CIMAT Centro de Investigación en Matemáticas, A.C.

APLICACIÓN DE APRENDIZAJE AUTOMÁTICO PARA LA PREDICCIÓN DE CLIENTES POTENCIALES EN PROCESOS DE MERCADOTECNIA

T E S I S

Que para obtener el grado de

Maestro en Ingeniería de Software

Presenta

Pablo Mar Castillo Castañeda

Director de Tesis:

Dr. José Arturo Mora Soto

Autorización de la versión final

Zacatecas, Zac., 9 de Noviembre de 2017

CENTRO DE INVESTIGACIÓN EN MATEMÁTICAS, A.C.

BIBLIOTECA

AUTORIZACION
PUBLICACION EN FORMATO ELECTRONICO DE TESIS

El que suscribe

Autor(s) de la tesis: ----------

Institución y Lugar: ----------

Grado Académico: Licenciatura ( ) Maestría ( ) Doctorado ( ) Otro ( ) ----------

Año de presentación: ----------

Área de Especialidad: ----------

Director(es) de Tesis: ----------

Correo electrónico: ----------

Domicilio: ----------

Palabra(s) Clave(s): ----------

ORCID: ----------

Por medio del presente documento autorizo en forma gratuita a que la Tesis arriba citada sea divulgada y reproducida para publicarla mediante almacenamiento electrónico que permita acceso al público a leerla y conocerla visualmente, así como a comunicarla públicamente en la Página WEB del Repositorio Institucional y Repositorio Nacional.

La vigencia de la presente autorización es por tiempo indefinido a partir de la firma de presente instrumento, quedando en el entendido de que si por alguna razón el alumno desea revocar la autorización tendrá que hacerlo por escrito con acuse de recibo de parte de alguna autoridad del CIMAT

Atentamente

_____

Pablo Mar Castillo Castañeda

CALLE JALISCO S/N MINERAL DE VALENCIANA APDO. POSTAL 402
C.P. 36240 GUANAJUATO, GTO., MÉXICO
TELÉFONOS (473) 732 7155, (473) 735 0800 EXT. 49609 FAX. (473) 732 5749
E-mail: biblioteca@cimat.mx

"El fracaso nunca te sobrecogerá si tu determinación para alcanzar el éxito es lo suficientemente poderosa." – Pathros

Resumen

En la última década, la inversión en publicidad y mercadotecnia ha incrementado considerablemente en México, siendo Mercadotecnia Directa la técnica con mayor incremento utilizada por empresas. Mediante la identificación de clientes potenciales para responder a una campaña de mercadotecnia, se pueden minimizar el costo de las campañas y maximizar el retorno de inversión enviando anuncios publicitarios específicos de acuerdo con los intereses de los clientes. Para lograrlo, grandes cantidades de datos deben ser analizados con la finalidad de determinar el canal más efectivo de Mercadotecnia Directa para alcanzar al cliente.

Por otro lado, debido a que la cantidad de respuestas positivas de los clientes es muy baja como resultado de este tipo de campañas, el conjunto de datos obtenidos generalmente se encuentra desbalanceado dificultando la predicción de clientes potenciales.

El objetivo de la investigación es identificar una técnica de aprendizaje automático que ofrezca el mejor rendimiento en la predicción de qué clientes tienen mayor probabilidad de comprar un producto o servicio como resultado de Mercadotecnia Directa. El conjunto de datos utilizado en esta investigación contiene información demográfica y socioeconómica de los clientes de una empresa en particular. La predicción de clientes potenciales ha sido formulada como una tarea de clasificación y regresión, utilizando los algoritmos de aprendizaje automático: (1) Random Forest, (2) Gradient Boosting y (3) eXtreme Gradient Boosting. Los resultados obtenidos muestran que el modelo eXtreme Gradient Boosting tiene un mejor desempeño frente a los otros modelos en este contexto. Así mismo, los resultados encontrados en la aplicación de este modelo ayudarán a los analistas de mercadotecnia a desarrollar mejores campañas.

Palabras clave: aprendizaje automático, predicción, clientes, mercadotecnia directa, marketing directo, clasificación, regresión.

Agradecimientos

Quiero dar las gracias a todos las personas que de una forma u otra, ayudaron a la culminación de esta tesis.

Primeramente, quiero dar gracias a Gaby, quien cruzó el Mar de Cortés junto a mi para hacer este sueño realidad. Vivir juntos en Zacatecas ha sido toda una aventura. Quiero además expresar mi gratitud a mis padres quienes me apoyaron desde la distancia en cada aspecto todo este tiempo. Este trabajo esta dedicado a ustedes. A mis hermanas Carolina y Kenia así como a mis pequeños sobrinos.

Estoy agradecido con mi director de tesis el Dr. Arturo Mora, quien siempre estuvo disponible para mi cuando necesité su ayuda, gracias por el tiempo que compartió conmigo durante su estancia en Zacatecas, fue lo máximo.

Quiero dar las gracias a todo el personal de CIMAT Zacatecas, a los Maestros y Doctores que fueron mis profesores durante mi ciclo escolar y a mis compañeros de generación, muchas gracias por la enseñanza que me han dejado y la experiencia de compartir un salón de clases con ustedes. Principalmente, quiero agradecer a mis amigos Francisco, Miguel, Ángel Farid, Daniel, Carlos, Freddy y Luis Ángel. Gracias por todo lo que me brindaron durante mi estancia en Zacatecas, gracias por el tiempo que pasamos juntos y todas las aventuras que compartimos.

Tambien, quiero dar las gracias a Pablo Martín y Pablo Bastidas por su amistad todos estos años. Gracias por sus consejos y apoyo para hacer más llevadera mi estancia.

Quiero agradecer al Maestro Pepe Hernández, por esta oportunidad de ir a Zacatecas a hacer mi Maestría. Recuerdo la llamada telefónica que sostuve con usted y la Maestra Lorena Barba cuando me entrevistaron y me dieron la noticia de haber sido aceptado en el programa de Maestría. Gracias Pepe por todo lo que me enseñó, pero principalmente por su amistad, por todos esos momentos que compartió conmigo, recordaré por siempre su compañía recorriendo los caminos de la Bufa en las frías mañanas de Zacatecas, muchas gracias.

Por último pero no menos importante, quiero dar las gracias al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico otorgado durante la realización de mis estudios de Maestría y el desarrollo de la presente tesis. Así mismo, quiero dar las gracias al Centro de Investigación en Matemáticas A.C. (CIMAT) por los apoyos en los congresos donde asistí.

Índice

Índice de Figuras

Índice de Tablas

Capítulo 1: Introducción

1.1 Introducción

Este capítulo presenta un panorama general del proyecto de investigación realizado. Se incluyen el contexto sobre el cual se desenvuelve el proyecto, la problemática presentada, el objetivo general, las preguntas de investigación y finalmente una descripción de la organización de la tesis.

1.2 Contexto

Mercadotecnia Directa (Direct Marketing en Inglés) es una técnica especial de mercadotecnia donde los vendedores seleccionan un grupo específico de consumidores, los cuales pueden encontrar un producto ya existente o un producto nuevo interesante (Barman et al, 2016).

De acuerdo Barman (2016) existen tres categorías de **Mercadotecnia Directa** dentro de las cuales se encuentran **Telemercadotecnia** (telemarketing en Inglés), **Mercadotecnia Directa por Correo Electrónico** (Email Direct Marketing en Inglés) y **Mercadotecnia Directa por Correo** (Direct Mail Marketing).

- En la Telemercadotecnia el vendedor intenta vender o informar sobre un producto por medio vía telefónica.
- En Mercadotecnia Directa por Correo Electrónico, el vendedor contacta al consumidor por medio de una dirección de correo electrónico.
- En Mercadotecnia Directa por Correo, el vendedor envía el catálogo de un producto, un díptico (brochure en Inglés) u otro tipo de publicidad impresa a la dirección domiciliaria del consumidor.

El uso de Mercadotecnia Directa para una campaña publicitaria resulta de gran interés para vendedores dado que puede ser medible fácilmente (Barman et al, 2016). Para medir el rendimiento de una campaña publicitaria se pueden aplicar diferentes métricas como la Tasa de respuesta, Costo por orden, Retorno de inversión, Tasa de cancelación de clientes (Churn en Inglés).

Generalmente son utilizados dos métodos en las campañas publicitarias, los cuales son Mercadotecnia Directa y Mercadotecnia Masiva (Mitik et al, 2017). La Mercadotecnia Masiva es una técnica de mercadotecnia en la cual los vendedores utilizan publicidad en medios

comunicación como lo son el periódico, la radio o la televisión para alcanzar tantos consumidores como sea posible (Kolarovszki et al, 2016).

1.3 Problemática

Según estudios de la Confederación de la Industria de la Comunicación Mercadotécnica en México (CICOM) el uso de Mercadotecnia Masiva ha disminuido en los últimos años (CICOM 2016). De acuerdo con el estudio CICOM 2016 la mercadotécnica creció en México 66.6% entre el 2010 y 2015, esto es una suma alrededor de $76.8 mdmdp.

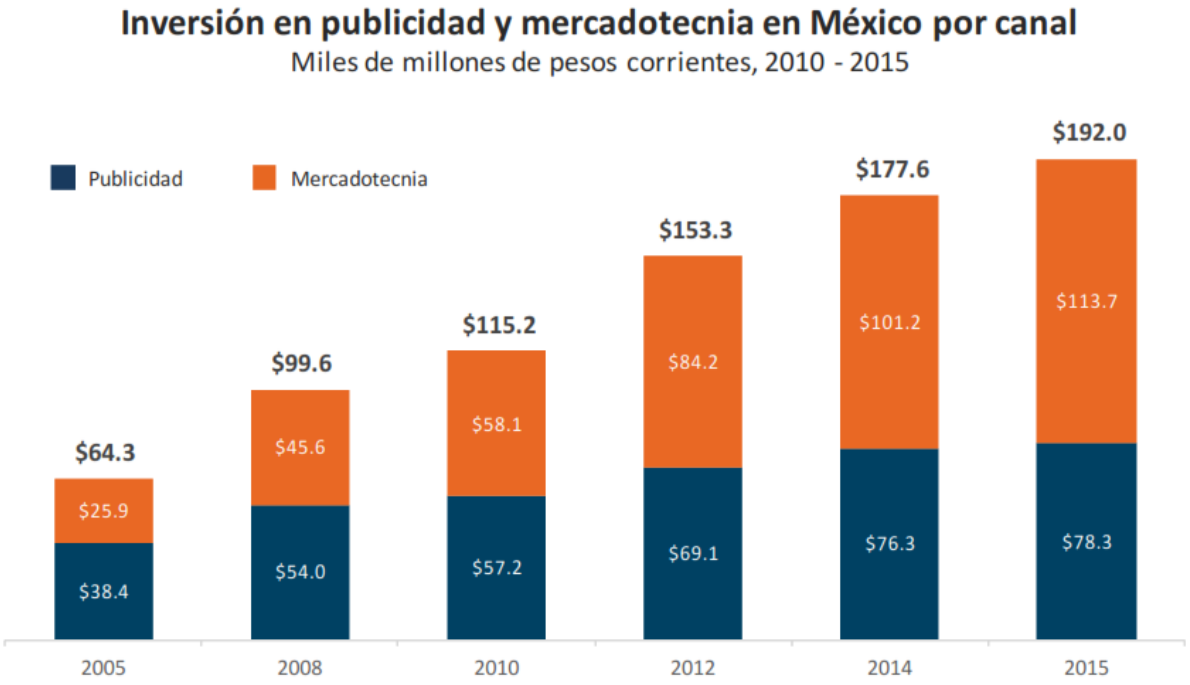

Figura 1. Inversión en publicidad y mercadotecnia en México por canal. CICOM 2016.

La Mercadotecnia Masiva ocupa el primer puesto con 41% de la inversión realizada en el 2015, seguido de promociones con un 35%, la Mercadotecnia Directa ocupa el tercer puesto con un 16% de inversión.

Sin embargo, el porcentaje de crecimiento de inversión entre 2005 y 2015 en publicidad y mercadotecnia en México para fue de 5% para la televisión de paga, un decremento del -26% para la radio, -46% para la televisión abierta y un -54% para periódicos, los cuales forman parte de la Mercadotécnica Masiva, comparado con un incremento del 40% para Mercadotecnia Directa.

El reporte Global Review of Data-Drive Marketing and Advertising (2017) realizado por Global Alliance of Data-Driven Marketing Associations (GDMA por sus siglas en Inglés), conformado por más de 18 mercados a nivel global, dentro de los cuales se encuentra México, aseguran

que los datos se han convertido en un recurso indispensable para la mercadotecnia. Una práctica común es la segmentación de la información para seleccionar y atraer de manera más efectiva a los clientes hacia ofertas de mercadotecnia. El reporte muestra que los avances en analítica predictiva y en segmentación ofrecen mayor potencial para expandir el uso de Mercadotecnia Directa impulsada por datos en los siguientes años.

Tradicionalmente, cuando se desarrollan nuevas campañas de mercadotecnia, los analistas de mercadeo frecuentemente basan sus decisiones en las acciones de los competidores, pero rara vez consideran las peticiones o las diferencias entre los clientes (Teng, He y Gu 2014). Como mencionan los autores Flici, Lü y Fearne (2011), los mercados con venta al detalle (retail en Inglés) operan en un ambiente altamente competitivo y volátil, donde el comportamiento de compra de los clientes está en constante cambio y es difícil de predecir. En mercados como estos, las técnicas tradicionales de mercadotecnia no son suficiente para desarrollar compañas efectivas, en su lugar el uso de Mercadotecnia Directa ha mostrado mejores resultados (Teng, He y Gu, 2014).

De acuerdo el reporte de GDMA (2017) el presupuesto de las empresas para mercadotecnia incrementa año tras año. Sin embargo, al utilizar Mercadotecnia Masiva se realiza una gran inversión que termina en un gasto para las empresas dado que sólo una pequeña porción de los clientes comprará el producto ofertado (Teng, He y Gu 2014). Lo anterior ha ocasionado que las empresas comiencen a invertir en mayor medida en el la Mercadotecnia Directa (CICOM, 2016) por el incremento en el rendimiento de las campañas de publicidad efectuadas usando ésta técnica.

La identificación de qué clientes son más probables a responder un correo o una llamada es un tema importante en la Mercadotecnia Directa (Teng, He y Gu 2014). Mediante la identificación de clientes prospectos, los vendedores pueden optimizar el rendimiento de sus campañas de mercadotecnia, maximizando sus ingresos y enviando una variedad de anuncios de acuerdo con los intereses de los clientes (Diapouli et al, 2016).

Modelos de aprendizaje automático pueden ser empleados para decidir qué productos o servicios ofrecer a los clientes basados en una probabilidad predicha (Teng, He y Gu 2014). Estos modelos en su mayoría se tratan de modelos predictivos para conocer la probabilidad de compra de un cliente o segmentar los clientes existentes (Tan et al, 2013). Los modelos construidos por lo general son modelos de aprendizaje automático supervisados, los cuales utilizan información histórica de los clientes como aspectos demográficos o socioeconómicos.

Los modelos de aprendizaje automático tienen además la capacidad de calcular un puntaje para cada cliente de acuerdo a la probabilidad de compra (Wong, 2010), de esta manera se pueden seleccionar los clientes potenciales que serán contactados por teléfono, correo

electrónico o dirección domiciliaria. Debido a que el monto destinado para compañas de mercadotecnia es limitado, es importante identificar a los prospectos más prometedores de compradores para enfocar los gastos de mercadotecnia, lo cual conlleva a una reducción significante del costo invertido (Mahdiloo, Noorizadeh y FarzipoorSaen, 2014).

El conocimiento obtenido en tales modelos permite un entendimiento más a fondo del comportamiento del cliente, lo cual apoya a la toma de decisiones en las empresas (Lessmann y Voß, 2008). Por medio de la obtención de patrones que conlleven a descubrir el comportamiento de los clientes al utilizar modelos de clasificación o regresión, como mencionan los autores Lessmann y Voß (2008), se puede ayudar a refinar y optimizar los procesos de negocios enfocados en los clientes para generar una estrategia que inmunice a los clientes contra las ofertas de los competidores y así obtener una ventaja competitiva en el mercado.

Las cuestiones mencionadas anteriormente enfatizan la importancia de tener un modelo de predicción preciso, capaz de identificar o predecir clientes que podrían estar interesados en un producto o servicio. Esos clientes identificados podrían ser contactados por medio de Mercadotecnia Directa.

Dentro de este ámbito, el objetivo de la investigación es identificar una técnica de aprendizaje automático que pueda ofrecer el mejor rendimiento en predecir qué clientes tienen mayor probabilidad de comprar un producto o servicio como resultado de Mercadotecnia Directa. En esta investigación además se busca evaluar las técnicas de aprendizaje automático de clasificación así como regresión con la finalidad de identificar aquella o aquellas que proporcionen el mejor rendimiento. Por último, se explora el mejor enfoque que ayude a identificar las características que puedan tener una influencia mayor en el desempeño del modelo.

1.4 Objetivo de investigación

Realizar un comparativo de técnicas de Aprendizaje Automático para predecir la probabilidad de compra de un producto por parte de un cliente para la empresa conocida en este estudio como Adventure Works Cycles.

1.5 Preguntas de investigación

1. ¿Es posible predecir la probabilidad de compra del cliente utilizando aprendizaje automático?

2. ¿Qué técnicas de aprendizaje automático son necesarias para generar modelos que predigan la probabilidad de compra?
3. ¿Qué características del cliente son útiles para predecir la probabilidad de compra?
4. ¿Cuál de los enfoques propuestos de aprendizaje automático es más preciso para la probabilidad de compra?

1.6 Alcance y limitaciones del estudio

Esta sección describe el alcance del proyecto y brevemente resalta las restricciones que limitan el enfoque de este proyecto de tesis.

- Conjunto de datos

El experimento fue realizado utilizando los conjuntos de datos proporcionados por el programa Data Science Professional Project (DAT102x) de Microsoft. Este conjunto de datos ha sido generado de forma sintética con la finalidad de simular un gran volumen de registros correspondientes a los clientes de la empresa Adventure Works Cycles.

- Algoritmos de clasificación

Los algoritmos de clasificación utilizados en los experimentos fueron limitados a Gradient Boosting, Random Forest y XGBoost (Extreme Gradient Boosting). Gradient Boosting, Random Forest fueron seleccionados por mostrar un mejor desempeño frente a otros algoritmos de clasificación (Brown y Mues, 2012). XGBoost fue seleccionado debido a que es una optimización del algoritmo Gradient Boosting, haciéndolo altamente eficiente, flexible y portable (Darmatasia & Arymurthy, 2016).

- Algoritmos de regresión

Los algoritmos de regresión utilizados en los experimentos fueron limitados a Gradient Boosting, Random Forest y XGBoost (Extreme Gradient Boosting). Gradient Boosting, Random Forest fueron seleccionados por mostrar un mejor desempeño frente a otros algoritmos de regresión (Huang y Zhou, 2016).

- Despliegue

El objetivo de esta tesis no es implementar un sistema completo, sino investigar las posibles mejoras para los modelos de clasificación y regresión utilizando las técnicas de selección de características e ingeniería de características. Por ello la puesta en ambiente de operación ha quedado fuera del alcance del proyecto.

- Caducidad

La antigüedad de los conjuntos de datos no es relevante, ya que los experimentos probados en este proyecto de tesis pueden aplicarse a nuevos conjuntos de datos sin ningún problema. Sin embargo, cuando un modelo es puesto en producción se recomienda actualizar el modelo. La frecuencia para actualizar varia del comportamiento de los datos, pudiendo ser diariamente en ambientes dinámicos hasta cada año en ambientes donde no cambian mucho los datos (Barga, Fontama y Tok, 2015:13). Debido a que la puesta en marcha no forma parte del alcance en este proyecto, la actualización del modelo también se ha dejado fuera.

1.7 Estructura de la tesis

Capítulo 2. Se presenta una revisión de la literatura existente para entender el contexto sobre el cual se desarrolla el presente trabajo de investigación. Se explica el concepto de ciencia de datos, la clasificación de los diferentes tipos de algoritmos de aprendizaje automático, el proceso que involucra su implementación, así como las tecnologías más utilizadas en la industria.

Capítulo 3. Se describe el experimento realizado para identificar clientes potenciales a partir de sus características socioeconómicas y demográficas. Se presenta el conjunto de datos, así como las técnicas utilizadas para la transformación de los datos. Finalmente, dos modelos de aprendizaje automático son propuestos: un modelo de clasificación binaria y un modelo de regresión para dar solución a la problemática presentada.

Capítulo 4. Se presentan las conclusiones de la investigación y los hallazgos principales. Se analizan los resultasdos respecto al objetivo propuesto. Finalmente, se presentan las contribuciones realizadas por el estudio, junto con las limitaciones y el posible trabajo futuro derivado de la presente investigación.

Un repositorio público de GitHub que contiene el código fuente de los scripts de Python utilizados para el entrenamiento y predicción de los modelos está disponible en https://github.com/caztillo/Aprendizaje-Automatico-para-la-Prediccion-de-Clientes-Potenciales-en-Procesos-de-Mercadotecnia

Capítulo 2: Estado de la Práctica

2.1 Introducción

En este capítulo se describen los conceptos para entender el contexto sobre el cual se desarrolla el presente trabajo de investigación. Se explica el concepto de ciencia de datos, la clasificación de los diferentes tipos de algoritmos de aprendizaje automático, el proceso que involucra su desarrollo, así como las tecnologías más utilizadas en la industria.

2.2 Ciencia de datos

El término ciencia de datos (*Data Science* en Inglés) fue acuñado en 1974 por Peter Naur como la Ciencia del Tratamiento de los Datos. Desde entonces la definición del término ha crecido hasta convertirse en un conjunto de disciplinas entre las que se encuentran "estadística, matemáticas, investigación de operaciones, procesamiento de señales, lingüística, bases de datos y almacenamiento, programación, aprendizaje automático y cómputo científico" (Fontana, Barga y Tok, 2015:3). Otra definición compartida por la mayoría de los practicantes de la ciencia de datos es "la transformación de la información usando matemáticas y estadística en ideas valiosas, decisiones y productos" (Foreman, 2013).

En la actualidad la ciencia de datos es utilizada para obtener conocimiento a partir de la información con que se cuenta y con ello, apoyar la toma de decisiones. Sin embargo, Ozdemir (2016:4) menciona que la ciencia de datos no reemplaza el cerebro humano sino que lo complementa, esta nueva ciencia no debe ser tratada como una solución final sino como una opinión, una opinión muy informada, pero una opinión al fin.

La ciencia de datos basa su fortaleza de analizar información y ayudar a la mejor toma de decisiones en el uso de distintos tipos de algoritmos, entre los cuáles los más comunes son:

- Aprendizaje Automático
- Clasificación Binaria
- Regresión
- Gradient Boosting
- Random Forest

Estos algoritmos y técnicas son presentados con mayor detalle a continuación.

2.3 Aprendizaje automático

El aprendizaje automático es definido según Murphy (2012:1) como "un conjunto de métodos que pueden automáticamente detectar patrones en la información y usar los patrones descubiertos para predecir la información futura o realizar otro tipo de tomas de decisiones bajo incertidumbre". Estos métodos utilizan la información recolectada para mejorar su rendimiento, basándose en el aprendizaje que obtienen del proceso automático de extraer patrones en la información.

Un ejemplo de la aplicación d este tipo de algoritmos en la vida cotidiana es la predicción del clima. Varios aspectos clave de la información histórica son: la presión atmosférica, la temperatura, el punto de rocío, entre otros son utilizados (Saba, Rehman y AlGhamdi, 2017) para entrenar los modelos de aprendizaje automático en la búsqueda de patrones en la información y de esta manera hacer predicciones acerca del clima futuro basados en información pasada.

2.3.1 Terminología clave en aprendizaje automático

Antes de presentar una definición formal de los distintos algoritmos de aprendizaje automático, es importante conocer algunos de los conceptos básicos relevantes que son empleados en la descripción de dichos algoritmos. En este apartado se presenta una definición de cada uno de estos conceptos.

Información. Es la fuente principal del aprendizaje automático (Gollapudi,2016:66). Puede estar en cualquier formato (imágenes, texto, información geoespacial, datos numéricos o inclusive audio) y ser de cualquier tamaño, desde el orden de kilobytes hasta de peta bytes. Por lo general cuando se trabaja con información en el aprendizaje automático, ésta se trabaja en forma de filas y columnas (Provost y Fawcett, 2016:38).

Conjunto de datos. Es una colección de muestras. En el contexto del aprendizaje automático supervisado hay diferentes tipos de conjuntos de datos usados para diferentes propósitos. Un algoritmo se ejecuta con diferentes conjuntos de datos en diferentes etapas para medir la exactitud del modelo. Existen dos tipos de conjuntos de datos: conjunto de datos de entrenamiento y conjunto de datos de prueba (Hackeling,2014:9). De acuerdo con los autores (Zinoviev, 2016; Raschka, 2015; Coelho y Richert, 2015) el conjunto de datos utilizado en el aprendizaje automático supervisado es dividido a una razón de 70% para el conjunto de entrenamiento y el otro 30% para el conjunto de datos de prueba.

Instancia. Se le conoce como instancia a cada punto de entrada de un conjunto de datos (Müller y Guido, 2017:4). Por ejemplo, las instancias de una base de datos serían cada fila que constituye una tabla.

Muestra. Se puede definir una muestra como un subconjunto de las unidades de tamaño n del conjunto de datos (Schutt y O'Neil, 2014:20). De acuerdo con Dean (2014:61), la muestra debe ser lo suficientemente grande para contener información relevante pero lo suficientemente pequeña para manipularla rápidamente.

Características o Atributos. Son unidades de información que describen las instancias (Müller y Guido, 2017:4). Así mismo se les conocen como predictores a las características que son usadas para hacer predicciones (Gollapudi, 2016:66).

Dimensión. Es un conjunto de atributos usados para describir una propiedad de la información (Gollapudi, 2016:66). Por ejemplo, la dimensión de un rectángulo consiste de los atributos: largo y altura.

Conjunto de datos de entrenamiento. Es la colección de datos que se utilizan para construir el modelo de aprendizaje. El conjunto de datos de entrenamiento comprende dos tipos de información: Los resultados que se quieren predecir y las características disponibles para hacer la predicción (Bowles, 2015:76).

Conjunto de datos de prueba. Es el conjunto de datos que ayudan a evaluar el rendimiento del modelo. Es importante que no se incluyan muestras del conjunto de datos de entrenamiento en el conjunto de datos de prueba, si esto sucede el algoritmo memorizará la información del conjunto de datos de entrenamiento y fallará en predecir los valores de nuevas muestras (Hackeling, 2014:11).

Conjunto de datos de validación. Son los datos usados como verificación final del modelo, los cuales pueden ser tratados como una prueba de aceptación de usuario (Gollapudi, 2016:66).

Variable objetivo. Es lo que se intenta predecir en los algoritmos de aprendizaje automático (Harrington,2012:9). Una variable objetivo reduce las tareas de búsqueda en la información para sólo enfocarse en encontrar algún patrón que describa el comportamiento de la variable objetivo.

Tipos de características. De acuerdo con Julian (2016:150) existen tres tipos de características: categóricas, ordinales y cuantitativas:

- **Características categóricas:** Son aquellas características que pueden tomar sólo un valor de un número limitado de valores. Algunas veces llamadas características nominales (Julian,2016:150) no tiene orden y por lo tanto no es posible utilizar un

resumen estadístico de sus valores a excepción de la moda. Ejemplo: El género (Hombre o Mujer).

- **Características Ordinales:** Son aquellas características que posee un orden y son representadas como número enteros. Ejemplo: Una escala de Likert.
- **Características cuantitativas:** Son aquellas características continuas que involucran números reales a los cuales se les puede aplicar estadística como la media o la desviación estándar. Ejemplo: El ingreso anual.

