creando conjuntos de subsistemas de software para el desarrollo de aplicaciones, entre los más reconocidos se encuentra MEAN Stack (acrónimo para: MongoDB, ExpressJs, AngularJs y NodeJS). Por otro lado, también se han elaborado frameworks que dan provecho a las capacidades de NodeJs de forma directa, entre los cuales se pueden mencionar a SailsJs, KoaJs y MeteorJs. [37]

Entre estas últimas opciones se pueden destacar a Meteor, el cual fue presentado en Diciembre del 2011 bajo el nombre de Skybreak [38], por su facilidad para crear prototipos y producir código multiplataforma para aplicaciones que funcionan en tiempo real.

Meteor es una tecnología de código abierto escrita usando NodeJs, se integra con MongoDB y utiliza el Protocolo de Datos Distribuidos (DDP, por sus siglas en inglés) y el patrón de publicación y suscripción para propagar automáticamente los cambios en los datos al cliente sin requerir que el desarrollador escriba código extra para la sincronización. Aparte, gracias a su sistema de plantillas similar a Handlebars, sus plantillas declarativas sin lógica embebida y su capacidad para recargar el sitio web de acuerdo a los cambios en código y estilo; Meteor se convierte en un framework bastante fácil de usar.

Con respecto a la renderización de contenido, Meteor soporta tres librerías para las vistas: React, Angular y Blaze. React fue creado por Facebook en el año 2013, Angular fue creado por Google en el año 2010 y Blaze fue creado como parte de Meteor durante su lanzamiento en el 2011. Estas tres librerías han sido usadas exitosamente por una buena cantidad de aplicaciones en producción. Blaze es considerado como el más fácil de aprender y tiene la mayor cantidad de paquetes full-stack de Meteor, sin embargo, Angular y React están más desarrollados y tienen comunidades de apoyo más grandes. [39]

Aparte, Meteor es completamente compatible con el Manejador de Paquetes de Node (NPM, por sus siglas en inglés), permitiéndole amplificar las capacidades del servidor reusando código creado por otros desarrolladores e integrándolo con paquetes y librerías para añadir más funciones, reducir tiempos de desarrollo y facilitar aún más su uso. No obstante, Meteor es también compatible con librerías y herramientas de terceros, como por ejemplo, Plotly.

Plotly es una herramienta para la visualización y análisis de datos que ofrece la capacidad de elaborar gráficos, análisis y estadísticas en línea. Esta construido usando Python y el framework Django. Con respecto a su vista para el usuario, hace uso de HTML, CSS, Javascript y la librería de visualización D3.js para producir las visualizaciones de datos dinámicas e interactivas por las cuales es reconocido [40]. Entre los servicios que Plotly ofrece se encuentran las librerías API para Python, R, MATLAB, NodeJs, Julia, Arduino y una API REST. Además, también puede ser usado para estilizar gráficos interactivos con IPython, una consola de comandos para computación interactiva en múltiples lenguajes, desarrollada originalmente para Python. [41]

Por otro lado tenemos a Mean, el cual es un conjunto de softwares que compromete a MongoDB, Express.js, Angular.js y Node.js. Como ya se ha explicado con anterioridad a Node.js y a MongoDB, se especificaran solamente a los dos componentes restantes:

- Express.js: es un framework por encima de Node.js que permite crear servidores web y recibir peticiones HTTP de una manera sencilla, lo que permite también crear APIs REST de forma rápida
- Angular.js: es un framework de Javascript para la parte cliente o Frontend de una aplicación web, que respeta el paradigma MVC y permite crear Single-Page Applications (Aplicaciones web que no necesitan recargar la página), de manera más o menos sencilla. Es un proyecto mantenido por Google y que actualmente está muy en auge.

Una de las principales ventajas de Mean es que emplea el mismo lenguaje de programación en todas las partes de la aplicación lo que permite que una persona pueda manejarse en todos los ámbitos de una aplicación web moderna aunque se especialice en uno de ellos. De esta manera se colabora más en los proyectos y el desarrollo es más continuo. Esto, añadido a las pruebas automatizadas y los test unitarios, los repositorios git como GitHub o Bitbucket, los servidores de integración continua y las PaaS (Plataformas como servicios por sus siglas en inglés) como Heroku o Nodejitsu hacen que el desarrollo web moderno sea más divertido y ágil.

En la siguiente tabla se señalaran los aspectos más relevantes entre estas dos opciones para desarrollar aplicaciones web con Javascript.

| | Meteor | Mean |
|-----------------|--|--|
| Características | Facilidad de uso | Curva de aprendizaje considerable |
| | Comunidad amplia y en crecimiento | Comunidad limitada |
| | Actualizaciones en tiempo real | Promueve la separación de las capas y la reutilización de código |
| | Recarga de código en caliente | Excelente rendimiento con respecto a los protocolos HTTP y TCP |
| | Actualización automática de plantillas reactivas | Javascript como único lenguaje de desarrollo |
| | Manejo de sesiones | Diversos casos de éxito |
| | Completación en el lado del servidor | Escalable en la nube |

Tabla 3. Meteor y Mean

Fuente: Propia

2.4. Minería de datos

La minería de datos, también llamada “Descubrimiento de conocimiento en bases de datos” (KDD, por sus siglas en ingles), es un sub-campo interdisciplinario de las ciencias de la computación que trata los procesos computacionales del descubrimiento de patrones sobre grandes conjuntos de datos, involucrando métodos de la ingeniería artificial, machine learning, análisis estadísticos y sistemas de bases datos. [42]

El propósito general del proceso de minería de datos es extraer información útil de un conjunto de datos y transformarla en una estructura entendible. Aparte

de las actividades de análisis crudo de datos, involucra aspectos del manejo de datos en bases de datos, procesamiento de datos, visualización, métricas, actualización en línea y consideraciones de modelos, inferencias y complejidad. [42]

2.4.1. Proceso de la minería de datos

El proceso de minería de datos se caracteriza por ser interactivo e interactivo por cada paso por el cual está conformado. [43]

2.4.1.1. Entendimiento del dominio del problema

Este es el paso inicial de preparación para todo el proceso, definiendo el ambiente para el entendimiento del problema y lo que se deberá hacer sobre distintos aspectos como la transformación de los datos, algoritmos, representación, entre otros. [43]

Las personas a cargo del proceso de minería de datos necesitan entender y definir las metas del usuario final y el ambiente en el cual tomara lugar el descubrimiento de conocimiento. A medida que el proceso avanza puede que ocurran revisiones y mejoramientos de este paso. [43]

2.4.1.2. Selección y creación del conjunto de datos

La determinación de los datos a usar para el descubrimiento de conocimiento incluye averiguar cuales datos están disponibles, obtener los datos necesarios adicionales e integrar toda la data en un mismo conjunto junto con los atributos que serán considerados durante el proceso. [43]

El éxito del proceso depende directamente de los atributos de los datos, tanto que si faltan algunos atributos importantes el estudio puede fallar, por ello es recomendable considerar la mayor cantidad de atributos posible. [43]

2.4.1.3. Procesamiento y limpieza

En esta etapa, la fiabilidad de los datos es mejorada. Aquí se incluyen las actividades de esclarecimiento de datos como el manejo de los valores faltantes y la eliminación del ruido y de los valores atípicos. Para realizar lo anteriormente dicho, puede que se deban usar varios métodos estadísticos complejos y algoritmos de minería específicos para el contexto. [43]

2.4.1.4. Transformación de los datos

Durante este paso se preparan, desarrollan y generan mejores datos para el proceso. Los métodos usados suelen incluir la reducción de dimensionalidad, como la selección y extracción de características relevantes y el muestreo de

registros; y la transformación de atributos, como la discretización de atributos numéricos y transformaciones funcionales. Esta etapa es crucial para el éxito de todo el proceso de minería y suele ser bastante específico para cada proceso en particular. [43]

2.4.1.5. Elección de las tareas de minería de datos

La elección de las tareas adecuadas depende casi en su totalidad de los objetivos de los procesos de minerías y de las etapas anteriores. Existen dos objetivos generales en la minería de datos: la predicción y la descripción. La predicción se refiere usualmente a la minería supervisada, mientras que la descripción incluye aspectos no supervisados y de visualización de la minería de datos. [43]

La mayoría de las técnicas de minería están basadas en el aprendizaje inductivo, donde un modelo es construido explícita o implícitamente mediante la generalización de un número suficiente de ejemplos de entrenamiento. La suposición subyacente del acercamiento inductivo es que el modelo entrenado puede ser aplicable para casos futuros. La estrategia también toma en cuenta el nivel de meta-aprendizaje para el conjunto particular de datos. [43]

2.4.1.6. Elección del algoritmo de minería de datos

Una vez que se decide la estrategia, se pueden decidir correctamente las tácticas. Esta etapa incluye la selección del método específico a ser usado para la búsqueda de patrones. [43]

Para cada estrategia de meta aprendizaje existen distintas posibilidades de cómo pueden ser alcanzados dicho aprendizaje. El mismo de concentran en explicar las que causa que un algoritmo de minería sea o no exitoso para un problema en particular. Por lo tanto, este acercamiento intenta entender las condiciones sobre las cuales un algoritmo de minería es más apropiado. [43]

2.4.1.7. Aplicación del algoritmo

En este paso se implementa el algoritmo de minería, el cual puede ser aplicado varias veces hasta que se obtenga un resultado satisfactorio. Por cada repetición se ajustan los parámetros de control del algoritmo, como por ejemplo el mínimo número de instancias en una hoja particular de un árbol de decisiones. [43]

2.4.1.8. Evaluación de los resultados

Durante esta etapa se realiza la evaluación e interpretación de los patrones obtenidos acorde con las metas definidas en la primera etapa. También, se

consideran posibles cambios sobre etapas anteriores según su posible efecto sobre los resultados del algoritmo. [43]

Este paso se concentra en la comprensibilidad, la usabilidad del modelo y la documentación del conocimiento obtenido para futuros usos. [43]

2.4.1.9. Uso del conocimiento obtenido

El éxito de esta etapa determina la efectividad de todo el proceso de minería de datos. El conocimiento se vuelve activo, en el sentido de que se pueden realizar cambios al sistema y medir los efectos obtenidos, así como también puede ser incorporado a otros sistemas. [43]

Existen distintos desafíos durante este paso, uno de los más importantes es la pérdida de las condiciones de laboratorio sobre las cuales el modelo ha operado. Por ejemplo, el conocimiento fue obtenido de estado estático de datos de ejemplo, pero ahora la data se vuelve dinámica. [43]

Algunas estructuras de datos pueden cambiar, ciertos atributos pueden no estar disponibles y el dominio de datos puede estar modificado, de modo que muchos atributos pueden tener valores que antes no eran asumidos durante el proceso de minería. [43]

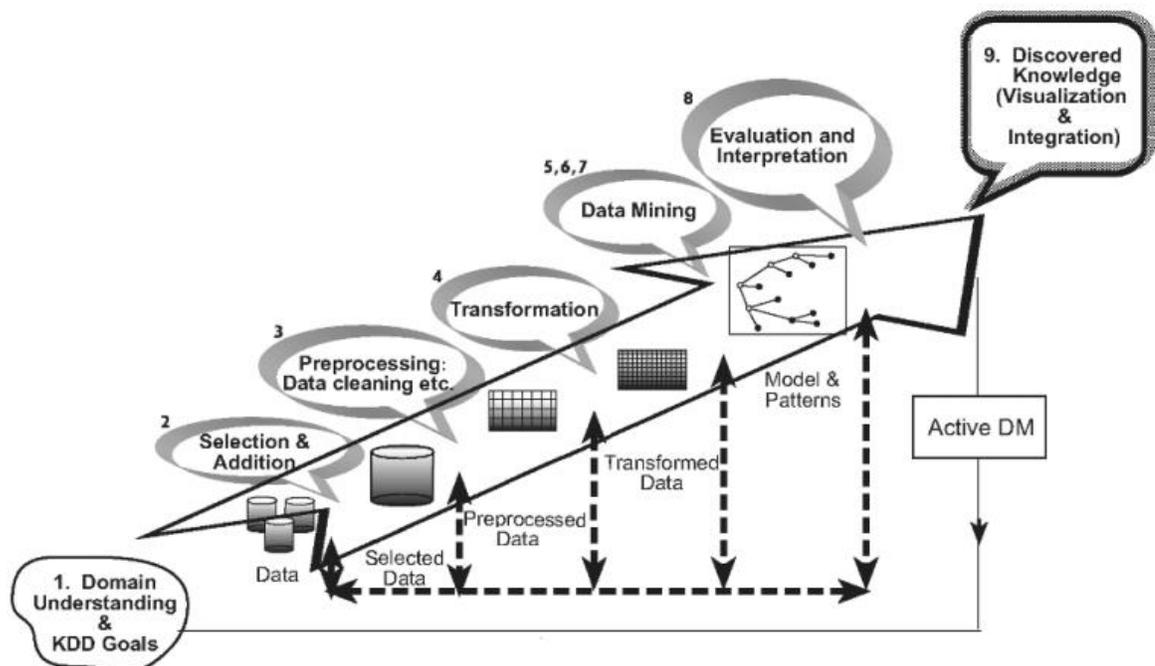


Figura 5. Etapas de un proceso de minería de datos

Fuente: Data Mining and Knowledge Discovery Handbook, Second Edition. Maimon, O. y Rokach, L. (2010).

2.4.2. Métodos, técnicas y algoritmos de la minería de datos

Existen muchos métodos de minería utilizados para distintos propósitos y objetivos. La taxonomía ayuda a entender la variedad de los mismos, su interrelación y agrupamiento. Es importante distinguir entre dos tipos principales de minería de datos: la minería orientada a la verificación, donde el sistema verifica las hipótesis del usuario; y la minería orientada al descubrimiento, donde el sistema encuentra nuevas reglas y patrones de forma autónoma. [43]

Los métodos de verificación lidian con la evaluación de una hipótesis propuesta por una fuente externa. Estos métodos incluyen los más comunes métodos de estadística como el test de hipótesis y análisis de varianza. [43]

Los métodos de descubrimientos son aquellos que identifican automáticamente patrones en los datos. Estos métodos se dividen en dos corrientes, los métodos de descripción y los métodos de predicción. [43]

Los métodos de descripción están orientados a la interpretación de los datos, concentrándose en entender la forma en la que la data se relaciona de forma subyacente con sus partes. [43]

Por otra parte, los métodos de predicción apuntan a construir un modelo de comportamiento, el cual obtiene nuevos ejemplos y es capaz de predecir valores de una o más variables relacionadas al ejemplo. [43]

2.4.2.1. Detección de anomalías

La detección de anomalías consiste en la identificación de elementos, eventos u observaciones que no se ajustan a un patrón esperado o a otros elementos de un conjunto de datos. [44]

Existen tres categorías de técnicas de detección:

- Detección no supervisada: Son técnicas que detectan anomalías en un conjunto de datos de prueba no categorizados bajo la suposición de que la mayoría de las instancias en los datos son normales mediante la búsqueda de instancias que parecen encajar menos con el resto. [45][46]
- Detección supervisada: Consisten en técnicas que requieren un conjunto de datos que han sido categorizados como “normales” y “anormales” e involucran el entrenamiento de un clasificador. [45][46]
- Detección semi-supervisada: Son técnicas que construyen un modelo que representa el comportamiento normal a partir de un conjunto de datos de

entrenamiento “normales” para luego probar el parecido de las instancias de prueba que serán generadas en el modelo de aprendizaje. [45][46]

2.4.2.2. Aprendizaje por reglas de asociación

Es un método para el descubrimiento de relaciones interesantes entre las variables en grandes conjuntos de datos mediante el uso de distintas medidas de interés. [47]

Muchos algoritmos para la generación de reglas de asociación han aparecido a lo largo de los años. Algunos de ellos son bastante conocidos, entre ellos están los algoritmos Apriori, Eclat y GP-Growth, pero solo hacen la mitad del trabajo ya que son algoritmos que encuentran conjuntos de elementos frecuentes. Por ello, es necesario ejecutar una segunda etapa para generar las reglas sobre dichos conjuntos. Algunos algoritmos usados para la generación de reglas son: AprioriDP, algoritmos de minería de reglas de asociación basada en contexto, algoritmos basados en conjuntos de nodos, búsqueda OPUS, entre otros. [47]

2.4.2.3. Análisis de clústers

El análisis de clústers consiste en el conjunto de tareas de agrupamiento de un conjunto de objetos en una forma específica que permita identificar distintas agrupaciones de objetos similares unos con otros. [48][49]

