# SISTEMA DE RECOMENDACIÓN PARA PRODUCTOS BANCARIOS CON TÉCNICAS DE MACHINE LEARNING

Sergio Martínez Lizarazo

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICOMECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2018

# SISTEMA DE RECOMENDACIÓN PARA PRODUCTOS BANCARIOS CON TÉCNICAS DE MACHINE LEARNING

Sergio Martínez Lizarazo

*Una tesis presentada en cumplimiento de los requisitos para el grado de Ingeniero de Sistemas e Informática*

Director:
Fabio Martínez Carrillo
PhD en Ingeniería de Sistemas e Informática

Codirector:
Raúl Ramos Pollán
PhD en Ingeniería en Informática

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICOMECÁNICAS
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2018

# *AGRADECIMIENTOS*

El autor expresa su agradecimiento:

Al grupo de investigación de Cómputo Avanzado y a Gran Escala (CAGE) por brindarme los espacios necesarios y los recursos computacionales suficientes, y en especial al profesor Raúl Ramos Pollán por la guía, dedicación y enseñanza integral que me brindó para la realización de este trabajo. Además, al profesor Fabio Martínez por el apoyo brindado en las etapas finales del proyecto y en las maniobras burocráticas de la escuela.

A todos mis amigos de la universidad, que me acompañaron en este proceso y que me han ayudado a crecer como persona y profesional de una manera integral. Una mención especial para aquellos del grupo de investigación CAGE y el semillero MACV.

A la Escuela de Ingeniería de Sistemas e Informática (EISI), a la Universidad Industrial de Santander (UIS) y a aquellos docentes que hacen la diferencia, que de manera sincera se dedican a brindar una formación integral y profesional a cada uno de los estudiantes.

Finalmente un agradecimiento especial para mis padres, mis hermanas y mi hermano, que me han apoyado en todo el curso de esta carrera universitaria, que seguramente será el inicio de nuevos caminos. Y hacer una mención especial a mi sobrino que es una gran motivación para seguir avazando y aportando a la sociedad.

# CONTENIDO

# LISTA DE FIGURAS

# LISTA DE TABLAS

# RESUMEN

**TÍTULO:** SISTEMA DE RECOMENDACIÓN PARA PRODUCTOS BANCARIOS CON TÉCNICAS DE MACHINE LEARNING[1]

**AUTOR:** SERGIO MARTÍNEZ LIZARAZO [2]

**PALABRAS CLAVE:** SISTEMA DE RECOMENDACIÓN, ANÁLISIS DE DATOS, APRENDIZAJE AUTOMÁTICO, INGENIERÍA DE CARACTERÍSTICAS, ÁRBOLES DE DECISIÓN, BOSQUES ALEATORIOS

**DESCRIPCIÓN:**

El comercio electrónico se ha incrementado en estos últimos años, lo que ha llevado que varios sectores comerciales usen los recursos tecnológicos disponibles para mejorar sus estrategias de marketing y ventas. Uno de los sectores que poco a poco ha incursionado en la tecnología es el sector bancario, brindando diferentes canales de comunicación para sus clientes, esto con el fin de mejorar la interacción que tienen con el banco a través de plataformas web y móviles. Poder ofrecer todos los productos y servicios que un banco tiene disponible de una manera personalizada a cada uno de sus clientes, es un reto, esto debido a la gran cantidad de tiempo y dinero que pueden demandar estas tareas. Una de las soluciones para este problema son los sistemas de recomendación, que pueden brindar sugerencias personalizadas a cada uno de los clientes de una manera automática. Por estas razones, este proyecto presenta una propuesta para la recomendación personalizada de productos bancarios a través de técnicas de aprendizaje automático, o *machine learning*. Este trabajo está basado en la competencia de recomendación de productos bancarios ofrecida por el Banco Santander a través de *Kaggle*, que tiene como objetivo la recomendación efectiva de productos bancarios que un cliente pueda añadir a los productos que ya tiene. Para lograr esto, se plantea un flujo de tareas, que incluye analítica y procesamiento de datos, y la creación y validación de modelos predictivos. Para la validación de estos modelos se usaron los datos que ofrece la competencia, en los que se encuentran registros de un poco más de 950.000 clientes en un período de un año y medio. El mejor modelo que se obtuvo tiene un rendimiento del 69,12% como promedio de pruebas realizadas con datos de seis meses diferentes.

---

[1] Trabajo de Grado

[2] Facultad de Ingenierías Físicomecánicas. Escuela de Ingeniería de Sistemas e Informática. Director: Fabio Martínez Carrillo, PhD en Ingeniería de Sistemas e Informática. Codirector: Raúl Ramos Pollán, PhD en Ingeniería en Informática.

# ABSTRACT

**TITLE:** RECOMMENDER SYSTEM FOR BANKING PRODUCTS WITH MACHINE LEARNING TECHNIQUES[1]

**AUTHOR:** SERGIO MARTÍNEZ LIZARAZO [2]

**KEYWORDS:** RECOMMENDER SYSTEM, DATA ANALYTICS, MACHINE LEARNING, FEATURE ENGINEERING, DECISION TREE, RANDOM FOREST.

**DESCRIPTION:**

The e-commerce has growth in the last few years, for this reason some commercial and business sectors have migrated to the new technologies to improve their marketing and sales strategy. The banking industry has gradually implemented new channels of communications, through apps and web platforms, to offer a better experience to its customers. Today is a challenge to offer new banking products to the customers in a personalized way, because it needs many resources such time and money. A recommender system might be a solution to this problem, because it can generate recommendations in an personalized and automatic way. Therefore, this project is an approach to a recommender system for banking products with Machine Learning techniques. This project is based on the Kaggle competition about banking product recommendation, it is offered by Santander Bank, where the goal is to predict which new banking products were most likely to buy. The proposed method to achieve this goal is to workflow that includes data analysis and processing and predictive models making. The validation of the predictive models used data offered by the competition that consisted of 1.5 years of nearly 950000 customers behavior data from Santander bank. The best predictive model has a 69.12% performance as per the tests with six different month data.

---

[1] Bachelor Thesis.