Modelo. Un modelo es el resultado de los algoritmos aplicados a un conjunto de datos (Gollapudi,2016:68).

Sobreajuste. Memorizar el conjunto de datos de entrenamiento es llamado sobreajuste (*Overfitting* en Inglés). Sobreajuste significa que el modelo ha sido entrenado para ajustarse a un conjunto de datos de entrenamiento muy bien, pero no generaliza bien al predecir resultados de observaciones no usados en el conjunto de datos de entrenamiento (Dean,2014:97).

Sobre-generalización. A diferencia que el sobreajuste, la sobre-generalización (*Underfitting* en Inglés) ocurre cuando el modelo no funciona bien incluso en el conjunto de datos de entrenamiento (Grus, 2015:143).

Compensación Sobreajuste/Sobregeneralización. Entre más complejo sea el modelo mejor será para predecir usando el conjunto de datos de entrenamiento, pero si se vuelve muy complejo no generalizará bien la nueva información. Esta compensación (*trade-off* en Inglés) se traduce en términos de sesgo (*Bias* en Inglés) y varianza. La varianza mide la consistencia o variabilidad de la predicción del modelo para una muestra particular, el sesgo mide que tan alejada esta la predicción de los valores correctos (Raschka,2015:66). Si un modelo sufre de sobreajuste, se dice que el modelo tiene una alta varianza esto debido a que tiende a ser muy complejo teniendo muchos parámetros. Por otro lado, si un modelo sufre de sobregeneralización tiende a tiene un alto sesgo.

De acuerdo con Marsland (2015:35) los modelos más complejos tienen a mejorar el sesgo de los datos, pero al costo de una varianza mayor, mientras haciendo el modelo más específico reduce la varianza, pero incrementa el sesgo. No es posible reducir a cero el valor del sesgo y la varianza, por lo tanto encontrar un balance entre el sobreajuste y la sobregeneralización es una tarea que consiste en encontrar un punto clave entre estos dos conceptos. La Figura 1 muestra la compensación existente entre la exactitud y la complejidad del modelo.

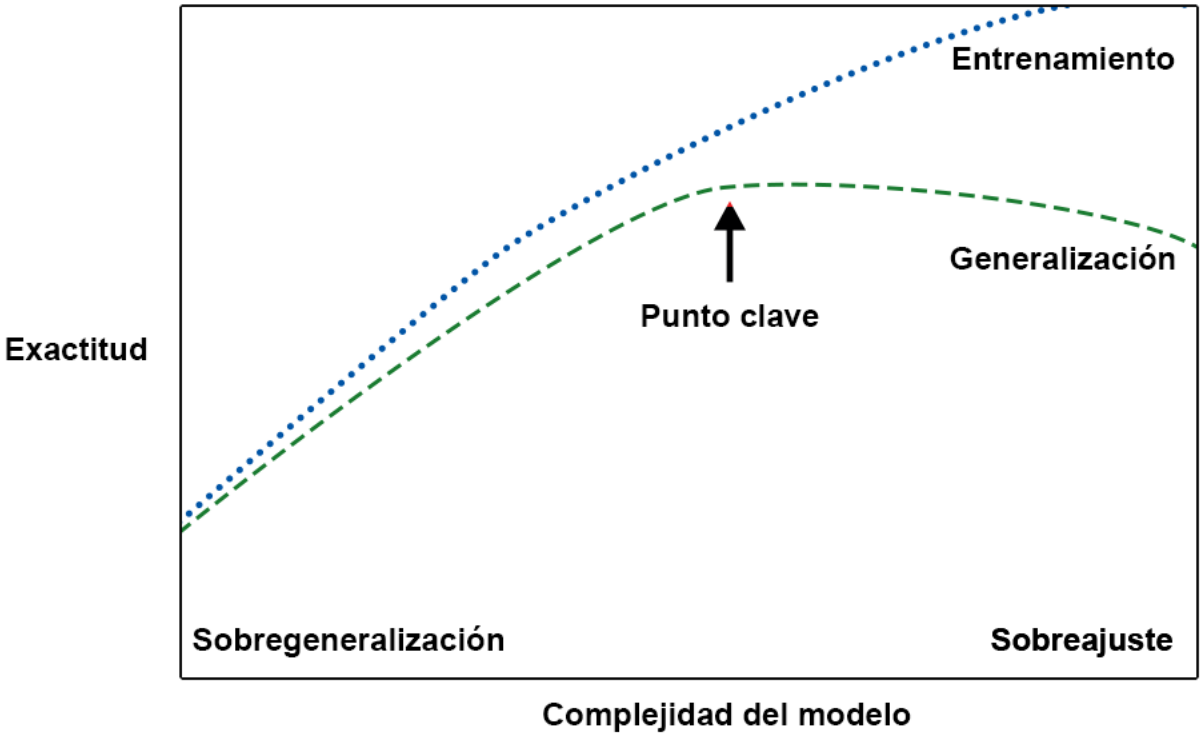

Figura 2. Compensación entre la complejidad del modelo contra la exactitud del conjunto de datos de entrenamiento y el conjunto de datos de prueba. Adaptado de "Introduction to Machine Learning with Python" por Müller A. y Guido S., 2017, p. 29. Copyright 2017 O'Reilly.

Verdadero Positivo. Con la analogía de una alarma contra incendios, si la alarma es activada en caso de incendio es un verdadero positivo. En este ejemplo, el incendio es positivo y la predicción hecha por el sistema es verdadera.

Falso Positivo. Si la alarma es activada y no hay incendio se trata de un falso positivo. En este caso el sistema predijo que el fuego sería positivo, pero hizo una predicción errónea, por lo que la predicción es falsa.

Verdadero Negativo. Similarmente, si la alarma no se activa y no hay fuego, se trata de un verdadero negativo. El fuego es negativo y la predicción es verdadera.

Falso Negativo. Finalmente, si la alarma no es activada, pero hubo un incendio se trata de un falso negativo. El sistema predijo que el fuego sería negativo, lo cual fue falso desde que en realidad hubo un incendio.

Exactitud. Del Inglés *Accuracy,* es el número correcto de predicciones (**V**erdaderos **P**ositivos y **V**erdaderos **N**egativos) dividido por el número de muestras (Müller y Guido, 2017:282). Es calculada con fórmula de la Ecuación 1:

$$Exactitud = \frac{VP + VN}{VP + VN + FP + FN}$$

Ecuación 1. Fórmula para calcular la exactitud en un modelo de aprendizaje automático.

En otras palabras, la exactitud es el número de predicciones correctas (**V**erdaderos **P**ositivos y **V**erdaderos **N**egativos) dividido entre el número de todas las instancias.

Precisión. Es la fracción de registros que fueron positivos del grupo que el algoritmo predijo que fueran positivos (Hackeling, 2014:15). Esta métrica ayuda a limitar el número de falsos positivos. Es calculado con la fórmula de la Ecuación 2:

$$Precisión = \frac{VP}{VP + FP}$$

Ecuación 2. Fórmula para calcular la precisión en un modelo de aprendizaje automático.

Exhaustividad. Del Inglés *Recall,* es la fracción de registros positivos que el clasificador predijo correctamente (Harrington,2012:164). Esta métrica ayuda a identificar todos los registros positivos cuando es importante evitar falsos negativos. Es calculado con la fórmula de la Ecuación 3:

$$Exhaustividad = \frac{VP}{VP + FN}$$

Ecuación 3. Fórmula para calcular la exhaustividad en un modelo de aprendizaje automático.

La exhaustividad es usada como una métrica de rendimiento cuando se necesita saber todas las instancias positivas, lo que sirve para evitar falsos negativos.

Compensación Precisión-Exhaustividad. Se puede construir un clasificador que logre una alta precisión y una alta exhaustividad, pero no ambos. Predecir todas las muestras como positivas resultará en muchos falsos positivos por lo tanto la exhaustividad será alta pero la precisión será mala, crear un clasificador que maximice tanto precisión como exhaustividad es un reto (Harrington,2012:144).

Raíz del Error Cuadrático Medio. La Raíz del Error Cuadrático Medio (Root Mean Square Error en Inglés) puede ser definida como lo muestra la Ecuación 4**:**

$$RMSE = \sqrt{\frac{1}{N}\sum_{n=1}^{N}(I_{pred,n} - I_{meas,n})^2}$$

Ecuación 4. Fórmula para calcular la raíz del error cuadrático medio.

Donde, $I_{pred,n}$ es el enésimo valor predicho, $I_{meas,n}$ es el enésimo valor medido y N es el número total de observaciones. La raíz del error cuadrático medio siempre es positiva, un valor cero es ideal. Esta métrica proporciona información del rendimiento de la correlación al ordenar una comparación de la desviación actual entre los valores calculados y los medidos.

Entre más pequeño sea el valor, mejor es el rendimiento del modelo (Maroof, Jamil y Jamil, 2016).

Validación cruzada. Es un método para evaluar el rendimiento de la generalización del modelo. Consiste en particionar el conjunto de datos de entrenamiento, el algoritmo es entrenado usando todas las particiones excepto una y probado en las particiones restantes. Las particiones son rotadas varias veces así el algoritmo es entrenado y evaluado en todo el conjunto de datos (Hackeling, 2014:12). El diagrama de la Figura 3 muestra el proceso de validación cruzada para cinco particiones.

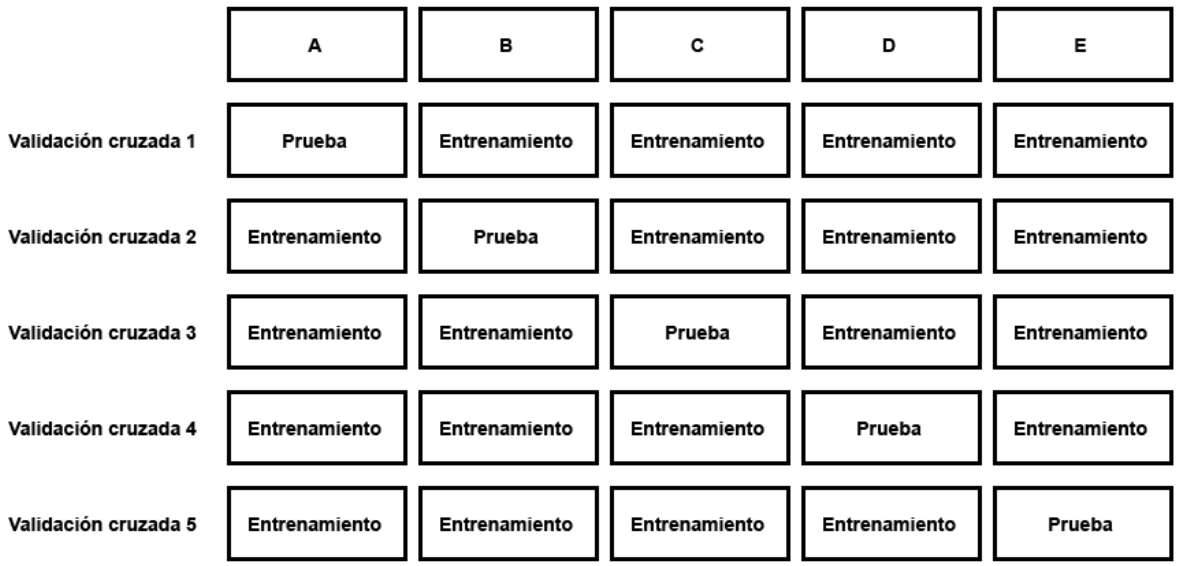

Figura 3. Validación cruzada para cinco particiones. Adaptado de "Mastering Machine Learning with scikit-learn" por Hackeling G., 2014, p. 12. Copyright 2014 Packt.

El conjunto de datos original se particiona en cinco subconjuntos de tamaños iguales A,B,C,D,E. En la primera validación el modelo es entrenado en las particiones B,C,D, y E y probado en la partición A. En las siguientes iteraciones de la validación cruzada, las particiones son rotadas hasta que el modelo ha sido entrenado y probado en todas las particiones. La validación cruzada ofrece una estimación más precisa en el rendimiento del modelo que sólo probar en una sola partición del conjunto de datos.

Ingeniería de características o *Feature engineering*. Es un proceso iterativo, para seleccionar características, hacer combinaciones entre ellas y determinar su potencial óptimo para hacer predicciones. Representar la información de la manera correcta puede tener una influencia mayor en el rendimiento de un modelo supervisado que los parámetros que se escojan (Coelho & Richert, 2015:42).

Boruta. Es una biblioteca construida a partir del algoritmo *Random Forest* que evalúa la importancia de las características creando variables "sombra" artificiales de una muestra aleatoria del conjunto de datos (Poona & Ismail, 2013). La importancia es asignada comparando las variables originales con las aleatorias. Se utilizan modelos *Random Forest* iterativamente hasta que las variables con clasificadas como **C**onfirmadas, **R**echazadas o **T**entativas.

2.3.2 Clasificación de los algoritmos del aprendizaje automático

En esta sección se presenta una descripción detallada de los algoritmos de clasificación del aprendizaje automático, los cuales se dividen en categorías:

1. Aprendizaje supervisado,
2. Aprendizaje no supervisado y
3. Aprendizaje por reforzamiento.

2.3.2.1 Aprendizaje supervisado

El aprendizaje supervisado según Ozdemir (2016:206) es una técnica que "encuentra asociaciones entre las características de un conjunto de datos y una variable objetivo (...) estas asociaciones le permiten al aprendizaje supervisado hacer predicciones basados en información pasada". El término supervisado es utilizado debido a que los resultados (la variable objetivo) son conocidos o supervisados en el conjunto de datos (Raschka, 2015:3), (Müller y Guido,2017:2). De acuerdo con Brink, Richards Y Fetherolf (2017:78), la meta principal del aprendizaje supervisado es una predicción exacta. Se desea que el modelo sea lo más exacto posible cuando nueva información es presentada para la variable objetivo es desconocida, de este modo cuando sea lanzado a producción se tenga la seguridad que las predicciones del modelo son confiables. Sin embargo, una de las principales desventajas del aprendizaje supervisado como lo menciona Ozdemir (2016:208) es la necesidad de información etiquetada lo cual es costoso, y aunado a esto, Marsland (2015:10) menciona, que el aprendizaje supervisado requiere la intervención de expertos en el ámbito que se este desarrollando el modelo para construir el conjunto de datos de entrenamiento y una inversión considerable de tiempo.

2.3.2.2 Aprendizaje no supervisado

El aprendizaje no supervisado es aquel según Marsland (2015:6) donde "las respuestas correctas no son proporcionadas, sino en su lugar el algoritmo intenta identificar similitudes entre la información de entrada de tal manera que al tener algo en común son categorizadas

en conjunto.". Este tipo de aprendizaje se asemeja más el de un ser humano, como menciona Murphy (2012:9) al compararlo con el aprendizaje supervisado, debido a que los humanos descubren la estructura del mundo observando y no se tiene un banco de información con las características de cada objeto. Una de las ventajas del aprendizaje no supervisado es la ausencia de información etiquetada. Ozdemir (2016:213) enfoca esa ventaja a la facilidad para obtener información de una manera más fácil y económica. Así mismo, Murphy (2012:10) menciona que la información etiquetada contiene además poca información y ciertamente no es confiable para modelos complejos. Por otro lado, la principal desventaja de este tipo de aprendizaje según los autores Ozdemir(2016:213), Mohri, Rostamizadeh y Talwalkar (2012:7) es que no existe un conjunto de datos con los resultados esperados por lo que evaluar el rendimiento del algoritmo resulta difícil al no poder comprar las respuestas del modelo con las respuestas correctas, lo que requiere la interpretación de un humano.

2.3.2.3 Aprendizaje por reforzamiento

El aprendizaje por reforzamiento es descrito por Raschka (2015:6) como "el desarrollo de un sistema (agente) que mejora su rendimiento basado en interacciones con el entorno". El entorno es el encargado de proporcionar información acerca de que tan buena es la estrategia través de una función de recompensa (Marsland, 2015:231). Como mencionan los autores Mohri, Rostamizadeh y Talwalkar (2012:7) la función del agente es maximizar la recompensa a través de un curso de acciones e iteraciones con el entorno. Esto significa que las interacciones con el ambiente se basan en prueba y error, lo cuál, es una de las principales diferencias con el aprendizaje supervisado debido a que el agente desconoce qué acciones están bien y cuales mal (Kulkarni, 2012:32), la recompensa como menciona Ozdemir (2016:214) simplemente alienta (o desalienta) diferentes acciones. Por lo tanto, como describe el Gollapudi (2016:78) la meta principal del aprendizaje por reforzamiento es medir la compensación entre explorar y explotar la información. Este dilema, como lo llaman los autores Mohri, Rostamizadeh y Talwalkar (2012:7), sitúa al agente entre explorar acciones desconocidas para obtener más información contra explotar la información ya obtenida.

2.4 Clasificación Binaria

Los algoritmos de clasificación son comúnmente usados para clasificar elementos en uno o varios grupos. La clasificación binaria es el tipo más común de modelo predictivo (Dean,2014:64). El objetivo de los algoritmos de clasificación según Murphy (2012:3) es aprender a trazar las entradas x en salidas y, donde $y \in \{1, \dots, C\}$, con C es el número de

clases. Si C = 2, se trata de clasificación binaria, si C > 2 es se trata de una clasificación multiclase.

Una definición más formal de clasificación binaria es expuesta por Zhou & Lai (2009) de la siguiente manera: suponga $x = (x_1, x_2, \dots, x_m)^T$ es un conjunto de m variables variables que describe una instancia. El valor de las variables m para una instancia particular k puede ser denotado como $x_k = (x_{k1}, x_{k2}, \dots, x_{km})^T$. Una función $f(x)$ puede ser construida de un grupo de muestras de entrenamiento seleccionadas $\{x_{k,}y_k\}_{k=1}^N$, con una entrada de datos $x_k \in R^m$ y sus correspondientes salidas $y_k \in R$ y $y_k \in \{1, -1\}$, lo cual traza las características para medir la probabilidad de clasificarla en una clase. El resultado de una instancia es denotada por y es terminado por el valor de $f(x)$ y valor de corte f_c. El valor de y puede ser determinado por la Ecuación 5:

$$y = \begin{cases} +1, si\ f(x) \geq f_c \\ -1, de\ lo\ contrario \end{cases}$$

Ecuación 5., Fórmula de Clasifinación Binaria

2.5 Regresión

Es una técnica de aprendizaje automático para la predicción de valores numéricos sean continuos o categóricos. El objetivo de la regresión es predecir un valor numérico de un conjunto de observaciones sin etiquetar. De acuerdo con Das (2016), se trata de un enfoque para expresar la relación entre una variable objetivo (dependiente) y una o más variables independientes (predictores).

La relación derivada ayuda a predecir los valores desconocidos de la variable objetivo. Algunas de las técnicas de regresión más comúnmente usadas incluyen regresión lineal, árboles de decisión, redes neuronales y gradient boosting (Barga, Fontama, Tok, 2014:16). Una definición formal por Birks, Lotter, Juggins, et al. (2012) es mostrada en la Ecuación 6:

$$y = E_y + \varepsilon$$

Ecuación 6. Fórmula simple de Regresión.

Donde y es el valor de la variable de objetivo, E_y es el valor esperado de la variable respuesta para los valores particulares de las variables predictoras, y ε es la variabilidad de los valores observados reales de y alrededor de los valores esperados E_y, llamado el error aditivo aleatorio.

Los valores esperados de la variable respuesta son descritos como una función f, de las variables predictoras $x_1, \dots, x_q$

$$E_y = f(x_1, \dots, x_q)$$

Ecuación 7. Fórmula de Regresión para las variables predictoras.

La parte E_y es llamada la parte sistemática o componente sistemático y ε es el la parte error o componente del modelo de regresión.

2.6 Gradient Boosting

Gradient Boosting es una técnica de aprendizaje automático para la construcción de modelos de clasificación y regresión (Friedman, 2001). El término gradient se refiere a la optimización del algoritmo durante el periodo de aprendizaje del modelo (Aler, Galván, Ruiz, et al., 2017). Como mencionan los autores Lu, Zhang, Bi, et al. (2016), la teoría básica de *Gradient Boosting* es producir un modelo predictivo construido por un conjunto de aprendices débiles o base, comúnmente árboles de decisión donde cada árbol crece secuencialmente usando la información del árbol anterior. Una definición formal del algoritmo de Gradient Boosting según menciona los autores Laradji, Alshayed y Ghouti (2014) es la siguiente: Dado un conjunto de árboles de decisión, $\{DT_1, DT_2, \dots, DT_n\}$, el algoritmo produce una suma ponderada de la salida de cada árbol de decisión como se aprecia en la fórmula de la Ecuación 8.

$$f(x) = w_0 + w_1 h_1(x) + w_2 h_2(x) + \cdots + w_x h_x(x)$$

Ecuación 8. Formula de Gradient Boosting.

Donde $h_i(x)$ es la salida de la decisión del enésimo árbol individual. Los pesos, w_i, aplicados en las decisiones individuales son optimizados minimizando una función de pérdida diferenciable.

2.7 Random Forest

Random Forest es una técnica de aprendizaje automático introducida por Breiman en 2001. La idea principal detrás *Random Forest* como mencionan Gondane y Devi (2015) es combinar muchos árboles de decisión usando un subconjunto de muestras de entrenamiento y escoger la mejor característica como punto de partida a través del conjunto aleatorio de características de cada nodo dividido. *Random Forest* puede ser utilizado para realizar predicciones sobre un valor nominal (Clasificación) o un valor numérico (Regresión) (Vens, 2013). Una definición formal es descrita por Huynh, Gao, Kang et al. (2015) se muestra a continuación. Sea $u \in \subset R^q$ un vector característica de entrada y $v \in V \subset Z$ sea su variable objetivo correspondiente

en el problema de clasificación. Para un conjunto de muestras $S_j \subset U \times V$ que llegan al nodo j, la ganancia de la información obtenida por escoger la k-enésima característica es calculada por la Ecuación 9:

$$I_j^k = H(S_j) - \frac{|S_{j,L}^k|}{|S_j|} H(S_{j,L}^k) - \frac{|S_{j,R}^k|}{|S_j|} H(S_{j,R}^k),$$

$$H(S) = -\sum_v p_v \log(p_v),$$

Ecuación 9. Fórmula para obtener la ganancia obtenida en Random Forest.

Donde L y R denotan los nodos hijos izquierdo y derecho, $S_{j,L}^k = \{(u, v) \in S_j | u^k < \theta_j^k\}$, $S_{j,R}^k = S_j \backslash S_{j,L}^k, u^k$ es la k-enésima característica en u, θ_j^k es el umbral de particionamiento escogido para maximizar la ganancia de la información I_j^k, y $|\cdot|$ es la carnalidad del conjunto. $H(S)$ denota la entropía de los valores objetivos en S, con p_v la fracción de los elementos en S teniendo valor en v. Para un problema de regresión $(V \subset R)$, la entropía es remplazada por la varianza de por la Ecuación 10:

$$H(S) = \sum_v p_v (v - \bar{v})^2,$$

$$\bar{v} = \sum_v p_v{}^v$$

Ecuación 10. Fórmula para obtener la ganancia obtenida en Random Forest utilizando la varianza.

2.8 XGBoost

XGBoost (eXtreme Gradient Boosting) es un proyecto de código abierto desarrollado inicialmente por Chen y Guestrin en un estudio realizado en la Universidad de Washington (Chen & Guestrin, 2016). XGBoost fue diseñado para optimizar el algoritmo Gradient Tree Boosting haciéndolo altamente eficiente, flexible y portable (Darmatasia & Arymurthy, 2016), capaz de resolver problemas de regresión y clasificación utilizando árboles. XGBoost goza de popularidad dentro de la comunidad científica por su gran desempeño frente a otros algoritmos (Brown & Mues, 2012).

2.9 El proceso de la ciencia de datos

Todo proyecto de ciencia de datos se conforma de una serie de pasos, los cuales, se realizan de forma iterativa. No todos los proyectos de ciencia de datos siguen el siguiente proceso, en algunas empresas se deberá seguir un orden estricto ya establecido y en otras habrá más libertad para desarrollar un proyecto (Cielen, Meysman y Ali, 2016:24) sin embargo, la serie de pasos que se presentan en la Figura 4 son una guía general que puede ser utilizada como punto de partida por aspirantes a científicos de datos.

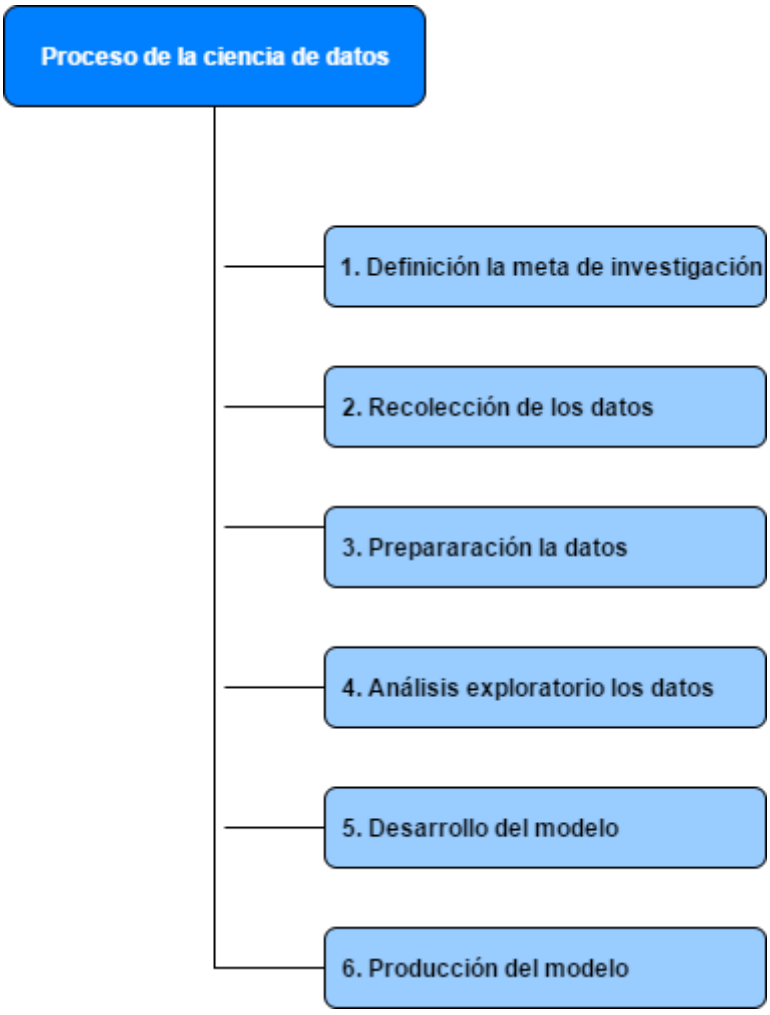

Figura 4. El proceso de la ciencia de datos.

Definición de la meta de investigación. Antes de construir cualquier modelo, es importante tener una meta de investigación, un buen entendimiento del contexto del proyecto, entregables bien definidos y un plan de acción con un calendario (Cielen, Meysman y Ali, 2016:24). Para ello es necesario como mencionan Barga, Fontama y Tok (2015:11) trabajar

muy de cerca con la persona u organización que requiere de un proyecto de esta naturaleza, y de esta manera, lograr identificar lo que se intenta resolver y así evitar trabajar meses en un problema mal comprendido.

Recolección de los datos. En muchas ocasiones la información ya se encuentra almacenada en las bases de datos de las empresas y este paso no es necesario. Cuando este no es el caso, la información puede ser adquirida de diversas fuentes externas ya sean abiertas o privadas.