Representa una de las principales tareas de la minería de datos de exploración y una técnica bastante común para el análisis estadístico de datos. Es usada en un muchos campos, incluyendo machine learning, reconocimiento de patrones, análisis de imágenes, recuperación de información y bioinformática. [48][49]

El análisis de clústers no es un algoritmo propiamente, sino más bien la actividad general a realizar. Puede ser alcanzado mediante distintos algoritmos que difieren significativamente en su noción de lo que constituye a un clúster y como se pueden encontrar de forma eficiente. [48][49]

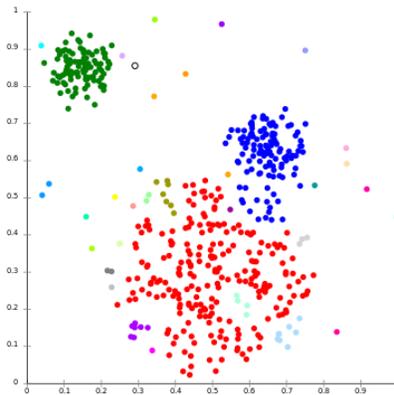


Figura 6. Ejemplo de grupos en conjuntos de datos analizados

Fuente: https://en.wikipedia.org/wiki/Cluster_analysis

2.4.2.4. Análisis de regresión

El análisis de regresión es un proceso estadístico usado para estimar las relaciones existentes entre distintas variables. Incluye muchas técnicas para el modelado y análisis de muchas variables cuando la concentración se basa en la relación entre una variable dependiente y una o más variables independientes. Más específicamente, el análisis de regresión ayuda a entender como los valores típicos de una variable dependiente cambian cuando alguna de las variables independientes varía y las otras se mantienen controladas. [50]

Para el análisis de la regresión han sido desarrollados muchos métodos, entre los más familiares están la regresión lineal y la regresión de mínimos cuadrados ordinarios, los cuales son métodos paramétricos donde la función de regresión es definida en términos de un número finito de parámetros desconocidos que son estimados mediante los datos. Las regresiones no paramétricas se refieren a las técnicas que permiten a la función de regresión basarse en conjunto específico de funciones, el cual puede ser infinitamente dimensional. [50]

2.4.2.5. Minería de texto

La minería de texto se refiere al proceso de derivar información de gran calidad a partir de texto. Esta información se deriva usualmente a través de la elaboración de patrones y tendencias por medios como el aprendizaje estadístico por patrones. La minería de texto involucra el proceso de estructuración de los datos de entrada, que consiste en la adición y eliminación de particularidades lingüísticas de los datos de entrada y su almacenamiento en una base de datos;

derivar patrones dentro de la data estructurada y, finalmente, evaluar e interpretar las posibles salidas. [51]

Las tareas típicas dentro de la minería de datos incluyen la categorización del texto, cauterización del texto, extracción de conceptos y entidades, producción de taxonomías granulares, análisis de sentimientos, sumarización de documentos y modelación de relaciones entre entidades. [51]

El análisis del texto involucra la recuperación de información, análisis léxico para estudiar la frecuencia de la distribución de las palabras, reconocimiento de patrones, elaboración de etiquetas, extracción de información, técnicas de minería de datos como el análisis de asociaciones, visualización y análisis predictivos. [52]

El objetivo principal es, esencialmente, convertir el texto en datos para su análisis mediante la aplicaciones de procesos naturales de lenguaje y métodos analíticos. [51][52]

2.4.2.6. Clasificador bayesiano ingenuo

El Clasificador bayesiano ingenuo es una técnica de clasificación probabilística y predicción que construye modelos que predicen la probabilidad de posibles resultados. Está basado fundamentado en el teorema de Bayes y algunas hipótesis simplificadoras adicionales. Esta técnica asume que la presencia o ausencia de una característica particular no está relacionada con la presencia o ausencia de cualquier otra característica, dada la clase variable.

Para otros modelos de probabilidad, los clasificadores de Bayes ingenuo se pueden entrenar de manera muy eficiente en un entorno de aprendizaje supervisado. En muchas aplicaciones prácticas, la estimación de parámetros para los modelos Bayes ingenuo utiliza el método de máxima verosimilitud, en otras palabras, se puede trabajar con el modelo ingenuo de Bayes sin aceptar probabilidad bayesiana o cualquiera de los métodos bayesianos.

Aun cuando existen estudios que demuestran de forma teórica la eficacia del clasificador bayesiano a pesar de su diseño y sus suposiciones sobresimplificadas, también existen investigaciones que comparan a esta tecnica con otros algoritmos, una de ellas [66] demostró que el rendimiento del clasificador bayesiano puede ser superado por otros acercamientos, como por ejemplo los bosques aleatorios o "Random Forests".

Una ventaja del clasificador de Bayes ingenuo es que solo se requiere una pequeña cantidad de datos de entrenamiento para estimar los parámetros necesarios para la clasificación.

2.3.2.7. Bosques aleatorios

El término bosques aleatorios o “Random Forests” se refiere al método de aprendizaje en conjunto para clasificaciones, regresiones y entre otras tareas. Este algoritmo fue introducido por primera vez por Tin Kam Ho en 1995, usando el método de subespacios aleatorios. Más adelante fue extendido por Leo Breiman en el 2001. El método consiste en una modificación sustancial del meta-algoritmo Agregación de Bootstrap ó “bagging”, construyendo múltiples árboles de decisión en tiempo de entrenamiento y mostrando la moda de las clases, para el caso de la clasificación; ó la predicción media de cada árbol individual para el caso de la regresión.

La idea esencial del bagging es promediar muchos modelos ruidosos pero aproximadamente imparciales y, por tanto, reducir la variación. Los árboles son los candidatos ideales para el bagging, dado que ellos pueden registrar estructuras de interacción compleja en los datos y si crecen lo suficientemente profundo obtienen una parcialidad relativamente baja. Producto de que los árboles son notoriamente ruidosos, ellos se benefician grandemente al promediar.

Entre las ventajas del random forests se puede destacar que es uno de los algoritmos de aprendizaje más certeros que hay disponible dado que para un ser de datos lo suficientemente grande produce resultados bastante aceptables. Aparte, puede manejar cientos de variables de entrada sin excluir ninguna de ellas, dando estimados de que variables son importantes en la clasificación.

Con respecto a sus desventajas, el random forests tiende a sobreajustarse en ciertos grupos de datos con tareas de clasificación o regresión ruidosas. Por otro lado, si los datos contienen grupos de atributos correlacionados con similar relevancia para el rendimiento, entonces los grupos más pequeños están favorecidos sobre los grupos más grandes.

2.5. Redes sociales

De acuerdo con Boyd y Ellison (2007), una red social se define como un servicio que permite a los individuos:

- Construir un perfil público o semipúblico dentro de un sistema delimitado.
[53][54]
- Articular una lista de otros usuarios con los que comparten una conexión.
[53][54]
- Ver y recorrer su lista de las conexiones y las realizadas por otros dentro del sistema. [53][54]

Según Bartolomé (2008), las redes sociales reflejan lo que en otros tiempos se mostraba mediante sociogramas: una serie de puntos representando individuos, notablemente personas, unidos mediante líneas que representan relaciones. Esta forma de representación puede también denominarse grafo. El carácter de una red social puede ser muy variado así como el motivo aglutinador: desde el sexo a la afición por los viajes, las redes sociales mueven el mundo, aunque evidentemente, algunas los mueven más que otras. [53][54]

Tichy, Tushman y Frombrun [55][56] han observado que las redes pueden caracterizarse por diversas dimensiones:

A. Contenido Transaccional

- a. Tipo de intercambio en la red: Expresión de afecto, influencia, intercambio de información, intercambio de recursos o de bienes y servicios.

B. Naturaleza de los nexos

- a. Intensidad: Fuerza de la relación.
- b. Reciprocidad: Grado en que la relación es comúnmente percibida por todas las partes relacionadas.
- c. Claridad de las expectativas: Grado de expectativas claramente definidas.
- d. Multiplicidad: Grado en que los individuos se vinculan por relaciones múltiples.

C. Dimensiones

- a. Tamaño: Número de personas en la red.
- b. Densidad o conectividad: Número de nexos reales en la red como proporción de los nexos totales posibles.
- c. Agrupamiento: número de regiones densas o de conglomerados en la red.
- d. Centralidad: Grado de jerarquía y restricción a la comunicación en la red.
- e. Estabilidad: Grado en que el patrón de la red cambia en el tiempo.
- f. Accesibilidad: Número promedio de nexos entre dos individuos cualesquiera en la red.
- g. Apertura: Número de nexos externos reales como proporción de los nexos externos totales posibles.
- h. Estrella: Individuo con el número más alto de nombramientos.
- i. Puente: Individuo miembro de múltiples enracimados en una red.
- j. Árbitro: Estrella que vincula también la red con redes externas.
- k. Aislado: Individuo con pocos (o nulos) nexos con otros en la red.

2.5.1. Técnicas para el análisis de redes sociales

Los elementos básicos del Análisis de Redes Sociales (ARS) son los provistos por la teoría de grafos para caracterizar redes: nodos y arcos. Los nodos en la red pueden ser personas, organizaciones, eventos o lugares. Los arcos representan las relaciones entre los nodos. Esos arcos pueden ser direccionales y mostrar la frecuencia o fortaleza de la relación (Scott, 2000; Wasserman & Faust, 1994). Las relaciones entre los nodos se pueden traducir en notación matricial para luego aplicar un conjunto de medidas derivadas de la teoría de grafos. [57][58][59]

Aunque las aproximaciones más deterministas normalmente enfatizan que el análisis de redes permite el estudio de cómo la estructura de relaciones sociales alrededor de una persona, grupo u organización afecta a su conducta y actitudes, las acciones intencionales estructuralmente limitadas de los individuos también pueden afectar la estructura social. Las redes sociales son a la vez la causa y el resultado de las conductas de los individuos. [60]

Luis Sanz Menéndez (2003) se refiere al ARS como:

Herramienta de medición y análisis de las estructuras sociales que emergen de las relaciones entre actores sociales diversos (individuos,

organizaciones, naciones, etc.). El ARS es un conjunto de técnicas de análisis para el estudio formal de las 6 relaciones entre actores y para analizar las estructuras sociales que surgen de la recurrencia de esas relaciones o de la ocurrencia de determinados eventos. [61]

Es importante dejar en claro que el análisis de redes sociales generalmente estudia la conducta de los individuos a nivel micro, los patrones de relaciones (la estructura de la red) a nivel macro, y las interacciones entre los dos niveles (Pfeffer, 1992) [62] [63]. De esta manera se tiene que los tipos de datos que se toman en el ARS son dos: atributos y relaciones. Cuando se está mirando la realidad desde el ARS se está buscando dar cuenta en forma sistemática del establecimiento de relaciones entre sujetos y los intercambios a través de dichas relaciones entre estos, y qué características tiene cada uno de esos sujetos u organizaciones, para saber cómo están influyendo o siendo influenciados los comportamientos de los mismos. [63]

2.5.1.1. Muestreo de relaciones

Dado un conjunto de actores o nodos, hay algunas estrategias para decidir cómo actuar en la recolección de medidas de las relaciones existentes. Al final de la gama de enfoques, se encuentran los métodos de “redes completas”. Este enfoque aporta el máximo de información, pero también es costoso y difícil de utilizar y puede ser de difícil generalización. En el otro extremo se encuentran métodos muy parecidos a los utilizados en la investigación empírica convencional. Estos enfoques aportan considerablemente menos información sobre la estructura de la red, pero a menudo son menos costosos y también facilitan la generalización desde las observaciones en la muestra hacia el total de la población. Sin embargo, no existe un método “correcto” para todos los problemas y preguntas de investigación.

2.5.1.1.1. Métodos de redes completas

Requieren que se recoja información acerca de los lazos de cada actor con los demás. En esencia, este enfoque utiliza un censo de los lazos en una población de actores (más que un muestreo de éstos). Por ejemplo, preguntando a cada niño en un grupo de juegos que identifique a sus amigos entre los demás. [64]

Una vez recolectada la información sobre lazos entre pares o díadas, los datos de redes completas aportan una fotografía completa de las relaciones en la población. [64]

Los datos de redes completas nos conducen a descripciones muy potentes y a análisis de estructuras sociales. Desafortunadamente, pueden también ser muy costosos y difíciles de obtener. Recolectar datos de cada miembro de una

población y tener cada rango o índice de cada uno de los demás, pueden ser tareas desafiantes. [64]

2.5.1.1.2. Métodos bola de nieve

Inicia focalizando un actor o conjunto de actores, a cada uno de los cuales se les pregunta por algunos de sus lazos con otros actores. Entonces, se toman todos los actores mencionados (que no sean parte de la lista original) y se les pregunta de nuevo por algunos de sus lazos. El proceso continúa hasta que no se identifiquen nuevos actores o hasta que se decida detenerlo (a menudo por motivos de tiempo y recursos). [64]

Hay dos grandes restricciones y debilidades potenciales de los métodos de bola de nieve. En primer lugar, los actores que no están conectados (es decir, “aislados”) no se pueden describir a través de este método. Su presencia y cantidad, puede ser una característica muy importante de las poblaciones para algunos propósitos analíticos. [64]

2.5.1.1.3. Método ego-céntrico

En muchos casos no será posible (o necesario) trazar la totalidad de las redes comenzando con nodos focales (como en el método de bola de nieve). Un enfoque alternativo es comenzar con una selección de nodos focales (egos), e identificar los nodos con los que ellos están conectados. Entonces determinamos cuáles de los nodos identificados en la primera fase están conectados con los demás. [64]

Este tipo de enfoque puede resultar muy efectivo para obtener un formulario de datos relacionales de poblaciones grandes y puede combinarse con enfoques basados en los atributos. Por ejemplo, se puede tomar una muestra simple aleatoria de estudiantes universitarios, y pedirles que identifiquen a sus amigos más cercanos y cuáles de éstos conoce a otros. Este tipo de enfoque puede dar una imagen buena y fiable de los tipos de redes (o al menos de los vecindarios locales) en los cuales se encuentran insertos los individuos. Se pueden obtener resultados tales como cuántos nodos de conexiones tienen y hasta qué puntos esos nodos forman núcleos fuertes. [64]

2.5.3. Usuarios

Las plataformas online se encuentran expuestas a distintos tipos de usuarios, generalmente su objetivo son personas, consumidoras o generadoras de contenido. Sin embargo, hablando específicamente de los tipos de usuarios posibles dentro de una red social y tomando como algunas de sus características

definitorias, su comportamiento en la red social y el contenido que genera, se pueden identificar al menos otras 2 categorías posibles:

- **Bots:** La generación de contenido se encuentra automatizado de acuerdo a algunos parámetros. Existen Bots “benévolos” que generan noticias o alertan de eventos y Bots “malignos” que llenan la red de Spam o contenido malicioso.
- **Cíborgs:** La generación de contenido es compartida entre humanos y bots. Puede ser humano asistida por bots o bots asistida por humanos.

Capítulo 3

Método de Desarrollo

3.1 Metodología fundacional para la ciencia de datos

Al igual que los científicos tradicionales, los científicos de datos requieren de una metodología fundamental, que pueda servir como una guía estratégica para la resolución de problemas. Esta metodología, la cual es independiente de herramientas o tecnologías particulares, provee un marco de trabajo para desarrollar los procesos requeridos para obtener respuestas y resultados. Basado en lo anteriormente descrito, IBM ha propuesto la Metodología Fundamental para la Ciencia de Datos (Foundational methodology for data science), desarrollada por el científico de datos John Rollins. [65]

Esta metodología, tal como la describe su autor, se compone por diez etapas que representan un proceso iterativo que va desde la concepción de la solución hasta la implementación de la misma, incluyendo el respectivo feedback y refinamiento.

Fue esta misma metodología la empleada para el desarrollo de la solución, cumpliendo con los pasos especificados para la ejecución correcta del proyecto.