[2] School of Physical-Mechanical Engineering. Department of Systems Engineering and Informatics. Advisor: Fabio Martínez Carrillo, PhD in Systems Engineering and Computer Science. Advisor: Raúl Ramos Pollán, PhD in Computer Science Engineering

# *INTRODUCCIÓN*

Los sistemas de recomendación son herramientas de software que tienen como función principal hacer sugerencias de ítems o productos a los usuarios de una aplicación [1]. Las recomendaciones se relacionan a situaciones de toma de decisiones, tales como ¿Qué producto compro?, ¿Cuál película veo? o ¿Qué noticia leo?, esto por citar algunos ejemplos. En la actualidad se pueden observar sistemas de recomendación en varias aplicaciones web y móviles, un ejemplo muy popular es la plataforma de películas y series vía *streaming*, Netflix, que ofrece sugerencias personalizadas a sus usuarios según su interacción con la aplicación, que se basa en el contenido que el usuario ha consumido en el pasado. También existen las recomendaciones no personalizadas, que son más simples para generar, porque se basan en tendencias o *rankings* de productos populares en un sector, como las películas más vistas del 2017, o el *top 10* de los cantantes más escuchado el último mes. Los sistemas de recomendación se hacen necesarios hoy en día por lo que se denomina el fenómeno de *long tail* [2], que hace referencia a la gran cantidad de productos que no se pueden ofrecer a un cliente, porque se tiene un inventario muy amplio, y sólo se sugieren los productos que son más populares, es por esto que la mayoría de tiendas *online* implementan sistemas de recomendación personalizados para cada cliente, y así poder ofrecerle productos que sean de su interés, sin necesidad de que estos sean los más populares.

Robin Burke en [3] distingue diferentes técnicas de sistemas de recomendación, en las que se destacan el filtrado colaborativo, o *collaborative filtering*, técnica que usa las preferencias conocidas de un grupo de usuarios para hacer recomendaciones o predicciones de las preferencias desconocidas de otro grupo de usuarios, así pues, si dos personas califican items similares, o tienen comportamientos parecidos, se podría inferir que tiene otros gustos similares [4]. Un ejemplo claro del uso de esta técnica son las redes sociales como Facebook, que sugiere contactos con base a los amigos en común que tenga cada

usuario con los contactos sugeridos; otra técnica son los sistemas de recomendación basados en contenido, o *content-based*, que basa sus recomendaciones en las características o perfil de los ítems del sistema y las preferencias del usuario, un ejemplo para este tipo de sistema es Netflix, que hace sus recomendaciones según las características de las películas que el usuario ve y califica; existen otras técnicas como los sistemas de recomendación demográficos, que se usa para generar recomendaciones según las características socio-económicas de los usuarios. En varias ocasiones estas técnicas son usadas en conjunto para hacer sistemas de recomendaciones híbridos, tal y como lo indica [5]. Es necesario notar que estas técnicas comparten algunas limitaciones, una de ellas es el problema del *cold start*, que se refiere al momento en el que el sistema no puede hacer recomendaciones de forma eficiente sobre nuevos usuarios porque no tiene la información necesaria y suficiente.

El estado del arte ha propuesto algunas implementaciones de sistemas de recomendación en el sector bancario y financiero. Gallego y Huecas en [6] presentan un modelo que puede hacer recomendaciones de servicios y productos de terceros con base en los datos de transacciones bancarias hechas con tarjetas de crédito de los clientes del banco, este sistema está implementado a través de una aplicación móvil que le entrega información adicional del cliente como su ubicación, y con estos datos sugieren sitios, como museos o restaurantes, según el contexto en el que se encuentre el cliente. En [7] se propone un sistema de recomendación personalizado de productos que ofrece un banco a través de la técnica de filtrado colaborativo, la información con la que se nutre el sistema proviene de los diferentes canales de interacción que el cliente tiene con el banco, como la plataforma web, aplicación móvil, e incluso los cajeros electrónicos. El resultado de implementar un sistema de recomendación en cierto contexto, como lo es el sector bancario y de finanzas, tiende a ser rentable, pues se ha probado la utilidad que tienen los sistemas de recomendación para sugerir inversiones de activos así como se demuestra en [8]. Así mismo, en [9] se puede ver que hay una buena aceptación por parte de los cliente de un banco con la implementación de un sistema de recomendación, pues les brindaría opciones personalizadas según sus necesidades. Las propuestas que se acaban de mencionar, usan enfoques tradicionales de los sistemas de recomendación, como lo son el filtrado colaborativo y las recomendaciones basadas en contenido, estos enfoques usualmente se usan con sistemas basados en memoria, es decir, se hacen las recomendaciones según los cálculos que realicen sobre todo el conjunto de datos. Esto puede ser un problema cuando se tiene un gran conjunto de datos,

pues puede haber una disminución en su rendimiento al hacer los cálculos correspondientes y además presentar ciertas limitaciones de escalabilidad.

En la literatura revisada, apenas se encontraron implementaciones de técnicas de *machine learning* en el sector bancario, en [10] se muestra un enfoque de *data mining* para predecir el éxito que tiene el *telemarketing* en la venta de productos bancarios, además de esto, no se encontraron implementaciones de sistemas de recomendación con técnicas de *machine learning* para este sector, es posible que por esto se lanzó la competencia de Kaggle [11] en la que se basa este proyecto. La utilización de este tipo de técnicas podrían hacer más efectivos los sistemas de recomendación, así como propone Paradarami *et. al.* en [12], donde se evidencia que usando un sistema híbrido de técnicas tradicionales y redes neuronales se puede mejorar en las sugerencias que genera el sistema, además que éste tipo de enfoques tienen capacidad de escalabilidad y recomendación en tiempo real que mejoraría la experiencia del usuario final.

La principal contribución de este proyecto es la construcción de un modelo predictivo que pueda hacer recomendaciones de nuevos productos bancarios que un cliente pueda adquirir el siguiente mes, esto basado en información que se tenga del comportamiento del cliente en el banco. Para poder realizar este objetivo, primero se debe hacer una exploración y análisis de los datos originales que brinda la competencia de Kaggle [11], esto con el fin de poder construir una serie de *datasets* que sirvan para implementar ciertas técnicas de *machine learning* como algoritmos de clasificación binaria. La validación de estos modelos construidos se hacen a través de la métrica MAP@7, *Mean Average Precision @ 7*, que es la que propone la competencia de Kaggle [11].