Preparación de los datos. La limpieza de la información es considerada la tarea que más tiempo consume del proceso de la ciencia de datos, en un proyecto de ciencia de datos típicamente del 75 al 80% del tiempo es invertido en la preparación de la información (Barga, Fontama y Tok , 2015:12). Este paso consiste en remover errores de la información para que sea verdadera y consistente (Cielen, Meysman y Ali, 2016:30). Si la información proviene de diversas fuentes, será necesario combinar las diversas fuentes para mantener una misma estructura. Finalmente, diversos modelos de aprendizaje automático requieren que la información esté una forma en particular, para ello en algunas ocasiones se necesita transformar la información y en otras reducir el número de variables.

Análisis exploratorio de los datos. De acuerdo con los autores Cielen, Meysman y Ali (2016:43), la información se vuelve mucho más fácil de entender cuando se ve en una imagen. Este paso consiste en usar diversas técnicas de visualización de datos para tener un entendimiento más claro de cómo interactúan las características en la información.

Desarrollo del modelo. Después del paso de análisis exploratorio de los datos, se obtienen las características que se usarán en el modelo de aprendizaje automático. El desarrollo del modelo es un proceso iterativo, donde se experimenta con diferentes modelos para encontrar el más predictivo (Barga, Fontama y Tok 2015:12). Para ello es necesario evaluar la exactitud del modelo lo cual requiere la intervención de expertos en el área (Marsland, 2015:12)

Producción del modelo. Este es el último paso del proceso y consiste en entregar el modelo al ambiente de producción de la empresa. Según mencionan los autores Barga, Fontama y Tok (2015:12), cuando el modelo se encuentra en producción es necesario tener una constante supervisión de su rendimiento ya que con el tiempo éste se degrada por diversas de razones. Cuando el rendimiento se degrada significativamente puede ser necesario simplemente desarrollar de nuevo el modelo o empezar de nuevo el proceso de la ciencia de datos.

Capítulo 3: Propuesta de Solución

3.1 Introducción

En este capítulo se describe el experimento realizado para identificar clientes potenciales a partir de sus características socioeconómicas y demográficas. Esto, a través de la implementación del proceso de la ciencia de datos (Cielen y Meysman, Ali, 2016:24) así como el análisis e interpretación de los resultados obtenidos. Se presenta en primera instancia el origen y características de los datos con los que se trabajarán en el experimento. Enseguida se describe el proceso de limpieza y transformación de los datos que servirán como entrada para el modelado de la solución de aprendizaje automático. Luego, se describe la exploración de la información utilizando diferentes técnicas de visualización y estadística. Finalmente, dos modelos de aprendizaje automático son empleados: un modelo de clasificación binaria y un modelo de regresión para dar solución a la problemática presentada.

3.1.1 Descripción del problema

Para promover la venta de sus productos por medio de la identificación de clientes potenciales la organización Adventure Works Cycles ha decidido realizar el análisis de la información histórica con que cuenta utilizando técnicas de aprendizaje automático. Particularmente la organización está interesada en analizar la información de sus clientes para determinar:

- Cualquier relación aparente entre las características demográficas conocidas de sus clientes y la probabilidad de que un cliente compre una bicicleta.
- Conocer si es posible predecir el gasto mensual promedio del cliente en la empresa a partir de sus características.

Los puntos anteriores abordan la definición de la meta de investigación, primera fase el proceso de Ciencia de Datos como lo mencionan los autores (Cielen, Meysman y Ali, 2016:24).

3.1.2 Conjunto de datos

Siguiendo el proceso de la Ciencia de Datos, la recolección de los datos se realizó utilizando el conjunto de datos proporcionados por el Data Science Professional Project (DAT102x), el cual es requisito final para la certificación Microsoft Professional Program for Data Science. Este conjunto de datos ha sido generado de forma sintética con la finalidad de simular un

gran volumen de registros correspondientes a los clientes de la organización, se incluyen características demográficas e información de sus compras.

El conjunto de datos descargado del sitio Web del proyecto contiene los siguientes dos archivos:

- **AWCustomers.csv** – Información de los clientes utilizada en Adventure Works Cycles. (18,361 registros y 24 columnas).
- **AWSales.csv** - Información de ventas existentes en Adventure Works Cycles. (18,355 registros y 3 columnas)

3.2 Preparación de la información

La tercera fase del proceso de la Ciencia de Datos, la preparación de los datos fue realizada haciendo limpieza y transformación inicial del conjunto de datos con ayuda de la biblioteca Pandas (Wes McKinney, 2010), utilizada ampliamente para análisis y manipulación de datos (Bulut, 2016; Bloice & Holzinger, 2016; Unpingco, 2016) en Python, el lenguaje de programación más popular para la Ciencia de Datos (Puget, 2016; Raschka, 2015).

3.2.1 Limpieza de la información

3.2.1.1 Valores faltantes

El primer paso para la limpieza de la información es la identificación de los valores faltantes. Una evaluación con Pandas del conjunto de datos determinó que el archivo AWSales no contiene ningún valor faltante. Por otro lado, se encontró que el archivo AWCustomers carece de información para las siguientes características categóricas relacionadas con la identificación del cliente:

- **Title**
- **MiddleName**
- **Suffix**
- **AddressLine2**

Finalmente, se corroboró que todas las características demográficas del cliente esenciales están completas utilizando la biblioteca Pandas.

3.2.1.2 Registros duplicados

Cada cliente es identificado por un valor único del campo CustomerID en el conjunto de datos. Una revisión al archivo AWSales utilizando la biblioteca Pandas confirmó que no contiene ningún registro duplicado. Sin embargo, el análisis identificó seis filas con valores duplicados

para la columna CustomerID en el archivo AWCustomers. Cada uno de los seis valores CustomerID tenía dos registros; cada par con un valor de 2017-03-06 y 2017-03-07 para la columna LastUpdated.

Los registros más antiguos (LastUpdated 2017-03-06) fueron removidos del conjunto de datos con propósito del análisis de la información más actual. (Esos registros contenían cuatro números telefónicos distintos, un estado marital diferente, así como un ingreso anual distinto). La Tabla 1 muestra un resumen de los 12 registros encontrados como duplicados en la revisión.

Tabla 1. Resumen de los registros duplicados en AWCustomers.

CustomerID	FirstName	LastName	City	PhoneNumber	Marital Status	Income	LastUpdated
13385	Danielle	Cook	Issaquah	140-555-0112	Single	84367	2017/03/07
13385	Danielle	Cook	Issaquah	140-555-0112	Single	79123	2017/03/06
21647	Blake	Brown	Bremerton	687-555-0187	Single	33966	2017/03/07
21647	Blake	Brown	Bremerton	687-555-0185	Single	33966	2017/03/06
23192	Emily	Garcia	Palo Alto	300-555-0120	Married	82359	2017/03/07
23192	Emily	Garcia	Palo Alto	300-555-0122	Married	82359	2017/03/06
23770	Jamie	Carlson	Maidenhead	500-555-0165	Single	30526	2017/03/07
23770	Jamie	Carlson	Maidenhead	500-555-0163	Single	30526	2017/03/06
24334	Alexa	James	Novato	897-555-0196	Married	79332	2017/03/07
24334	Alexa	Jones	Novato	897-555-0196	Single	79332	2017/03/06
26829	Morgan	Evans	Bottrop	500-555-0110	Married	54675	2017/03/07
26829	Morgan	Evans	Bottrop	500-555-0119	Married	54675	2017/03/06

3.2.1.3 Información sobrante

Las siguientes columnas han sido excluidas del análisis del conjunto de datos debido a que presentan información irrelevante para el mismo debido a que son específicas para cada cliente y no se puede generalizar conocimiento a partir de ellas.

- **Title**
- **FirstName**
- **MiddleName**
- **LastName**
- **Suffix**
- **AddressLine1**
- **AddressLine2**
- **PhoneNumber**

3.2.2 Transformación de la información

3.2.2.1 Ingeniería de características

Para obtener más información útil a partir de las características en el conjunto de datos, se realizó el proceso de ingeniería de características (Coelho & Richert, 2015:42) para algunas variables. Las siguientes características calculadas fueron agregadas al conjunto de datos para propósitos de análisis:

- **Age:** Edad del cliente en años calculada a partir de la columna BirthDate y la fecha actual.
- **AgeGroup:** Cinco categorías de edades de los clientes en los rangos: <19, 19-25, 25-30, 30-50, >50

3.2.2.3 Unificación de conjuntos de datos

Después de la limpieza y manipulación inicial de la información, los conjuntos de datos AWCustomers y AWSales fueron unidos por medio de la columna en común CustomerID. Lo que resultó en un único conjunto de datos adecuado para el siguiente paso del proceso de la ciencia de datos; el análisis de los datos.

3.3 Análisis de los datos

Después de la limpieza y transformación de los datos, como cuarta fase del proceso de la Ciencia se Datos, el análisis exploratorio de los datos se realizó haciendo uso de la biblioteca

Pandas para el manejo del conjunto de datos y la biblioteca Seaborn para las visualizaciones de los mismos.

3.3.1 Estadística descriptiva

3.3.1.1 Características del conjunto de datos

Los conjuntos de datos fueron inspeccionados utilizando Pandas para determinar sus características como lo es la categoría y tipo de dato. Esta información se resume en la Tabla 2, Tabla 3 y Tabla 4.

Tabla 2. Descripción de los campos en AWCustomers.

#	Campo	Categoría	Tipo de dato	Descripción
1	CustomerID	Llave primaria	integer	Identificador único del cliente.
2	Title	Identidad	string	Título formal del cliente (Mr., Ms., Sr., Mrs., o Ms.) .
3	FirstName	Identidad	string	Primer nombre del cliente.
4	MiddleName	Identidad	string	Segundo nombre del cliente.
5	LastName	Identidad	string	Apellido del cliente.
6	Suffix	Identidad	string	Un sufíjo para el nombre del cliente (Jr, Sr, etc.).
7	AddressLine1	Geográfico	string	La primera línea de la dirección del cliente.
8	AddressLine2	Geográfico	string	La segunda línea de la dirección del cliente
9	City	Geográfico	string	La ciudad donde el cliente vive.
10	StateProvinceName	Geográfico	string	El estado o provincial donde el cliente vive.
11	CountryRegionName	Geográfico	string	El país donde el cliente vive.
12	PostalCode	Geográfico	string	El código postal para la dirección del cliente.
13	PhoneNumber	Geográfico	string	El número telefónico del cliente.
14	BirthDate	Demográfico	date	La fecha de nacimiento del cliente en el formato YYYY-MM-DD.

15	Education	Demográfico	string	El nivel más alto de educación alcanzado por el cliente (Partial High School, High School, Partial College, Bachelors, o Graduate Degree).
16	Occupation	Demográfico	string	El tipo de empleo del cliente (Manual, Skilled Manual, Clerical, Management, o Professional).
17	Gender	Demográfico	string	El género del cliente es hombre (M) o mujer (F).
18	MaritalStatus	Demográfico	string	Indica si el cliente es casado (M) o soltero (S)
19	HomeOwnerFlag	Demográfico	integer	Indica si el cliente tiene una casa (1) o no (0)
20	NumberCarsOwned	Demográfico	integer	Indica el número de carros que posee el cliente
21	NumberChildrenAtHome	Demográfico	integer	Indica el número de hijos que el cliente tiene viviendo en casa.
22	TotalChildren	Demográfico	integer	El número total de hijos del cliente.
23	YearlyIncome	Demográfico	decimal	El ingreso anual del cliente.
24	LastUpdated	Sistema	date	La fecha cuando el registro del cliente fue actualizado por última vez.

Tabla 3. Descripción de los campos en AWSales.

#	Campo	Categoría	Tipo de dato	Descripción
1	CustomerID	Llave foránea	integer	Identificador único del cliente.
2	AvgMonthSpend	Transaccional	decimal	La cantidad promedia de dinero que el cliente ha gastado en AWC cada mes
3	BikeBuyer	Transaccional	integer	Indica si el cliente ha comprador anteriormente una bicicleta (1) o no (0).

Tabla 4. Descripción de los datos adicionales calculados.

#	Campo	Categoría	Tipo de dato	Descripción
1	Age	Demográfico	decimal	Edad en años del cliente calculada a partir del campo BirthDate.
2	AgeGroup	Demográfico	string	Rangos del campo Age: <19, 19-25, 25-30, 30-50, >50.

3.3.1.2 Estadísticas individuales de las características

La exploración de la información inició aplicando estadística descriptiva a las características en el conjunto de datos. Resumen estadístico para máximo, media, mediana, mínimo y desviación estándar fueron calculados para las columnas numéricas mostradas en la Tabla 5:

Tabla 5. Resumen estadístico para las columnas numéricas en el conjunto de datos.

Columna	Máximo	Media	Mediana	Mínimo	Desviación Estándar	Valores Únicos
Age	86.6	35.2	33.6	16.3	11.25	8,230
NumberCarsOwned	5	1.27	1	0	0.914	6
NumberChildrenAtHome	3	0.34	0	0	0.569	4
TotalChildren	3	0.85	0	0	0.927	4
YearlyIncome	139,115	72,759	61,851	25,435	30,688	15,355
AvgMonthSpend	65.3	51.8	51.4	44.1	3.44	1,830

La columna ***AvgMonthSpend*** es de particular interés para este análisis como variable objetivo. La pequeña diferencia entre la media y la mediana indican que la distribución es relativamente simétrica.

La columna ***YearlyIncome*** tiene una media más grande que la mediana, lo cual implica una distribución sesgada. La desviación estándar muestra una variación grande en el ingreso de los clientes.

La columna ***Age*** tiene una media mayor a la mediana, lo que también implica una distribución sesgada. La desviación estándar de 11.25 muestra una varianza grande en la edad de los clientes.

3.3.2 Visualización de la información

3.3.2.1 Variables numéricas

Para verificar las afirmaciones de la columna **AvgMonthSpend**, **YearlyIncome** y **Age**, las distribuciones de **AvgMonthSpend, YearlyIncome** y **Age** fueron trazadas en la Figura 1, Figura 2 y Figura 3:

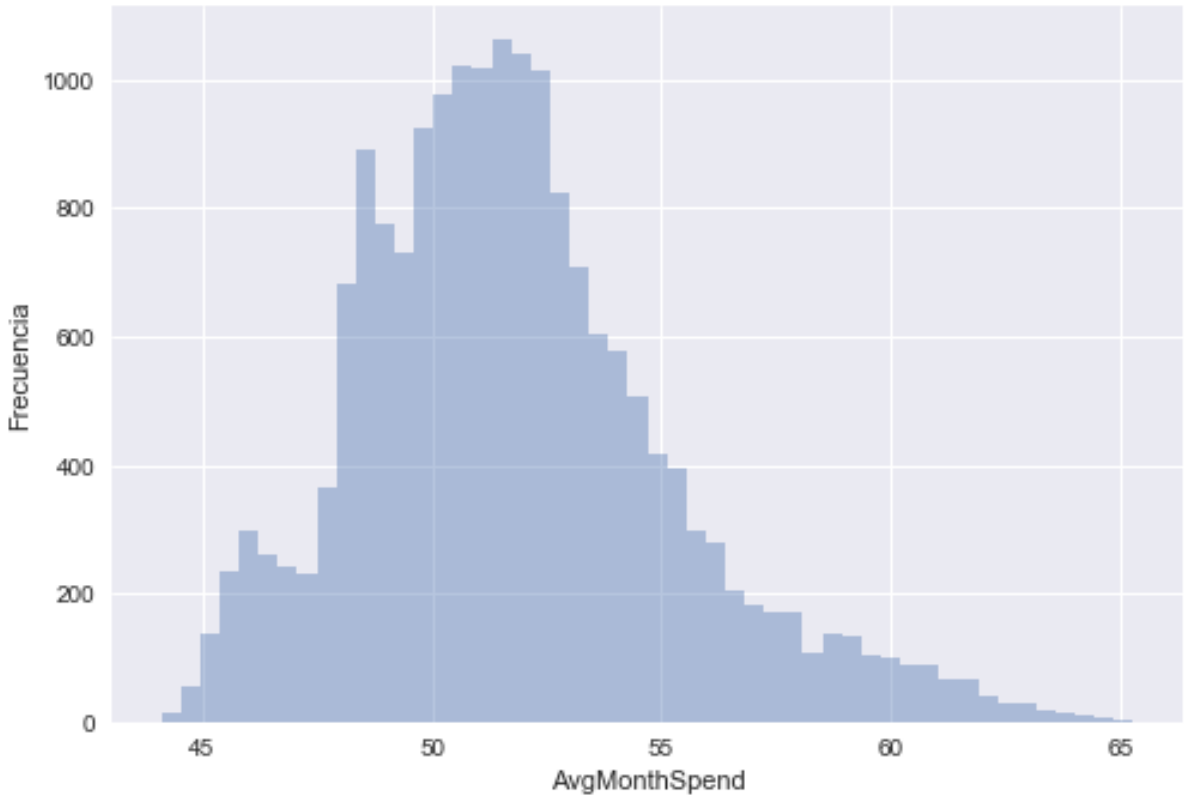

Figura 5. Distribución de AvgMonthSpend.

La Figura 5 muestra la distribución del gasto mensual promedio del cliente en la empresa un poco sesgada a la derecha, lo que representa la posible presencia de valores atípicos en los datos.

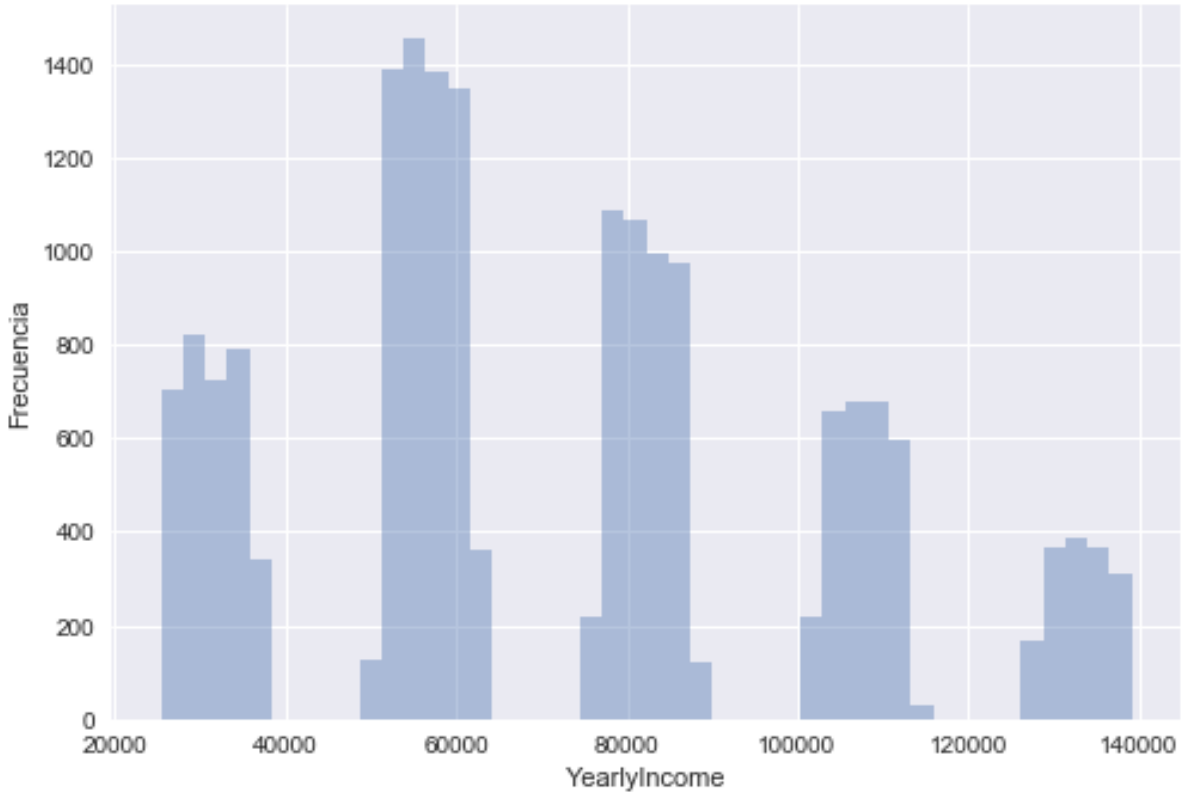

Figura 6. Distribución de YearlyIncome.

La Figura 6 representa la distribución del ingreso anual del cliente, generando 5 grupos, posiblemente por el tipo de empleo del cliente (Manual, Skilled Manual, Clerical, Management, o Professional).

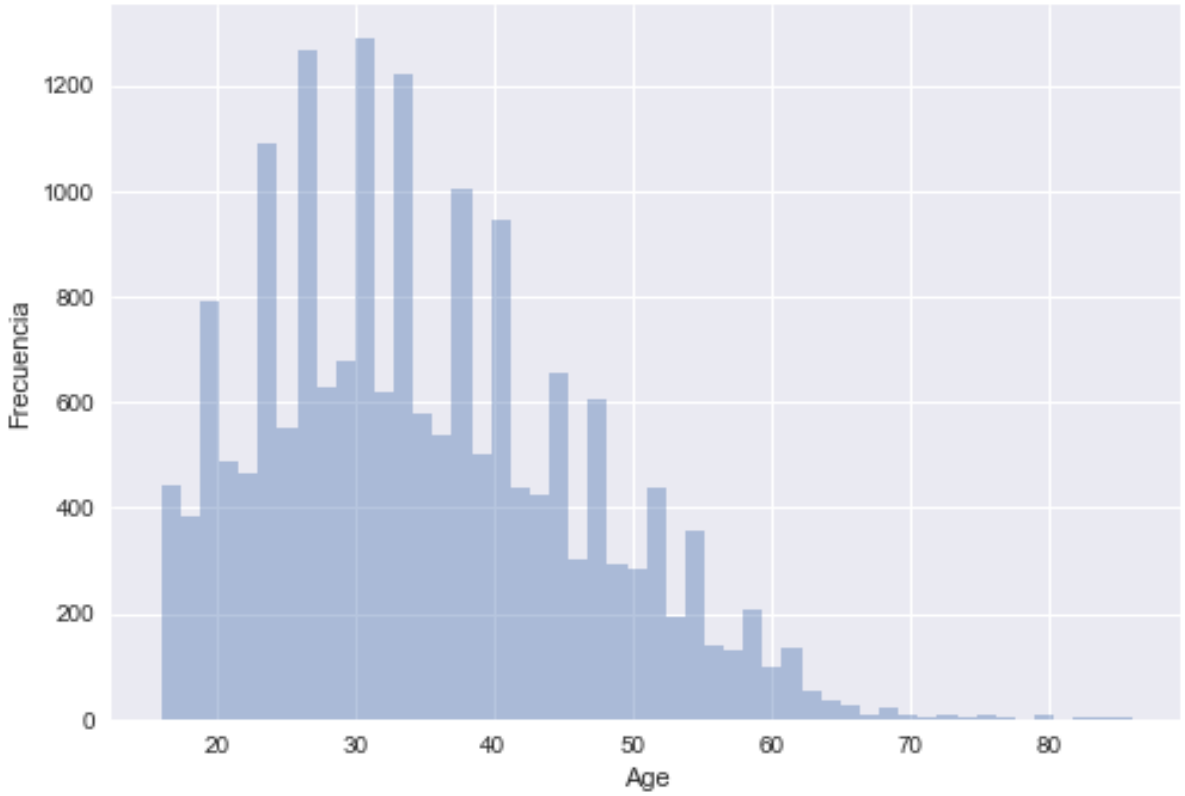

Figura 7. Distribución de la edad de los clientes.

La Figura 5 y Figura 7 confirman las afirmaciones de la distribución de de las comunas **AvgMonthSpend** y **Age**, excepto para **YearlyIncome**. De la distribución se pueden observar en la Figura 6 cinco categorías diferentes, este patrón sugiere que la característica fue sintetizada escogiendo cinco grupos y aplicando una variación aleatoria a los datos.

La Figura 8, Figura 9,Figura 10 y Figura 11 muestran la distribución del resto de las variables
numéricas.

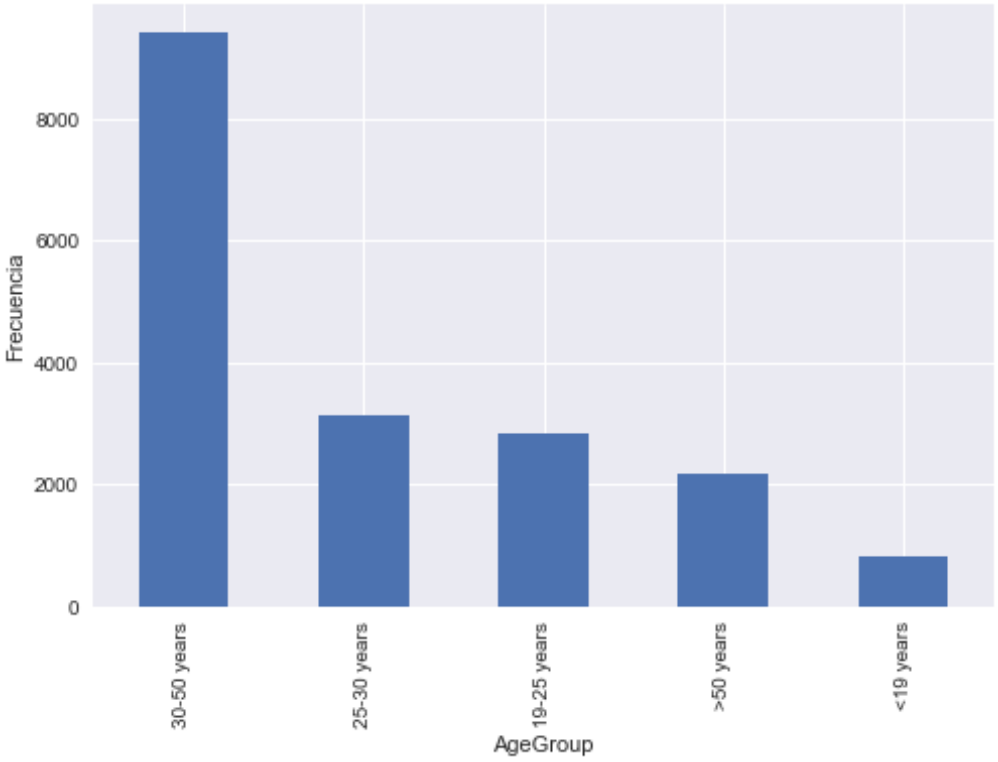

Figura 8. Distribución de la edad de los clientes por rangos.

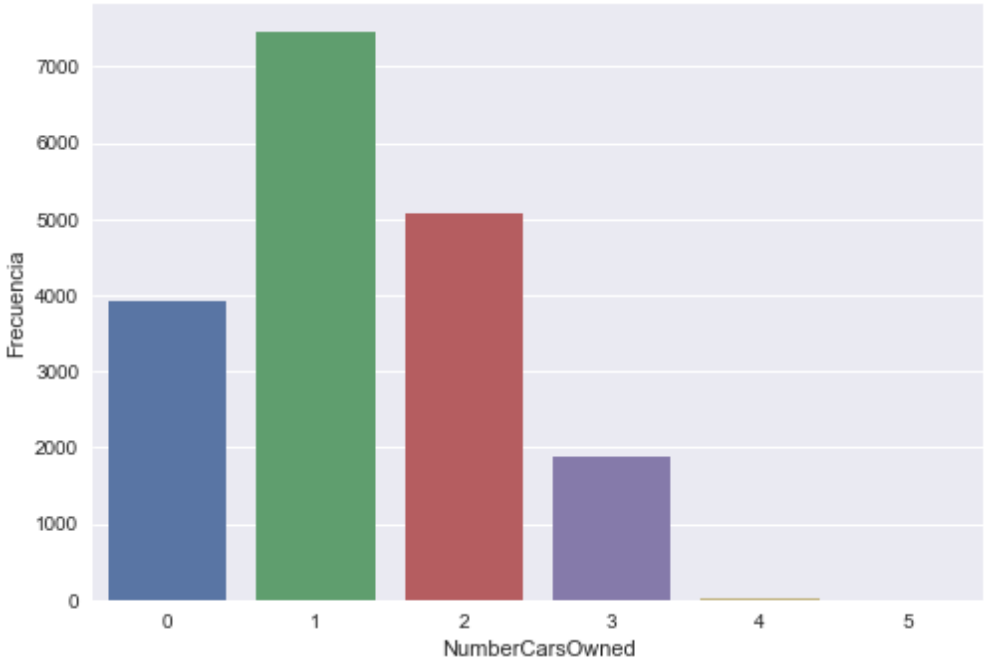

Figura 9. Distribución del número de automóviles de los clientes.