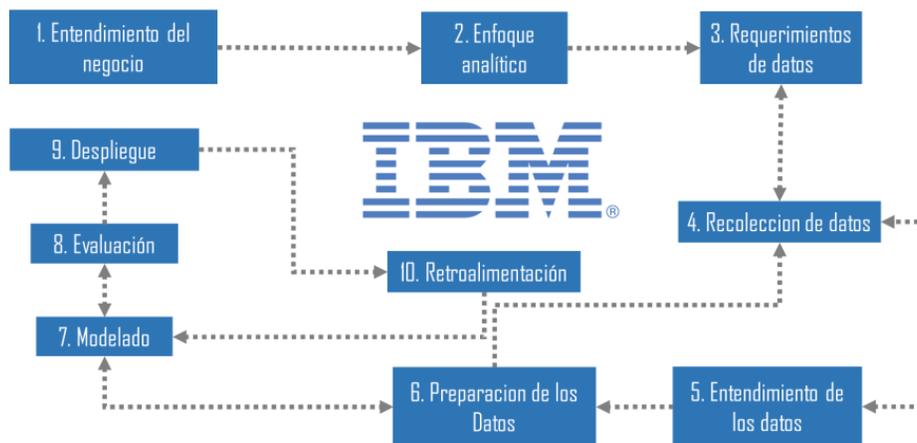


Figura 7. Etapas de la Foundational Methodology for Data Science

Fuente: Propia

1. Entendimiento del negocio (Business understanding): Cada proyecto, sin importar su tamaño, comienza con el entendimiento del negocio, el cual representa el fundamento para el éxito de la solución al problema. Los

representantes del negocio necesitan que la solución analítica juegue el rol crítico en esta etapa mediante la definición del problema, los objetivos del proyecto y los requerimientos de la solución.

2. **Enfoque analítico** (Analytic approach): Luego de definir claramente el problema, el científico de datos puede establecer el enfoque analítico para resolverlo. Realizar esta tarea involucra expresar el problema en el contexto de técnicas de Machine Learning para que el científico de datos logre identificar cuáles son las técnicas que mejor se adaptan para conseguir los resultados deseados.

3. **Requerimientos de datos** (Data requirements): La elección del enfoque analítico determina los requerimientos de datos. Los métodos analíticos a usar requieren contenido, formato y representaciones particulares de los datos.

4. **Recolección de datos** (Data collection): El científico de datos identifica y agrupa las fuentes de datos (estructurados, no estructurados, semi-estructurado) que son relevantes para el dominio del problema. En caso de encontrar brechas en la recolección de datos, podría ser necesario revisar los requerimientos de datos y recolectar más datos.

5. **Entendimiento de los datos** (Data understanding): Técnicas de visualización y estadísticas descriptivas pueden ayudar al científico de datos a entender el contenido de los datos, evaluar su calidad y descubrir características de interés en los datos.

6. **Preparación de los datos** (Data preparation): Esta etapa se compone de todas las actividades involucradas en la construcción de los datos que serán usados en la etapa de modelación. Estas incluyen limpieza de datos, combinación de múltiples fuentes y transformación de datos en variables más prácticas.

7. **Modelado** (Modeling): Iniciando de la primera versión del conjunto de datos preparado, el científico de datos usa conjuntos de entrenamiento (data histórica en la cual los resultados son conocidos), para desarrollar modelos predictivos o descriptivos utilizando el enfoque analítico anteriormente descrito. Esta etapa es altamente iterativa.

8. **Evaluación** (Evaluation): El científico de datos evalúa la calidad del modelo, y verifica si dirige el problema de forma completa y apropiada. Esto requiere de distintos diagnósticos y medidas computadas utilizando el conjunto de entrenamiento y el modelo predictivo.

9. **Despliegue** (Deployment): Luego del desarrollo de un modelo satisfactorio, aprobado por los representantes del negocio, el mismo es desplegado dentro del ambiente de producción u otro ambiente de prueba equiparable, con el fin de permitir la evaluación de su rendimiento.

10. **Retroalimentación** (Feedback): Mediante la recolección de los resultados generados por la implementación del modelo, la organización obtienen una retroalimentación sobre el rendimiento del modelo y observa como el mismo afecta el ambiente de producción. Analizar dicho feedback permite a los científicos de datos refinar el modelo, incrementar su exactitud y con esto su utilidad.

Capítulo 4

Desarrollo de la Solución

A continuación se describen cada una de las acciones ejecutadas en los pasos definidos por la metodología descrita en el Capítulo 3.

4.1 Entendimiento del negocio

Como se describió anteriormente en el primer capítulo del presente trabajo, Twitter es una red de microblogging destinada a la publicación de mensajes cortos no mayores a 140 caracteres de longitud. Twitter está integrado por una comunidad compuesta por más de 320 millones de usuarios activos al mes, cuyos gustos, géneros, y patrones de comportamiento son altamente variados.

Sin embargo, a pesar de la alta diversificación de los patrones de comportamiento de los usuarios, algunos estudios demostraron que se pueden identificar tres tipos principales de usuarios: humanos, bots y ciborgs. [7]

Cada uno de estos usuarios interactúa con la red de forma distinta, por lo que resulta interesante estudiar las diferencias entre estos tipos de comportamiento, particularmente aquellos que pueden considerarse dañinos o “maliciosos” para los integrantes de esta red social.

4.2 Enfoque analítico

Basado en el objetivo definido anteriormente, buscando categorizar automáticamente cuentas de usuarios de Twitter, se apuesta a la construcción, desarrollo e implementación de un modelo de clasificación que constará de 2 componentes principales:

4.2.1 Componente basado en el perfil de usuario

En el cual se estudiaron los siguientes puntos referentes al comportamiento del usuario dentro de Twitter:

- Año de registro de la cuenta en Twitter
- Si posee alguna descripción en el perfil de usuario
- Si el usuario posee la función de geolocalización activa
- Si la foto de perfil es la asignada por defecto

- Si posee alguna imagen de fondo en el perfil de usuario
- Si el perfil se encuentra verificado por Twitter
- Proporción de seguidores (N° seguidores/ N° seguidos)
- Número de tweets marcados como favoritos
- Número de listas registradas en el perfil
- Número de tweets desde el registro de la cuenta en Twitter
- Reputación de la cuenta (N° seguidores/[N° seguidores + N° seguidos])

4.2.2 Componente basado en el contenido publicado

Dentro de este componente se realizaron análisis sobre el contenido publicado por el usuario en Twitter.

- Proporción de URLs (N° URLs/ N° tweets)
- Promedio de diversidad de palabras tweets (N° palabras distintas/ N° palabras totales)
- Promedio de la cantidad de palabras utilizadas en los tweet (N° palabras totales/ N° tweets)
- Proporción de menciones (N° menciones/ N° de tweets)
- Promedio de etiquetas utilizadas por tweet (N° etiquetas/ N° tweets)
- Proporción de respuestas (N° respuestas/ N° de tweets).
- Promedio de la longitud de los tweets (N° caracteres totales/ N° tweets).
- Promedio de la diversidad lexicográfica en los tweets (N° de letras distintas/ N° letras totales).
- Entropía condicional corregida del tiempo entre los tweets. [67]
- Fuente desde la que fue publicado el tweet (web, móvil, terceros)
- Distribución de los tweets en la semana.
- Distribución de los tweets en el día.

Basado en el resultado de los componentes antes descritos, el clasificador deberá ser capaz de categorizar al usuario según el contenido publicado y su comportamiento dentro de la red social.

4.3 Requerimientos de los datos

Para la construcción del modelo de clasificación presentado, fue necesario disponer de información sobre cada perfil de usuario perteneciente a la muestra de datos, así como también se requirió el contenido de los tweets publicados junto a la metadata correspondiente.

Los archivos de datos se almacenaron en formato JSON, el cual se presentó como una alternativa flexible y versátil con respecto a los posteriores análisis realizados con las herramientas seleccionadas.

4.4 Recolección de datos

Se utilizó un método de recolección y muestreo de datos ego-céntrico [64] aplicado en dos fases:

4.4.1 Construcción del árbol de cuentas

1. Se eligieron ocho nodos focales de gran interés para la población venezolana, apuntando a que la mayoría de sus seguidores, y los seguidores de estos, estuvieran relacionados con Venezuela. Las cuentas utilizadas fueron las siguientes:
 - a. @metro_caracas (Metro de Caracas)
 - b. @UNoticias (Periódico Ultimas Noticias)
 - c. @noticierovv (Noticiero Venevisión)
 - d. @trafficMIRANDA (T tráfico en el Edo. Miranda)
 - e. @BcodeVenezuela (Banco de Venezuela)
 - f. @ifetren (Ferrocarriles Venezuela)
 - g. @SomosMovilnet (Operadora Movilnet)
 - h. @MeridianoTV (Canal de TV Meridiano)
2. Se utilizó un algoritmo de Búsqueda por Anchura (Breadth-first search, o BFS por sus siglas en inglés) [68] para construir el árbol de cuentas.
3. Se escogieron como máximo los primeros 80 seguidores devueltos por el API de Twitter por cada cuenta obtenida para limitar la amplitud del árbol.
4. Se extrajo el ID de cada cuenta y se enlistó el ID de sus seguidores para la posterior extracción de sus timelines.

5. Se detuvo el proceso de recolección al alcanzar alrededor de 700.000 mil cuentas de usuario.

4.4.2 Obtención de timelines de usuarios

1. Se recorrió el árbol de cuentas construido previamente utilizando un algoritmo BFS.
2. Se extrajeron, como máximo, todos los datos de los últimos 800 tweets del timeline de cada usuario mediante el API de Twitter.

El proceso de extracción de datos estuvo siempre limitado por las restricciones establecidas en los términos de uso del API de Twitter. Estas limitaciones se encuentran claramente expresadas la página oficial para desarrolladores de Twitter [69]. Agregando a estas limitaciones los retrasos generados por fallas eléctricas y problemas con el ISP, ocasionando que la recolección de los datos, la cual se distribuyó en 2 máquinas que se encontraban en constante extracción de contenido, tomara 5 meses para culminar.

4.5. Entendimiento de los datos

Se desarrollaron distintos programas y herramientas para visualizar de forma útil los datos de extraídos, los cuales fueron obtenidos completamente en formato JSON. A continuación se presentan algunas observaciones interesantes sobre la data procesada:

4.5.1. Volumen de tweets

La Figura 8 presenta la Función de Distribución Acumulada (FDA) del número de tweets para cada usuario perteneciente a las tres categorías de humanos, ciborgs y bots. Se puede notar claramente que los usuarios ciborg realizan en su mayoría más publicaciones que los usuarios bot y los usuarios humanos. Este volumen considerable de publicaciones se atribuye a los propósitos comerciales que suelen tener este tipo de cuentas. Aunque cierta parte de la gestión de estas cuentas es realizada por empleados, la mayoría de las publicaciones son realizadas por herramientas automatizadas. Por otro lado, las cuentas de usuarios humanos y bots tienen un comportamiento similar en cuanto al conteo de sus publicaciones. Sin embargo, el volumen de tweets de ambos grupos no necesariamente pudo ser generado en el mismo instante de tiempo. Los períodos de activación de las cuentas bots nivelan el volumen de publicaciones con respecto al generado por el comportamiento constante de los usuarios humanos.

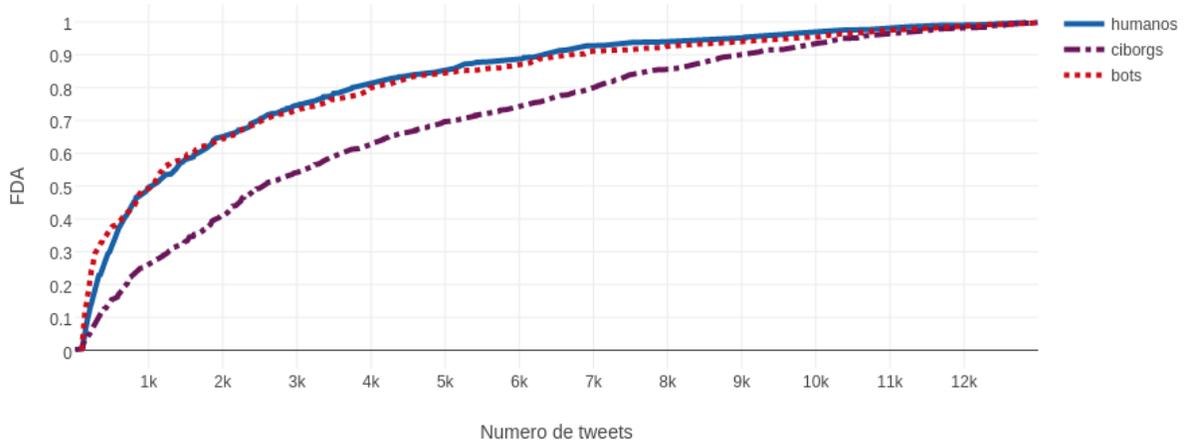


Figura 8. FDA del número de tweets

Fuente: Propia

4.5.2. Longitud de Tweets

Una característica insigne de Twitter es el límite de caracteres permitidos para cada tweet. Actualmente, este límite establecido en 140 caracteres no contempla los caracteres relacionados al contenido multimedia o a los nombres de usuarios (respuestas y menciones). En la Figura 9 se calcula la FDA de la longitud promedio de los tweets para cada usuario de las distintas categorías. La mayoría de los usuarios humanos solamente utilizan la cantidad de caracteres necesaria para expresar sus ideas u opiniones. En contraparte a este resultado, los usuarios ciborgs aprovechan al máximo la cantidad de caracteres límite con el fin de incluir toda la información posible en sus publicaciones, en su mayoría con fines publicitarios. Los bots obtienen un resultado intermedio en esta medida ya que sus publicaciones dependen fundamentalmente del tipo de contenido que se dedique a publicar cada cuenta.

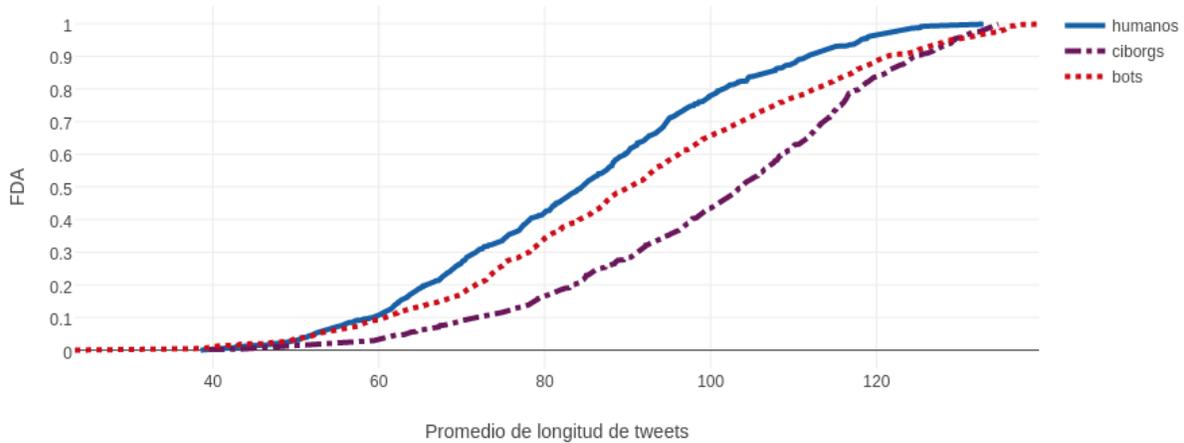


Figura 9. FDA del promedio de longitud de tweets

Fuente: Propia

4.5.3. Uso de enlaces

Se evaluó la frecuencia con la que se encontraron enlaces externos en el contenido de los tweets publicados por los distintos tipos de usuarios. Como se puede observar en la Figura 10, los bots tienden a incluir enlaces en sus publicaciones con mayor frecuencia respecto a los otros usuarios. Este comportamiento tiene como propósito redirigir a los usuarios a las páginas de interés para el administrador de la cuenta. En muchos casos, los bots suelen incluir más de un enlace en cada tweet. Los ciborgs siguen de cerca a los bots respecto a la proporción de enlaces publicados en sus tweets. Un gran número de ciborgs suele integrar su timeline con fuentes RSS o actualizaciones de blogs, generando tweets con títulos de artículos seguidos por enlaces a la página web que ofrece el resto de la información. Los humanos poseen la menor cantidad de enlaces externos por tweet publicado, debido a que generalmente sus publicaciones describen lo que está haciendo, pensando o lo que sucede a su alrededor, lo cual es descrito en su mayoría con solo texto, sin ningún tipo de enlace a otros sitios web.