# *Capítulo 1*
# *PLANTEAMIENTO Y JUSTIFICACIÓN DEL PROBLEMA*

Identificar las necesidades que tienen los clientes del sector bancario, y poder satisfacerlas, representan un reto hoy en día, esto quizá se debe a que este tipo de tareas demandan mucho esfuerzo, tiempo y dinero. Muchas veces el resultado de esto, es que pocos clientes son los que tienen recomendaciones de productos y servicios del banco, y como consecuencia podría haber una experiencia inconsistente para el resto de clientes. Por esto, automatizar las recomendaciones personalizadas para cada usuario es fundamental para ahorrar recursos, y además poder ofrecer una mejor experiencia a una gran mayoría de clientes.

En la literatura que se ha revisado, no se encontró implementaciones de sistemas de recomendación con técnicas de *machine learning* en el sector bancario. Esto es quizá por las prácticas que tienen ciertas entidades privadas para mantener en secreto sus métodos predictivos y de análisis de datos. Ahora bien, poder aplicar este tipo de técnicas para los sistemas de recomendación podría mejorar la precisión en las recomendaciones que se generen.

# *Capítulo 2*
# *OBJETIVOS*

## 2.1 OBJETIVO GENERAL

Construir y evaluar modelos predictivios para la recomendación de productos bancarios basado en el perfil y comportamiento del cliente

## 2.2 OBJETIVOS ESPECÍFICOS

- Hacer un análisis descriptivo de los datos suministrados por el Banco Santander para la competencia de analítica de datos de Kaggle
- Establecer una metodología de preprocesado de datos para la construcción de datasets preparados para su uso en el entrenamiento de los modelos predictivos
- Construir modelos predictivos para la recomendación de productos bancarios según el perfil de cada cliente
- Establecer las métricas necesarias que permitan evaluar los modelos predictivos que se construyan
- Validar el desempeño del modelo predictivo basado en las métricas previamente establecidas

# *Capítulo 3*
# *MARCO CONCEPTUAL*

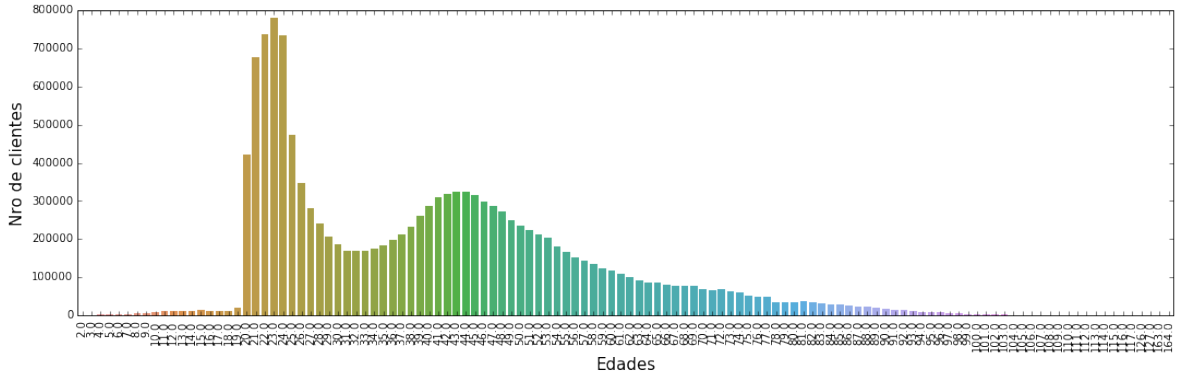
## 3.1 EXPLORACIÓN Y PROCESAMIENTO DE DATOS

**3.1.1 Exploración de datos** En cualquier tarea que se incluya la aplicación de técnicas de *machine learning* se debe iniciar por la exploración de los datos. El fin de esa tarea es dar un primer vistazo a la consistencia de los datos, en muchas ocasiones hay datos faltantes que pueden introducir ruido al conjunto de datos. Además de esto, se hace una primera aproximación al entendimiento de los datos a partir de la estadística descriptiva que se le aplique a cada una de las características o *features* que tenga el *dataset*, esto sirve para observar si existen algunos patrones que puedan servir la construcción del modelo predictivo, por esto, en esta fase es fundamental la visualización de los datos, de tal forma que sea más cómodo hacer el análisis que se desee. La exploración de datos también se le conoce como minería de datos, o *data mining*, en [13] se mencionan técnicas de exploración, preprocesamiento, y análisis de datos relacionadas con el campo de los sistemas de recomendación. Algunas técnicas que se usaron en la etapa de exploración de datos son las siguientes:

- **Estadística descriptiva:** Es la técnica más básica en la exploración de datos, en esta fase se revisa cada una de los *features* para saber su tipo (numérico o categórico), cantidad, media y otras estadísticas. Frecuentemente se usan representaciones gráficas para identificar ciertas tendencias o comportamientos que puedan tener elementos del conjunto de datos, así como se describe en [14]. Un ejemplo está ilustrado en la Figura 1 que muestra la distribución de edades de los clientes que están presentes en el *dataset* que se usó en este trabajo.

- **Medidas de similitud:** En [13] mencionan que definir estas medidas apropiada-

Figure 1. Distribución de edad. A través de la estadística descriptiva se puede ver que esta distribución es bimodal.




mente es importante para comprender qué tan similares pueden ser los usuarios o ítems que estén presentes en un conjunto de datos, las métricas de distancia que se pueden aplicar son la distancia Euclidiana o la distancia coseno. La similitud entre ítems se puede dar por su correlación, que mide las relaciones lineales entre objetos, en este caso la medida más usada es el coeficiente de correlación de Pearson y que se puede definir como un índice que mide el grado de relación entre dos variables siempre y cuando sean continuas y cuantitativas. A continuación, se presenta la ecuación con la que se puede calcular el coeficiente de correlación de Pearson para dos conjuntos de datos, $X$ y $Y$, de tamaño $n$:

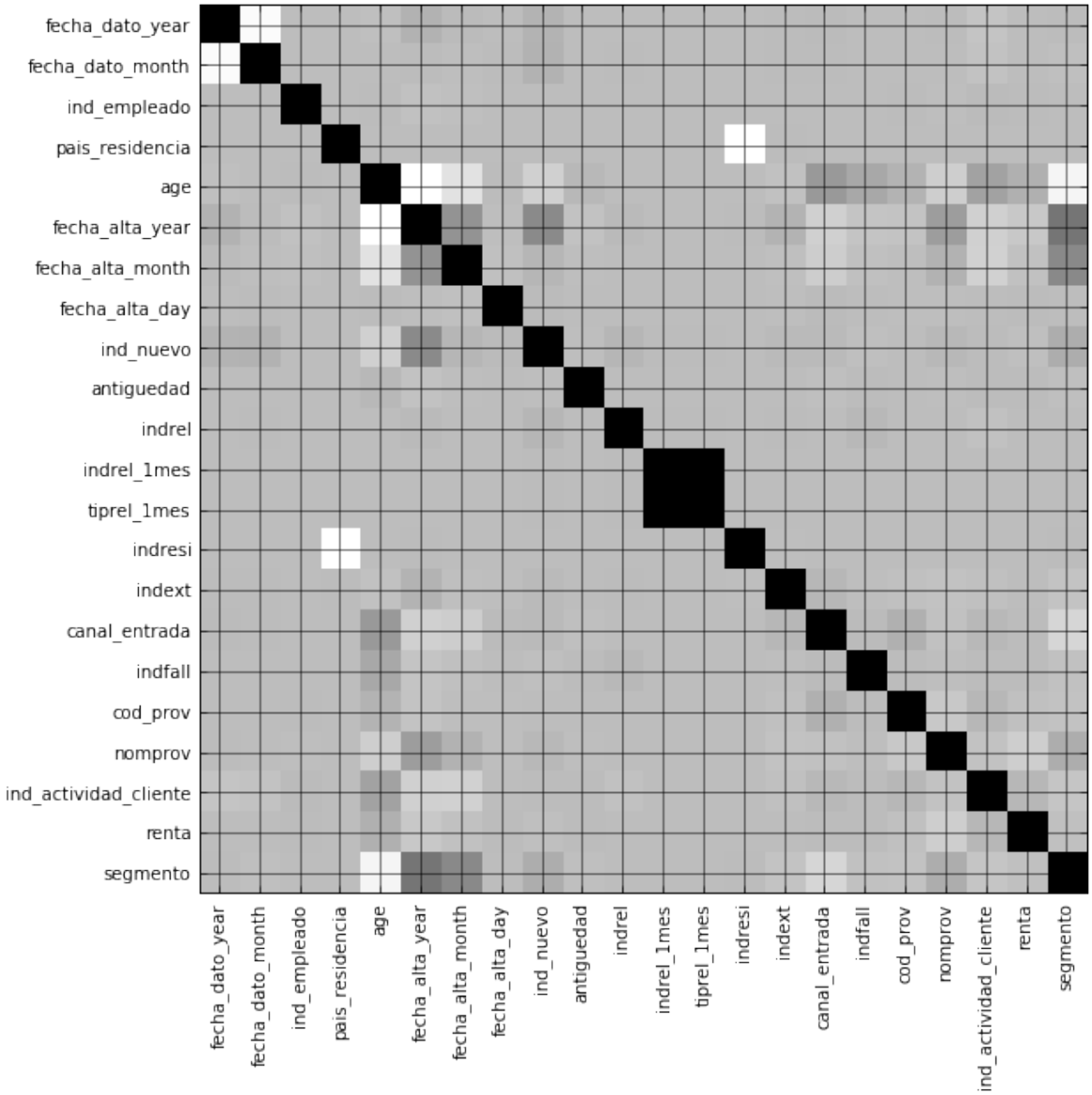
$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{3.1}$$

Donde $n$ es el tamaño del conjunto de datos, $x_i, y_i$ son los valores individuales de cada conjunto de datos indexados con $i$, y $\bar{x}$ y $\bar{y}$ son las medias de cada conjunto de datos. Una forma de visualizar la correlación de los datos es la matriz de correlación, en la que se observa en una gráfica la interdependencia entre cada pareja de variables, esto se ilustra en la Figura 2

### 3.1.2 Procesamiento de datos

Tiene como fin la construcción de datasets preparados para su uso en el entrenamiento de modelos predictivos. Son variadas las

Figure 2. Matriz de correlación de las características de un conjunto de datos. En esta representación es necesario notar que entre más oscura sea la intersección entre las parejas de variables hay una mayor relación de dependencia entre ellas.



técnicas que se usan para el procesamiento de datos, no necesariamente se deben aplicar las mismas técnicas para cada problema, pues se debe tener en cuenta la naturaleza del problema y la exploración previa que se haya hecho del conjunto de datos. Las tareas de procesamiento que se listan a continuación son las que se usaron para el desarrollo de este proyecto.

- **Eliminación de ruido:** En la fase de exploración de datos se define la cantidad de datos faltantes en el conjunto de datos. Es necesario decir que el ruido no sólo son datos faltantes, sino también pueden ser *outliers* o valores anómalos que están presentes en los datos, esto se puede apreciar en la fase de exploración como se menciona en [13]. El fin de eliminar el ruido es que no genere resultados no deseados a la hora de entrenar el modelo predictivo. Sin embargo, hay que tener en cuenta, que algunas veces los datos faltantes se pueden inferir con base a la información existente y así se puede completar considerablemente el conjunto de datos, esto con el fin de tener un *dataset* más robusto y confiable, que pueda describir mejor el problema.

- **Conversión del tipo de *feature*:** En [14] se mencionan dos tipos de *features*, los numéricos y los categóricos, los primeros como su nombre lo indica son datos cuantitativos, mientras que los segundos son datos cualitativos que indican características de un registro, un ejemplo del tipo categórico es el sexo de una persona que puede ser hombre o mujer, y además se puede observar que ninguno de los dos tiene una relación de orden, es decir, no hay uno que tenga mayor valor que el otro, este tipo de *feature* se puede cambiar a numérico asignando un valor cuantitativo a cada uno de los posibles valores categóricos.

- ***Principal Component Analysis* (PCA):** Reducir la dimensionalidad de los datos, ha sido considerado un enfoque que se debe tener en cuenta para el diseño de sistemas de recomendación, esto se debe a que en varios *datasets* los datos se encuentran muy dispersos. Entre los algoritmos para la reducción dimensionalidad que menciona [13] se encuentra *Principal Component Analysis*, o PCA. Esta técnica de reducción de dimensionalidad es capaz de reducir el número de variables considerablemente manteniendo mucha de la información que hay en el *dataset* original. Para lograr esto, se supone que hay $n$ medidas en un vector $x$ que tiene $p$ variables aleatorias, y se desea reducir la dimensión de $p$ a $q$, donde $q < p$. PCA encuentra combinaciones lineales, $a_1'x, a_2'x, ..., a_q'x$, llamadas componentes

principales, en la que la cantidad de varianza tomada por el primer componente es mucho mayor que la del segundo, y así sucesivamente, haciendo que los componentes no estén correlacionados con los componentes $a'_k x$s previos. Al resolver este problema de maximización se encuentran los eigenvectores, o vectores propios, $a_1, a_2, ..., a_q$ de la matriz de covarianza, $S$, de los datos. Los eigenvectores dan la varianza de sus respectivos componentes, y la relación de la suma de los primeros $q$ eigenvalores con la suma de las varianzas de todos los $p$ valores originales representa la proporción de la varianza total en el *dataset* original [15].