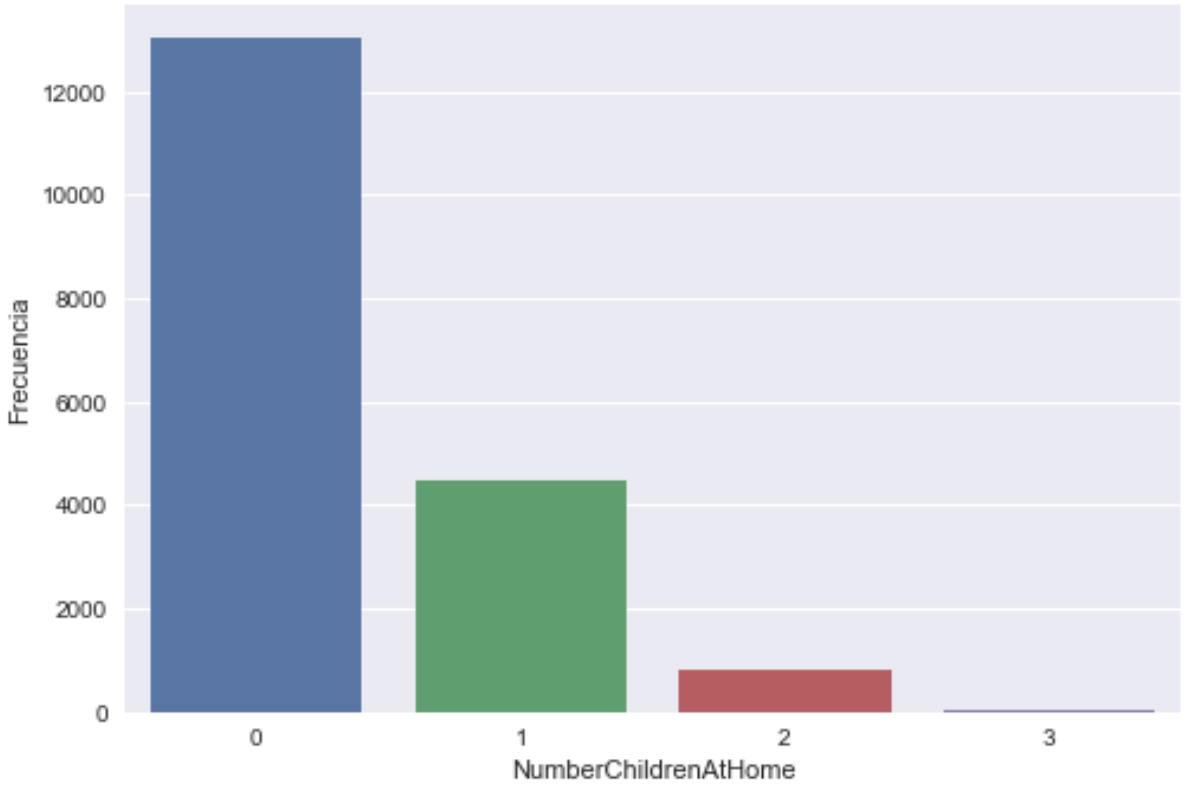

Figura 10. Distribución del número de hijos en el hogar de los clientes.

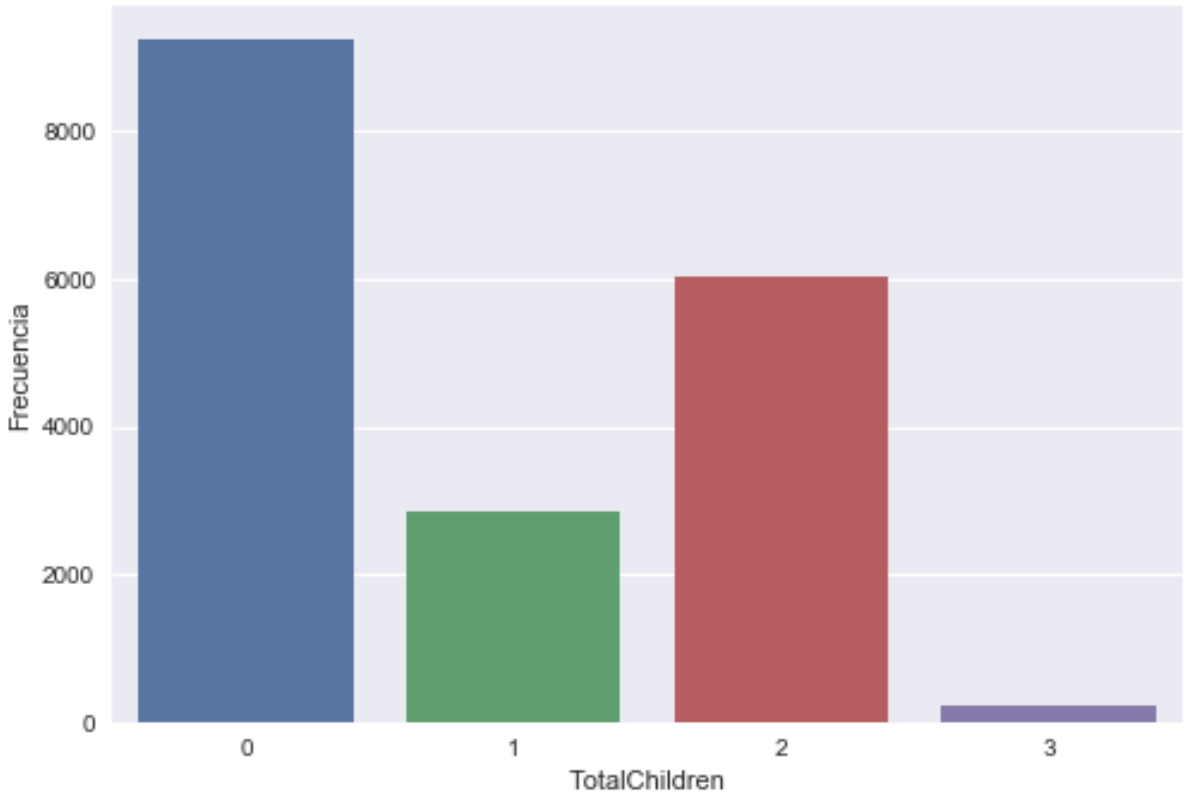

Figura 11. Distribución del total de hijos de los clientes.

Las Figuras mostradas confirman los siguientes argumentos:

- El rango de edad más frecuente entre los clientes es de 30-50 años (Figura 8).
- Por lo regular, la mayoría de los clientes cuenta con 1 automóvil. Por otro lado, muy pocos tienen más de 3 (Figura 9).
- La gran mayoría de los clientes no tiene hijos viviendo en sus casas (Figura 10).
- La cantidad de hijos por familia más frecuente es 2 (Figura 11).

3.3.2.2 Variables Categóricas

Posteriormente se trazaron las distribuciones de las variables categóricas, las cuales son mostradas en la Figura 12, Figura 13, Figura 14 y Figura 14:

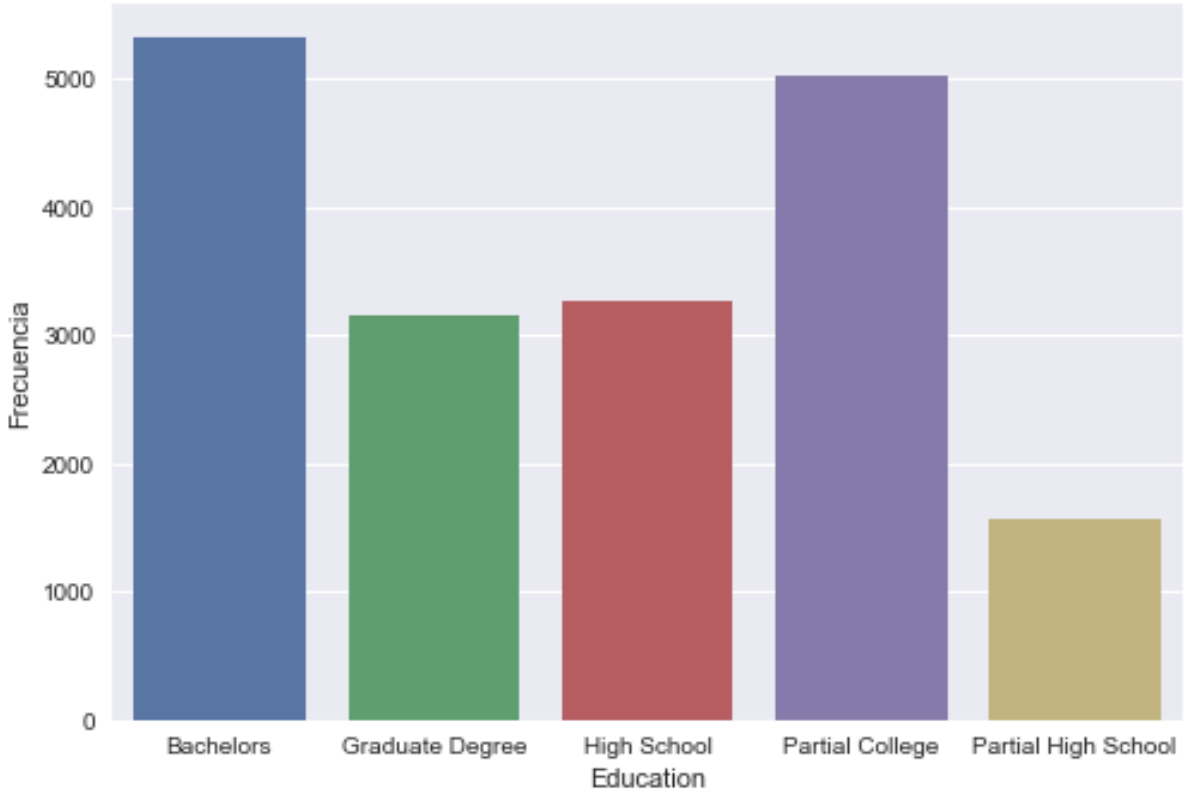

Figura 12. Distribución del nivel educativo de los clientes.

La Figura 12 muestra un gran número de clientes con título profesional (Bachelors), lo cual sugiere que la gran mayoría de los clientes tiene un empleo, seguido de un nivel educativo como estudiante universitario (Partial College).

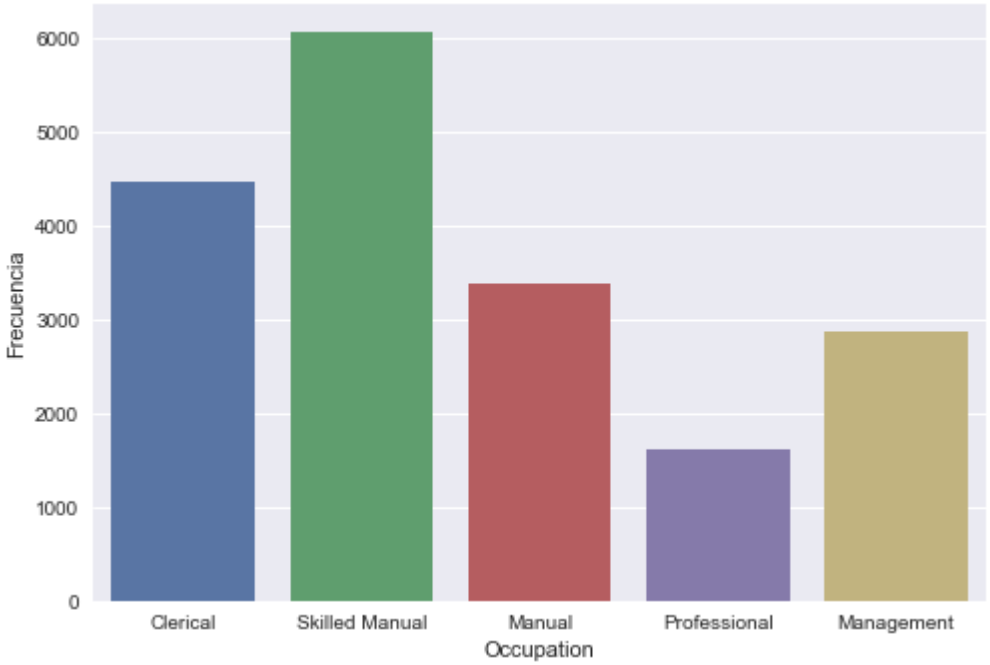

Figura 13. Distribución del tipo de empleo de los clientes.

Como se observa en la Figura 13, un empleo muy común entre los clientes son los trabajos que involucran habilidad manual como plomero o electricista, contrario a lo que sugiere la distribución de empleos donde la mayoría tiene un título profesional.

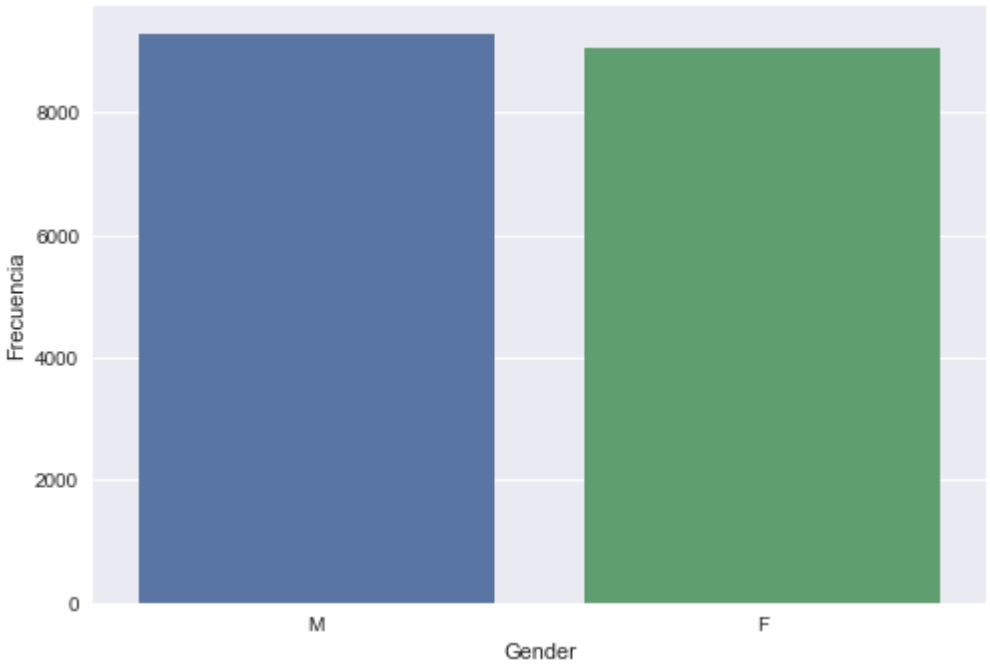

Figura 14. Distribución del género de los clientes.

La mayoría de los clientes son del género masculino (M).

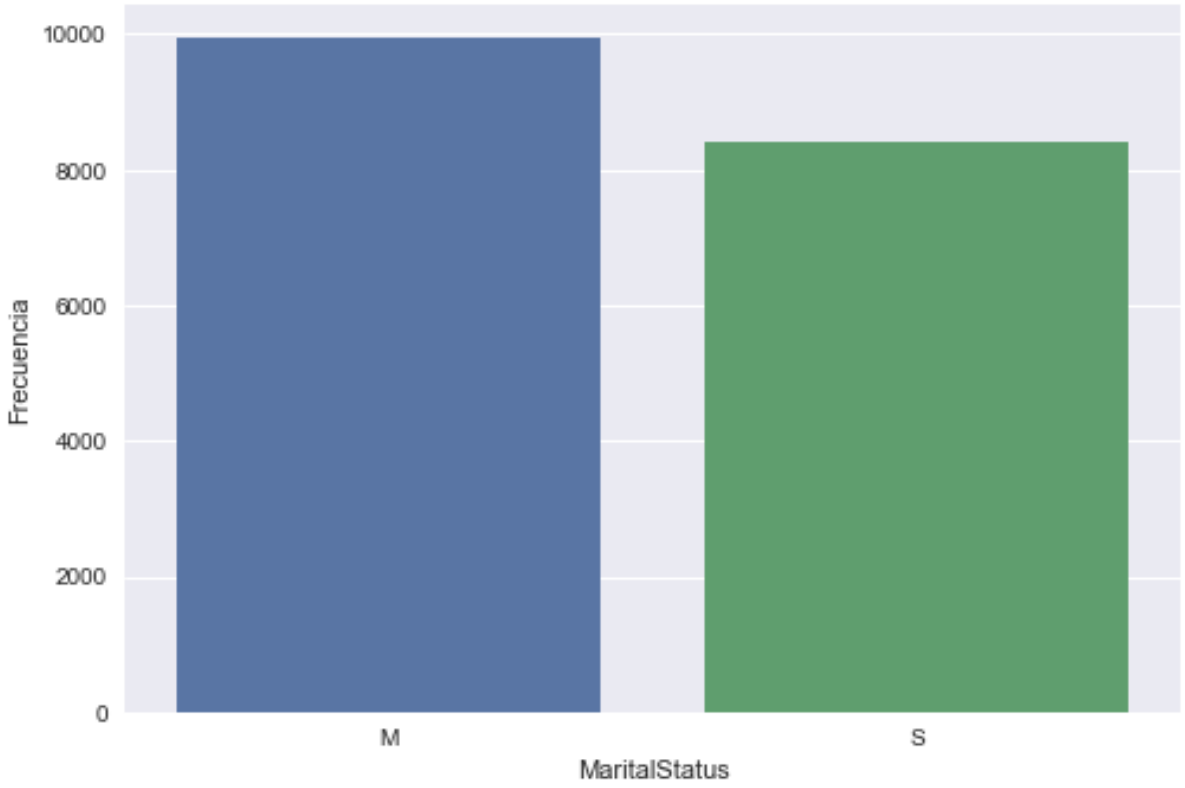

Figura 15. Distribución del estado civil de los clientes.

La mayoría de los clientes tiene un estado civil de casado (M).

Las figuras anteriores confirman los siguientes argumentos:

- El nivel educativo más común dentro de los clientes es ***Bachelors*** o título profesional.
- El tipo de trabajo más practicado es ***Skilled Manual*** *o* manual calificado.
- El género de los clientes está levemente inclinado hacia los hombres.
- El estado civil de los clientes está moderadamente inclinado hacia los casados.

3.3.3 Correlación y relaciones

Después de explorar las características individuales, el análisis de los datos tuvo como objetivo identificar las relaciones entre las características, en particular entre ***AvgMonthSpend*** y las demás características para descubrir alguna posible correlación correlación entre variables que describan el comportamiento del cliente.

3.3.3.1 Relaciones numéricas

La matriz de correlación mostrada en la Figura 16 indica los coeficientes de correlación entre las características numéricas, donde la tonalidad roja más oscura define una correlación fuerte positiva y la tonalidad azul más oscura una correlación fuerte negativa entre cada variable. Esta matriz muestra una moderada correlación entre ***AvgMonthSpend*** y

YearlyIncome, seguida por una correlación positiva con ***HomeOwnerFlag*** y ***NumberCarsOwned***.

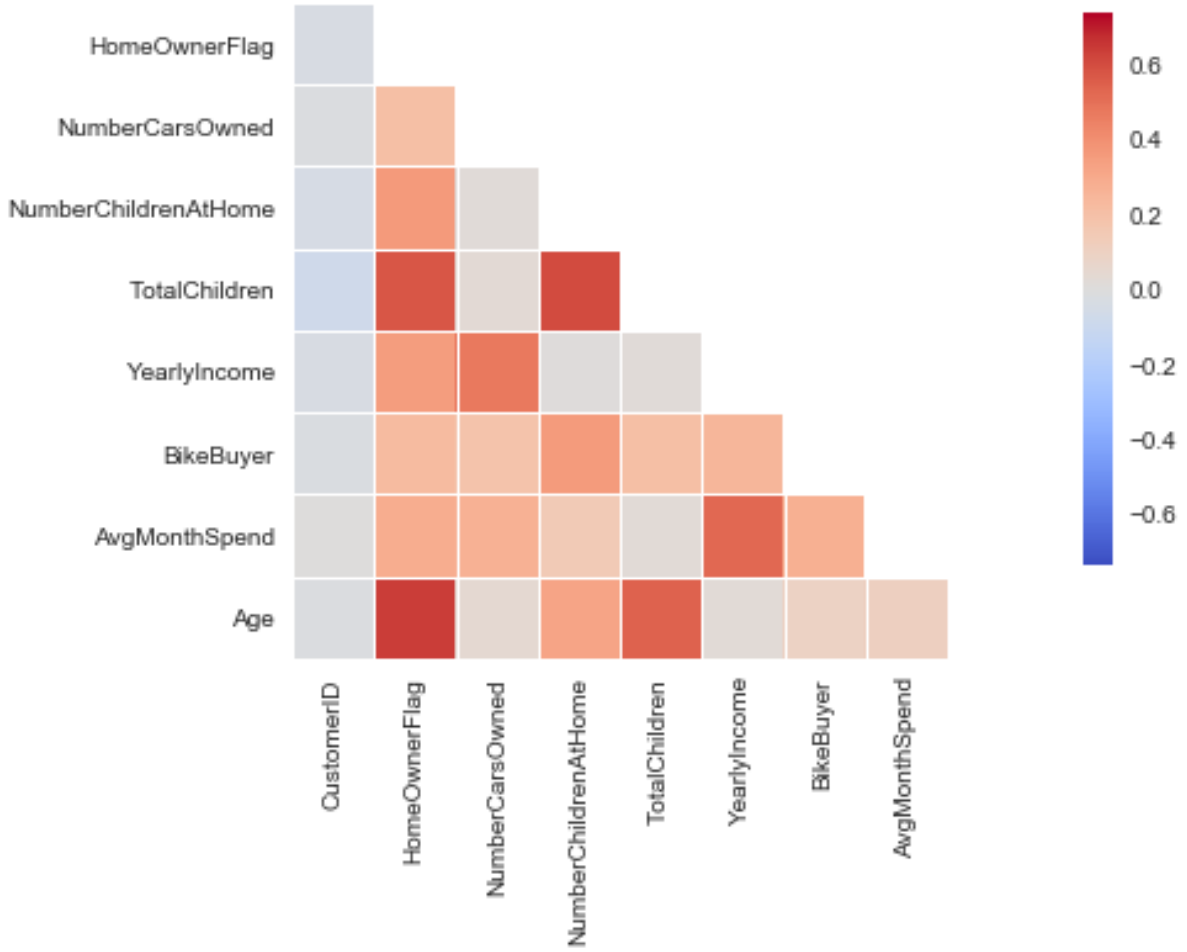

Figura 16. Matriz de correlación de coeficientes entre columnas numéricas.

La matriz de correlación de coeficientes muestra en tonalidades más rojas una fuerte correlación entre variables, y en tonalidades más tenues una correlación más débil. Por ejemplo, se observa una correlación aparente ente el ingreso anual del cliente y el número de autos que éste posee.

3.3.3.2 Relaciones categóricas

Habiendo explorado las relaciones entre ***AvgMonthSpend*** y las características numéricas, se utilizaron diagramas de caja para discernir cualquier relación aparente entre las características categóricas y ***AvgMonthSpend***. Los siguientes diagramas de caja en la Figura

17, Figura 18, Figura 19, Figura 20 y Figura 21 facilitan las comparaciones entre las características categóricas para exhibir una relación con ***AvgMonthSpend***.

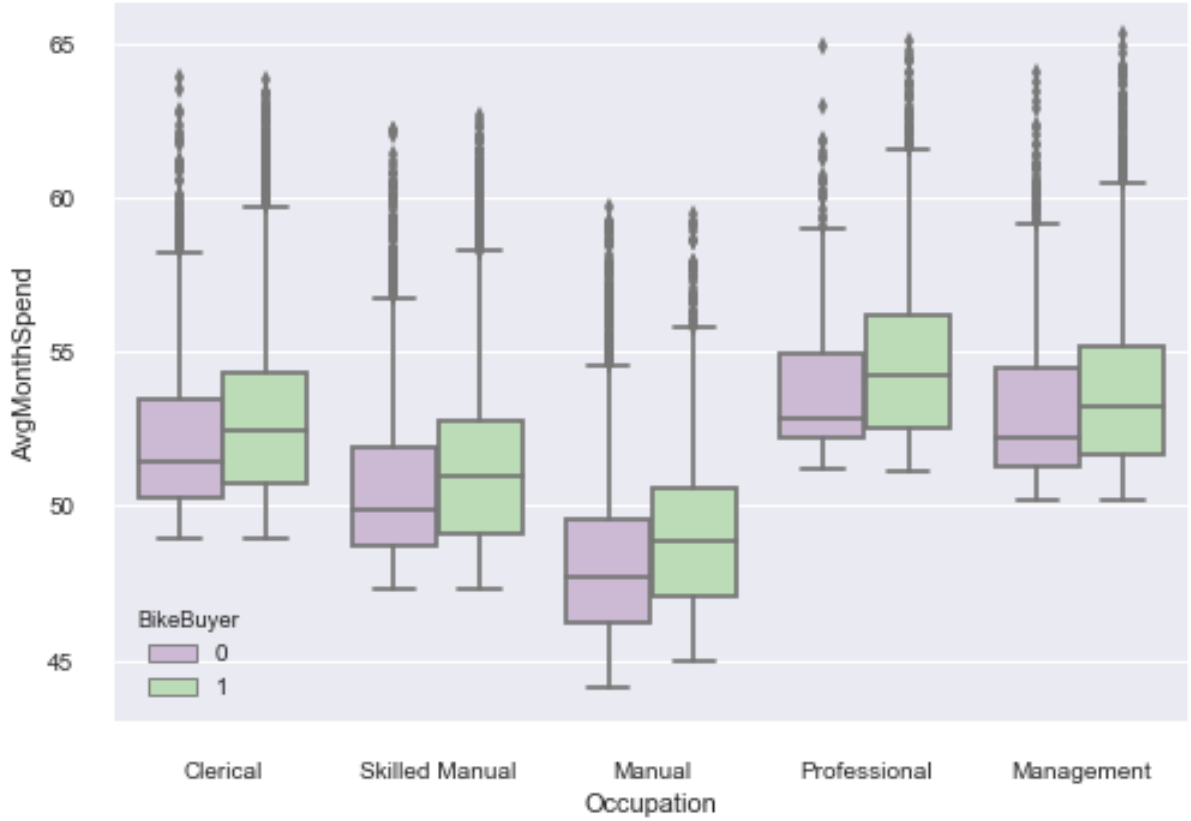

Figura 17. Diagrama de caja del gasto mensual promedio por tipo de trabajo y tipo de comprador.