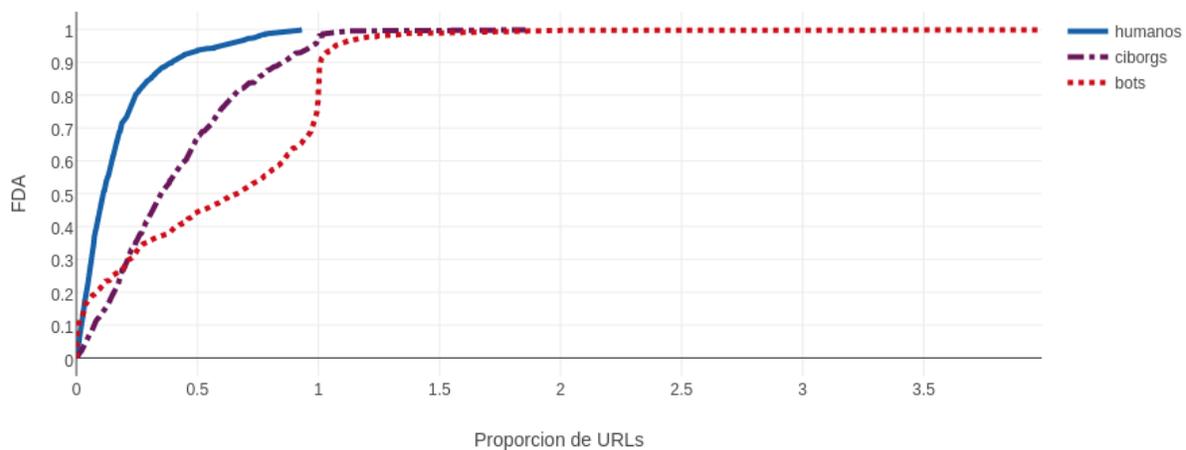


Figura 10. FDA de la proporción de enlaces a fuentes externas

Fuente: Propia

4.5.4. Contenido spam

En la Figura 11 se evalúan los resultados obtenidos durante la detección de contenido spam, los cuales demuestran una clara diferencia de comportamiento entre las tres categorías de usuarios. Se puede destacar a los usuarios bots como los mayores generadores de contenido spam. Este resultado está relacionado al propósito que suele tener este tipo de cuentas con respecto a la generación de contenido no deseado y publicidad engañosa. Luego se encuentran los ciborgs que, debido a su naturaleza híbrida, poseen una proporción intermedia entre los humanos y los bots.

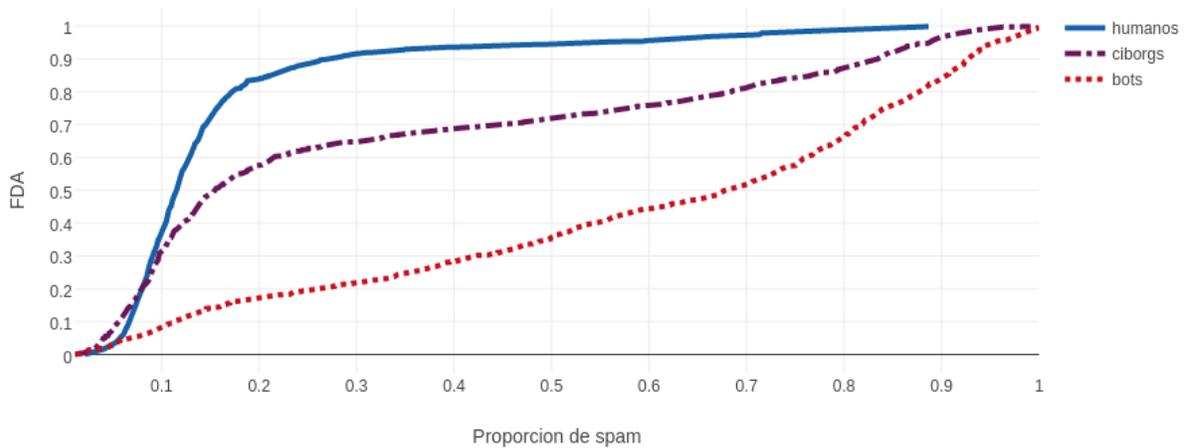


Figura 11. FDA de la proporción de contenido spam

Fuente: Propia

4.5.5. Dispositivos de publicación

Se realizó un proceso de agrupación sobre las diversas fuentes disponibles para la publicación de tweets, de donde resultaron tres categorías. El uso del sitio web oficial de Twitter es la única fuente categorizada como “Uso web”. Cualquier publicación proveniente del cliente oficial de twitter para dispositivos móviles (Blackberry, Windows Phone, iOS, Android, entre otros) se catalogaron como “Uso móvil”. Por último, cualquier publicación proveniente de una fuente no incluida en las dos categorías anteriores (SmarTV, TweetDeck, RSS, entre otros) se catalogaron como “Uso de Terceros”, refiriéndose al uso del API de Twitter desde terceros para la publicación de los tweets.

Tal como se puede apreciar en la Figura 12, los humanos prefieren el uso de los dispositivos móviles para la publicación de sus mensajes, relegando al sitio web de Twitter como la segunda opción y el uso desde terceros como la alternativa menos utilizada. Los bots, en total contraste respecto a los humanos, tienen como medio predilecto de publicación a las fuentes terceras, debido a las capacidades de automatización que muchas de ellas ofrecen. Respecto a los ciborgs se puede observar un comportamiento parecido al de los humanos, destacando un mayor uso en las fuentes de terceros, compartiendo una ligera similitud con los bots, demostrando su naturaleza heterogénea.

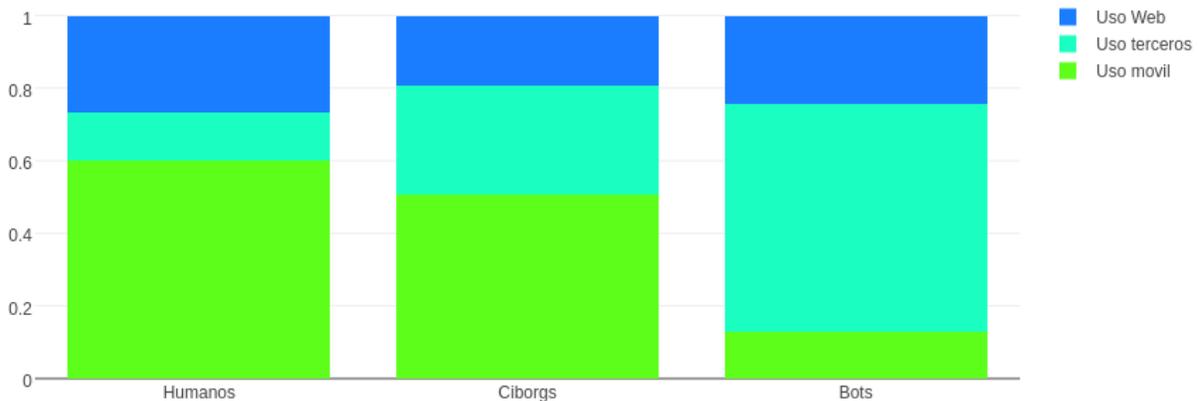


Figura 12. Proporción de uso de medios de publicación

Fuente: Propia

4.5.6. Menciones a otros usuarios

Un usuario es capaz de mencionar a otro usuario específico dentro de una publicación con el propósito de compartir contenido de una forma más directa. En la Figura 13 se señala el comportamiento de las tres categorías de usuario con respecto a esta funcionalidad. Los usuarios humanos y ciborgs presentan gran similitud pero se diferencian drásticamente de los usuarios bots, esto puede explicarse de la siguiente forma: 1) los humanos tienden a interactuar con mayor frecuencia con otros individuos dentro de la plataforma que los usuarios de las otras categorías, 2) las cuentas ciborgs suelen pertenecer a compañías, marcas registradas o proveedores de servicio ya que requieren un componente humano para atender y dar respuesta las exigencias de sus seguidores, y 3) la interacción constante y fluida con otros usuarios es difícil de automatizar por lo que muchas cuentas bots carecen de esta capacidad.

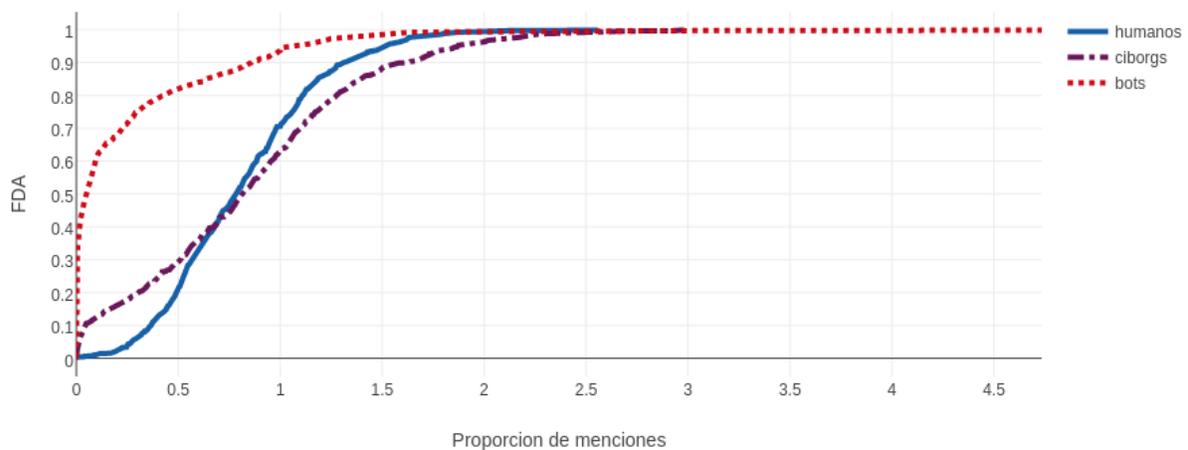


Figura 13. FDA de la proporción de menciones

Fuente: Propia

4.5.7. Respuestas a publicaciones

Una publicación es considerada como una “respuesta” si comienza con el nombre de usuario (@username, por ejemplo) de la persona a quien va dirigida dicha respuesta. En la Figura 14 se puede notar una separación considerable entre las tres categorías de usuarios en relación a esta funcionalidad de Twitter. Los usuarios humanos destacan con la mayor proporción de respuestas entre sus publicaciones, debiéndose esto a que la mayoría de la interacción de este tipo en Twitter es realizada precisamente por usuarios humanos. Solamente una cantidad minúscula y excepcional de bots realizan respuestas a otros usuarios. Por otra parte, los usuarios ciborg se encuentran entre los humanos y los bots, el cual es el comportamiento esperado para este tipo de cuentas mixtas.

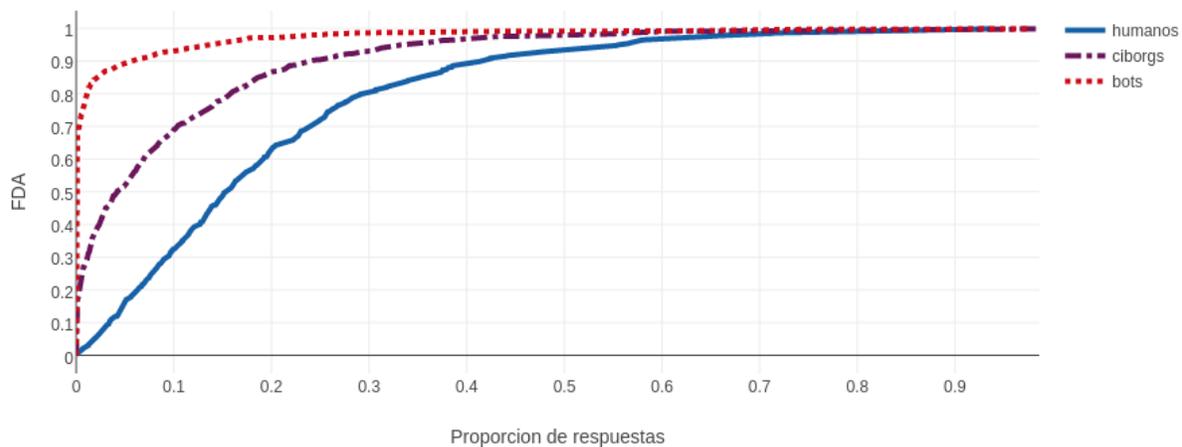


Figura 14. FDA de la proporción de respuestas

Fuente: Propia

4.5.8. Relación entre seguidores y amigos

Dentro de la plataforma de twitter, cada usuario tiene la capacidad de seguir distintas cuentas (amigos) y de ser seguido por otros usuarios (seguidores). Para medir la relación entre la cantidad de seguidores y la cantidad de amigos de un usuario calculamos el valor de reputación de una cuenta, el cual definimos como:

$$Reputación\ de\ cuenta = \frac{\#seguidores}{\#seguidores + \#amigos}$$

Los valores de reputación más altos (cerca de uno) corresponden a los usuarios seguidos por muchas cuentas pero que siguen a pocos usuarios, como también a los usuarios con muy pocos seguidores pero con una cantidad considerable de amigos. Este último comportamiento se presenta claramente para la categoría de usuarios humanos en la Figura 15, donde se muestra la FDA de la reputación de los usuarios de cada grupo. Por otro lado, la categoría ciborg presenta el comportamiento opuesto, donde a partir del percentil 30 de los

usuarios que conforman este grupo se observa una reputación mayor o igual a 0.5.

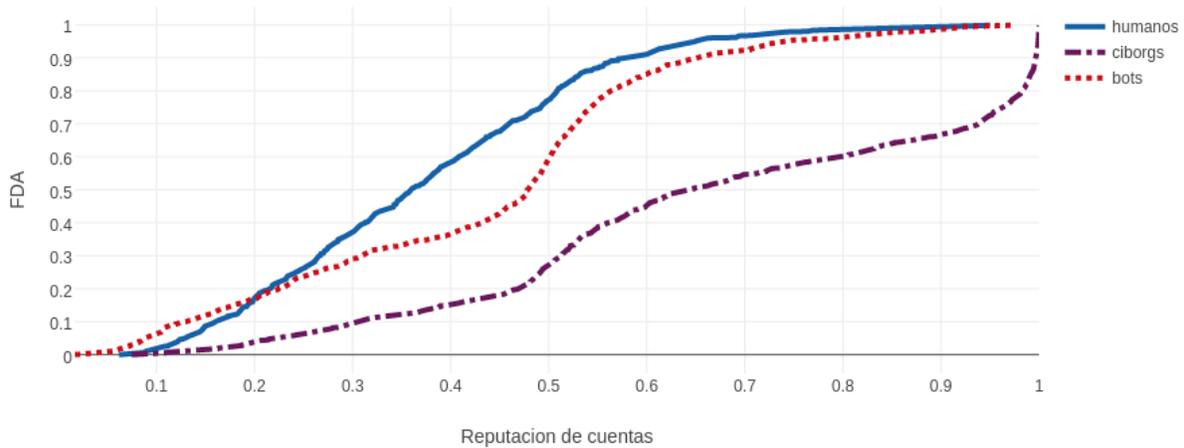


Figura 15. FDA de la reputación de las cuentas

Fuente: Propia

4.5.9. Entropía de los grupos

El valor entrópico calculado permite medir y representar el nivel de irregularidad que tiene un usuario con respecto a la frecuencia de sus publicaciones. Los usuarios humanos, en su gran mayoría, tienden a comportarse de forma irregular en contraste con las publicaciones programadas periódicamente de los usuarios ciborg y bot. Se puede observar en la Figura 16, como el grupo de usuarios humanos se diferencia en su mayoría de los usuarios ciborgs y bots, los cuales tienden a solaparse.

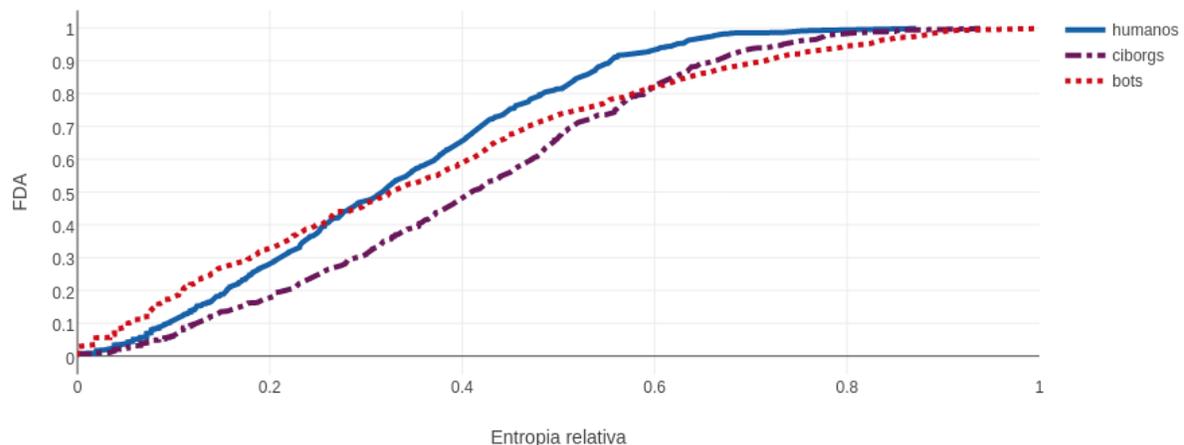


Figura 16. FDA de la entropía relativa

Fuente: Propia

4.5.10. Horarios de alto tráfico

Dentro del estudio se determinó que la hora y el día en el que un tweet es publicado permiten diferenciar de una forma interesante a los usuarios de cada grupo. En la Figura 17, Figura 18 y Figura 19 se puede visualizar cuales son las horas de cada día (en GMT-0), de los siete días de la semana, en los que cada grupo presenta mayor actividad de publicación. Se puede resaltar de forma impresionante que los usuarios humanos tienen fuertes picos de actividad entre las 00:00 horas y las 04:00 horas, en especial el día domingo. Por otra parte, los usuarios ciborg tienen un amplio periodo de actividad entre aproximadamente las 14:00 horas y las 02:00 horas en todos los días de la semana. Los usuarios bots poseen un comportamiento similar a los ciborg, concentrando su actividad en los días miércoles, jueves y viernes; pero con un decrecimiento considerable en la cantidad de sus publicaciones los días lunes y martes. Los tres grupos de usuarios generan la menor cantidad de volumen por día en el periodo comprendido por las 06:00 horas y las 10:00 horas.