- ***Feature Engineering*:** Esta fase de ingeniería de características, o *feature engineering*, es fundamental previo a la construcción y entrenamiento de los modelos predictivos, y supone ser el paso inmediatamente siguiente de la exploración de datos. El fin principal de esta fase es crear nuevas características o *features* en base a las existentes que puedan representar de mejor manera los datos que se tienen, para que se puedan ajustar mejor el modelo predictivo que se construya y poder tener mayor precisión al momento de hacer las predicciones.

## 3.2 MACHINE LEARNING

El concepto de *Machine learning* usualmente se refiere a los cambios que tienen los sistemas que ejecutan tareas asociadas con inteligencia artificial. Tales tareas implican reconocimiento, diagnóstico, planeación, predicción y otras relacionadas. Un objetivo del *machine learning* es que las máquinas sean capaces de ajustarse así mismas para poder generar resultados deseados con respecto a la tarea que deba cumplir, estos ajustes se hacen según el ambiente en el que se encuentren, ambientes que pueden ser cambiantes en el tiempo [16]. Existen dos tipos de aprendizaje en *machine learning*, el aprendizaje supervisado y el no supervisado, el primero consta en hacer predicciones o clasificaciones de objetos con base en datos etiquetados con los que se nutre el modelo predictivo, es decir, a un algoritmo de clasificación se le puede entrenar con fotos de gatos y perros, y cada una de estas fotos tiene una etiqueta indicando cual animal tiene, en este tipo hay dos tareas que se pueden realizar, que son regresión linea y clasificación de objetos; por otra parte, el segundo tipo consiste en entrenar un modelo predictivo con datos que no tienen etiquetas con el fin de encontrar alguna estructura del conjunto de datos que se suministra. En la actualidad se usa *machine learning* en varias áreas del conocimiento y la industria, en el sector bancario poco se ha visto la incidencia de estas

técnicas, aunque han habido acercamientos como se puede ver en [12].

### 3.2.1 Clasificación en *Machine Learning*

Es una de las ramas del *machine learning* supervisado, en el que se nutre el algoritmo elegido con los datos que tienen las características de los objetos que se desean clasificar y las etiquetas que identifica a cada uno de los objetos, es decir, cada objeto o registro del conjunto de datos tiene un conjunto de características propias y además se le asocia una etiqueta o *label*. Un ejemplo de esto sería un conjunto de datos que tiene información personal de clientes de un banco y a cada uno se le asigna una etiqueta indicando si puede o no recibir un crédito. Cuando se tienen los modelos predictivos entrenados, se le ingresan datos nuevos para generar una predicción. La salida de estos modelos de clasificación tienen dos elementos la etiqueta del objeto en cuestión, y la probabilidad de que ese objeto pertenezca a la clase que se predijo. Además hay que agregar que existen dos tipos de clasificación, la binaria, que tiene sólo dos etiquetas o clases, y la multiclase, en la que hay más de dos clases para etiquetar al objeto.

### 3.2.2 Algoritmos de clasificación

A continuación, se listan sólo dos de los algoritmos más destacados en clasificación, estos algoritmos se pueden usar tanto para clasificación binaria, como para clasificación multiclase. Solamente están listados los algoritmos que se tuvieron en cuenta para el desarrollo de este proyecto.

#### 3.2.2.1 Árboles de decisión

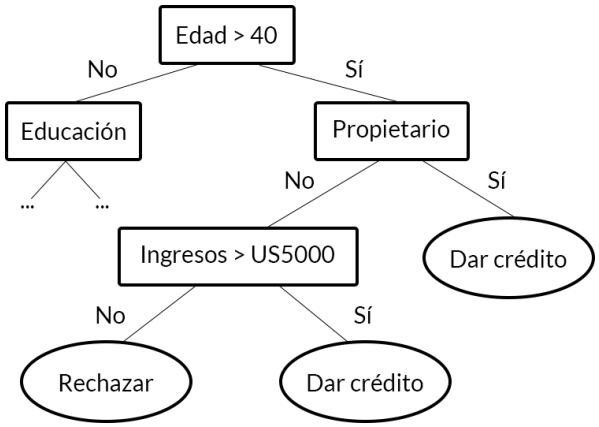
Son algoritmos de predicción que pueden ser usados para modelos de clasificación o regresión. Los árboles de decisión son usados para clasificar un objeto en un conjunto predefindo de clases basado en sus características, estas clasificaciones se hacen con base a unas reglas de decisión simple que se encuentran en cada nodo [17]. Generalmente los árboles de decisión simples y pequeños se pueden representar gráficamente a través de un grafo como se ve en la Figura 3. Un árbol de decisión tiene tres tipos de nodos:

- **Nodo raíz:** Es el nodo inicial del árbol, al tener un nodo raíz se indica que es un árbol dirigido. Este nodo no recibe ninguna entrada.
- **Nodo de deicisión:** También llamados nodos internos, es en donde están las reglas simples de decisión, aquí se testean los atributos del objeto para determinar

por cuál rama debe seguir. Cada nodo divide el objeto en dos o más de acuerdo a una función discreta que se defina. Estos nodos siempre reciben una entrada.

- **Hojas:** Son los nodos finales de cada rama, y es allí en donde está la etiqueta del objeto que se ha clasificado, algunas veces se encuentra en esta rama la probabilidad con la que se puede clasificar el objeto.