El diagrama de caja es un método para visualizar los valores máximos y mínimos de los datos mediante cuartiles, así como los valores atípicos. La Figura 17 sugiere que los clientes que más gasta mensualmente son aquellos con un trabajo profesional como Doctor o Contador, además comprueba la existencia de valores atípicos en los datos que necesitan ser excluidos para evitar que los modelos de predicción generalicen de forma errónea.

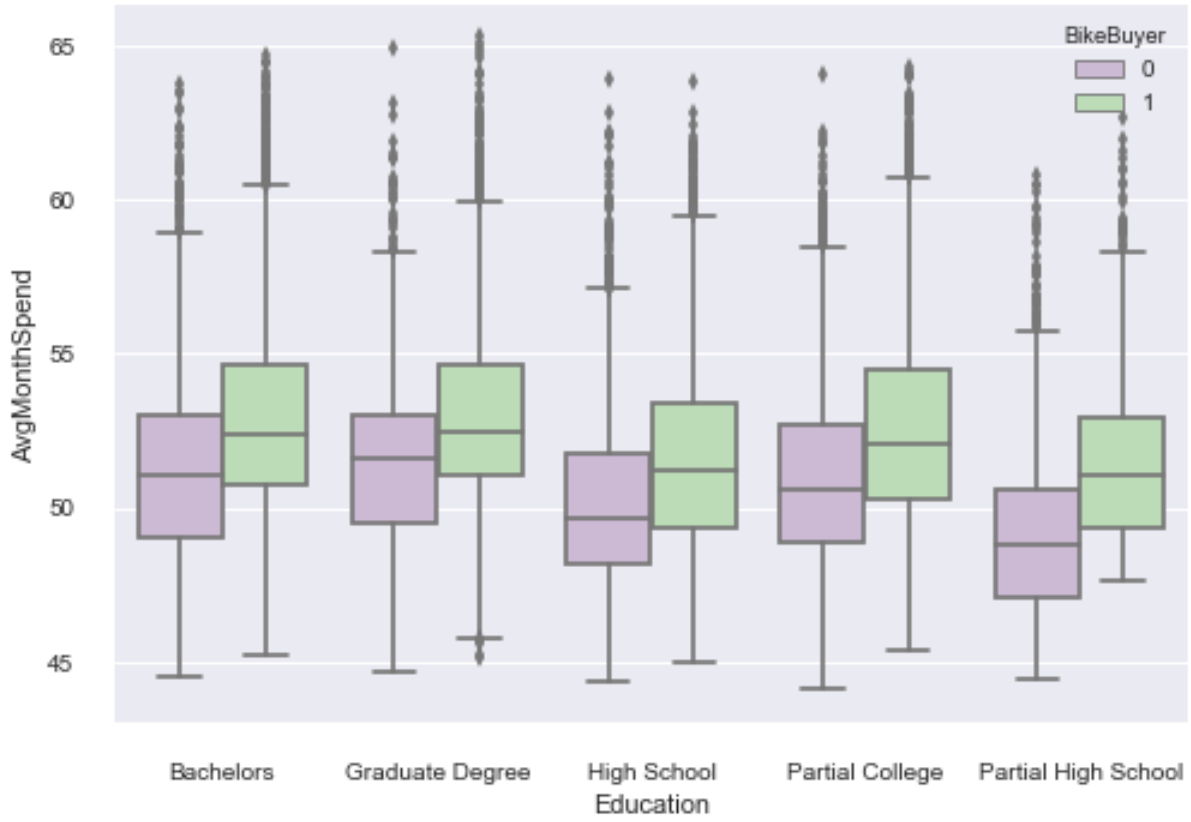

Figura 18. Diagrama de caja del gasto mensual promedio por nivel educativo y tipo de comprador.

La Figura 18 sugiere que los clientes tienen a gastar más cuando su nivel educativo es más alto como estudiante universitario, profesionista o posgrado.

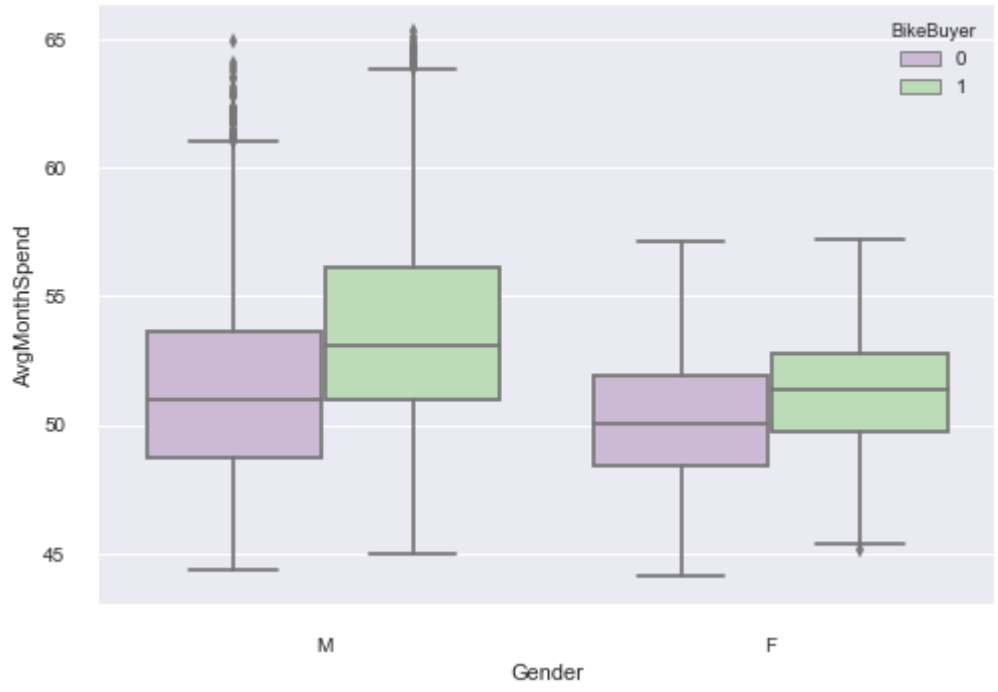

Figura 19. Diagrama de caja del gasto mensual promedio por género y tipo de comprador.

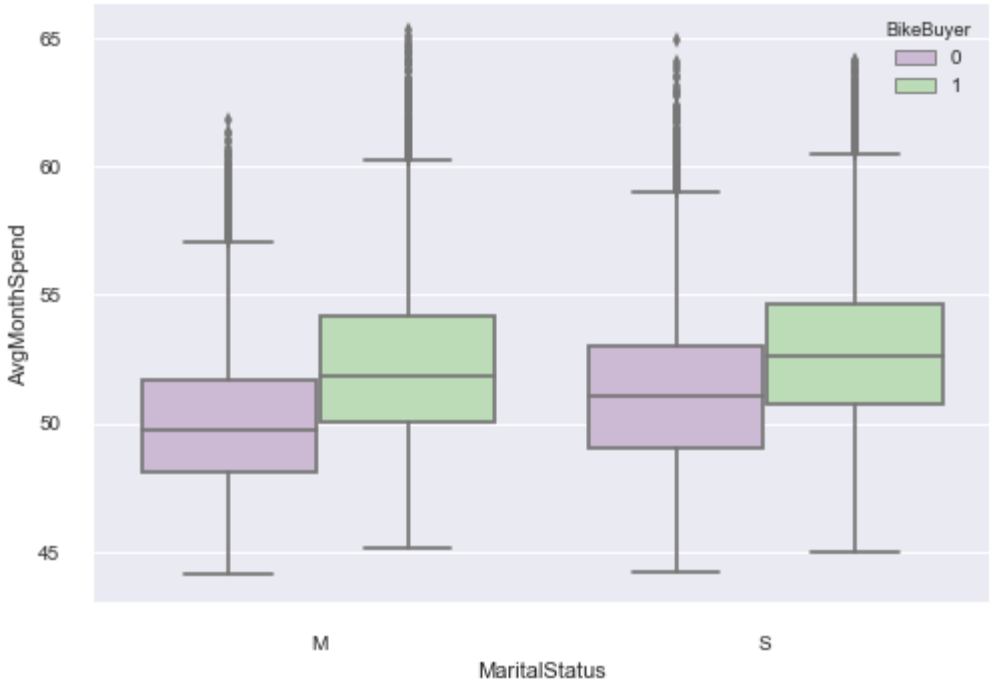

Figura 20. Diagrama de caja del gasto mensual promedio por estado civil y tipo de comprador.

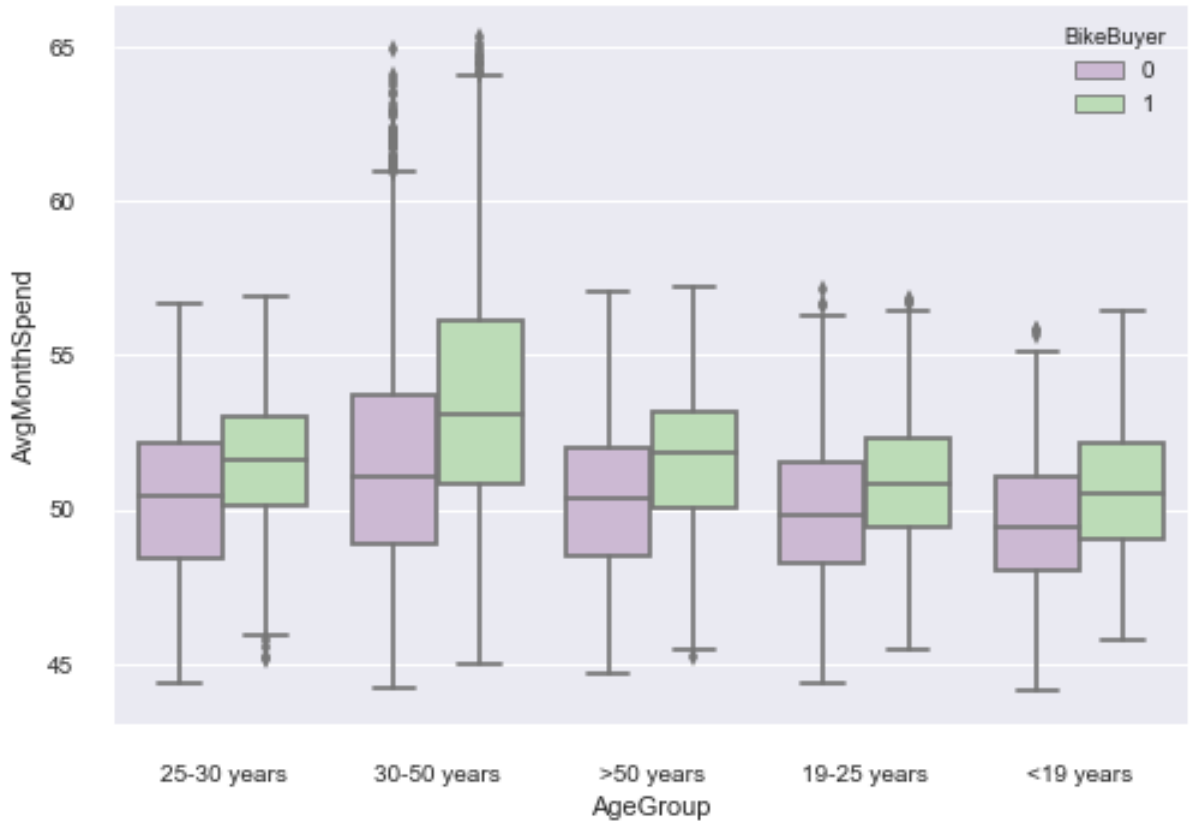

Figura 21. Diagrama de caja del gasto mensual promedio por rango de edades y tipo de comprador.

Los anteriores diagramas de la Figura 17, Figura 18, Figura 19, Figura 20 y Figura 21 muestran las variables categóricas que parecen exhibir una relación con ***AvgMonthSpend*** y ***BikeBuyer***:

- Los clientes con trabajos manuales o habilidades manuales tienen a gastar menos que aquellos con trabajos no manuales.
- Los clientes que han comenzado la universidad gastan más al mes que aquellos que sólo han terminado la preparatoria o están en preparatoria aún. Aunque la relación es pequeña, los diagramas de caja muestran también que los clientes gastan más cuanto más alto sea su nivel de educación.
- Los hombres tienden a gastar más al mes en promedio que las mujeres.
- No hay una diferencia significante entre el gasto mensual entre casado o solteros.
- Los clientes entre 30 y 50 años tienen un gasto promedio mensual mayor.

Se puede apreciar en todos los diagramas de caja valores atípicos, los cuales pueden ser un problema para el análisis de la información. Para minimizar este problema, se decidió borrar los registros arriba del brazo superior del diagrama de caja usando la siguiente regla (Sirca & Horvat, 2012:212): cualquier registro mayor que 1.5 * (Q3 - Q1) + Q3 (donde Q3 es el rango superior de la caja o el tercer cuartil y Q1 es el rango inferior de la caja o primer cuartil) ha sido removido del conjunto de datos.

3.4 Aprendizaje Automático

Los hallazgos del análisis de la información tales como las características y relaciones de las variables y los registros del conjunto de datos, sirvieron como base para la construcción, pruebas y optimización de modelos predictivos de aprendizaje automático enfocados en alcanzar el objetivo de la investigación.

Todos los experimentos fueron realizados en una sola máquina corriendo el sistema operativo Windows 10. La máquina contiene un procesador Intel Core i7 Q720 1.60 GHz y 16 GB de memoria. Los experimentos fueron programados en Python y los algoritmos proporcionados por las bibliotecas de aprendizaje automático en Python Scikit-Learn y XGBoost.

La quinta fase del proceso de la Ciencia de Datos, el desarrollo del modelo fue realizada implementando los siguientes modelos:

- Un modelo de clasificación que predice si un nuevo cliente comprará o no una bicicleta.
- Un modelo de regresión que predice el gasto mensual promedio de un nuevo cliente.

3.4.1 Modelo de clasificación

3.4.1.1 Selección del modelo

Brown y Mues (2012) estudiaron el rendimiento de 10 técnicas de clasificación: Least Square Support Vector Machines (LS-SVM), Decision Trees (C4.5), Random Forest, Gradient Boosting, Neuronal Networks (NN), K-Nearest Neighbors (*k*-NN10, *k*-NN100), Logistic Regression (LOG) y Linear Discriminant Analysis (LDA, QDA), para un problema de clasificación binaria de puntaje crediticio. Como mencionan los autores Brown y Mues (2012), el rendimiento de las técnicas de clasificación varía dependiendo del nivel de desbalanceo en los datos. El desbalanceo de los datos en el puntaje crediticio se encuentra en que el conjunto de datos generalmente contiene una cantidad mucho más pequeña de persona moratorias que aquellas que pagan a tiempo. El experimento mostró el desempeño de cada clasificador en cada uno de los cinco conjuntos de datos utilizados en varios niveles de desbalance.

Tabla 6. Posición promedio de las técnicas de clasificación en los diferentes conjuntos de datos. Adaptado de "An experimental comparison of classification algorithms for imbalanced credit scoring data sets" por Brown I. y Mues C., 2012, 39(3), 3446–3453. Copyright 2012 Expert Systems with Applications.

	Nivel de desbalance					
	30%	15%	10%	5%	2.5%	1%
LOG	5.4	4.5	4.4	5.5	6.3	7.7
C4.5	9.1	8.2	8.4	7.6	7.0	6.9
NN	5.7	5.7	5.8	5.0	4.6	5.6
Gradient Boosting	3.7	**2.3**	3.2	3.4	**2.8**	3.5
LDA	3.8	3.2	3.2	**2.6**	3.0	3.4
QDA	8.5	9.2	8.4	7.9	8.3	7.9
Random Forest	3.2	2.7	**2.2**	3.2	3.0	**1.6**
***k*-NN10**	7.7	7.9	6.8	7.0	7.0	6.5
***k*-NN100**	6.7	4.6	4.6	3.6	3.8	3.1
Lin LS-SVM	**1.2**	6.7	8.0	9.2	9.2	8.8

Los resultados de su investigación muestran en la Tabla 6 que los algoritmos Gradient Boosting y Random Forest tienen un mejor rendimiento frente a otros clasificadores. A partir de este criterio, se seleccionaron los algoritmos Gradient Boosting y Random Forest, así como el algoritmo XGBoost, el cual es una optimización del algoritmo Gradient Boosting.

3.4.1.2 Selección de características

No todas las características del conjunto de datos proporcionan el mismo valor de información para la variable objetivo **BikeBuyer** del modelo de clasificación (Joshi P., Hearty J., Sjardin B. et al., 2016:48). Durante el proceso de transformación de las características para poder alimentar a los algoritmos se clasificación, se terminó obteniendo un total de 8659 características.

Para poder conocer el valor que proporciona cada característica se obtuvo la importancia que cada una proporciona a los algoritmos antes descritos haciendo uso de los métodos de relevancia de características incluidos en cada algoritmo. La calificación está entre 0 y 1, donde 0 significa “no se usa en lo absoluto” y 1 significa “perfectamente predice la variable objetivo”. Sin embargo, si una característica obtiene un valor bajo no significa que no aporte valor al algoritmo, sino que otra característica proporciona la misma información y se inclina más por la otra (Müller y Guido, 2017:78).

Antes de realizar el proceso de selección se características, los algoritmos fueron alimentados sin ninguna selección de características para evaluar su mejora posterior. Para evaluar el desempeño de los algoritmos de clasificación, se utilizó la métrica exactitud (accuracy en Inglés), la cual es ampliamente utilizada en problemas de clasificación (Gupta, Chetty & Shukla, 2015; Al-Thubaity & Al-Subaie, 2015; Vidulin, Luštrek & Gams, 2007).

La siguiente lista muestra la exactitud alcanzada inicialmente por los algoritmos:

- Gradient Boosting: 78.75 %
- Random Forest: 71.55 %
- XGBoost: 78.65 %

A continuación,en la Tabla 7 se muestran los lugares obtenidos por las 10 características mejor calificadas del conjunto de datos para determinar cuáles aporta mayor valor para predecir la variable objetivo.

Tabla 7. Importancia de las características para los modelos de clasificación.

Lugar	Gradient Boosting	Random Forest	XGBoost
1	NumberCarsOwned	YearlyIncome	YearlyIncome
2	YearlyIncome	NumberChildrenAtHome	NumberCarsOwned
3	NumberChildrenAtHome	AvgMonthSpend	NumberChildrenAtHome
4	Gender_M	NumberCarsOwned	Age
5	Age	Age	MaritalStatus_M
6	Education_Partial High School	TotalChildren	Education_Partial High School
7	AvgMonthSpend	Occupation_Manual	Gender_F
8	MaritalStatus_M	MaritalStatus_S	AvgMonthSpend
9	MaritalStatus_S	Gender_F	TotalChildren
10	Gender_F	MaritalStatus_M	City_Sulzbach Taunus

Al utilizar las características de la Tabla 7 en cada algoritmo de clasificación se obtuvieron los siguientes puntajes de exactitud:

- Gradient Boosting: 79.19 %
- Random Forest: 77.52 %
- XGBoost: 78.80 %

Se puede observar que haciendo uso de las primeras 10 características listadas como las más importantes de cada algoritmo de clasificación ayudó a incrementar el puntaje de exactitud de cada uno.

3.4.1.3 Optimización del modelo

El conocer las características que aportan valor para predecir la variable objetivo ayuda a realizar de nueva cuenta el proceso de ingeniería de características para encontrar nuevas características que aporte valor a partir de las características encontradas.

Se realizó el proceso de ingeniería de características para crear las siguientes columnas con el fin de obtener un mejor puntaje en la exactitud del algoritmo de clasificación:

- **PostalCode2**: Se utilizó el código de 3 dígitos del país de la columna CountryRegionName y se concatenó a la columna PostalCode.

- **Education2**: Se utilizó un orden numérico a la columna del 1 al 5. Donde 1 corresponde al nivel educativo Partial High Education y 5 corresponde a Graduate Degree.
- **Occupation2**: Se utilizó un orden numérico a la columna del 1 al 5. Donde 1 corresponde al tipo de empleo Manual y 5 corresponde a Professional.
- **TotalChildren2**: Se utilizó un valor numérico 0 si no tenía hijos y 1 si tenía uno o más.
- **NumberChildrenAtHome2**: Se utilizó un valor numérico 0 si no tenía hijos viviendo en casa y 1 sin tenia al menos 1 hijo viviendo en casa.
- **NumberCarsOwned2**: Se utilizó un valor numérico 0 si no tenía automóvil y 1 si tenía al menos un automóvil.
- **Gender2**: Se utilizó un valor numérico 0 si el género era masculino y 1 si era femenino.
- **MaritalStatus2**: Se utilizó un valor numérico 0 si estaba casado y 1 si era soltero.
- **TotalChildren3**: Se utilizó un valor cadena “0” cuando no tenía hijos, “1” cuando tenía un hijo y “2+” cuando tenía dos o más hijos.
- **NumberChildrenAtHome3**: Se utilizó un valor cadena “0” cuando no tenía hijos viviendo en casa y “1+” cuando tenía al menos un hijo viviendo en casa.
- **NumberCarsOwned3**: Se utilizó un valor cadena “0” cuando no tenía automóvil, “1” cuando tenía un automóvil, “2” cuando tenía dos automóviles y “3+” cuando tenía tres o más automóviles.
- **Age2**: Se transformaron los valores utilizando la función de logaritmo natural.
- **AvgMonthSpend2**: Se transformaron los valores utilizando la función de logaritmo natural.
- **YearlyIncome2**: Se transformaron los valores utilizando la función de logaritmo natural.
- **NumberCarsOwned4**: Se estandarizaron los valores utilizando la función z-score.
- **NumberChildrenAtHome4**: Se estandarizaron los valores utilizando la función z-score.
- **TotalChildren4**: Se estandarizaron los valores utilizando la función z-score.
- **Gender4**: Se estandarizaron los valores utilizando la función z-score.
- **MaritalStatus4**: Se estandarizaron los valores utilizando la función z-score.
- **Education4**: Se estandarizaron los valores utilizando la función z-score.
- **Occupation4**: Se estandarizaron los valores utilizando la función z-score.

Después de haber realizado la ingeniería de características, las nuevas características fueron valoradas para obtener su importancia en los algoritmos de clasificación. La Tabla 8 muestra

el valor de importancia de las características después del proceso de selección de los algoritmos de clasificación.

Tabla 8. Importancia de las características después del proceso de ingeniería de características.

Lugar	Gradient Boosting	Random Forest	XGBoost
1	YearlyIncome	YearlyIncome	YearlyIncome
2	YearlyIncome2	YearlyIncome2	NumberCarsOwned3_1
3	NumberCarsOwned3_1	AvgMonthSpend	Age
4	Age2	AvgMonthSpend2	NumberChildrenAtHome
5	Gender_F	NumberChildrenAtHome2	Gender2
6	NumberCarsOwned3_2	Age	AvgMonthSpend
7	Age	Age2	Education2
8	NumberChildrenAtHome3_0	NumberChildrenAtHome	MaritalStatus2
9	AvgMonthSpend	NumberChildrenAtHome4	TotalChildren
10	NumberChildrenAtHome2	NumberCarsOwned4	PostalCode2_V8Y 1L1-124

Sin embargo, los algoritmos de la Tabla 8 no proporcionan un método para seleccionar las características importantes sino sólo miden la importancia de las características a través de un listado. Se realizó la selección de las características importantes por medio de la biblioteca Boruta por su capacidad de reducir el número de características al seleccionar sólo aquellas que proporcionan poder de predicción al modelo (Poona & Ismail, 2013; Yang, Jin, Zhang et al., 2017; Cananedo, Feldheim & Deramaix, 2017).

Utilizando Boruta con un porcentaje de características importantes del 80%, se logró reducir el número de características para alimentar a los algoritmos de clasificación, la Tabla 9 muestra las características importantes seleccionadas como resultados del proceso Boruta.

Tabla 9. Características seleccionadas por el proceso Boruta.

Lugar	Gradient Boosting	Random Forest
1	YearlyIncome	NumberCarsOwned
2	AvgMonthSpend	NumberChildrenAtHome
3	Age	YearlyIncome
4	NumberChildrenAtHome2	AvgMonthSpend
5	Age2	Age
6	AvgMonthSpend2	Education2
7	YearlyIncome2	Occupation2
8	Gender_F	NumberChildrenAtHome2
9	MaritalStatus_M	Age2
10	NumberChildrenAtHome3_0	AvgMonthSpend2
11	NumberCarsOwned3_1	YearlyIncome2
12	NumberCarsOwned3_2	NumberCarsOwned4
13		NumberChildrenAtHome4
14		Education4
15		Occupation4
16		NumberChildrenAtHome3_0
17		NumberChildrenAtHome3_1
18		NumberCarsOwned3_1

La biblioteca XGBoost no implementa completamente la API de SciKit-Learn para poder utilizar el proceso Boruta, las características seleccionadas por el algoritmo Gradient Boosting fueron utilizadas también en el algoritmo XGBoost.

Después de seleccionar las características más importantes para cada algoritmo, se realizaron repetidas ocasiones el proceso de optimización de hiper parámetros para obtener los valores óptimos de cada algoritmo de clasificación.

Al utilizar las características del listado anterior en cada algoritmo de clasificación se obtuvieron los siguientes puntajes de exactitud:

- Gradient Boosting: 79.47 %
- Random Forest: 79.07 %
- XGBoost: 79.70 %

3.4.1.4 Evaluación del modelo

Utilizando el análisis de la información de las ventas existentes y la información de los clientes, un modelo predictivo fue construido para clasificar a clientes en dos categorías (comprador / no comprador). La selección del modelo fue hecha comparando la exactitud para clasificar de los siguientes algoritmos:

- Gradient Boosting
- Random Forest
- XGBoost

El algoritmo XGBoost mostró mejor rendimiento con el conjunto de datos existente y fue utilizado para los siguientes cálculos. El modelo fue entrenado con 70% de la información. La información fue estratificada como medida de validación cruzada debido a que no está normalmente distribuida. Se probó el modelo con el 30% restante de la información y se optimizaron los parámetros del algoritmo produciendo la siguiente matriz de confusión de la Tabla 10:

Tabla 10. Matriz de confusión para el modelo de clasificación.

Verdaderos Positivos	**Falsos Negativos**
1654	726
Falsos Positivos	**Verdaderos Negativos**
318	2446

La métrica de la curva ROC (acrónimo de Received Operator Characteristic) fue utilizada como se muestra en la Figura 22. La línea azul indica el desempeño de los diferentes valores de clasificación y la línea diagonal muestra los valores esperados de adivinar al azar.