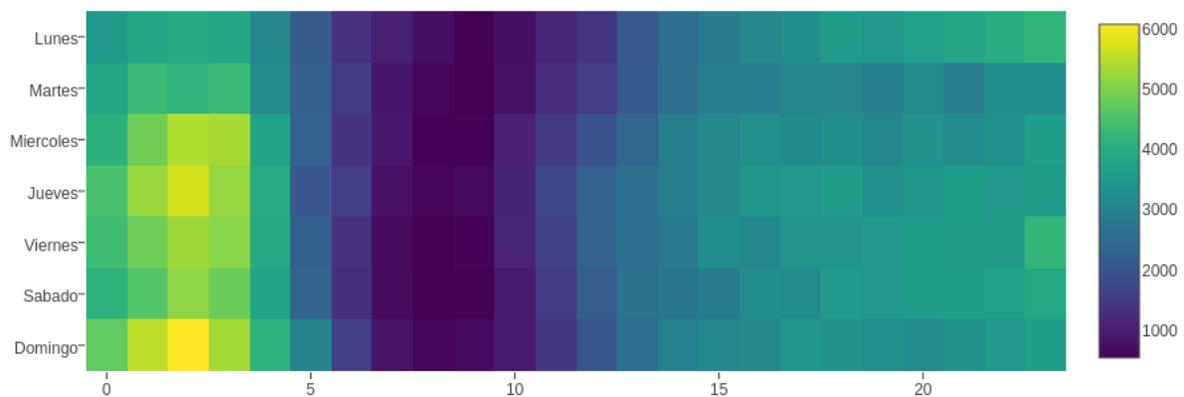


Figura 17. Publicaciones de humanos por hora en la semana (GMT-0)

Fuente: Propia

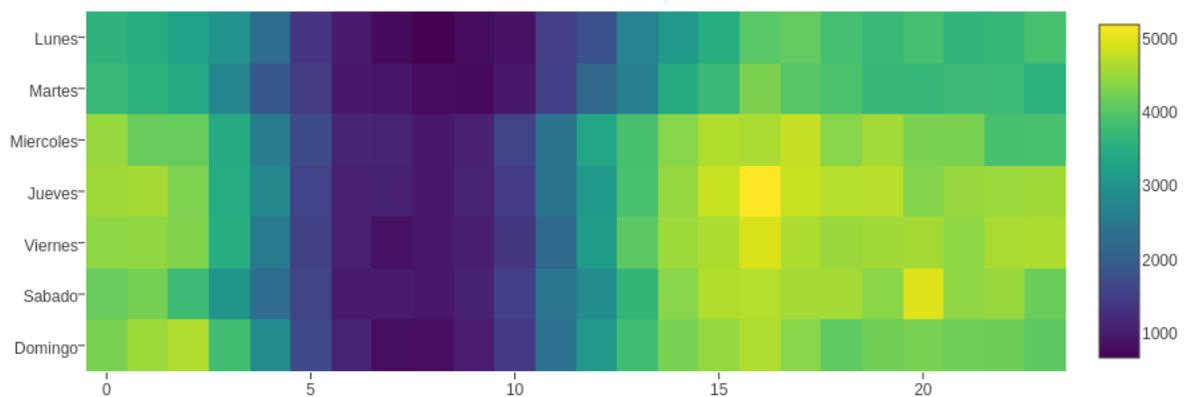


Figura 18. Publicaciones de ciborgs por hora en la semana (GMT-0)

Fuente: Propia

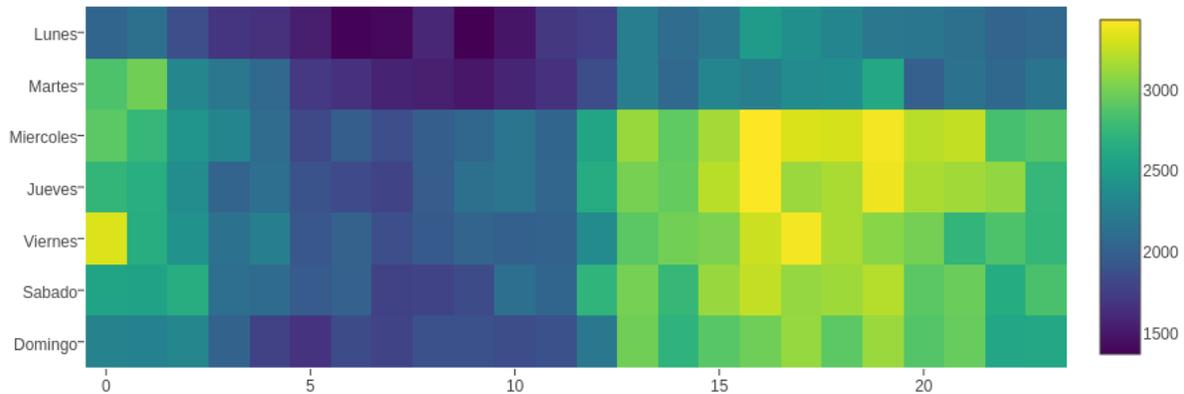


Figura 19. Publicaciones de bots por hora en la semana (GMT-0)

Fuente: Propia

4.6. Preparación de los datos

El proceso de preparación de los datos recolectado estuvo constituido por las siguientes etapas:

4.6.1. Calculo de la muestra de datos

De acuerdo al Ministerio del Poder Popular para la Comunicación e Información de Venezuela (MINCI) [70], al cierre del año 2015 Venezuela contaba con al menos 16 millones de personas con acceso a internet. Es natural pensar que no todas y cada una de las personas con acceso a internet en Venezuela poseen cuentas en la plataforma de Twitter. Para ajustar aún más la cifra, según el estudio realizado por Statista (Digital Market Outlook) [71] referente al número de usuarios activos en Twitter para mayo del 2016, reflejado en la Figura 20, el último país entre las primeras once naciones con más usuarios en la plataforma de Twitter es España, con aproximadamente 7.52 millones de usuarios. Razón por la cual es sensato considerar que en Venezuela existe una cantidad inferior a 7.52 millones de usuarios. Sin embargo, debido a la carencia de información certera sobre esta cifra, se decidió realizar los cálculos y evaluaciones asumiendo que Venezuela cuenta con, como máximo, 10 millones de usuarios activos en Twitter, la cual es una cantidad incluso superior a la presentada por México, el cual ocupa el 8vo lugar entre los países con más usuarios dentro de la plataforma.

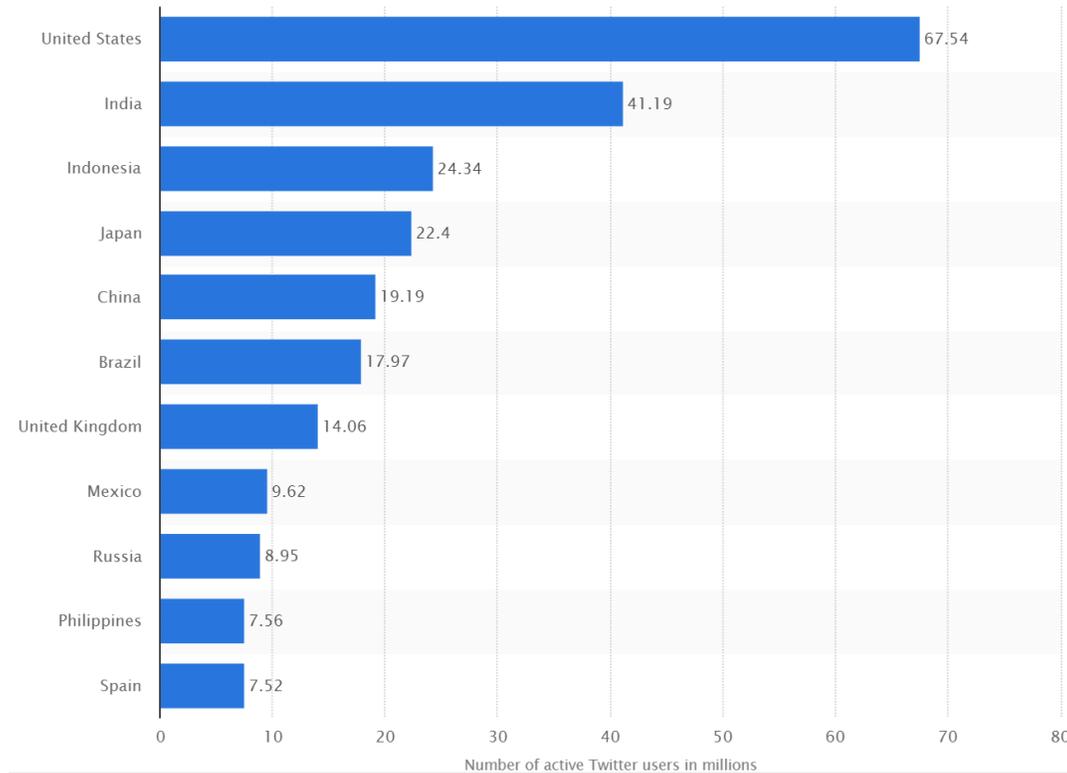


Figura 20. Número de usuarios activos de Twitter por Statista

Fuente: <https://www.statista.com/statistics/242606/number-of-active-twitter-users-in-selected-countries/>

Para establecer una cantidad de usuarios que pueda definirse como representativa basada en el número de usuarios que se asumen activos en Twitter para Venezuela (N), se realizó el cálculo para el tamaño de la muestra [72] asumiendo la constante 0.5 como la desviación estándar (σ) de la población, un nivel de confianza (Z) de 99% que deriva en el valor 2.58 y un límite aceptable de error muestral (e) establecido en 5%.

$$\frac{N\sigma^2Z^2}{e^2 N - 1 + \sigma^2Z^2} \approx 665.595$$

Resultando en, aproximadamente, 666 como cantidad de usuarios suficiente para satisfacer los criterios definidos para el cálculo de la muestra. Sin embargo, para esta investigación se decidió escoger a 1.000 usuarios, provenientes de los 700.000 timelines recaudados, como muestra representativa para cada categoría de usuarios, reduciendo el límite del error muestral a 4% y obteniendo un total de 3.000 cuentas de Twitter para conformar el set de datos inicial.

Cada una de estos usuarios se clasificó manualmente por un componente humano entre los tres grupos previamente definidos. Para cada categoría de usuarios, el 80% (800 usuarios) se utilizó como set de datos de entrenamiento y el 20% restante (200 usuarios) fue utilizado para probar la eficacia del modelo.

4.6.2. Clasificación manual de usuarios

Para el proceso de clasificación manual de cada usuario se realizó el siguiente conjunto de actividades:

- Se visitó la página principal del usuario (<http://twitter.com/username>).
- Se revisaron las características asociadas al perfil del usuario, tales como la cantidad de tweets publicados, número de seguidores, número de amigos, número de publicaciones favoritas, fecha de creación de la cuenta (en caso de ser pública), imagen de perfil por defecto y respuestas a publicaciones.
- Se tomó en cuenta la coherencia del contenido publicado con respecto al perfil general del usuario, la frecuencia de respuestas a otras publicaciones sospechosamente automatizadas.
- Se inspeccionó el timeline del usuario para examinar características adicionales como los dispositivos de publicación.

Un usuario es clasificado como humano si se obtuvo evidencia de que el contenido publicado es inteligente, original, coherente y similar al contenido que podría publicar un humano. Por otra parte, un usuario es clasificado como bot si: el contenido publicado carece de originalidad, cantidad excesiva de publicaciones automáticas, la existencia de tweets duplicados, y si la cantidad de seguidores y amigos es exageradamente alta para un corto periodo de tiempo. Por último, un usuario es clasificado como ciborg si no puede clasificarse como humano pero tiene suficiente contenido original como para suponer que se trata de una cuenta asistida (refiérase tanto a una cuenta bot asistida por un humano o una cuenta de un humano con cierto grado de automatización).

4.6.3. Creación de conjuntos de Tweets spam y no spam

Se elaboraron dos conjuntos de datos de forma manual a partir de tweets que cumplían características específicas: uno correspondiente a aquellos tweets que fueron catalogados como spam, y otro correspondiente a los tweets clasificados como no spam. Se consideraron como spam aquellos tweets provenientes de cuentas bots, con enlaces externas maliciosas o con publicidad no deseada. Algunas cuentas bots “avanzadas” esconden los tweets spam entre tweets no spam; este tipo de tweets fueron ignorados. Se consideró como no spam a los tweets provenientes de usuarios humanos, sin enlaces externos o archivos multimedia. Como medida conservadora, el set de datos no spam no contiene tweets de bots o ciborgs.

4.6.4. Consideraciones

Los usuarios cuyos timelines que estaban protegidos al momento de la recolección de datos fueron excluidos del estudio, así como también aquellos usuarios cuyo conteo de publicaciones era inferior 100, ya que se consideraron como cuentas con poca actividad para realizar su respectiva categorización.

4.7. Modelado

Gracias a investigaciones relacionadas [73] y a los resultados de las pruebas realizadas sobre los set de datos de publicaciones spam y no spam, los cuales se pueden apreciar en la Tabla 4, se puede concluir que entre los algoritmos correspondientes al clasificador Bayesiano para clasificaciones [74] y el Random Forest para clasificaciones [75] (ambos definidos en el paquete ML de Spark), este último presenta la menor tasa de error, razón por la cual se optó por el Random Forest para clasificar los tweets de los usuarios entre spam y no spam.

| | Bayesiano | Random Forest |
|-----------|-----------|---------------|
| Exactitud | 0.764 | 0.811 |

Tabla 4. Comparación entre la exactitud del algoritmo Bayesiano y el Random Forest

Fuente: Propia

Posteriormente, para la implementación del juez de usuarios se optó igualmente por la utilización de un modelo Random Forest, principalmente debido a su eficacia en procesos de clasificación en casos que involucran más de dos categorías, además de ser capaz de manejar una gran cantidad de características o variables, pudiendo descartar aquellas que no proporcionen suficiente información para discriminar entre las categorías; y de haber demostrado buenos resultados en estudios relacionados [7]. Este juez utiliza la lista de características respectivas de cada usuario para realizar su predicción, juzgándolo como humano, bot o ciborg.