Figure 3. Árbol de decisión simple para predecir si se le da o no un crédito a una persona




#### 3.2.2.2 Bosques aleatorios

El algoritmo de bosques aleatorios, o *Random Forest*, es un modelo ensamblado, esto quiere decir que es un algoritmo que combina varios modelos predictivos base para poder mejorar la precisión de predicción, por esta razón, el algoritmo de bosques aleatorios es una combinación de árboles de decisión. Los árboles de decisión que forman este bosque son independientes entre ellos y cada uno tiene la misma distribución que el resto. Después de que un gran número de árboles se hayan generado, se hace la predicción con el más popular entre los árboles, es decir, con el resultado que se repita más. [18]

## 3.3 MÉTRICA DE DESEMPEÑO

Para poder validar los modelos predictivos que se realizan en problemas de *machine learning*, se deben usar métricas que se adapten al problema en cuestión. En problemas como los sistemas de recomendación, siempre se quiere sugerir con mayor precisión los productos a los clientes, por esa razón es que se generan varias recomendaciones para que el cliente pueda elegir entre las diferentes opciones, sin embargo, esto también se

hace de forma que los productos se le presenten en orden de relevancia, el que probablemente le interese más irá de primero y así sucesivamente. La métrica que se describe a continuación es la que propone la competencia Kaggle [11] , *Mean Average Precision* @ 7, o MAP@7, es ideal para poder validar este tipo de listas donde los elementos están ordenados según su relevancia. La métrica MAP@7 está definida en la siguiente ecuación:

$$MAP@7 = \frac{1}{|U|} \sum_{u=1}^{|U|} \frac{1}{min(m,7)} \sum_{k=1}^{min(n,7)} P(k) \tag{3.2}$$

Donde $|U|$ es el número de clientes que están presentes en el mes de prueba y en el mes anterior, $P(k)$ es la precisión de las recomendaciones de un usuario, está dada por $p(k) \cdot r(k)$, en donde $p(k)$ es el porcentaje de productos correctos entre los $k$ primeros productos, y $r(k)$ es un valor binario de uno o cero que indica si el producto es o no relevante respectivamente, $n$ es el número de productos que se predijeron y $m$ es la cantidad de productos que el cliente realmente añade al siguiente mes. [19, 20]

# *Capítulo 4*
# *MÉTODO PROPUESTO*

El método propuesto para el desarrollo de este proyecto inicia en la exploración descriptiva de los datos originales que brinda la competencia de Kaggle [11]. Esto con el fin de limpiar los datos, pues algunas veces se presentan inconsistencias y datos faltantes, y así poder construir nuevos conjuntos de datos a partir de los datos limpios, en los que se podrán añadir nuevas características o *features*. Posteriormente se usan ciertas estrategias para la creación de modelos predictivos con algoritmos de clasificación binaria. En base a estos modelos se podrá hacer la validación respectiva con diferentes conjuntos de datos de prueba para poder definir un modelo predictivo final que tenga la mayor precisión posible según la métrica, MAP@7, que se estipuló inicialmente. Como se puede ver en la Figura 4, este método tiene un ciclo de realimentación, este ciclo, se ve reflejado en la experimentación que se lleva a cabo en el proyecto, y con la cual se puede generar un modelo predictivo final.

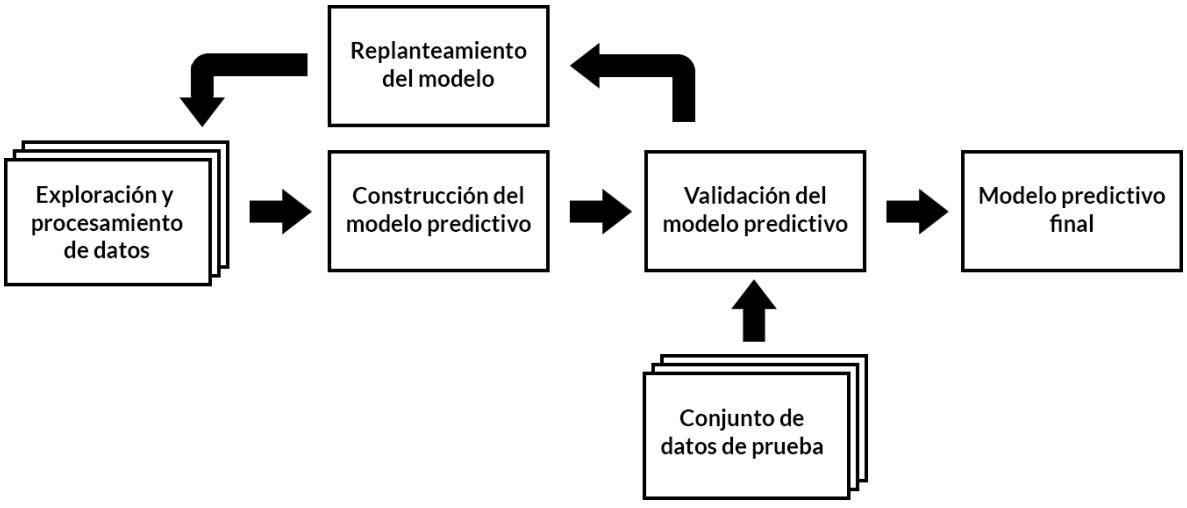



Figure 4. Flujo del desarrollo del proyecto.

## 4.1 EXPLORACIÓN Y PROCESAMIENTO DE DATOS

Esta es la primera fase del proyecto, en el que se usaron las diferentes técnicas descritas anteriormente, desde la estadística descriptiva, la eliminación de ruido, la conversión del tipo de *feature*, PCA y otros. Todo esto con el fin de tener un mayor entendimiento de todo el conjunto de datos que se tiene y poder preparar los *datasets* para entrenar los algoritmos. A esta fase se puede regresar tantas veces se necesite, pues el método que se propone lo permite, ya que no es un modelo estrictamente en cascada donde hay un paso a paso, sino más bien, se puede hacer un ciclo realimentación que sirva para ajustar mejor los datos al modelo que se necesite.

## 4.2 CONSTRUCCIÓN DEL MODELO PREDICTIVO

Para efectos de este proyecto se usarán las técnicas de *machine learning* de clasificación binaria. Esto debido a que el problema que se está trabajando consiste en dar la probabilidad que un cliente del banco compre un producto el siguiente mes, entonces, hay presentes dos clases identificadas que son la compra del producto y la elección de no comprar el producto, que se representan por 1 y 0 respectivamente, así pues, el resultado de este proyecto está dado por la probabilidad que un cliente pertenezca a una de esas dos clases. Como se dijo anteriormente, los modelos predictivos que se construirán en las fases de experimentación se harán con los algoritmos de árboles de decisión y bosques aleatorios, que se describieron previamente.