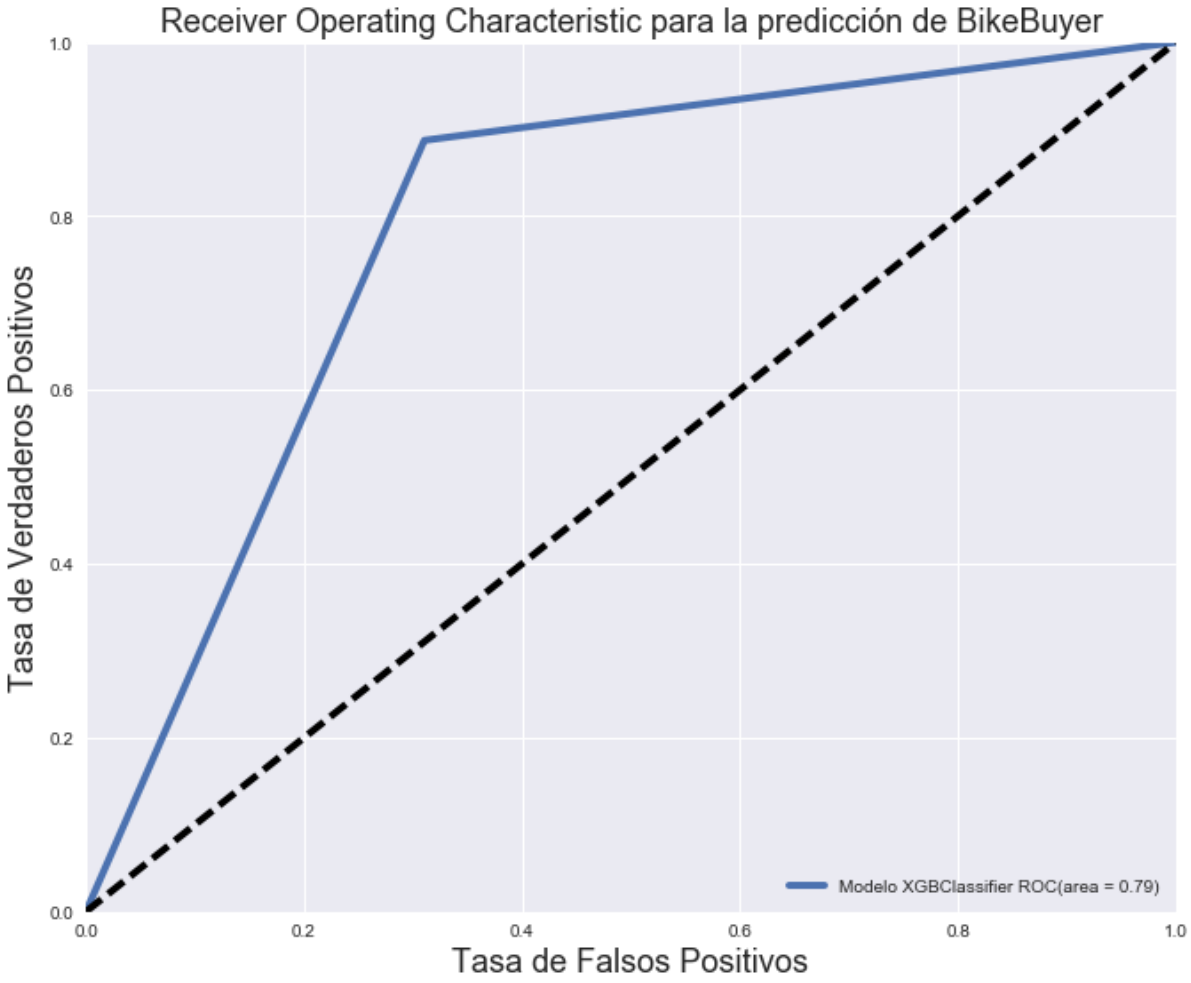

Figura 22. Curva ROC del modelo de clasificación.

El área de la curva ROC de 0.79, puede ser interpretada con las siguientes métricas de rendimiento para el modelo de clasificación:

- Exactitud: 79.70% (proporción de compradores de bicicleta correctamente clasificados)
- Precisión: 80% (proporción de positivos)
- Exhaustividad: 80% (tasa de positivos verdaderos)
- Puntaje F1: 80% (balance entre las métricas de precisión y exhaustividad)

El desempeño del modelo es ligeramente menor al 80% lo cual sugiere una carencia de generalización en el poder de predicción, lo cual se debe principalmente al sobre-generalización (underfitting) de la distribución actual. Otra causa potencial es la existencia de otra variable predictora oculta que no está disponible en el conjunto de datos de entrenamiento.

3.4.1.5 Hallazgos

El análisis de los datos sirvió como base de la construcción, optimización y prueba del modelo de clasificación binaria. La evaluación del modelo proporción los siguientes hallazgos:

- Las características más importantes para la predicción de la compra del cliente son:
    - YearlyIncome (Ingreso anual)
    - NumberChildrenAtHome (Núm. de hijos en casa)
    - Age2 (Edad)
    - AvgMonthSpend (Gasto mensual promedio)
    - Education2 (Nivel educativo)
    - Gender (Género)
    - MaritalStatus (Estado civil)
    - NumberCarsOwned3 (Núm. de autos poseídos)
- La selección de las características más importantes con la biblioteca Boruta contribuyó a la selección del conjunto óptimo para mejorar la exactitud de predicción.
- El algoritmo de clasificación binaria que produjo el puntaje de exactitud más alto fue XGBoost.
- El puntaje más alto de exactitud obtenido fue de 79.70%.

La Figura 23 resume los resultados obtenidos de las actividades realizadas para optimizar el modelo. En el área azul se muestra el puntaje de exactitud más alto obtenido en cada iteración del modelo de clasificación, en los recuadros de la derecha las principales actividades realizadas para optimizarlo.

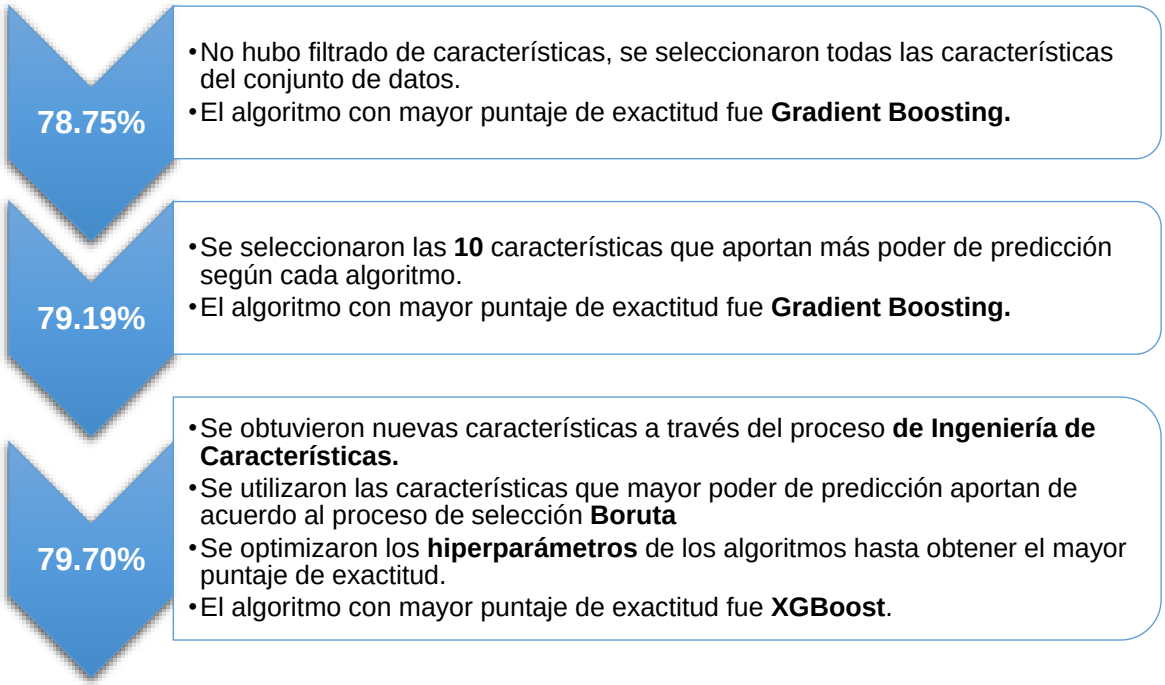

Figura 23. Proceso de optimización del modelo de clasificación.

3.4.2 Modelo de regresión

3.4.2.1 Selección de características

Huang, Zhou, Gu, et al. (2016) estudiaron el desempeño de 5 técnicas de regresión: Support Vector Machines con kernel polinominal, lineal y RBF, Random Forest y Gradient Boosting. La Tabla 11 muestra el rendimiento de las diferentes técnicas de regresión utilizadas en su investigación utilizando como métrica la raíz del error cuadrático medio o RMSE en Inglés, dónde más bajo el valor RMSE mejor es la técnica para predecir.

Tabla 11. Comparación del rendimiento de las técnicas de regresión en los diferentes conjuntos de datos. Adaptado de "Comparison of regressors on 3D visual discomfort prediction" por Huang, Zhou, Gu et al., 2016, pp. 1–6. Copyright 2016 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB).

Características	Modelos	RMSE
Simple Statistical Features	Linear SVR	1.722
	Poly SVR	4.850
	RBF SVR	1.017
	Random Forest	**0.8012**
	GBDT	**0.8778**
Statistical Distribution Features	Linear SVR	1.746
	Poly SVR	3.047
	RBF SVR	0.9403
	Random Forest	**0.8202**
	GBDT	**0.8223**
Neural Response Features	Linear SVR	**0.6599**
	Poly SVR	1.301
	RBF SVR	0.8407
	Random Forest	0.9132
	GBDT	**0.7644**
ALL	Linear SVR	0.9383
	Poly SVR	2.302
	RBF SVR	**0.8709**
	Random Forest	**0.8397**
	GBDT	0.8963

Los resultados de su investigación mostraron que los algoritmos Gradient Boosting y Random Forest tenían un mejor rendimiento frente a otras técnicas de regresión. A partir de este criterio, se seleccionaron los algoritmos Gradient Boosting y Random Forest, así como el algoritmo XGBoost, el cual es una optimización del algoritmo Gradient Boosting.

3.4.2.2 Selección de características

Siguiendo el procedimiento del modelo de clasificación anteriormente descrito para la selección de características, los algoritmos de regresión fueron alimentados sin ninguna selección de características para evaluar su posterior mejora respecto a la variable objetivo **AvgMonthSpend**. Para evaluar el desempeño de los algoritmos de regresión, se la raíz del error cuadrático medio como métrica, la cual es un estándar en la evaluación de problemas de regresión (Geng & Ma, 2008;Li, Xu, Zhao et al, 2009; Xu & Liu, 2011).

La siguiente lista muestra los puntajes de la raíz del error cuadrático medio obtenidos:

- Gradient Boosting: 1.7521
- Random Forest: 1.8833
- XGBoost: 1.7483

A continuación, en la Tabla 12 se muestran los lugares obtenidos por las 10 características mejor calificadas del conjunto de datos para determinar cuáles aporta mayor valor para predecir la variable objetivo.

Tabla 12. Importancia de las características para los modelos de regresión.

Lugar	Gradient Boosting	Random Forest	XGBoost
1	YearlyIncome	YearlyIncome	YearlyIncome
2	Age	Age	Age
3	NumberChildrenAtHome	NumberChildrenAtHome	Gender_F
4	Gender_F	Gender_F	MaritalStatus_M
5	Gender_M	Gender_M	NumberChildrenAtHome
6	MaritalStatus_S	MaritalStatus_M	BirthDate_1986-12-19
7	MaritalStatus_M	MaritalStatus_S	BirthDate_1973-06-13
8	BirthDate_1972-03-05	NumberCarsOwned	BirthDate_1972-03-05
9	BirthDate_1973-06-13	TotalChildren	BirthDate_1968-04-22
10	BirthDate_1980-03-04	Education_High School	NumberCarsOwned

Al utilizar las características del listado anterior en cada algoritmo de clasificación se obtuvieron los siguientes puntajes de RMSE:

- Gradient Boosting: 1.7335
- Random Forest: 1.7747
- XGBoost: 1.7313

Haciendo uso de las primeras 10 características listadas como las más importantes de cada algoritmo de regresión ayudó a incrementar el puntaje de RMSE de cada uno.

3.4.2.3 Optimización del modelo

Con el propósito de mejorar el puntaje de RMSE para los algoritmos de regresión, se utilizaron las mismas características obtenidas del proceso de ingeniería de características realizado

en el modelo de clasificación. La Tabla 13 muestra el valor de importancia de las características después del proceso de selección de los algoritmos de regresión.

Tabla 13. Importancia de las características después del proceso de ingeniería de características.

Lugar	Gradient Boosting	Random Forest	XGBoost
1	YearlyIncome	YearlyIncome	Age
2	YearlyIncome2	YearlyIncome2	YearlyIncome
3	Gender_F	Age2	Gender2
4	Age	Age	NumberChildrenAtHome
5	Age2	Gender_M	MaritalStatus2
6	Gender_M	Gender2	PostalCode2_4551-036
7	MaritalStatus_M	Gender_F	TotalChildren
8	Gender4	NumberChildrenAtHome3_0	Education2
9	Gender2	Gender4	PostalCode2_98020-840
10	NumberChildrenAtHome2	NumberChildrenAtHome4	NumberCarsOwned

De la misma manera que el modelo de clasificación realizó el proceso de selección de características, se utilizó el proceso Boruta para seleccionar las características dentro de un porcentaje mayor o igual a 80% de importancia, la Tabla 14 muestra las características importantes seleccionadas como resultados del proceso Boruta.

Tabla 14. Características seleccionadas por el proceso Boruta.

Lugar	Gradient Boosting	Random Forest
1	YearlyIncome	NumberChildrenAtHome
2	Age	YearlyIncome
3	Gender2	Age
4	Age2	NumberChildrenAtHome2
5	YearlyIncome2	Gender2
6	Gender4	Age2
7	Gender_F	YearlyIncome2
8	Gender_M	NumberChildrenAtHome4
9	MaritalStatus_M	Gender4
10	NumberChildrenAtHome3_0	Gender_F
11		Gender_M
12		MaritalStatus_M
13		MaritalStatus_S
14		NumberChildrenAtHome3_0

La biblioteca XGBoost no implementa completamente la API de SciKit-Learn para poder utilizar el proceso Boruta, las características seleccionadas por el algoritmo Gradient Boosting fueron utilizadas también en el algoritmo XGBoost.

Después de seleccionar las características más importantes para cada algoritmo, se realizó repetidas ocasiones el proceso de optimización de hiperparámetros para obtener los parámetros óptimos de cada algoritmo de regresión.

Al utilizar las características del listado anterior en cada algoritmo de clasificación se obtuvieron los siguientes puntajes de RMSE:

- Gradient Boosting: 1.7320
- Random Forest: 1.7628
- XGBoost: 1.7294

3.4.2.4 Evaluación del modelo

Utilizando el análisis de la información de las ventas existentes y la información de los clientes, un modelo de regresión fue construido para predecir el gasto mensual promedio del cliente. La selección del modelo fue hecha comparando el puntaje de la raíz del error cuadrático medio (RMSE) los siguientes algoritmos:

- Gradient Boosting
- Random Forest
- XGBoost

El algoritmo XGBoost mostró mejor rendimiento con el conjunto de datos existente y fue utilizado para los siguientes cálculos. El modelo fue entrenado con 70% de la información. Se probó el modelo con el 30% restante de la información y se optimizaron los parámetros del algoritmo. Un diagrama de dispersión mostrando el gasto previsto y el real se muestra a continuación en la Figura 24.

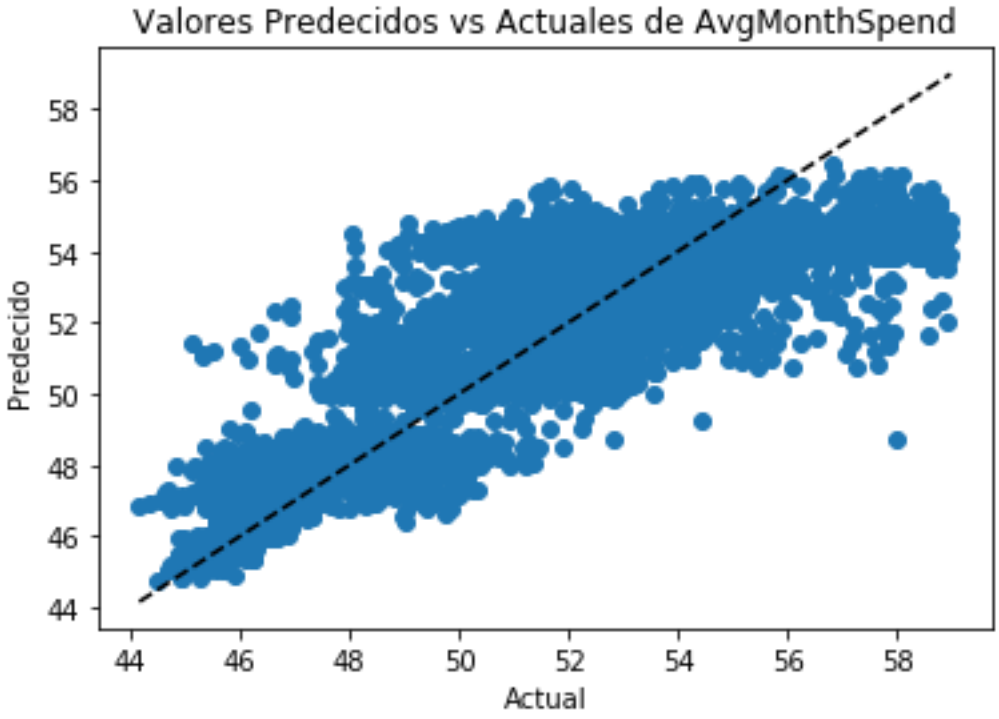

Figura 24. Diagrama de dispersión de los valores previstos y los reales del gasto mensual promedio.

El diagrama de dispersión muestra una clara relación lineal entre los valores previstos y los reales en el conjunto de datos de prueba. El error más pequeño de la raíz del error cuadrático medio (RMSE) de los resultados fue de: 1.729404. Este valor obtenido representa la diferencia entre los valores predichos por el modelo y los valores actualmente observados.

3.4.2.5 Hallazgos

El análisis de los datos sirvió como base de la construcción, optimización y prueba del modelo de clasificación binaria. La evaluación del modelo proporción los siguientes hallazgos:

- Las características más importantes para la predicción de la compra del cliente son:
    - YearlyIncome (Ingreso anual)
    - Age (Edad)
    - NumberChildrenAtHome (Núm. de hijos en casa)
    - Gender (Género)
    - MaritalStatus (Estado civil)

- La selección de las características más importantes con la biblioteca Boruta contribuyó a la selección del conjunto óptimo para mejorar el puntaje de la raíz del error cuadrático medio (RMSE).
- El algoritmo de regresión que produjo el puntaje de la raíz del error cuadrático medio más bajo fue XGBoost.
- El puntaje más bajo del error de la raíz del error cuadrático medio obtenido fue de 1.7294.

La Figura 25 resume los resultados obtenidos de las actividades realizadas para optimizar el modelo. En el área azul se muestra el puntaje más pequeño de la raíz del error cuadrático medio obtenido en cada iteración del modelo de regresión, en los recuadros de la derecha las principales actividades realizadas para optimizarlo.

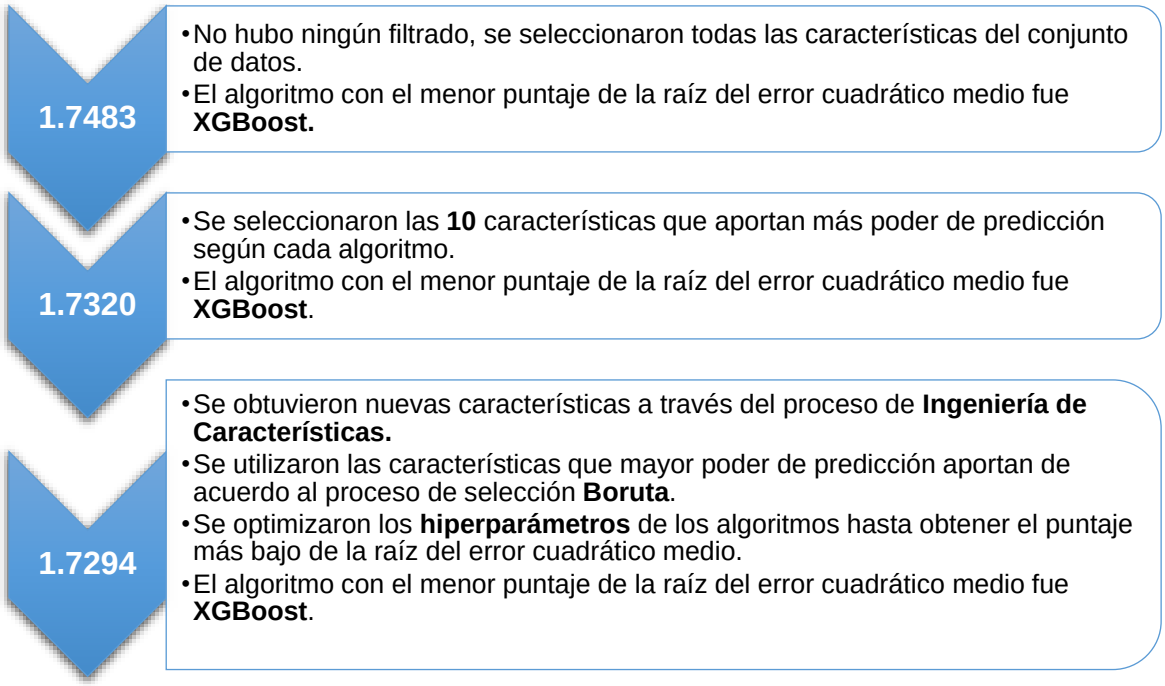

Figura 25. Proceso de optimización del modelo de regresión.

Capítulo 4: Conclusión

1.2 Introducción

Este capítulo resume las conclusiones de la investigación. Se presentan los principales hallazgos, revela las limitaciones del trabajo e identifica áreas apropiadas que pueden ser desarrolladas como trabajos futuros.

1.3 Conclusión

Cumpliendo el objetivo de la investigación cuatro preguntas fueron definidas, las cuales serán discutidas a continuación.

1.3.1 Pregunta 1 de investigación

¿Es posible predecir la probabilidad de compra del cliente utilizando aprendizaje automático?

La investigación muestra que técnicas de aprendizaje automático pueden ser teóricamente empleadas para realizar predicciones efectivas en la probabilidad de compra del cliente. El estudio ha mostrado que es posible predecir si un cliente realizará una compra en el futuro de acuerdo a sus características socioeconómicas y demográficas con una exactitud de 79.70%. Así mismo el estudio ha mostrado además que es posible conocer el ingreso mensual promedio del cliente con un puntaje de la raíz del error cuadrático medio de 1.72, lo cual muestra que la predicción del modelo esta cercana a los valores reales. Lo anterior significa que es posible seleccionar a los clientes prospectos de la empresa para una campaña de mercadotécnica directa. Al conocer qué clientes son más probables a responder un correo o una llamada conlleva a una reducción significante del costo invertido en las campañas de mercadotecnia, lo cual se traduciría a mayores ingresos para la empresa.

1.3.2 Pregunta 2 de investigación

¿Qué técnicas de aprendizaje automático son necesarias para generar modelos que predigan la probabilidad de compra?

Uno de los principales problemas de los modelos predictivos es encontrar un conjunto óptimo de características. Como se menciona en la sección 2.3.1, el incluir demasiadas características conlleva al sobreajuste del modelo. El encontrar el conjunto óptimo de características puede ayudar a mejorar el rendimiento de los modelos de aprendizaje automático. Durante esta investigación se utilizó la biblioteca Boruta para encontrar las

características óptimas del conjunto de datos, lo que también ayudó al ahorro del consumo de recursos utilizados y principalmente a incrementar la exactitud de predicción del modelo.

Por otra parte, otra técnica que juega un papel crucial en los resultados del modelo es la ingeniería de características. Esta técnica es una de las tareas que más tiempo consumen en el proceso de la ciencia de datos debido a que normalmente es realizada manualmente. La ingeniería de características es donde la intuición y la creatividad son tan importantes como el conocimiento técnico. La utilización de esta técnica durante el experimento ayudó a la creación de nuevas características para el modelo a partir de las existentes en el conjunto de datos. Al realizar lo anterior, los modelos mostraron una mejora significativa en su rendimiento, para el modelo de clasificación la mejora fue un incremento del 0.95% en la exactitud de la predicción y para el modelo de regresión la mejora fue de un decremento del puntaje de la raíz del error cuadrático medio de 0.019 .

1.3.3 Pregunta 3 de investigación

¿Qué características del cliente son útiles para predecir la probabilidad de compra?

Dados los resultados presentados en las secciones 3.4.1.5 y 3.4.2.5, se observa que las características más importantes para predecir la probabilidad de compra del cliente son: El ingreso anual, el número de hijos en casa, la edad del cliente, el género y el estado civil. Es entendible que la característica más importante sea el ingreso anual del cliente para determinar si comprará en un futuro una bicicleta, los clientes con mayor poder adquisitivo tienden a comprar más.

La segunda y tercer características más importantes para la predicción de la probabilidad de compra es el número de hijos en casa del cliente y la edad del cliente, de acuerdo a la matriz de correlación mostrada en la Figura 16, existe una correlación moderada entre la variable BikeBuyer y NumberChildrenAtHome, así como una leve correlación entre la variable Age y BikeBuyer. Las correlaciones anteriores señalan, que la variable BIkeBuyer esta relacionada con las variables NumberChildrenAtHome y Age, esto significa que a esas variables tienen influencia en que sea comprador un cliente. Por otra parte, dada la distribución de los datos se observa que la mayoría de los clientes tienden a no tener hijos en su hogar, así mismo la edad de la mayoría de los clientes está entre los 30 y los 50 años.

Las características restantes que proporcionan poder de predicción a los modelos son el género del cliente y su estado civil. Dada la distribución de los datos, existe muy poca diferencia entre la distribución del género del cliente y el estado civil. Sin embargo, dadas las relaciones mostradas en las Figuras 19 y 20, los clientes solteros y en particular los hombres tienden a gastar más en bicicletas.

Se puede concluir que las características de los clientes que aportan mayor poder de predicción para la posibilidad de comprar una bicicleta representan a cliente del sexo masculino solteros, que al no tener hijos en casa tienen más tiempo libre para realizar actividades recreativas como andar en bicicleta.

1.3.4 Pregunta 4 de investigación

¿Cuál de los enfoques propuestos de aprendizaje automático es más preciso para la probabilidad de compra?

El objetivo del experimento fue crear modelos para predecir la probabilidad de compra del cliente. Para el experimento de clasificación binaria, la métrica utilizada para encontrar al mejor modelo fue el puntaje de exactitud más alto. Para validar los modelos, la información fue estratificada como medida de validación cruzada. Los datos inicialmente no eran adecuados para ser utilizados en los algoritmos de aprendizaje automático por lo cual tuvieron que pasar por múltiples transformaciones para ser apropiados para el entrenamiento y pruebas de los modelos. Tres modelos fueron entrenados y probados, los algoritmos seleccionados fueron Gradient Boosting, Random Forest y XGBoost. Con un puntaje de 79.70% de exactitud, el modelo construido utilizando Gradient Boosting fue el modelo obtuvo el porcentaje más alto de los tres.

Así mismo, para el experimento de regresión la métrica utilizada para encontrar al mejor modelo fue el puntaje de la raíz del error cuadrático medio (RMSE) más bajo. Siguiendo los pasos de entrenamiento, pruebas y validación realizados en el experimento de clasificación binaria, fueron utilizados los algoritmos Gradient Boosting, Random Forest y XGBoost para abordar el problema de regresión. Con un puntaje de 1.7294 de la raíz del error cuadrático medio, el modelo Gradient Boosting resultó también para el problema se regresión ser el mejor modelo al obtener el puntaje más bajo.

Con este estudio se probó que los modelos aplicados al conjunto de datos disponible pueden ayudar a los departamentos de mercadotécnica de las empresas en la predicción de la probabilidad de compra del cliente y de esta manera, seleccionar clientes prospecto para una campaña de mercadotecnia que conlleve a optimizar las ganancias del presupuesto invertido en la campaña.

1.4 Limitaciones de la investigación

Esta investigación como todas las investigaciones, no está exenta de sus limitaciones. Las siguientes son algunas limitaciones encontradas durante la investigación:

- Dado que el marco de tiempo del conjunto de datos utilizado es de 2 días, esto puede influenciar de una forma positiva o negativa al modelo de aprendizaje automático. Sería interesante investigar qué marcos de tiempo producen el mejor resultado y cómo la eficiencia de los modelos es influenciada.