La Tabla 5 señala la relevancia medida en pesos de las características utilizadas por el API para la clasificación de los usuarios.

| CARACTERÍSTICA | PESO | CARACTERÍSTICA | PESO | CARACTERÍSTICA | PESO |
|--------------------------|--------|--------------------------|--------|-------------------------|--------|
| Promedio de spam | 0.0935 | Actividad a las 2:00 am | 0.0099 | Actividad el lunes | 0.0074 |
| Proporción de respuestas | 0.0904 | Actividad a las 4:00 pm | 0.0099 | Actividad a las 5:00 pm | 0.0074 |
| # de listas | 0.0831 | Actividad a la 1:00 am | 0.0095 | Actividad el jueves | 0.0072 |
| Reputación | 0.0777 | Diversidad lexicográfica | 0.0089 | Actividad a las 7:00 am | 0.0071 |
| Publicaciones por | 0.0522 | Actividad a las 12:00 pm | 0.0089 | Actividad a las 11:00 | 0.0071 |

| móvil | | | pm | | |
|---------------------------------|--------|--------------------------|--------|--|--------|
| Año de registro | 0.0494 | Actividad a las 8:00 pm | 0.0088 | Actividad a las 6:00 am | 0.007 |
| Proporción de menciones | 0.0481 | Actividad a las 2:00 pm | 0.0087 | Actividad el viernes | 0.0069 |
| Proporción de seguidores | 0.038 | Actividad a las 3:00 pm | 0.0087 | Actividad el martes | 0.0066 |
| Proporción de enlaces | 0.0348 | Actividad a las 6:00 pm | 0.0084 | Actividad a las 5:00 am | 0.0066 |
| # de favoritos | 0.0332 | Actividad a las 9:00 pm | 0.0084 | Actividad a las 11:00 am | 0.0066 |
| Promedio de hashtags | 0.0278 | Actividad el domingo | 0.0083 | Actividad a las 4:00am | 0.0064 |
| Publicaciones por terceros | 0.0261 | Publicaciones por web | 0.0081 | Actividad el sábado | 0.0062 |
| Longitud de tweets promedio | 0.018 | Actividad a las 10:00 am | 0.0079 | Geolocalización activa | 0.0055 |
| # de tweets | 0.0177 | Actividad a la 1:00 pm | 0.0079 | Perfil verificado | 0.0021 |
| Actividad a las 9:00 am | 0.0159 | Actividad el miércoles | 0.0078 | Perfil con imagen por defecto | 0.001 |
| Diversidad de palabras promedio | 0.0127 | Actividad a las 12:00 am | 0.0078 | Perfil con imagen de fondo por defecto | 0.001 |
| Perfil con descripción | 0.012 | Actividad a las 8:00 am | 0.0078 | Actividad a las 7:00 pm | 0.0007 |
| Actividad a las 3:00 am | 0.0107 | Entropía | 0.0077 | | |
| Promedio de palabras | 0.0102 | Actividad a las 10:00 pm | 0.0076 | | |

Tabla 5. Tabla de pesos de las características de los usuarios

Fuente: Propia

La relevancia de cada característica indica que tan importante es la misma para el modelo, lo que quiere decir que, a mayor relevancia, más diferenciable es un usuario al ser evaluado por dicha característica. El cálculo de los pesos fue realizado en base al índice Gini [76] para el cual, en cuanto mayor sea la medida, más variabilidad aporta la característica independiente que está siendo evaluada. El cálculo consiste en los siguientes dos pasos:

- Importancia de la característica j = suma (sobre los nodos en los cuales se encuentra repartida la característica j) de la ganancia de información, donde la misma es escalada por el número de instancias pasadas a través del nodo.
- Normalización de las importancias del árbol para igualar su suma a uno (1).

Se puede observar que ninguna característica es determinante de forma absoluta, sino que es la suma de todas ellas permite realizar una buena categorización.

4.8. Evaluación

Para obtener el mejor modelo se genera una tabla de parámetros que Spark podrá utilizar para crear el modelo del Random Forest. Spark se encarga de realizar una validación cruzada usando el set de datos de entrenamiento, el cual está constituido por el 80% de los datos del set inicial, utilizando las distintas configuraciones disponibles entre los parámetros de la tabla y retornando aquel modelo cuya configuración de parámetros haya proporcionado la mayor exactitud durante la evaluación.

El set de datos de prueba, representado por el 20% de datos restante, es utilizado para calcular la matriz de confusión y la exactitud del modelo. Se tomó la medida de calcular la exactitud del modelo y la matriz de confusión con un set de datos completamente independiente para garantizar robustez y confianza en los resultados, aun cuando el modelo fue realizado con una validación cruzada.

En la Tabla 6 se presentan los resultados de ejecución del Juez sobre los usuarios de Twitter categorizados manualmente en los tres grupos definidos. En ella las filas denotadas como “Reales” son la categoría real a la cual pertenecen los usuarios. La columna “Clasificados” expresa el resultado del juez. Por ejemplo, en la intersección de la fila y la columna “Humanos” se señala que 180 usuarios que son humanos, fueron clasificados correctamente como humanos. En cambio, en la intersección de la fila “Bots” y la columna “Ciborgs” se muestra que 17 usuarios que son “Bots” fueron clasificados erróneamente como “Ciborgs”. En general, se obtuvo una exactitud de 86.5%.

| | | Clasificados | | | Total | Exactitud |
|--------|---------|--------------|------|---------|------------|-----------|
| | | Humanos | Bots | Ciborgs | | |
| Reales | Humanos | 180 | 8 | 12 | 200 | 0,9 |
| | Bots | 0 | 183 | 17 | 200 | 0,915 |
| | Ciborgs | 15 | 29 | 156 | 200 | 0,78 |
| | | | | | avg | 0,865 |

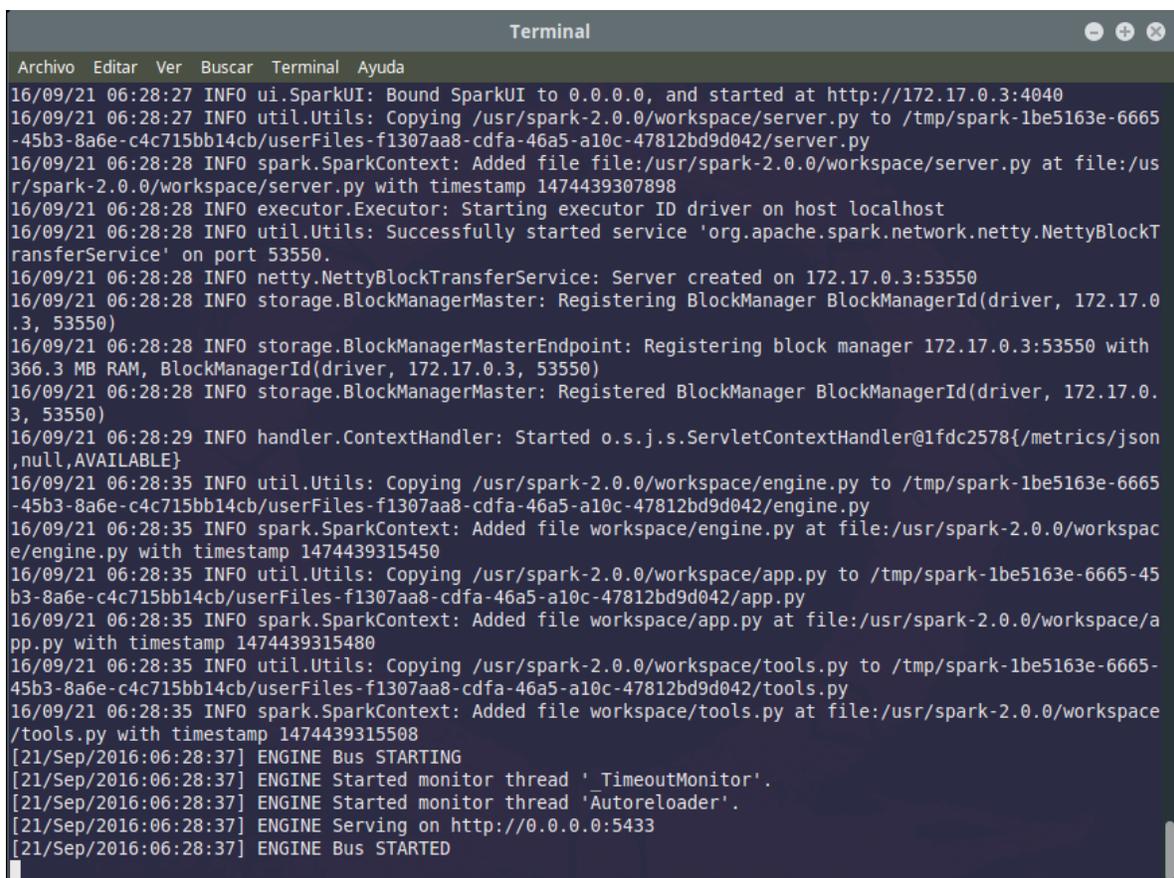
Tabla 6. Matriz de confusión sobre los resultados del Juez

Fuente: Propia

4.9. Despliegue

Una vez obtenido el modelo de clasificación con los parámetros que proporcionaron los mejores resultados para el set de datos utilizado, se procedió a integrar los procedimientos referentes al entrenamiento y clasificación en una API REST, ofreciendo flexibilidad en cuanto a la forma en la que sus funcionalidades puedan ser consumidas. La misma se desarrolló con el API de Python ofrecido por Spark 2.0.0, haciendo uso de la librería de Machine Learning definida para el uso con DataFrames. Este servicio web es desplegado en un contenedor de Docker que contiene a Spark 2.0.0 con las librerías de Hadoop 2.7, además de las dependencias de Python requeridas para el correcto funcionamiento del Servicio Web.

Gracias a las tecnologías utilizadas para la creación del API, el proceso de despliegue de la misma involucra solamente unos pocos pasos, los cuales se pueden encontrar entre los anexos de la presente investigación.



```
Terminal
Archivo Editar Ver Buscar Terminal Ayuda
16/09/21 06:28:27 INFO ui.SparkUI: Bound SparkUI to 0.0.0.0, and started at http://172.17.0.3:4040
16/09/21 06:28:27 INFO util.Utils: Copying /usr/spark-2.0.0/workspace/server.py to /tmp/spark-1be5163e-6665-45b3-8a6e-c4c715bb14cb/userFiles-f1307aa8-cdfa-46a5-a10c-47812bd9d042/server.py
16/09/21 06:28:28 INFO spark.SparkContext: Added file file:/usr/spark-2.0.0/workspace/server.py at file:/usr/spark-2.0.0/workspace/server.py with timestamp 1474439307898
16/09/21 06:28:28 INFO executor.Executor: Starting executor ID driver on host localhost
16/09/21 06:28:28 INFO util.Utils: Successfully started service 'org.apache.spark.network.netty.NettyBlockTransferService' on port 53550.
16/09/21 06:28:28 INFO netty.NettyBlockTransferService: Server created on 172.17.0.3:53550
16/09/21 06:28:28 INFO storage.BlockManagerMaster: Registering BlockManager BlockManagerId(driver, 172.17.0.3, 53550)
16/09/21 06:28:28 INFO storage.BlockManagerMasterEndpoint: Registering block manager 172.17.0.3:53550 with 366.3 MB RAM, BlockManagerId(driver, 172.17.0.3, 53550)
16/09/21 06:28:28 INFO storage.BlockManagerMaster: Registered BlockManager BlockManagerId(driver, 172.17.0.3, 53550)
16/09/21 06:28:29 INFO handler.ContextHandler: Started o.s.j.s.ServletContextHandler@1fdc2578{/metrics/json,null,AVAILABLE}
16/09/21 06:28:35 INFO util.Utils: Copying /usr/spark-2.0.0/workspace/engine.py to /tmp/spark-1be5163e-6665-45b3-8a6e-c4c715bb14cb/userFiles-f1307aa8-cdfa-46a5-a10c-47812bd9d042/engine.py
16/09/21 06:28:35 INFO spark.SparkContext: Added file workspace/engine.py at file:/usr/spark-2.0.0/workspace/engine.py with timestamp 1474439315450
16/09/21 06:28:35 INFO util.Utils: Copying /usr/spark-2.0.0/workspace/app.py to /tmp/spark-1be5163e-6665-45b3-8a6e-c4c715bb14cb/userFiles-f1307aa8-cdfa-46a5-a10c-47812bd9d042/app.py
16/09/21 06:28:35 INFO spark.SparkContext: Added file workspace/app.py at file:/usr/spark-2.0.0/workspace/app.py with timestamp 1474439315480
16/09/21 06:28:35 INFO util.Utils: Copying /usr/spark-2.0.0/workspace/tools.py to /tmp/spark-1be5163e-6665-45b3-8a6e-c4c715bb14cb/userFiles-f1307aa8-cdfa-46a5-a10c-47812bd9d042/tools.py
16/09/21 06:28:35 INFO spark.SparkContext: Added file workspace/tools.py at file:/usr/spark-2.0.0/workspace/tools.py with timestamp 1474439315508
[21/Sep/2016:06:28:37] ENGINE Bus STARTING
[21/Sep/2016:06:28:37] ENGINE Started monitor thread 'TimeoutMonitor'.
[21/Sep/2016:06:28:37] ENGINE Started monitor thread 'Autoreloader'.
[21/Sep/2016:06:28:37] ENGINE Serving on http://0.0.0.0:5433
[21/Sep/2016:06:28:37] ENGINE Bus STARTED
```

Figura 21. API de Clasificación desplegada

Fuente: Propia

Posteriormente, se desarrolló una interfaz web llamada “Lambda”, la cual permite interactuar con el API de forma sencilla y ofrece gráficos que facilitan el entendimiento de los resultados obtenidos. La interfaz fue realizada con el framework de desarrollo de aplicaciones web Meteor, en su versión 1.3.3, tomando como base el lenguaje Javascript en conjunto con la librería de Plotly para el respectivo lenguaje.

Siguiendo el orden de ideas para el despliegue del API de clasificación, el despliegue de la aplicación web que cumple el rol de interfaz es igualmente simple y se encuentra también anexado al final del presente documento.

A través de dicha interfaz se pueden realizar un conjunto determinado de acciones que se explican a continuación.

4.9.1. Configuración de Credenciales

Lambda necesita registrar las credenciales de acceso y seguridad de una aplicación de Twitter creada bajo ciertos parámetros, los cuales son explicados detalladamente en la interfaz de usuario de Lambda. Lambda utiliza estas credenciales para conectarse con el API de Twitter y ejecutar la mayoría de sus funciones. Este paso es estrictamente necesario para poder iniciar sesión en la aplicación web.

4.9.2. Inicio de sesión por medio de Twitter

Haciendo uso de las credenciales suministradas, Lambda permite a sus usuarios iniciar sesión por medio de la plataforma de Twitter, ingresando su correo electrónico y contraseña, y aceptando los permisos que la aplicación de Twitter solicita para que Lambda pueda ejecutar sus procesos exitosamente. Estas credenciales pueden ser reconfiguradas más adelante en la aplicación.

4.9.3. Entrenamiento del modelo

Antes de poder ejecutar algún proceso de clasificación, Lambda exige que se realicen los siguientes pasos en el orden establecido:

1. Suministrar dirección del servidor: Es necesario que el usuario indique cual es la dirección del servidor en el cual se encuentra desplegado el API de clasificación.
2. Entrenar el clasificador spam: Algunas características en el proceso de clasificación son calculadas por este componente. Por ello es necesario suministrar la ubicación de los conjuntos de datos spam y no spam. Además, es necesario definir la cantidad y el nivel de los árboles con los que trabajará el algoritmo Random Forest de ese componente. Una vez

culminado el entrenamiento, la interfaz señalará la exactitud obtenida por el componente, permitiendo al usuario volver a realizar el entrenamiento con otros set de datos y parámetros si no se encuentra conforme con el resultado obtenido.

3. Entrenar el juez: Este componente se encarga de evaluar a cada usuario según las características calculadas en el proceso de clasificación. Para que funcione de forma correcta se debe suministrar la ubicación de los conjuntos de usuarios humanos, ciborgs y bots clasificados manualmente. Al igual que el componente de clasificación spam, se debe suministrar la cantidad y el nivel de los árboles a usar por el algoritmo Random Forest, aparte de la dirección en la cual se guardará un respaldo automático del modelo resultante. Al igual que para el componente anterior, la interfaz anunciará la exactitud obtenida por el juez.



Figura 22. Pantalla de configuración y entrenamiento

Fuente: Propia

4.9.4. Clasificar usuarios

Lambda puede clasificar usuario de tres formas distintas:

1. Clasificación individual: Suministrando el nombre de usuario, la cantidad de tweets a descargar y la cantidad de tweets a solicitar por cada solicitud al API de Twitter; Lambda descarga la información del usuario y la transfiere al API de clasificación para ser procesada.
2. Clasificación de seguidores: Lambda puede descargar la información de un número determinado de seguidores del usuario que se encuentra logueado en la aplicación. Al igual que para la clasificación individual, debe suministrarse la cantidad de tweets a descargar y la cantidad de

tweets por solicitud al API de Twitter, además del número de seguidores a evaluar.

3. Clasificación por lotes: En caso de que el usuario posea los timelines de los usuarios que desea evaluar, puede suministrar a Lambda la ubicación del repositorio remoto (alguna de las distribuciones de hdfs) que los contiene para que sean evaluados por el API de clasificación.

Figura 23. Pantalla de clasificación individual de usuario

Fuente: Propia

4.9.5. Visualizar resultados

La visualización de resultados divide a los usuarios procesados en dos categorías: “Mis Seguidores” y “Otros Usuarios”.

En el apartado “Otros Usuarios” se listan todos los usuarios clasificados de forma individual y por lotes, especificando su nombre de usuario, el estado de su clasificación (Clasificado, Procesando y Error), y la posibilidad de ver los resultados de la clasificación del usuario o eliminarlo.