## 4.3 VALIDACIÓN DEL MODELO PREDICTIVO

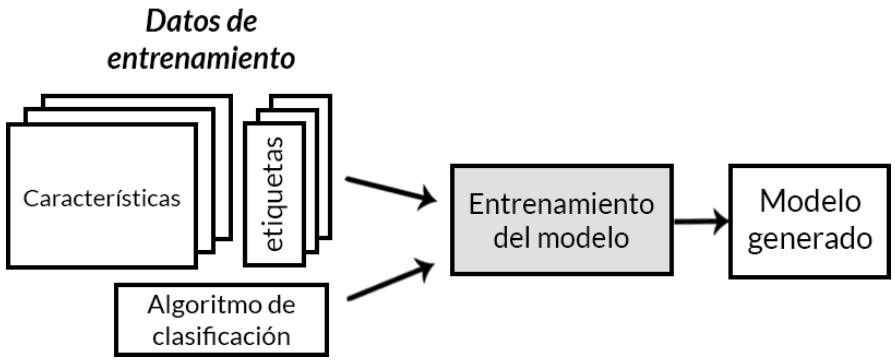
Para la validación de los modelos predictivos que se construyan se tiene una métrica de evaluación en la que se evalúa la calidad de las recomendaciones que se le presente al cliente. Hay que tener en cuenta que la predicción que se genere será la de los posibles productos que el cliente agregue en el futuro cercano, es decir, al siguiente mes. La métrica que se usa para esta validación es la de *Mean Average Precision @ 7*, MAP@7. Por esto, para cada cliente se generará una recomendación con siete productos ordenados de tal manera que el primero tiene una mayor relevancia que el resto.

## 4.4 *FRAMEWORK* DE AUTOMATIZACIÓN

Para poder realizar el flujo de trabajo planteado en la Figura 4 se desarrolló un *framework* de trabajo en el que se toma los datos y el algoritmo elegido para crear el modelo predictivo y poder validarlo con datos de prueba. El *framework* que se desarolla, se hace con el fin de tener una automatización de todo el método propuesto, y de esta forma hacer más fácil la fase de experimentación. A continuación, se muestra todo el proceso que se lleva a cabo para obtener el resultado de un experimento, a través, del *framework* de trabajo.

1. **Construcción del modelo predictivo:** En esta fase se deben tener los datos de los *features*, o características, con el preprocesado correspondiente, ya sea con un PCA o con *features* adicionales. Además se deben tener las etiquetas o *labels* de cada objeto con las que se va a entrenar el algoritmo de clasificación que se elija. Los datos de entrenamiento, *features* y *labels*, y el algoritmo de clasificación serán las entradas necesarias para la construcción del modelo predictivo. Hay que resaltar que los algoritmos que se eligieron para este proyecto son los de árboles de decisión y bosques aleatorios. El modelo predictivo que se genera, está listo para hacer las predicciones que se validarán posteriormente. En la Figura 5 se puede ver el esquema de esta primera fase.

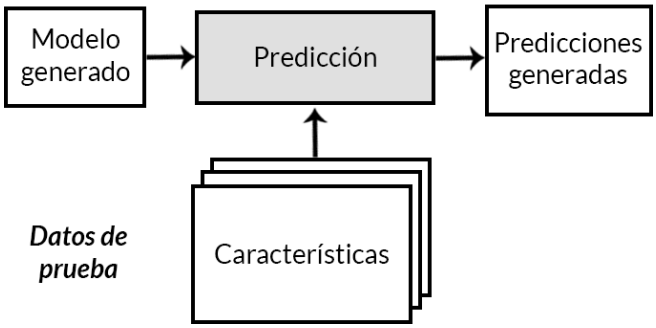
Figure 5. Construcción del modelo predictivo




2. **Generación de predicciones:** Cuando se tiene entrenado el modelo predictivo se prosigue a hacer las predicciones con base a unos datos de prueba, que deben tener el mismo preprocesado que se le realiza a los datos de entrenamiento. Es

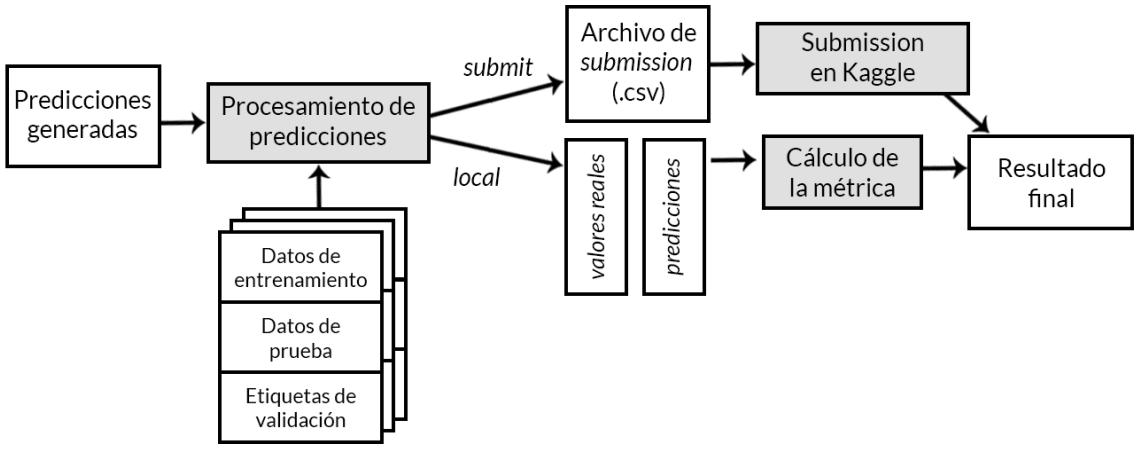
importante mencionar que los datos de prueba solamente son las características de los objetos a los cuales se les va a hacer la predicción de su etiqueta. Al final de esta fase se crea una estructura de datos en los que se almacenan las predicciones y sus probabilidades correspondientes, esto con el fin de tener la información necesaria para la siguiente etapa, así como se ilustra en la Figura 6.

Figure 6. Generación de predicciones




3. **Validación de las predicciones:** Para poder hacer las validaciones de los experimentos que se plantean en este proyecto, se debe crear una estructura de datos ó un archivo para poder calcular la precisión del modelo predictivo con base a la métrica propuesta. En esta fase se presentan dos salidas posibles, como se puede ver en la Figura 7, esto depende de cuál opción se eliga, además esto sucede porque hay dos fases de experimentación, y la validación de cada una se hace en ambientes distintos, pero con la misma métrica.

Figure 7. Validación del modelo predictivo.