- La mayoría de la información presente en el conjunto de datos corresponde a Estados Unidos (42%), las diferencias culturales de los países del conjunto de datos pueden influir en el comportamiento del modelo. Esto significa que los resultados no pueden ser generalizados ya que pueden ser diferentes en otros países. Sin embargo, los algoritmos de aprendizaje automático utilizados para extraer conocimiento siguen siendo los mismos. Por lo tanto, el modelo debe ser editado apropiadamente antes de su uso en tales casos.

- La falta de información psicográfica (actitudes, intereses) y conductuales (tasa de utilización del producto, fidelidad a la marca) de los clientes en el conjunto de datos fue otra limitación en la realización de esta investigación. Debido a esto, no fue posible incluir estos factores en el modelo. Sería de gran interés agregar esas variables en el conjunto de datos y ver si mejoran los modelos.

- Finalmente, los modelos desarrollados en esta investigación no fueron desplegados en ambiente real. Los modelos fueron solamente utilizados como una herramienta para obtener y validar los resultados. Liberar software en la industria es un proceso largo y complejo, el cual va más allá del alcance de esta investigación.

1.5 Trabajo futuro

Las ideas presentadas en el presente trabajo de tesis pueden ser mejoradas o ampliadas en diferentes direcciones. Se proponen las siguientes líneas de trabajo futuro.

- Utilizar más datos. Como se menciona en las limitaciones, el conjunto de datos empleado en este estudio utiliza un marco de tiempo muy corto. Se recomienda utilizar un conjunto de datos cuyo marco de tiempo sea mayor, por ejemplo de 6 meses a 1

año por lo menos para así observar más patrones en los datos , esto permitirá al modelo considerar más casos para generalizar mejor sus predicciones.

- Utilizar más fuentes de datos. Los conjuntos de datos de esta tesis utilizan la información proporcionada por el programa Data Science Professional Project (DAT102x) de Microsoft. El uso de una variedad mayor de conjunto de datos de diversas fuentes puede ayudar a los modelos a encontrar nuevos patrones en los datos.

- Utilizar más algoritmos. Debido a las restricciones de cómputo y tiempo, se seleccionaron 3 de los algoritmos más populares y adecuados para construir los modelos. Sería interesante expandir el número de algoritmos utilizados y comparar su rendimiento.

- Usar valores óptimos para los parámetros de cada algoritmo. A pesar de la utilización del método Random Search durante el proceso de optimización de hiper parámetros para obtener los valores óptimos de cada algoritmo de clasificación, sería interesante probar otra técnica de optimización de hiper parámetros más exhaustiva como Grid Search y ver si es posible encontrar una mejora en el desempeño de los algoritmos.

- Utilizar diferentes métricas. Después de explorar diversas métricas de evaluación, los valores de exactitud para el experimento de clasificación y la raíz del error cuadrático medio para el de regresión fueron seleccionados como los más relevantes. Sería interesante evaluar el comportamiento de los modelos con diferentes métricas.

Anexo A: Lenguajes de programación para Aprendizaje Automático

En la actualidad existen diversos lenguajes de programación para realizar aprendizaje automático. En un estudio realizado para conocer los principales lenguajes de programación para aprendizaje automático utilizados en la industria, se observaron las tendencias de empleos del motor de búsqueda de empleos Indeed.com (Puget, 2016). El sitio Web de búsqueda de empleo con mayor tráfico en los Estados Unidos. Para obtener las principales bibliotecas de cada lenguaje de programación se utilizó información de las bibliotecas de aprendizaje automático más populares alojadas en GitHub (Misiti, 2015), la plataforma de codificación social y servicio de alojamiento de proyectos más popular. De la misma manera en el caso del lenguaje de programación R, la información de las bibliotecas de aprendizaje automático más populares se obtuvo del portal del proyecto R. La Figura 26 resume las principales bibliotecas de Aprendizaje Automático para los lenguajes más populares de encontrado en el estudio.

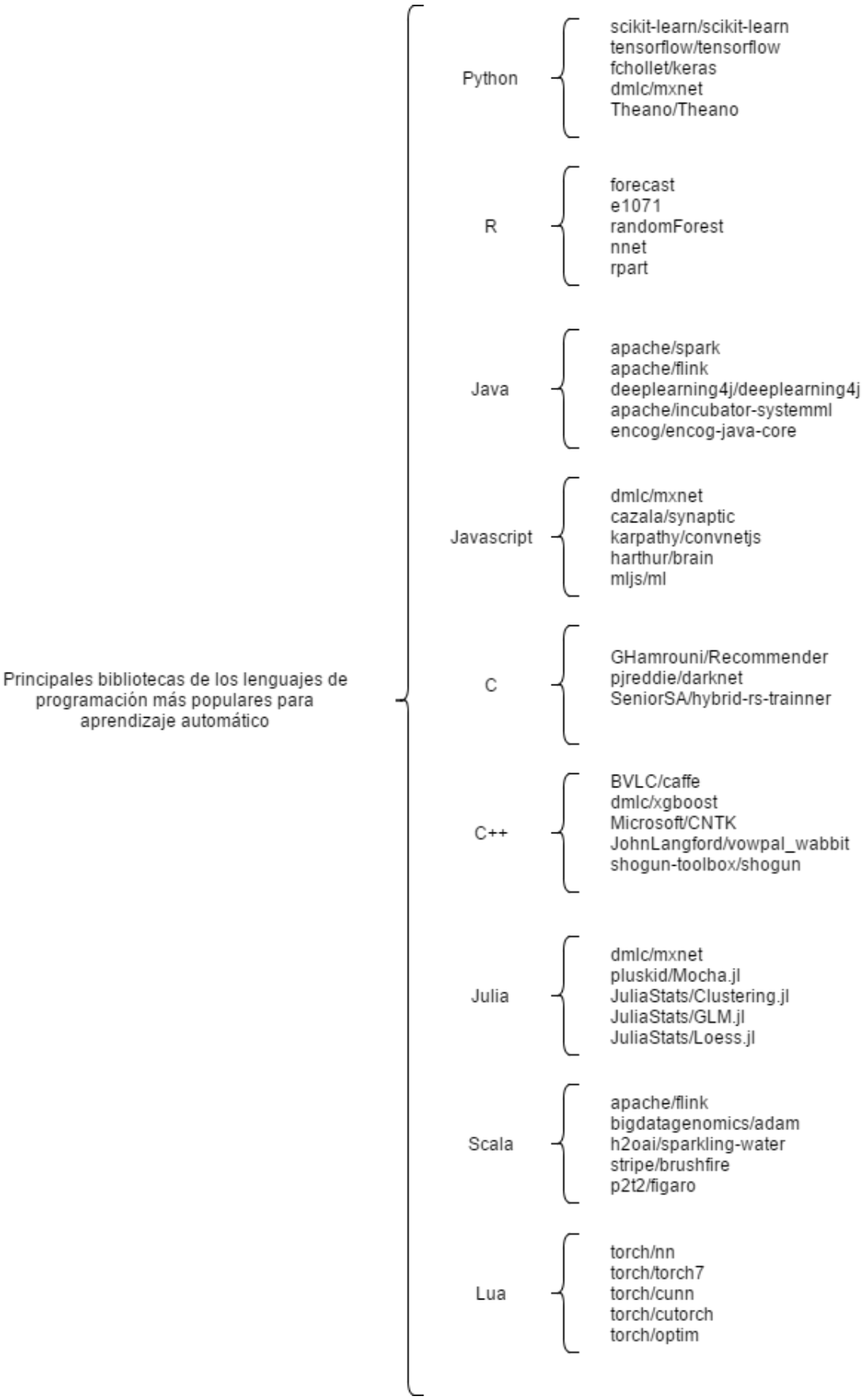

Figura 26. Principales bibliotecas de los lenguajes de programación más populares para aprendizaje automático.

1. Python

Creado en 1991, Python se ha convertido en uno de los lenguajes de programación más populares para la Ciencia de datos (Raschka, 2015:13). Dentro de las principales razones para elegir Python como lenguaje de programación para trabajar con aprendizaje automático se puede mencionar el vasto desarrollo de la comunidad científica. Python cuenta con un ecosistema maduro de paquetes especializados para el análisis y procesamiento de información (Joshi P., Hearty J., Sjardin B. et al., 2016:600).

Aunque el rendimiento de Python frente a lenguajes como C o Java es menor (Harrington, 2012:13), las bibliotecas para realizar análisis y procesamiento de información han sido construidas en Fortran y C para operaciones rápidas con vectores en matrices (Raschka, 2015:13), otorgando a Python ventaja sobre otros lenguajes populares como R y MATLAB (Joshi P., Hearty J., Sjardin B. et al., 2016:600).

1.1. Scikit-Learn

Inició en el 2007 como un proyecto Google Summer of Code por David Cournapeu (Massaron & Boschetti, 2016). Actualmente Scikit-Learn es un proyecto de código abierto bajo la licencia BSD, lo que significa que es libre de usarse, distribuirse, y cualquiera puede acceder al código fuente. Desde su lanzamiento en 2007, se ha convertido en una de las bibliotecas más populares de código abierto para aprendizaje automático en Python (Hackeling, 2014) así como ampliamente utilizada en la industria y la academia (Müller y Guido, 2017:6). Gracias a la gran comunidad de usuarios activos que constantemente desarrollan y mejoran Scikit-Learn, cuenta con una documentación muy elaborada de cada algoritmo de aprendizaje automático supervisado y no supervisado.

1.2. Pandas

Pandas es una biblioteca de alto rendimiento para el análisis de datos en Python desarrollada inicialmente por Wes Mckinney en el 2008 (Anthony, 2015:6) y al igual que otras bibliotecas utilizadas en este trabajo es un proyecto de código abierto bajo la licencia BSD. Como lo describe su autor Wes Mckinney (2012:4), Pandas proporciona estructuras de datos robustas así como funciones diseñadas para trabajar con datos estructurados de forma rápida, fácil y expresiva.

1.3. Seaborn

A pesar de que Matplotlib es la biblioteca más popular en Python para plasmar gráficos y otras visualizaciones en 2D (McKinney, 2012:5), muchas veces se requiere una cantidad significante de optimización manual para generar figuras limpias, de alta calidad y listas para publicar (Rossant, 2015:67). La biblioteca Seaborn ofrece una interfaz de alto nivel para graficar figuras en pocas líneas de código, además se adapta en gran manera con la biblioteca Pandas.

1.4. TensorFlow

Originalmente, los investigadores de Google que formaban parte de Google Brain Team desarrollaron TensorFlow, liberada al público en noviembre de 2015 (Sjardin, Boschetti y Massaron, 2016:26). TensorFlow es una biblioteca de código abierto para cómputo numérico (Abrahams, Hafner, Erwitt y Scarpinelli,2016:23), la cual proporciona funciones para trabajar con las último en arquitecturas de redes neuronales convolucionales y recurrentes para procesamiento de imágenes y texto (Bonnin, 2016:30).

1.5 Keras

Es una biblioteca minimalista y altamente modular para redes neuronales escrita en Python y capaz se ser ejecutada en marcos de trabajo como Theano o TensorFlow (Boschetti y Massaron, 2016:49). El desarrollo de Keras comenzó a principios del 2015 por François Chollet un investigador de aprendizaje automático durante su estancia en Google. Keras permite utilizar la unidad de procesamiento gráfico (*GPU* por sus siglas en Inglés) para acelerar el entrenamiento de las redes neuronales. Es una gran herramienta para el prototipado rápido de ideas (Ketkar, 2017:95) gracias a su muy intuitiva interfaz de programación de aplicaciones (*API* por sus siglas en Inglés), lo cual permite la implementación de redes neuronales con tan sólo pocas líneas de código (Raschka,Julian y Hearty, 2016:456).

1.6 Theano

Es una biblioteca que creada por académicos que ha sido usada a gran escala en cómputo intensivo desde 2007 (Theano Development Team, 2017). Theano permite definir, optimizar y evaluar expresiones matemáticas que involucran arreglos multidimensionales eficientemente (Boaschetti y Massaron, 2016:48), específicamente brinda facilidad al momento de implementar redes neuronales (Zocca, Spacagna, Slater, et al., 2017:121). De acuerdo con Bergstra, Breuleux, Bastien, et al. (2010) , su principal ventaja es la utilización

de la unidad de procesamiento gráfico lo cual permite ejecutar código en paralelo, de esta forma en 2010 Theano superó 1.8 veces el rendimiento sobre la biblioteca de cómputo numérico de Python por defecto NumPy, al ser ejecutado sobre la unidad central de procesamiento (*CPU* por sus siglas en Inglés) y superó 11 veces el rendimiento de NumPy al ser ejecutado en la unidad de procesamiento gráfico.

2. R

R fue originalmente desarrollado por Robert Gentleman y Ross Ihaka del Departamento de Estadística en la Universidad de Auckland en 1993 (Rodríguez, 2015:61). Inicialmente se basó en el lenguaje de programación S creado por John Chambers en los laboratorios Bell en 1976, el cual fue creado con la finalidad de trabajar con estadística de forma más fácil (Mueller y Massaron, 2016:46). Dentro de los beneficios del lenguaje R se encuentran según Rodríguez (2015:61):

- R es software de código abierto
- Es la herramienta más utilizada en la comunidad científica
- Contiene un repositorio llamado CRAN (*Comprenhensive R Archive Network*, en Inglés) el cual proporciona tanto el código fuente y la versión compilada de R así como una gran cantidad de paquetes para diferentes propósitos.
- Posee una comunidad muy activa para consultar con otros y resolver problemas.

Dentro del entorno de trabajo (framework) y bibliotecas de R más descargados en el repositorio CRAN (Misiti, 2015) se encuentran:

2.1 Forecast

Es una biblioteca para R creada y mantenida por el profesor Rob Hyndman de la Universidad de Monash (Smith, 2016). Forecast es un pronóstico estadístico, se utiliza en casi todos los ámbitos de análisis de datos desde predecir el futuro precio de las acciones de la bolsa o tratar de anticipar los cambios climáticos.

2.2 e1071

Es una biblioteca que reagrupa un conjunto de funciones del Departamento de Estadística (antiguamente llamado e1017) de la Universidad TU Wien (R Core Team, 2017) escrita originalmente por David Meyer, Evgenia Dimitriadou, Kurt Hornik, Andreas Weingessel, y Friedrich Leisch. Dentro de las funciones que proporciona se encuentran Transformaciones de Fourier, Clustering difuso, Máquinas de vectores de soporte, entre otras.

2.3 randomForest

Es una biblioteca traída a R por Liaw y Matthew Wiener originada de la versión en Fortan del 2002 de Breiman y Adele Cutler (R Core Team, 2015). La biblioteca randomForest proporciona técnicas de aprendizaje automático como clasificación y regresión basado en bosques de árboles utilizando entradas aleatorias.

2.4 nnet

Es una biblioteca que permite desarrollar y validad los tipos más comunes de redes neuronales ("Visualizing Neural Networks",2013). La biblioteca posee suficiente flexibilidad a través de sus múltiples parámetros para alcanzar un modelo óptimo.

2.5 rpart

Es una biblioteca que implementa la funcionalidad recursiva para particionar árboles de regresión y clasificación publicados en 1984 por Breiman, Friedman, Olshen and Stone (R Core Team, 2017).

3 Java

Java es un lenguaje de programación orientado a objetos que fue creado en 1990 por James Gosling cuando trabajaba en Sun Microsystems. Es uno de los lenguajes de programación más ampliamente usados en el mundo, y ha sido excepcionalmente exitoso en el cómputo empresarial y de negocios (Evans y Flanagan, 2015:3). A pesar de que Java no es muy popular para realizar ciencia de datos como lo son otros lenguajes como Python y R, el autor Grigorev (2017) menciona que las principales ventajas de usar Java frente a sus competidores son:

- Java es un lenguaje de tipado estático, por lo que es más fácil mantener el código y cometer menos errores.
- Su librería estándar para procesamiento de datos es muy rica
- El código de Java es típicamente más rápido en ejecución que el código en lenguajes de script usados para ciencia de datos (como R y Python)
- La mayoría de los marcos de trabajo para procesamiento escalable de datos están escritos en Java o en lenguajes JVM, como lo son Apache Hadoop, Apache Spark o Apache Flink.

- Muy a menudo los sistemas en producción son escritos en Java y el construir modelos en otros lenguajes agrega niveles innecesarios de complejidad. Crear los modelos en Java hacer que sea más fácil integrarlos al producto.

3.1 Apache Spark

Spark comenzó como un proyecto de investigación en el laboratorio AMP de la Universidad de California en Berkeley en 2014 (Dua, Ghotra y Pentreath, 2017:37) donde su principal motivación fue la creación de algoritmos de aprendizaje automático distribuidos. Apache Spark se caracteriza por ser un marco de trabajo que proporciona paralelización al procesamiento de grandes cantidades de datos en los lenguajes de programación como Scala, Java, Python y R. Algunos autores como Shams (2017:297) y Watson (2015:35) mencionan que Apache Spark va un paso adelante comparado con otros sistemas como son Hadoop o el paradigma Map Reduce de Google, esto se debe a que Spark utiliza la caché en memoria para extraer conjuntos de datos, procesarlos y repetir el proceso en lugar de trabajar directamente con datos almacenados en disco. Entre los módulos más populares de Apache Spark como mencionan los autores Sugomori, Kaluza, Soares, et .al (2017:301) se encuentran GraphX para procesamientos de grafos, Spark Streaming para procesamiento en tiempo real de flujos de datos y MLlib como biblioteca de aprendizaje automático para regresión, clasificación, filtrado colaborativo, clústering, entre otros.

3.2 Apache Flink

Comenzó como un proyecto de Apache Incubator en 2014 bajo el nombre de Stratosphere (Deshpande, 2017:30) con la meta de construir la siguiente plataforma de análisis masivo de datos (*Big Data* en Inglés). Apache Flink es un marco de trabajo para el procesamiento de flujos para aplicaciones distribuidas, de alto rendimiento, alta disponibilidad y con flujo de datos precisos (The Apache Software Foundation,2017). Como mencionan los autores John y Misra (2017:164) la principal razón para moverse de Apache Spark a Apache Flink son los cuellos de botella que ocurren en Spark debido a que éste opera en micro lotes para manejar datos lo cual introduce cierta latencia, Flink en cambio procesa eventos en casi tiempo real.

3.3 Deeplearning4j

Deeplearning4j o DL4J es una biblioteca de código abierto para aprendizaje profundo (*Deep Learning* en Inglés) desarrollada por la empresa Skymind, la cual brinda soporte comercial. La descripción de Deeplearning4j por la propia empresa Skymind (Skymind, 2017) menciona

a Deeplearning4j como la primera biblioteca de grado comercial, de código abierto, para aprendizaje profundo distribuido escrita en Java y Scala. Deeplearning4j integrado con Hadoop y Spark, está diseñada para ser usada en ambientes de negocios, en lugar de ser usada como herramienta de investigación. Deeplearning4j incluye soporte para varias estructuras de redes neuronales tales como redes neuronales prealimentadas (*Feedfowawrd Neuronal Networks* en Inglés), redes neuronales máquina de Boltzmann restringida (*Restricted Boltzmann Machine* en Inglés), redes neuronales convolucionales, entre otro tipos (Kaluza, 2016:90)

3.4 SystemML

Es un sistema escalable y flexible de aprendizaje automático. Dentro de las características más importantes de SystemML (Apache Software Foundation, 2017) proporciona personalización de algoritmos utilizando R o Python, múltiples modos de ejecución (Spark MLContext, Spark Batch, Hadoop Batch, entre otros) y optimización automática basada en datos y características de clúster para asegurar tanto la eficiencia como la escalabilidad.

3.5 Encog

Es un marco de trabajo avanzado para aprendizaje automático que ha estado en desarrollo activo desde 2008. Encog soporta una variedad de algoritmos avanzados, así como clases para normalizar y procesar datos (Heaton Research, 2017). De acuerdo con el sitio Web oficial, Encog soporta algoritmos tales como máquinas de vectores de soporte *(Support Vector Machines* en Inglés), redes neuronales, redes bayesianas, algoritmos genéticos, entre otros. Los algoritmos de entrenamiento son multi-hilo y escalan bien en hardware multi-núcleo.

4 JavaScript

Es un lenguaje de alto nivel, dinámico y no tipado (Zaidi,2017:1). JavaScript fue creado en 1995 por Brandon Eich mientras trabajaba en Netscape incorporándolo al navegador Web Navigator 2.0 (Refsnes et al., 2010:2). Javascript fue considerado tan sólo un lenguaje para añadir mejoras y pequeñas aplicaciones (*Widgets* en Inglés) a los sitios Web, sin embargo, ha evolucionado hasta tener su propio ecosistema completo con la creación de Node.js (Krol, 2014:47). Node.js fue desarrollado en 2009 por Ryan Dahl utilizando el entorno de tiempo de ejecución (runtime en Inglés) V8 de Google (Dayley, 2014:58). Node.js hace uso del moto V8 para embeberla en una aplicación autónoma (*Standalone* en Inglés) que puede ejecutar JavaScript fuera de un navegador Web (Cummings, 2016:32). La principal ventaja de Node.js

es su tecnología sin bloque de dispositivos de entrada y salida, por lo que es muy rápido y usa cada ciclo de la unidad central de procesamiento de forma óptima (Mejía, 2015:48).

4.1 MXNet

Es una biblioteca de aprendizaje profundo permitiendo mezclar programación simbólica e imperativa para maximizar eficiencia y productividad (DMLC, 2017). MXNet utiliza paralelización en sus operación simbólicas e imperativas al vuelo, así mismo su capa de grafos optimiza la ejecución de operaciones simbólicas de forma rápida y eficiente. Finalmente, MXNet es una biblioteca portable y ligera que puede escalar efectivamente en múltiples unidades de procesamiento gráfico.

4.2 Synaptic

Synaptic es una biblioteca de redes neuronales para Node.js así como para navegadores utilizado el algoritmo para redes neuronales recurrentes de Monner (2010). Se describe como un algoritmo de arquitectura libre, capaz de construir y entrenar cualquier tipo de arquitecturas de redes neuronales de primer orden incluso de segundo orden (Cazala, 2017).

4.3 ConvNetJS

Es una biblioteca para entrenar modelos de aprendizaje profundo (redes neuronales) en el navegador. Al ser ejecutada en el navegador no requiere de instalación de software, compiladores, unidades de procesamiento gráfico, sólo se necesita abrir una pestaña en el navegador para comenzar a entrenar (Karpathy, 2017). Actualmente soporta algoritmos de clasificación y regresión, redes neuronales convolucionales así como un módulo experimental para aprendizaje por reforzamiento.

4.4 ml.js

Es un conjunto de bibliotecas desarrolladas por una organización de GitHub de aprendizaje automático y herramientas para análisis numérico en JavaScript para Node.js y el navegador denominada ml.js (ml.js, 2017). Dentro de las bibliotecas para aprendizaje automático se encuentran algoritmos para máquinas de vectores de soporte, Naive Bayes, k vecinos más próximos, K-means, entre otros.

5 C

El lenguaje de programación C fue desarrollado en 1970 por Dennis Ritchie cuando trabajaba en AT&T Bell Laboratories (Kochan, 2015). C es un lenguaje de programación de propósito general, el cual puede ser considerado un lenguaje de nivel intermedio debido a que contiene características de lenguajes de alto nivel como Pascal y de lenguajes de bajo nivel como Ensamblador (Pragada,2015). C no está específicamente diseñado para un área de aplicación en particular, gracias a la rapidez del lenguaje es adecuado para aplicaciones científicas y de negocios (Cearley, 2016).

5.1 Recommender

Es una biblioteca en C para recomendaciones/sugerencias de un producto utilizando filtrado colaborativo (Hamrouni, 2017). La biblioteca analiza la retroalimentación de los usuarios y sus preferencias por ciertos artículos, de esta manera aprende un patrón y predice los productos más adecuados para un usuario en particular. Dentro de las características sobresalientes de la biblioteca se encuentra el filtrado colaborativo, no dependencias externas, ejecución rápida (81 segundos para 10 millones de registros) así como uso de poca memoria de acceso aleatorio (*RAM*, por sus siglas Inglés) (160 MB para 10 millones de registros).

5.2 Darknet

Es un marco de trabajo de código abierto para redes neuronales escrito en C y en CUDA (siglas del Inglés *Compute Unified Device Architecture*). Dentro de las características ofrecidas por Darknet se encuentra la rapidez de instalación, detección en tiempo real de objetos, clasificación de imágenes y redes neuronales recurrentes (Redmon, 2017).

5.3 Hybrid RS Trainner

Es una biblioteca para entrenar diferentes sistemas de recomendación como filtrado colaborativo, filtrado basado en contenido o híbrido al vuelo (Senior Sistemas S.A, 2016). Por el momento la biblioteca sólo soporta filtrado colaborativo de usuario x usuario, el cual utiliza una matriz de usuarios x artículos donde cada fila representa una calificación de un usuario sobre un artículo, las recomendaciones son generadas observando las similitudes entre los usuarios.

6. C++

Es un lenguaje de programación compilado, multipropósito, estáticamente tipado y con distinción entre mayúsculas y minúsculas, soporta programación genérica, de procedimientos y orientada a objetos (Johansen,2015:59). C++ nació como una mejora del lenguaje de programación C al incluir clases en 1980 por Bjarne Stroustrup (McGrath,2011:8). La principal razón por la cual C++ ha tenido tanto éxito es la rapidez del lenguaje, C++ fue creado como un lenguaje de bajo nivel cercano al nivel de programación binaria (Acodemy, 2015:7), lo cual lo hace mucho más rápido y flexible que otros lenguajes como Java, C# o VB.

6.1 Caffe

Es un marco de trabajo para aprendizaje profundo enfocado en modularidad, rapidez y expresión (Berkeley AI Research, 2017). Dentro de las cualidades que ofrece caffe se encuentran rapidez de procesamiento, así como una arquitectura que le permite cambiar entre unidad de procesamiento gráfico y unidad central de procesamiento de forma sencilla. Caffe ha sido aceptado rápidamente por la comunidad de código abierto que durante su primer año fue bifurcado (del Inglés *Forked*) por más de 1000 desarrolladores quienes contribuyeron ampliamente a las mejoras de caffe.

6.2 CNTK

Es CNTK (siglas del Inglés *Cognitive Toolkit*) un conjunto de herramientas de aprendizaje profundo unificado, el cual utiliza redes neuronales como una serie de pasos computacionales a través de un grafo dirigido (Microsoft, 2017). CNTK pretende ser una alterativa a TensorFlow de Google, presentando una rapidez 5 veces mayor en redes recurrentes, una API (siglas del Inglés *Application Programming Interface*) para lenguajes de bajo nivel y de alto nivel, escalabilidad sobre miles de unidades de procesamiento gráfico así como su extensibilidad hacia otro lenguajes de aprendizaje automático como Python.

6.3 Vowpal Wabbit

Es una biblioteca que implementa aprendizaje automático en línea (del Inglés *Online Machine Learning*) donde la información los datos están en forma secuencial y son actualizados para generar el predictor opuesto a las técnicas de aprendizaje por lotes donde el predictor es generado aprendiendo del conjunto de datos entero (Langford, 2017). El aprendizaje automático en línea es utilizado cuando es computacionalmente impráctico entrenar sobre todo el conjunto de datos, esta característica puede además de evitar que el programa falle cuando el conjunto de datos es muy grande, ayudar a localizar errores en segundos que

requerirían minutos u horas debido a la falta de retroalimentación de otras técnicas de aprendizaje automático.

6.4 Shogun

Es una caja de herramientas para aprendizaje automático con una amplia gama de métodos eficientes y unificados (Shogun, 2017). Shogun soporta una gran variedad de lenguajes de programación (Python, Octave, R, Scala, Lua, etc.), múltiples plataformas (Linux/Unix, MacOS y Windows) así como un entorno en la nube para probarlo.