En el apartado “Mis seguidores” presentan la misma información del apartado anterior, añadiendo la capacidad de poder bloquear y desbloquear de la plataforma de Twitter a los seguidores procesados del usuario logueado.

En la sección “Matriz de Confusión” se señala en detalle la exactitud del proceso de entrenamiento del juez con respecto a la distribución de los usuarios procesados entre las tres categorías.

El resultado de la clasificación de un usuario señala la categoría a la cual el usuario pertenece, teniendo como opciones solamente tres grupos distintos:

humano, ciborg y bot. Además, se visualizaran los valores específicos que obtuvo el usuario para cada medida utilizada durante el proceso de clasificación, aparte de un conjunto de graficas que comparan el comportamiento del usuario suministrado con el comportamiento de los usuarios bots, ciborgs y humanos utilizados durante el entrenamiento del juez.

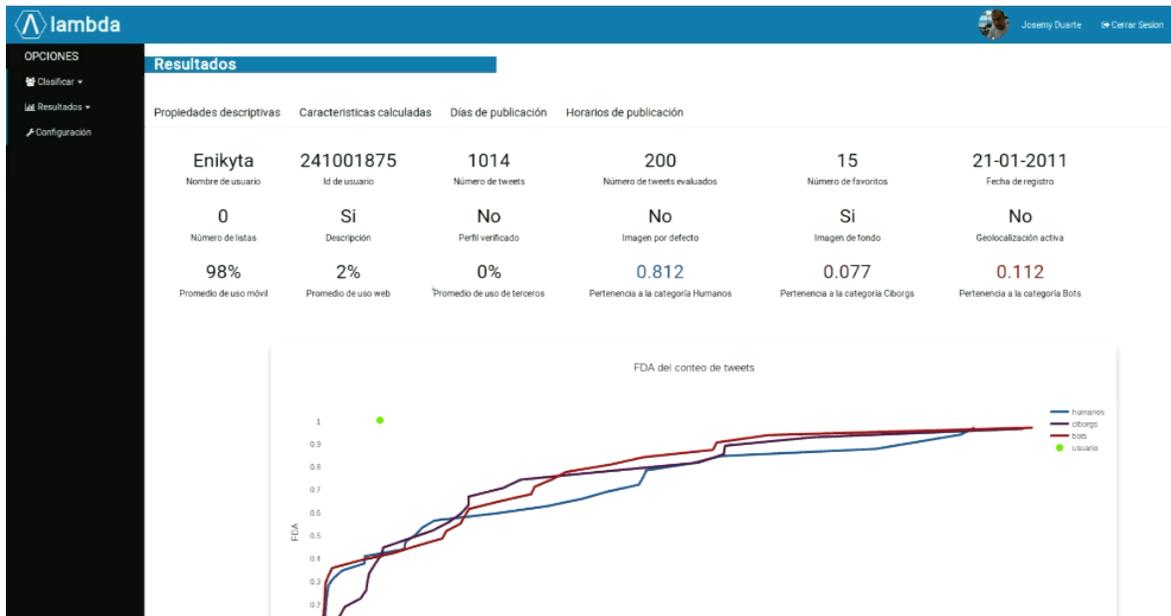


Figura 24. Pantalla de resultados de clasificación

Fuente: Propia

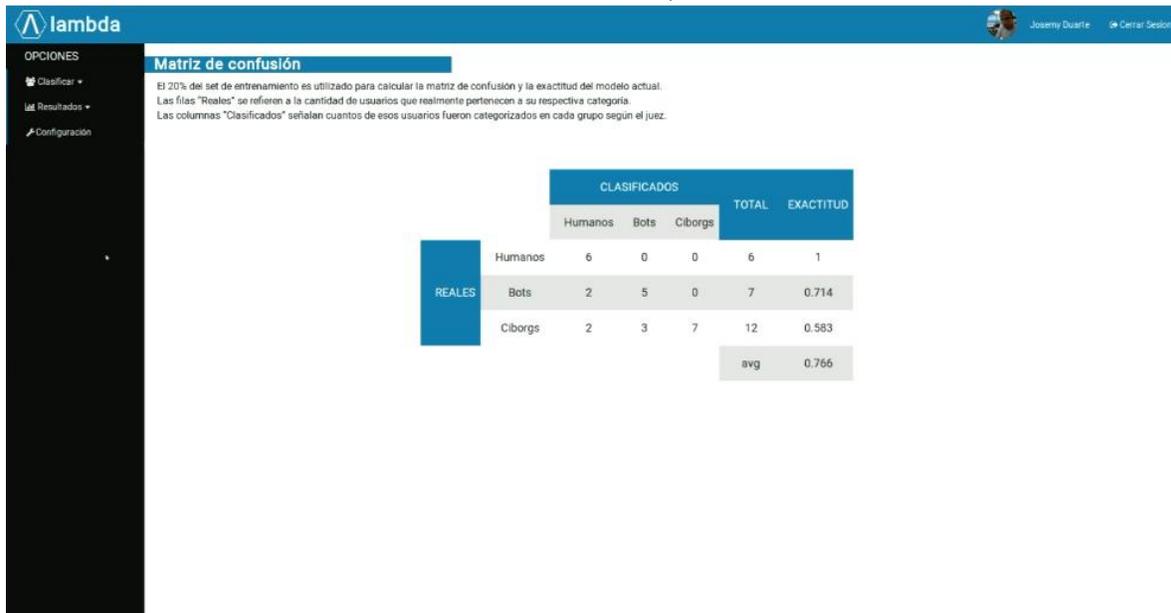


Figura 25. Pantalla con matriz de confusión del juez entrenado

Fuente: Propia

4.10. Retroalimentación

Posterior a la culminación de la herramienta, se realizó una revisión general del trabajo realizado, buscando aspectos que pudieran ser mejorados para aumentar el rendimiento y la efectividad.

Se lograron recaudar las siguientes recomendaciones referentes a características que se pudieran agregar para la evaluación de la clasificación, estas son:

- Timelines que posean tweets repetidos.
- Ratio de hashtags repetidos.
- Comprobación de enlaces maliciosos en los tweets.
- Diversidad de enlaces.
- Evaluar interrelación de los usuarios, considerando que quizás los usuarios de una misma categoría suelen relacionarse con una frecuencia mayor a la normal.
- Calcular el gusto de las cuentas (Promedio de la reputación de los amigos).

Debido a la forma en la que funciona el algoritmo del Random Forest, es posible que al aumentar el set de entrenamiento de los usuarios y evaluar otros parámetros de ejecución puedan generarse mejores resultados.

Este mismo argumento es aplicable al set de datos perteneciente a los tweets spam y no spam. Igualmente es recomendable que este set de datos se actualice cada cierto tiempo, ya que la forma en la interactúan los usuarios es propensa a cambiar y quizás las palabras utilizadas varíen en el tiempo.

Es probable que otros algoritmos de Machine Learning sean capaces de generar mejores resultados que los obtenidos por esta herramienta, por lo que sería recomendable evaluar las distintas alternativas para lograr obtener una mayor eficacia.

Capítulo 5

Conclusiones

El proceso de extracción de los 700.000 timelines de Twitter, sin duda representó un desafío y una pieza fundamental en la persecución de los objetivos planteados, esto debido a las restricciones impuestas por el API de Twitter para la extracción de datos. Sin embargo, no se puede negar que dicha API resultó ser de gran utilidad y probablemente ahorró una gran cantidad de trabajo que de no existir, hubiera sido necesario un proceso de extracción mucho más complejo.

Cada usuario de Twitter es libre de publicar el contenido que desee (texto, imágenes, videos, URLs a otras páginas...), permitiéndole expresarse libremente. Luego del estudio realizado, es posible afirmar que el contenido de las publicaciones hechas por cada tipo de usuario, sea humano, bot o ciborgs, suele diferir entre cada categoría. Sea por que los humanos con frecuencia publican mensajes más cortos que los bots o ciborgs, o porque los bots incluyen con mayor frecuencia URLs en sus publicaciones, o la cantidad de spam presente en las publicaciones de los bots respecto a los otros... No se puede negar que el contenido de las publicaciones sugiere que existen diferencias entre cada categoría.

Tomando en cuenta que el contenido publicado por los usuarios realmente marcaba una diferencia entre ellos, se procedió a estudiar el comportamiento. En este caso, también se observaron diferencias entre las categorías. Se observó por ejemplo, que los humanos suelen realizar sus publicaciones desde dispositivos móviles con mayor frecuencia que los bots, que suelen preferir publicar desde aplicaciones de terceros. También que la frecuencia de respuestas a las publicaciones al igual que las horas de actividad, son características que definen diferencias bien demarcadas entre los distintos tipos de usuario.

Luego de estudiar el contenido de las publicaciones y el comportamiento de los usuarios (humanos, bots y ciborgs) en Twitter, se observó que existían una serie de características que podrían ser útiles para la clasificación automática de las distintas categorías de usuarios. Fueron estas características las utilizadas para el entrenamiento y evaluación de un modelo Random Forest que logró una exactitud de un 86%, cuya debilidad se puede decir son los ciborgs, especulando que esto es debido a su naturaleza híbrida.

Finalmente, se obtuvo una aplicación web que permite entrenar el modelo con distintos sets de entrenamiento, clasificar de forma sencilla y directa distintas

cuentas, incluida la opción de clasificar los seguidores de la cuenta del usuario registrado en la herramienta, además de la posibilidad de bloquearlos. Ofreciendo de forma resumida y grafica las características evaluadas, la categoría predicha y la probable pertenencia a las otras dos.

5.1. Contribución

En la actualidad, muchos estudios que involucran el análisis de la interacción de los individuos en las redes sociales se pueden interpretar como indicadores de la realidad social. Bien sea porque estudian la forma en la que se expresan en las redes como un indicador de la opinión general, o por referirse a las relaciones entre usuarios de la red social relacionadas de alguna forma con la vida real, en cualquier caso no se puede negar la paridad por la que se suelen realizar estos estudios.

El desarrollo de este proyecto busca contribuir con ese tipo de estudios, ofreciéndoles una herramienta que les permita descartar aquellos tipos de usuarios que quizás no aporten información valiosa a la investigación o que generen cierto ruido en los datos. Igualmente, su uso no queda limitado al descrito anteriormente, sino al ingenio del investigador y a cualquier función que le pueda encontrar a este proyecto.

Finalmente, este proyecto queda abierto a cualquier aporte que pueda ampliar sus funciones o mejorar las existentes, teniendo en cuenta que el propósito final es y será siempre contribuir a la comunidad científica.

5.2. Recomendaciones

El porcentaje de casos correctamente categorizados se verá afectado según la calidad del set de entrenamiento utilizado. Se recomienda prestar mucha atención a la selección de las cuentas a utilizar para el entrenamiento, al igual que a los tweets de tipo spam y no spam.

El comportamiento de los usuarios y el contenido publicado puede variar con el pasar del tiempo, por lo que actualizar el set de entrenamiento con cierta frecuencia puede mejorar las predicciones pasado un periodo de tiempo.

Para una experiencia fluida en la utilización de la herramienta se recomienda su despliegue en un ambiente distribuido y con buena capacidad de procesamiento. El rendimiento, la memoria principal, la capacidad de procesamiento y la cantidad de datos a procesar están estrechamente relacionados, viéndose afectados los tiempos de espera cuando la cantidad de datos es amplia y la memoria principal o capacidad de cómputo es baja.

Aunque se recomiendan conjuntos de entrenamiento lo suficientemente numerosos (hasta cierto punto) para mejorar la exactitud del modelo, los conjuntos de usuarios humanos, ciborgs y bots para el entrenamiento deben tener, como mínimo, cuatro usuarios cada uno.

Con respecto a la clasificación de usuarios, se recomienda un número mínimo de 120 tweets a descargar. Así mismo, el número de tweets por solicitud no puede exceder la cantidad de 200 tweets debido a las limitaciones del API de Twitter. [64]

5.3. Trabajos futuros

La inclusión de nuevas características discriminatorias pudiera mejorar las predicciones hechas, al igual que la utilización de otros modelos de clasificación o la mezcla de los mismos.

Por otro lado, agregar la capacidad de aplicar acciones a lotes de usuarios en la aplicación web mejoraría la gestión de los mismos para los escenarios donde se trabajen con grandes cantidades de usuarios.

Anexos

1. Instrucciones para la creación del contenedor

Instrucciones del Dockerfile

```
FROM gettyimages/spark:2.0.0-hadoop-2.7

MAINTAINER Josemy Duarte <duartejosemy@gmail.com>

ENV SPARK_VERSION 1.6.1
ENV HADOOP_VERSION 2.6
ENV MONGO_HADOOP_VERSION 1.5.2
ENV MONGO_HADOOP_COMMIT r1.5.2

RUN apt-get update \

    && apt-get install -y wget libffi-dev libssl-dev build-essential python-dev libxml2-dev libxslt-
dev python3-dev\

    && apt-get clean \

    && rm -rf /var/lib/apt/lists/*

ENV MONGO_HADOOP_URL https://github.com/mongodb/mongo-
hadoop/archive/${MONGO_HADOOP_COMMIT}.tar.gz

ENV MONGO_HADOOP_LIB_PATH /usr/local/mongo-hadoop/build/libs

ENV MONGO_HADOOP_JAR ${MONGO_HADOOP_LIB_PATH}/mongo-hadoop-
${MONGO_HADOOP_VERSION}.jar

ENV MONGO_HADOOP_SPARK_PATH /usr/local/mongo-hadoop/spark

ENV MONGO_HADOOP_SPARK_JAR
${MONGO_HADOOP_SPARK_PATH}/build/libs/mongo-hadoop-spark-
${MONGO_HADOOP_VERSION}.jar

ENV PYTHONPATH
$PYTHONPATH:${MONGO_HADOOP_SPARK_PATH}/src/main/python

ENV SPARK_DRIVER_EXTRA_CLASSPATH
${MONGO_HADOOP_JAR}:${MONGO_HADOOP_SPARK_JAR}
```

```
ENV CLASSPATH ${SPARK_DRIVER_EXTRA_CLASSPATH}
ENV JARS
${MONGO_HADOOP_JAR},${MONGO_HADOOP_SPARK_JAR}

RUN wget -qO - ${MONGO_HADOOP_URL} | tar -xz -C /usr/local/ \
  && mv /usr/local/mongo-hadoop-${MONGO_HADOOP_COMMIT} \
  /usr/local/mongo-hadoop \
  && cd /usr/local/mongo-hadoop \
  && ./gradlew jar

RUN echo "spark.driver.extraClassPath  ${CLASSPATH}" >
${SPARK_HOME}/conf/spark-defaults.conf

RUN wget https://bootstrap.pypa.io/get-pip.py \
  && python2.7 get-pip.py \
  && rm get-pip.py

RUN pip2 --no-cache-dir install requests numpy flask python-dateutil
cherry pyopenssl ndg-httpsclient paste pymongo
```

2. Instrucciones de despliegue del API de clasificación y la aplicación web

Despliegue de contenedor con Mongo

Especificación de variables:

- MONGO_NAME: Alias del contenedor con mongo.

```
docker run -p 27017:27017 --name $MONGO_NAME mongo
```

Despliegue de contenedor con API

Especificación de variables:

- MONGO_NAME: Alias del contenedor con mongo.
- API_SPARK_HOME: Ubicación local del código del API

```
docker run -ti --rm --link $MONGO_NAME:mongo -p 5433:5433 -v  
$API_SPARK_HOME:/usr/spark-2.0.0/workspace josemyd/spark2-  
mongohadoop bin/spark-submit workspace/server.py
```

Despliegue de la aplicación web

Especificación de variables:

- APP_HOME: Ubicación local del código de la aplicación.

```
cd $APP_HOME  
  
meteor
```

3. Peticiones al API de clasificación

Archivo de configuración del API de clasificación

Especificación de variables:

- Spark
 - name: Nombre de la aplicación en Spark.
- Server
 - host: Dirección en la cual se desplegará el API.
 - port: Puerto en el cual se desplegará el API.
- Database
 - host: Dirección en la cual se encuentra desplegada la instancia de mongoDB a usar.
 - port: Puerto en el cual se encuentra desplegada la instancia de mongoDB a usar.
 - collection: Nombre de la colección en la que se almacenarán los resultados de las clasificaciones.
 - collection_training: Nombre de la colección en la que se almacenarán los resultados de los entrenamientos.
 - ttl: Tiempo de vida de los datos en la instancia de mongoDB.

```
[spark]
```

```
name = ExtraerCaracteristicas
```

```
[server]
```

```
host = 0.0.0.0
```

```
port = 5433
```

```
[database]
```

```
host = mongo
```

```
port = 27017
```

```
db = db
```

```
collection = caracteristicas
```

```
collection_training = entrenamiento
```

```
ttl = 2000
```

Entrenar clasificador spam

Especificación de variables:

- SPAM: Ruta de la ubicación del set de datos spam en el contenedor. Ej: workspace/datasets/spam.
- NO_SPAM: Ruta de la ubicación del set de datos no spam en el contenedor. Ej: workspace/datasets/no_spam.
- NUM_TREE: Cantidad de árboles para el Random Forest.
- MAX_DEPTH: Nivel de árboles para el Random Forest.
- SERVIDOR: Dirección del servidor en el cual se encuentra desplegada el API de clasificación. Ej: http://localhost:5433

```
curl -H "Content-Type: application/json" -X POST -d
 '{"spam":$SPAM,"no_spam":$NO_SPAM,"num_tree":$NUM_TREE,"max_d
 epth":$MAX_DEPTH}' $SERVIDOR/entrenar_spam/
```

Entrenar juez

Especificación de variables:

- HUMANOS: Ubicación de la carpeta con los timelines de usuarios humanos en el contenedor. Ej: workspace/humanos
- CIBORGS: Ubicación de la carpeta con los timelines de usuarios ciborgs en el contenedor. Ej: workspace/ciborgs
- BOTS: Ubicación de la carpeta con los timelines de usuarios bots en el contenedor. Ej: workspace/bots
- NUM_TREE: Cantidad de árboles para el Random Forest.
- MAX_DEPTH: Nivel de árboles para el Random Forest.
- DIR_JUEZ: Dirección local al servidor donde se guardará un respaldo del juez entrenado
- SERVIDOR: Dirección del servidor en el cual se encuentra desplegada el API de clasificación. Ej: http://localhost:5433

```
curl -H "Content-Type: application/json" -X POST -d
 '{"humanos":$HUMANOS,"ciborgs":$CIBORGS,"bots":$BOTS,"num_tree":$
 NUM_TREE,"max_depth":$MAX_DEPTH,"dir_juez":$DIR_JUEZ}'
 $SERVIDOR/entrenar_juez/
```

Evaluar directorio de usuarios

Especificación de variables:

- DIRECTORIO: Ubicación del contenedor o remota (hdfs) de timelines de usuarios a evaluar. Ej: workspace/usuarios/*
- SERVIDOR: Dirección del servidor en el cual se encuentra desplegada el API de clasificación. Ej: http://localhost:5433

```
curl -H "Content-Type: application/json" -X POST -d  
'{"directorio":$DIRECTORIO}' $SERVIDOR/evaluar/
```

Evaluar usuario

Especificación de variables:

- TIMELINE: Timeline de usuario a evaluar en formato JSON.
- SERVIDOR: Dirección del servidor en el cual se encuentra desplegada el API de clasificación. Ej: http://localhost:5433

```
curl -H "Content-Type: application/json" -X POST -d  
'{"timeline":$TIMELINE}' $SERVIDOR/evaluar/
```

Referencias

- [1] McGiboney, M. (2009). *Twitter's Tweet Smell of Success*. Recuperado el 3 de octubre de 2016, de <http://www.nielsen.com/us/en/insights/news/2009/twitters-tweet-smell-of-success.html>
- [2] Twitter, Inc. (2016). *Empresa | About*. Recuperado el 3 de octubre de 2016, de <https://about.twitter.com/es/company>
- [3] Ferrara, E., Varol, O., Davis, C., Menczer, F., & Flammini, A. (2016). The rise of social bots. *Communications Of The ACM*, 59(7), 96-104.
- [4] Boshmaf, Y., Muslukhov, I., Beznosov, K., & Ripeanu, M. (2011, Diciembre). The socialbot network: when bots socialize for fame and money. En *Proceedings of the 27th Annual Computer Security Applications Conference* (pp. 93-102). ACM.
- [5] Murgia, A., Janssens, D., Demeyer, S., & Vasilescu, B. (2016, Mayo). Among the Machines: Human-Bot Interaction on Social Q&A Websites. En *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (pp. 1272-1279). ACM.
- [6] Penna, G. (2012). *Reality Mining in Twitter* (Maestría). Departamento de Computación del Imperial London College.
- [7] Chu, Z., Gianvecchio, S., Wang, H., & Jajodia, S. (2012). Detecting Automation of Twitter Accounts: Are You a Human, Bot, or Cyborg?. *IEEE Transactions On Dependable And Secure Computing*, 9(6), 811-824.
- [8] Leek, J. (2013, Diciembre 12). *The key word in "Data Science" is not Data, it is Science*. Recuperado el 3 de octubre de 2016, de <http://simplystatistics.org/2013/12/12/the-key-word-in-data-science-is-not-data-it-is-science/>
- [9] Dhar, V. (2013, Diciembre). Data Science and Prediction. *Communications of the ACM* (56) 12, 64-73.
- [10] Laney, D. (2001). 3D data management: Controlling data volume, velocity and variety. *META Group Research Note*, 6, 70.
- [11] Microsoft: News Center Latinoamerica. (2014, Octubre 2). Big Data: Volumen, Variabilidad y Velocidad. Recuperado el 3 de octubre de 2016, de <https://news.microsoft.com/es-xl/big-data-volumen-variabilidad-y-velocidad/>

- [12] Rijmenam, M. (s.f.) *Why The 3V's Are Not Sufficient To Describe Big Data*. Recuperado el 3 de octubre de 2016, de <https://datafloq.com/read/3vs-sufficient-describe-big-data/166>
- [13] McNulty, E. (2014). *Understanding Big Data: The Seven V's*. Recuperado el 3 de octubre de 2016, de <http://dataconomy.com/seven-vs-big-data/>
- [14] Rishe, N. (1992). *Database Design: The Semantic Modeling Approach*. School of Computer Science. Florida International University. Miami, Estado Unidos: McGraw-Hill.
- [15] Codd, E. F. (1990). *The relational model for database management: version 2*. Addison-Wesley Longman Publishing Co., Inc..
- [16] Rose, M. (2015, Enero). *Base de datos relacional: Definición*. Recuperado el 3 de octubre de 2016, de <http://searchdatacenter.techtarget.com/es/definicion/Base-de-datos-relacional>
- [17] Warden, P. (2011). *Big Data Glossary*. O'Reilly Media.
- [18] Silva, A. (2015). *Entendiendo el movimiento NoSQL (Not Only SQL)*. Universidad Central de Venezuela. Caracas, Venezuela.
- [19] Acens: Acens White Papers. (s.f.). Bases de datos NoSQL. Qué son y tipos que nos podemos encontrar. Telefónica.
- [20] Beal, V. (s.f.). *Structured Data*. Webopedia. Recuperado el 3 de octubre de 2016, de http://www.webopedia.com/TERM/S/structured_data.html
- [21] Andriole, S. (2015). *Unstructured Data: The Other Side of Analytics*. Recuperado el 3 de octubre de 2016, de <http://www.forbes.com/sites/steveandriole/2015/03/05/the-other-side-of-analytics/2/>
- [22] Abiteboul, S. (s.f.). *Querying Semi-Structured Data*. Recuperado el 3 de octubre de 2016, de <http://cs.brown.edu/courses/cs295-11/2006/semistructured.pdf>
- [23] Loudon, K. (2011). *Programming languages: principles and practices*. Cengage Learning.
- [24] Samment, J. (1972). *Programming Languages: History and Future*. IBM Corporation. Communication of the ACM (15).
- [25] Shuerer, K. Maufrais, C. Letodal, C. Deveaud, E. Petit, A. (2008). *Introduction to Programming using Python*. Pasteur Institute.
- [26] Sanner, M. F. (1999). *Python: a programming language for software integration and development*. J Mol Graph Model, 17(1), 57-61.

- [27] Python. (s.f.) *What is Python? Executive Summary*. Recuperado el 3 de octubre de 2016, de <https://www.python.org/doc/essays/blurbl/>
- [28] Martin Odersky (2011). *The Scala Language Specification*. Recuperado el 3 de octubre de 2016, de http://www-dev.scala-lang.org/old/sites/default/files/linuxsoft_archives/docu/files/ScalaReference.pdf
- [29] Mozilla Developer Network. (2015). *¿Qué es JavaScript?*. Recuperado el 3 de octubre de 2016, de https://developer.mozilla.org/es/docs/Web/JavaScript/Acerca_de_JavaScript
- [30] Apache Software Foundation: Hadoop. (2014). *Hadoop*. Recuperado el 3 de octubre de 2016, de <https://hadoop.apache.org/>
- [31] Apache Software Foundation: Hadoop. (2014). *Native Libraries Guide*. Recuperado el 3 de octubre de 2016, de <https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-common/NativeLibraries.html>
- [32] Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1), 107-113.
- [33] Hortonwrks. (s.f.). *Apache Spark & Hadoop*. Recuperado el 3 de octubre de 2016, de <http://hortonworks.com/hadoop/spark/>
- [34] Docker Inc. (2016). *What is Docker?*. Recuperado el 3 de octubre de 2016, de <https://www.docker.com/what-docker>
- [35] Wolf, G. (2015). *Los contenedores: Aislamiento, sí, pero... ¿distribución?*. Software Gurú, (48), 50-51.
- [36] Squillace, R. (2016). *Máquinas virtuales y contenedores de Azure*. Recuperado el 3 de octubre de 2016, de <https://azure.microsoft.com/es-es/documentation/articles/virtual-machines-windows-containers/>
- [37] Yang, C. (2015). Express, Koa, Meteor, Sails.js: *Four Frameworks Of The Apocalypse*. Recuperado el 3 de octubre de 2016, de <https://www.toptal.com/nodejs/nodejs-frameworks-comparison>
- [38] DeBergalis, M. (2012). *Skybreak is now Meteor*. Recuperado el 3 de octubre de 2016, de <http://info.meteor.com/blog/skybreak-is-now-meteor>
- [39] Meteor Development Group Inc. (2016). *User Interfaces*. Recuperado el 3 de octubre de 2016, de <https://guide.meteor.com/ui-ux.html>
- [40] Plotly. (2016). *Visualize Data, Together*. Recuperado el 3 de octubre de 2016, de <https://plot.ly/>

- [41] Plotly. (2016). *Librerías API de Plotly*. Recuperado el 3 de octubre de 2016, de <https://plot.ly/api/>
- [42] Chakrabarti, S., Ester, M., Fayyad, U., Gehrke, J., Han, J., Morishita, S., Piatetsky-Shapiro, G., Wang, W. (2006) *Data Mining Curriculum: A Proposal (Version 1.0)*. Intensive Working Group of ACM SIGKDD Curriculum Committee.
- [43] Maimon, O. y Rokach, L. (2010). *Data Mining and Knowledge Discovery Handbook, Second Edition*. Springer.
- [44] Handola, V., Banerjee, A., Kumar, V. (2009). *Anomaly detection: A survey*. ACM Computing Surveys 41 (3).
- [45] Hodge, V. J. y Austin, J. (2004). *A Survey of Outlier Detection Methodologies*. Artificial Intelligence Review 22 (2).
- [46] Dokas, P., Ertöz, L., Kumar, V., Lazarevic, A., Srivastava, J., & Tan, P. N. (2002, November). Data mining for network intrusion detection. En *Proc. NSF Workshop on Next Generation Data Mining* (pp. 21-30).
- [47] Piatetsky-Shapiro, G. (1991). Discovery, analysis, and presentation of strong rules. *Knowledge discovery in databases*, 229-238.
- [48] Bailey, K. (1994). *Numerical Taxonomy and Cluster Analysis*. Typologies and Taxonomies. p. 34.
- [49] Everitt, B. (2011). *Cluster analysis*. Chichester, West Sussex.
- [50] Cook, R. D., & Weisberg, S. (1982). Criticism and influence analysis in regression. *Sociological methodology*, 13, 313-361.
- [51] Cohen, K. B., & Hunter, L. (2008). Getting started in text mining. *PLoS Comput Biol*, 4(1), e20.
- [52] Grimes, S. (2007, Octubre 30). *A Brief History of Text Analytics*. Recuperado el 3 de octubre de 2016, de <http://www.b-eye-network.com/view/6311>.
- [53] Ellison, N. B. (2007). Social network sites: Definition, history, and scholarship. *Journal of Computer-Mediated Communication*, 13(1), 210-230.
- [54] Flores Cueto, J. J., Morán Corzo, J. J., & Rodríguez Vila, J. J. (2009). *Las Redes Sociales*. Universidad de San Martín de Porres, 1-15.
- [55] Tichy, N. M., Tushman, M. L., & Fombrun, C. (1979). *Social network analysis for organizations*. *Academy of management review*, pp. 507-519.
- [56] Brand, E., & Gómez, H. (2006). Análisis de redes sociales como metodología de investigación. Elementos básicos y aplicación.

- [57] Willging, P. A. (2008). Técnicas para el análisis y visualización de interacciones en ambientes virtuales. En *Redes: revista hispana para el análisis de redes sociales* (Vol. 14, pp. 000-0).
- [58] Scott, J. (2000). *Social Network Analysis: A Handbook. Second edition*. London: Sage Publications.
- [59] Wasserman, S., & Faust, K. (1994). *Social network analysis: Methods and applications*. New York: Cambridge University Press.
- [60] Sanz Menéndez, L. (2003). Análisis de redes sociales: o cómo representar las estructuras sociales subyacentes.
- [61] Sanz Menéndez, L. (2003, Junio). Análisis de redes sociales: o cómo representar las estructuras sociales subyacentes. En *Apuntes de Ciencia y Tecnología*, N° 7, pp. 21
- [62] Preffer, J. (1992). Organizaciones y teoría de las organizaciones. *Textos de economía*.
- [63] Tichy, N. M., Tushman, M. L., & Fombrun, C. (1979). Social network analysis for organizations. *Academy of management review*, pp. 507-519.
- [64] Hanneman, R. (2001). Introducción a los métodos del análisis de redes sociales.
- [65] Rollins, J. (2015). *Why we need a methodology for data science*. Recuperado el 3 de octubre de 2016, de <http://www.ibmbigdatahub.com/blog/why-we-need-methodology-data-science>
- [66] Caruana, R., & Niculescu-Mizil, A. (2006, Junio). An empirical comparison of supervised learning algorithms. En *Proceedings of the 23rd international conference on Machine learning* (pp. 161-168). ACM.
- [67] Porta, A., Baselli, G., Liberati, D., Montano, N., Cogliati, C., Gneccchi-Rusccone, T., Cerutti, S. (1998). Measuring regularity by means of a corrected conditional entropy in sympathetic outflow. *Biological cybernetics*, 78(1), 71-78
- [68] Moore, E. F. (1959). *The shortest path through a maze*. Bell Telephone System.
- [69] Twitter, Inc. (2016). *API rate limits*. Recuperado el 3 de octubre de 2016, de <https://dev.twitter.com/rest/public>
- [70] MINCI (2016). *Venezuela amplía acceso a servicios de Internet*. Recuperado el 3 de octubre de 2016, de <http://minci.gob.ve/2016/06/venezuela-amplia-acceso-a-servicios-de-internet/>

- [71] Statista (2016). *Number of active Twitter users in leading markets as of May 2016*. Recuperado el 3 de octubre de 2016, de <http://www.statista.com/statistics/242606/number-of-active-twitter-users-in-selected-countries/>
- [72] Smith, S. (2013). *Determining Sample Size: How to Ensure You Get the Correct Sample Size*. Recuperado el 3 de octubre de 2016, de <https://www.qualtrics.com/blog/determining-sample-size/>
- [73] Rios, G., & Zha, H. (2004, Julio). Exploring Support Vector Machines and Random Forests for Spam Detection. En CEAS.
- [74] Apache Spark. (2016). *PySpark ML Package, Naive Bayes*. Recuperado el 3 de octubre de 2016, de <http://spark.apache.org/docs/latest/api/python/pyspark.ml.html#pyspark.ml.classification.NaiveBayes>
- [75] Apache Spark. (2016). *PySpark ML Package, Random Forest Classifier..* Recuperado el 3 de octubre de 2016, de <http://spark.apache.org/docs/latest/api/python/pyspark.ml.html#pyspark.ml.classification.RandomForestClassifier>
- [76] Gini, C. (1912). Variabilità e mutabilità. *Reimprimido en Memorie di metodologica statistica (Ed. Pizetti E, Salvemini, T). Roma: Libreria Eredi Virgilio Veschi, 1.*
- [77] MongoDB. (s.f.). *MongoDB*. Recuperado el 3 de octubre de 2016, de <https://www.mongodb.com/es>.