7. Julia

Es un lenguaje de programación joven creado en 2009 por Stefan Karpinski, Jeff Bezanson y Viral Shah en grupo de Cómputo Aplicado del Instituto Tecnológico de Massachusetts bajo la supervisión del Prof. Alan Edelman (Joshi,2016:7). De acuerdo con el autor Balbaert (2015) Julia se distingue por ser un lenguaje de programación de alto nivel dinámico diseñado para ser efectivo tanto para cómputo numérico como científico, así mismo Julia posee un compilador virtual de bajo nivel basado en la técnica compilación en tiempo de ejecución (*Just In Time* en Inglés) que le permite obtener un rendimiento cercano a los lenguajes como C o C++.

7.1 Mocha

Mocha es un marco de trabajo de aprendizaje profundo para Julia, inspirado en el marco de trabajo Caffe de C++ (Zhang, 2017). Mocha posee una arquitectura modular para extender su funcionalidad agregando componentes personalizados además de la adopción del formato HDF5 para almacenar grandes cantidades de datos lo cual le permite tener compatibilidad con Matlab, Python (NumPy), así como Caffe.

7.2 JuliaStats

La activa comunidad de Julia ha concentrado sus trabajos en el grupo JuliaStats, donde se concentran bibliotecas sobre estadística y aprendizaje automático (JuliaStats, 2017). Dentro de las bibliotecas que ofrece JuliaStats se encuentra StatsBase, donde como su nombre lo indica proporciona las funcionalidades básicas para trabajar con estadística descriptiva. También se encuentra la biblioteca DataFrames para trabajar con datos tabulares como en Python. Finalmente ofrece gran variedad de algoritmos de aprendizaje automático como son clústering, modelos lineales generalizados, entre otros.

8. Scala

Es un lenguaje de programación multiparadigma que mezcla la programación funcional con la programación orientada a objetos en un lenguaje estáticamente tipado (Odersky, Spoon y Venners,2016:18) creado en 2004 por Martin Odersky cuando trabajaba como profesor en la École Polytechnique Fédérale de Lausanne en Suiza (Jančauskas,2016:43). Dentro de las características que destacan a Scala según los autores Wampler y Payne (2014:1) se encuentran:

- **Una Máquina Virtual de Java (*JVM* en Inglés) y el lenguaje JavaScript.** Scala toma ventaja del rendimiento y optimización de la MVJ, así como el ecosistema de herramientas y bibliotecas de Java. Existe además una implementación experimental llamada Scala.js para JavaScript.
- **Tipado estático**. Scala hace uso del tipado estático para reparar diversos defectos de Java al eliminar mucho el tipado innecesario.
- **Paradigma mezclado Orientado a Objetos.** Scala soporta completamente la Programación Orientada a Objetos añadiendo nuevas características como los rasgos (*Traits* en Inglés) y la implementación de tipos de datos usando la clase *mixin*.
- **Paradigma mezclado Funcional.** Scala soporta completamente la Programación funcional para solventar los problemas actuales de concurrencia y datos masivos.
- **Una sintaxis flexible y elegante.** Scala proporciona facilidades para la construcción de lenguajes de dominio específico (*Domain-Specific Language* en Inglés).
- **Arquitecturas escalables.** Scala posee la capacidad de escribir scripts pequeños que puede fácilmente ser integrados a sistemas más grandes distribuidos, gracias al uso de rasgos, miembros de tipado abstractos, clases anidadas y tipos de datos auto explícitos.

8.1 ADAM

Es una plataforma para el análisis genómico con formatos especializados usando Apache Avro, Apache Spark y Apache Parquet (Big Data Genomics, 2017). ADAM propone una solución a los problemas de escalabilidad e interoperabilidad que presentan las técnicas actuales para el manejo de las secuencias de ADN (Ácido desoxirribonucleico) y ARN (ácido ribonucleico). Utilizando Apache Spark, ADAM proporciona formatos especializados para estructuras estándar usadas en análisis genómico: lecturas mapeadas (*Mapped Read* en Inglés), representación de regiones genómicas y variantes por medio de cómputo en memoria y la tolerancia a fallos distribuida en paralelo sin utilizar los métodos clásicos que generan cuellos de botella como son las operaciones de lectura y escritura intermedias en disco.

8.2 Sparkling Water

Sparkling Water integra el motor de aprendizaje automático de H2O (una plataforma en memoria para aprendizaje automático, escalable y distribuidos) con Spark (H2Oai, 2017). Dentro de las funcionalidades que proporciona se encuentran el poder intercambiar estructuras de datos de Spark entre estructuras H2O así como construcción de bloques para crear aplicaciones de aprendizaje automático utilizando las interfaces de Spark y H2O.

8.3 Brushfire

Brushfire es un marco de trabajo de aprendizaje supervisado distribuido para modelos de árboles de decisión (*Decision Tree* en Inglés) en Scala (Brushfire, 2017). Actualmente Brushfire soporta clasificadores binarios y multi-clase, validación cruzada y bosques aleatorios (*Random Forests* en Inglés), importancia de características, así como cómputo distribuido en Hadoop/Scalding.

8.4 Figaro

Es un lenguaje de programación probabilístico que soporta el desarrollo de modelos probabilísticos robustos y proporciona algoritmos de razonamiento que pueden ser aplicados a esos modelos para sacar conclusiones útiles a partir de la evidencia (Figaro, 2017). Los modelos de Figaro son estructuras de datos de Scala, los cuales utilizan algoritmos de razonamiento incorporados en Figaro que pueden ser aplicados automáticamente en nuevos modelos.

9. Lua

Es un lenguaje de programación ligero, rápido e incrustable creado en 1993 en la Pontifícia Universidad Católica do Rio de Janeiro por Roberto Lerusalimschy, Waldermar Celes y Luiz Henrique de Figueiredo (Toal et al,2017:45). De acuerdo con uno de los autores (Lerusalimschy,2016), Lua fue diseñado para ser integrado con software escrito en C/C++ obteniendo una gran extensibilidad con otros lenguajes de programación como Java, C# y Python. La simplicidad de Lua lo convierten en un lenguaje fácil de aprender, pero también muy ligero. Finalmente, la portabilidad de Lua es muy grande gracias a la utilización del compilador estándar ISO de C, permitiendo ejecutar Lua en todos los ambientes UNIX, Windows, Android, iOS, OS X, consolas de videojuegos, etc.

9.1 Torch

Torch es un marco de trabajo de cómputo científico con un amplio soporte para algoritmos de aprendizaje automático que utilicen principalmente la unidad de procesamiento gráfico (Torch, 2017). Torch ofrece gran variedad de bibliotecas dentro de las cuales se encuentra aprendizaje automático, visión artificial, procesamiento de señales, cómputo paralelo, y otras más gracias al apoyo de la comunidad Lua. La principal característica de Torch son las bibliotecas que ofrece para redes neuronales utilizadas para aprendizaje profundo, las cuales pueden ser paralelizadas y ejecutadas sobre unidades de procesamiento gráfico, así como unidades centrales de procesamiento en forma eficiente.

Bibliografía

Abrahams, S., Hafner, D., Erwitt, E., & Scarpinelli, A. (2016). *Tensorflow for Machine Intelligence*. Bleeding Edge Press.

Acodemy. (2015). *C++: Learn C++ In A Day!* Acodemy.

Al-Thubaity, A., & Al-Subaie, A. (2015). Effect of Word Segmentation on Arabic Text Classification. *Journal of IEEE*, 127–131.

Aler, R., Galván, I. M., Ruiz-Arias, J. A., & Gueymard, C. A. (2017). Improving the separation of direct and diffuse solar radiation components using machine learning by gradient boosting. *Solar Energy*, *150*, 558–569. https://doi.org/10.1016/j.solener.2017.05.018

Anthony, F. (2015). *Mastering Pandas*. Packt Publishing.

Balbaert, I. (2015). *Getting started with Julia Programming*. Packt Publishing. https://doi.org/10.1007/s13398-014-0173-7.2

Barman, D., Kumar Shaw, K., Tudu, A., & Chowdhury, N. (2016). *Information Systems Design and Intelligent Applications*. (S. C. Satapathy, J. K. Mandal, S. K. Udgata, & V. Bhateja, Eds.), *Advances in Intelligent Systems and Computing* (Vol. 435). New Delhi: Springer India. https://doi.org/10.1007/978-81-322-2757-1

Bergstra, J., Breuleux, O., Bastien, F. F., Lamblin, P., Pascanu, R., Desjardins, G., … Bengio, Y. (2010). Theano: a CPU and GPU math compiler in Python. *Proceedings of the Python for Scientific Computing Conference (SciPy)*, (Scipy), 1–7.

Birks, H. J. B., Lotter, A. F., Juggins, S., & Smol, J. P. (2012). *Tracking Environmental Change Using Lake sediments: Volume 5: Data Handling and Numerical Techniques*. https://doi.org/10.1007/978-94-007-2745-8

Bloice, M., & Holzinger, A. (2016). *A Tutorial on Machine Learning and Data Science Tools with Python* (Vol. 9605). https://doi.org/10.1007/978-3-319-50478-0

Bonnin, R. (2016). *Building Machine Learning Projects with TensorFlow*. Packt Publishing.

Bowles, M. (2015). *Machine Learning in Python*. (Wiley, Ed.).

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Brown, I., & Mues, C. (2012). An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Systems with Applications*, *39*(3), 3446–3453. https://doi.org/10.1016/j.eswa.2011.09.033

Brushfire. (2017). Brushfire. Recuperado el 25 de julio de 2017, a partir de https://github.com/stripe/brushfire

Bulut, A. (2015). RightInsight: Open source architecture for data science. *Lecture Notes in Informatics (LNI), Proceedings - Series of the Gesellschaft fur Informatik (GI)*, *242*, 151–160.

Candanedo, L. M., Feldheim, V., & Deramaix, D. (2017). Data driven prediction models of energy use of appliances in a low-energy house. *Energy and Buildings*, *140*, 81–97. https://doi.org/10.1016/j.enbuild.2017.01.083

Cazala. (2017). synaptic. Recuperado el 6 de agosto de 2017, a partir de https://github.com/cazala/synaptic

Cearley, D. (2016). *C Programming: Language: The Ultimate Beginner's Guide*. EasyProgramming.

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Kdd*, 1–10. https://doi.org/10.1145/2939672.2939785

CICOM. (2016). *Valor de la Industria de Publicidad y Mercadotecnia en México 2015*.

Cielen, D., Meysman, A. D. B., & Ali, M. (2016). *Introducing Data Science*. Manning.

Cummings, H. (2916). *Learning Node.js for .NET Developers*. Packt Publishing.

Darmatasia, & Ary. (2016). Predicting the Status of Water Pumps Using Data Mining Approach, 57–64.

Das, S. (2016). *Data Science Using Oracle Data Miner and Oracle R Enterprise*. Berkeley, CA: Apress. https://doi.org/10.1007/978-1-4842-2614-8

Dayley, B. (2014). *Node.js, MongoDB, and AngularJS Web Development*. Addison-Wesley.

Dean, J. (2014). *Big Data, Data Mining, and Machine Learning*. (Wiley, Ed.).

Deshpande, T. (2017). *Learning Apache Flink*. Packt Publishing. https://doi.org/10.1002/ejoc.201200111

Diapouli, M., Petridis, M., Evans, R., & Kapetanakis, S. (2016). *Research and Development in Intelligent Systems XXXIII*.

DMLC. (2017). MXNet. Recuperado el 3 de agosto de 2017, a partir de https://github.com/apache/incubator-mxnet

Dua, R., Ghotra, M. S., & Pentreath, N. (2017). *Machine Learning with Spark*. Packt Publishing.

Figaro. (2017). Figaro. Recuperado el 28 de julio de 2017, a partir de https://github.com/p2t2/figaro

Flanagan, D., & Evans, B. (2015). *Java in a Nutshell*. O'Reilly Media.

Fontama, V., Barga, R., & Tok, W. H. (2015). *Predictive Analytics with Microsoft Azure Machine Learning* (2nd ed.). https://doi.org/10.1017/CBO9781107415324.004

Foreman, J. W. (2013). *Data Smart: Using Data Science to Transform Information into Insight*. (J. W. & S. Inc, Ed.) (1a ed.).

Foundation, A. S. (2017). Apache SystemML. Recuperado el 26 de julio de 2017, a partir de https://systemml.apache.org/

Foundation, T. A. S. (2017). Apache Flink. Recuperado el 5 de agosto de 2017, a partir de https://flink.apache.org/

Friedman, J. H. (2001). Greedy Function Aproximation: A Gradient Boosting Machine. *The Annals of Statistics*, *29*(5), 1189–1232. https://doi.org/10.1214/aos/1013203451

GDMA. (2017). *The Global Review of Data-Driven Marketing and Advertising*. https://doi.org/10.1057/dddmp.2015.7

Geng, L. G. L., & Ma, J. M. J. (2008). TSK Fuzzy Inference System Based GARCH Model for Forecasting Exchange Rate Volatility. *2008 Fifth International Conference on Fuzzy Systems and Knowledge Discovery*, *3*, 103–107. https://doi.org/10.1109/FSKD.2008.228

Genomics, B. D. (2017). ADAM. Recuperado el 8 de julio de 2017, a partir de https://github.com/bigdatagenomics/adam

Gollapudi, S. (2016). *Practical Machine Learning*. (Packt Publishing, Ed.).

Gondane, R., & Devi, S. (2015). *Classification Using Rough Random Forest*. Springer International Publishing.

Grigorev, A. (2017). *Mastering Java for Data Science*. Packt Publishing.

Grus, J. (2015). *Data Science from Scratch*. O'Reilly Media.

Gupta, A., Chetty, N., & Shukla, S. (2015). A classification method to classify high dimensional data. *2015 International Conference on Computing, Communication and Security, ICCCS 2015*. https://doi.org/10.1109/CCCS.2015.7374132

H2Oai. (2017a). Sparkling Water. Recuperado el 27 de julio de 2017, a partir de https://github.com/h2oai/sparkling-water

H2Oai. (2017b). Sparkling Water. Recuperado el 28 de julio de 2017, a partir de https://github.com/h2oai/sparkling-water

Hackeling, G. (2014). *Mastering Machine Learning with scikit-learn*.

Hamrouni. (2017). Recommender. Recuperado el 9 de agosto de 2017, a partir de https://github.com/GHamrouni/Recommender

Harrington, P. (2012). *Machine Learning in Action*. (Manning, Ed.).

Henrik Brink, Joseph Richards, M. F. (2017). *Real World Machine Learning*.

Huang, R., Zhou, J., Gu, X., Zhang, Y., & Bovik, A. C. (2016). Comparison of regressors on 3D visual discomfort prediction. En *2016 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)* (pp. 1–6). IEEE. https://doi.org/10.1109/BMSB.2016.7521925

Huynh, T., Gao, Y., Kang, J., Wang, L., Zhang, P., & Shen, D. (2015). Multi-source Information Gain for Random Forest: An Application to CT Image Prediction from MRI Data. En *IEEE Signal Processing Magazine* (Vol. 27, pp. 321–329). https://doi.org/10.1007/978-3-319-24888-2_39

Ierusalimschy, R. (2016). *Programming in Lua*. Lua.org.

Jancauskas, V. (2016). *Scientific Computing with Scala*. Packt Publishing.

John, T., & Misra, P. (2017). *Data Lake for Enterprises*. Packt Publishing.

Joshi, A. (2016). *Julia for Data Science*. Packt Publishing.

Joshi, P., Hearty, J., & Sjardin, B. (2016). *Python Real World Machine Learning*. Packt Publishing.

Julian, D. (2016). *Designing Machine Learning Systems with Python*. (Packt Publishing, Ed.).

JuliaStats. (2017). Julia Statistics. Recuperado el 9 de agosto de 2017, a partir de https://github.com/JuliaStats

Kaluza, B. (2016). *Machine Learning in Java*. Packt Publishing.

Karpathy. (s/f). ConvNetJS. Recuperado el 8 de agosto de 2017, a partir de http://cs.stanford.edu/people/karpathy/convnetjs/

Ketkar, N. (2017). *Deep Learning with Python*. Apress. https://doi.org/10.1007/978-1-4842-2766-4

Kochan, S. (2015). *Programming in C*. Addison-Wesley.

Kolarovszki, P., Tengler, J., & Majerþáková, M. (2016). The new model of customer segmentation in postal enterprises. *Procedia -Social and Behavioral Sciences*, *230*(May), 121–127. https://doi.org/10.1016/j.sbspro.2016.09.015

Krol, J. (2014). *Web Development With Mongodb and Node.js*. Packt Publishing.

Kulkarni, P. (2012). *Reinforcement and Systemic Machine Learning for Decision Making*. Wiley.

Langford, J. (2017). Vowpal Wabbit. Recuperado el 6 de agosto de 2017, a partir de https://github.com/JohnLangford/vowpal_wabbit

Laradji, I. H., Alshayeb, M., & Ghouti, L. (2015). Software defect prediction using ensemble learning on selected features. *Information and Software Technology*, *58*, 388–402. https://doi.org/10.1016/j.infsof.2014.07.005

Lessmann, S., & Voß, S. (2008). Supervised Classification for Decision Support in Customer Relationship Management. *Intelligent Decision Support*, 231–253. https://doi.org/10.1007/978-3-8349-9777-7_14

Li, D. Y., Xu, W., Zhao, H., & Chen, R. Q. (2009). A SVR based forecasting approach for real estate price prediction. *Proceedings of the 2009 International Conference on Machine Learning and Cybernetics*, *2*(July), 970–974. https://doi.org/10.1109/ICMLC.2009.5212389

Lu, J., Lu, D., Zhang, X., Bi, Y., Cheng, K., Zheng, M., & Luo, X. (2016). Estimation of elimination half-lives of organic chemicals in humans using gradient boosting machine. *Biochimica et Biophysica Acta - General Subjects*, *1860*(11), 2664–2671. https://doi.org/10.1016/j.bbagen.2016.05.019

Mahdiloo, M., Noorizadeh, A., & FarzipoorSaen, R. (2014). Optimal direct mailing modelling based on data envelopment analysis. *Expert Systems*, *31*(2), 101–109. https://doi.org/10.1111/exsy.12011

Maroof, M., Jamil, M., & Jamil, B. (2016). Development of Models for the Estimation of Global Solar Radiation Over Selected Stations in India. En P. Grammelis (Ed.), *Energy,Transportation and Global Warming* (pp. 149–160). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-30127-3

Marsland, S. (2015). *Machine Learning: An Algorithmic Perspective*. Chapman and Hall/CRC.

Massaron, L., & Boschetti, A. (2016). *Regression Analysis with Python*. Packt Publishing.

McGrath, M. (2011). *C++ Programming in Easy Steps*. In Easy Steps Limited.

Mckinney, W. (2012). *Python for Data Analysis*. O'Reilly Media.

Mehryar, M., Rostamizadeh, A., & Talwalkar, A. (2012). *Foundations of Machine Learning*. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial*

Intelligence and Lecture Notes in Bioinformatics) (Vol. 17). MIT Press.

Mejía, A. (2015). *Building an E-Commerce Application with MEAN*. Packt Publishing.

Microsoft. (2017). Microsoft Cognitive Toolkit (CNTK),. Recuperado el 9 de agosto de 2017, a partir de https://github.com/Microsoft/CNTK

Misiti, J. (2015). awesome-machine-learning. Recuperado el 28 de marzo de 2017, a partir de https://github.com/josephmisiti/awesome-machine-learning

Mitik, M., Korkmaz, O., Karagoz, P., Toroslu, I. H., & Yucel, F. (2017). Data Mining Based Product Marketing Technique for Banking Products. *IEEE International Conference on Data Mining Workshops, ICDMW*, 552–559. https://doi.org/10.1109/ICDMW.2016.0085

Ml.js. (2017). ml.js. Recuperado el 8 de agosto de 2017, a partir de https://github.com/mljs/ml

Monner, D., & Reggia, J. A. (2012). A generalized LSTM-like training algorithm for second-order recurrent neural networks. *Neural Networks*, *25*(301), 70–83. https://doi.org/10.1016/j.neunet.2011.07.003

Mueller, J. P., & Massaron, L. (2016). *Machine Learning for Dummies*. Wiley. https://doi.org/10.1017/CBO9781107415324.004

Müller, A. C., & Guido, S. (2016). *Introduction to Machine Learning with Python*. https://doi.org/10.1017/CBO9781107415324.004

Murphy, K. (2012). *Machine Learning: A Probabilistic Perspective*. *Machine Learning*. MIT Press.

O'Neil, C., & Schutt, R. (2014). *Doing Data Science*. (O'Reilly Media, Ed.).

Odersky, M., Spoon, L., & Venners, B. (2016). *Praise for the earlier editions of Programming in Scala*. Artima Inc.

Ozdemir, S. (2016). *Principles of Data Science* (1a ed.). Packt Publishing.

Poona, N. K., & Ismail, R. (2013). Reducing hyperspectral data dimensionality using random forest based wrappers. En *2013 IEEE International Geoscience and Remote Sensing Symposium - IGARSS* (Vol. 33, pp. 1470–1473). IEEE. https://doi.org/10.1109/IGARSS.2013.6723063

Pragada, S. (2015). *C Language for Beginners*. Srikanth Technologies.

Provost, F., & Fawcett, T. (2013). *Data science for business*. (Oreilly, Ed.).

Puget, J. (2016). The Most Popular Language For Machine Learning Is. Recuperado el 5 de marzo de 2017, a partir de https://www.ibm.com/developerworks/community/blogs/jfp/entry/What_Language_Is_Be

st_For_Machine_Learning_And_Data_Science

R Core Team. (2017). e1071. Recuperado el 25 de junio de 2007, a partir de https://cran.r-project.org/web/packages/e1071/index.html

Raschka, S. (2015). *Python Machine Learning*. Packt Publishing. https://doi.org/10.1007/s13398-014-0173-7.2

Raschka, S., Julian, D., & Hearty, J. (2016). *Python: Deeper Insights into Machine Learning*.

Redmon. (2017). Darknet. Recuperado el 10 de agosto de 2017, a partir de https://pjreddie.com/darknet/

Refsnes, S., Refsnes, K. J., & Refsnes, J. E. (2010). *Learn JavaScript and Ajax with w3Schools*. Wiley.

Research, B. A. (2017). Caffe. Recuperado el 8 de agosto de 2017, a partir de http://caffe.berkeleyvision.org/

Research, H. (2017). Encog Machine Learning Framework. Recuperado el 28 de julio de 2017, a partir de http://www.heatonresearch.com/encog/

Richert, W., & Coelho, L. P. (2015). *Building Machine Learning Systems with Python, Second Edition*. *Book*. https://doi.org/10.1007/s13398-014-0173-7.2

Rodriguez, E. (s/f). *Unsupervised Learning with R*.

Rossant, C. (2015). *Learning IPython for Interactive Computing and Data Visualization*. Packt Publishing.

S.A, S. S. (2016). Hybrid RS Trainner. Recuperado el 10 de agosto de 2017, a partir de https://github.com/SeniorSA/hybrid-rs-trainner

Saba, T., Rehman, A., & AlGhamdi, J. S. (2017). Weather forecasting based on hybrid neural model. *Applied Water Science*, 1–6. https://doi.org/10.1007/s13201-017-0538-0

Shams, R. (2017). *Java Data Science Cookbook*. Packt Publishing.

Shogun. (2017). The Shogun Machine Learning Toolbox. Recuperado el 8 de agosto de 2017, a partir de https://github.com/shogun-toolbox/shogun

Sjardin, B., Massaron, L., & Boschetti, A. (2016). *Large Scale Machine Learning with Python*. Packt Publishing.

Skymind. (2017). Open-Source Libraries. Recuperado el 28 de julio de 2007, a partir de https://skymind.ai/open-source

Smith, D. (2017). Updates to the “forecast” package for R. Recuperado el 25 de junio de 2017,

a partir de http://blog.revolutionanalytics.com/2016/06/updates-to-the-forecast-package-for-r.html

Souza, A. M., Soares, F. M., Kaluža, B., & Sugomori, Y. (2017). *Deep Learning: Practical Neural Networks with Java*. Packt Publishing.

Tan, D.-W., Yeoh, W., Boo, Y. L., & Liew, S.-Y. (2013). The Impact of Feature Selection: A Data-mining Application in Direct Marketing. *Intelligent Systems in Accounting, Finance and Management*, *20*(1), 23–38. https://doi.org/10.1002/isaf.1335

Team, R. C. (2015). RandomForest. Recuperado el 25 de junio de 2017, a partir de https://cran.r-project.org/web/packages/randomForest/randomForest.pdf

Team, R. C. (2017). rpart. Recuperado el 26 de julio de 2017, a partir de https://cran.r-project.org/web/packages/rpart/rpart.pdf

Teng, G., He, C., & Gu, X. (2014). Response model based on weighted bagging GMDH. *Soft Computing - A Fusion of Foundations, Methodologies & Applications*, *18*(12), 2471. https://doi.org/10.1007/s00500-014-1225-9

The Theano Development Team. (2016). Theano: A Python framework for fast computation of mathematical expressions, 1–19.

Toal, R., Rivera, R., Schneider, A., & Choe, E. (2017). *Programming Language Explorations*. Chapman and Hall/CRC.

Torch. (2017). Torch. Recuperado el 4 de agosto de 2017, a partir de https://github.com/torch

Unpingco, J. (2016). *Python for probability, statistics, and machine learning*. *Python for Probability, Statistics, and Machine Learning*. https://doi.org/10.1007/978-3-319-30717-6

Vens, C. (2013). *Encyclopedia of Systems Biology*. Springer International Publishing. https://doi.org/10.1007/978-1-4419-9863-7

Vidulin, V., Luštrek, M., & Gams, M. (2007). Training the genre classifier for automatic classification of web pages. *Proceedings of the International Conference on Information Technology Interfaces, ITI*, 93–98. https://doi.org/10.1109/ITI.2007.4283750

Visualizing neural networks. (2013). Recuperado el 28 de julio de 2017, a partir de https://www.r-bloggers.com/visualizing-neural-networks-from-the-nnet-package/

Wampler, D., & Payne, A. (2014). *Programming Scala*. O'Reilly Media.

Watson, M. (2016). *Power Java*. Leanpub.

Wong, M. L. (2010). Direct Marketing Modeling Using Evolutionary Bayesian Network Learning Algorithm, 273–294.

Xu, J., Liu, J., & Zhao, H. (2011). Financial forecasting: Comparative performance of volatility models in chinese stock markets. *Proceedings - 4th International Joint Conference on Computational Sciences and Optimization, CSO 2011*, 1220–1225. https://doi.org/10.1109/CSO.2011.136

Yang, Z., Jin, M., Zhang, Z., Lu, J., & Hao, K. (2017). Classification Based on Feature Extraction For Hepatocellular Carcinoma Diagnosis Using High-throughput Dna Methylation Sequencing Data. *Procedia Computer Science*, *107*(Icict), 412–417. https://doi.org/10.1016/j.procs.2017.03.130

Zaidi, R. (2017). *JavaScript Essentials for SAP ABAP Developers*. Apres.

Zhang, C. (2017). Deep Learning framework for Julia. Recuperado el 22 de julio de 2017, a partir de https://github.com/pluskid/Mocha.jl

Zhou, L., & Lai, K. K. (2009). Benchmarking binary classification models on data sets with different degrees of imbalance. *Frontiers of Computer Science in China*, *3*(2), 205–216. https://doi.org/10.1007/s11704-009-0027-1

Zinoviev, D. (2016). *Data Science Essentials in Python*. The Pragmatic Programmers.

Zocca, V., Spacagna, G., Slater, D., & Roelants, P. (2017). *Python Deep Learning*. Packt Publishing.