Para cada una de las salidas, es necesario aclarar, que las predicciones generadas tienen un procesamiento que consiste en organizar las recomendaciones según su relevancia y además remover los productos que el cliente tiene el mes anterior al mes de prueba. A continuación, se explica cómo se produce cada una de las salidas y sus propósitos:

- **submit:** La primera fase de experimentación usa los datos de entrenamiento y los dos algoritmos elegidos para hacer los modelos predictivos. Para la validación de estos, se usan los datos de prueba que son suministrados por la competencia de Kaggle [11], estos datos sólo tienen los *features*, pero no tiene sus etiquetas correspondientes. Así que, para poder hacer la validación con estos datos de prueba, se debe generar un archivo de formato .csv (*comma-separated values*) para poder subirlo al ambiente de pruebas de Kaggle, y al final se genera el *score* de precisión del experimento.

  Kaggle ofrece su propio ambiente de validación en el cual se debe subir un archivo de *submission*, o archivo final, que tiene la estructura que indica la Figura 8, el cual tiene dos columnas, una identificada con *ncodpers* que indica el código único de cada cliente, y la segunda, *added_products*, que tiene los productos recomendados para cada cliente. Este archivo se puede subir a través de un navegador web, o a través de línea de comando usando el API que Kaggle ofrece, por comodidad, en este proyecto se usa el segundo método de *submission.* En la Figura 9 se puede observar un ejemplo de *submission* automática a través del API de Kaggle, en la que se pasa como argumentos el archivo y un mensaje descriptivo del experimento.

Figure 8. Estructura del archivo de *submission*. Se puede observar que en algunas ocasiones se puede generar una recomendación vacía en la que no se le sugiere ningún producto.

```
ncodpers, added_products
15889, ind_tjcr_fin_ult1, ind_recibo_fin_ult1,...
15890, ind_tjcr_fin_ult1, ind_recibo_fin_ult1,...
15891, in_nomina_ult1, in_nom_pen_ult1, ind_re...
15892,
```

Figure 9. Ejemplo de submission automática al ambiente de pruebas de Kaggle. La interfaz que se observa pertence a Jupyter, una aplicación web que permite ejecución de código Python

```
!bash results/submissions/kaggle_submission "RF, Mayo 2016, PURCHASERS"
```

```
2018-07-24-h18-27-03_submission.csv
/home/sergioml/.local/lib/python2.7/site-packages/urllib3/contrib/socks.py:37: DependencyWarning: SO
CKS support in urllib3 requires the installation of optional dependencies: specifically, PySocks.  F
or more information, see https://urllib3.readthedocs.io/en/latest/contrib.html#socks-proxies
  DependencyWarning
Warning: Looks like you're using an outdated API Version, please consider updating (server 1.4.2 / c
lient 1.3.12)
Successfully submitted to Santander Product Recommendation
real    0m20.814s
user    0m1.396s
sys     0m0.376s
```

- **local:** La segunda fase de experimentación consiste en elegir los tres mejores experimentos que se hacen en la fase previa, y con estos se hace la validación correspondiente con información de seis meses diferentes. Para este caso, los datos de prueba se deben tomar de los datos de entrenamiento que se tienen originalmente, además, la validación de estos experimentos se pueden hacer de forma local, es decir, sin enviarlas al ambiente de pruebas de Kaggle, porque para cada conjunto de datos de prueba se tienen sus etiquetas o *labels* correspondientes.

  Esta etapa genera dos estructuras de datos para el cálculo de la precisión de las recomendaciones a través de la métrica MAP@7. Las estructuras de datos que se entregan son: las predicciones generadas por el método predictivo, y los valores reales, que son los productos que el cliente realmente adquiere el siguiente mes, por esto, esta etapa de validación recibe como entrada las "etiquetas de validación" para poder hacer la estructura de los valores reales. Con estas dos estructuras que se generan se hace comparación respectiva para calcular la precisión de las predicciones, todo esto se hace a través de la métrica propuesta.

En la Figura 10 se puede ver la ejecución de un experimento en el que se usó el *framework* descrito previamente. Como se puede observar, este *framework* presenta facilidad para la ejecución de diferentes tipos de experimentos, lo único que se debe hacer fuera de este, es el preprocesamiento de los datos que es específico para cada uno de los experimentos.

Figure 10. Pantallazo de implementación del Framework de automatización, en este ejemplo se puede ver que se genera un archivo de *submission* pues así se específica en los argumentos de la función de la última línea

```
%%time
x_train = df_purchasers[df_purchasers['fecha_dato'] == '2016-05-28']
y_train = df_targets.loc[x_train.index]

x = x_train.drop(['fecha_dato', 'fecha_alta'], axis=1).as_matrix()
y = y_train.as_matrix()

model = local.model(x, y, RandomForestClassifier(n_jobs=4))

x_test = df_test.drop(['fecha_dato', 'fecha_alta'], axis=1).as_matrix()
probs, preds = local.calculatePredsProbs(x_test, model)

subm = local.processPredictions(probs, preds, x_train, df_test, y_train, env='submit')
```

```
results/submissions/2018-07-20-h16-52-11_submission.csv
CPU times: user 2min 26s, sys: 4.21 s, total: 2min 30s
Wall time: 1min 21s
```

# *Capítulo 5*
# *DESARROLLO DEL PROYECTO*

En la siguiente sección, se presentará todo el desarrollo del proyecto, el cual se hizo según el método propuesto previamente, siguiendo cada una de las etapas desde la exploración y procesamiento de datos, construcción de modelos predictivos y su posterior validación. Además, se describirán las diferentes fases de experimentación que se llevaron a cabo para poder obtener un modelo predictivo final.

También, cabe mencionar que el desarrollo de este proyecto, se llevó a cabo sobre los recursos computacionales que ofrece el grupo de investigación de Cómputo Avanzado y a Gran Escala (CAGE) de la Escuela de Ingeniería de Sistemas e Informática (EISI). Se usó el clúster GUANE que está ubicado en el Parque Tecnológico Guatiguará. Este clúster tiene 16 nodos ProLiant SL390s G7, cada uno tiene una memoria RAM de 104 GB, disco SAS de 200GB, procesadores Intel(R) Xeon(R) de 12 y 8 Cores, y además 8 GPUs Tesla de diferentes modelos. En términos generales, estos recursos fueron suficientes para la realización y desarrollo satisfactorio de este proyecto de grado.

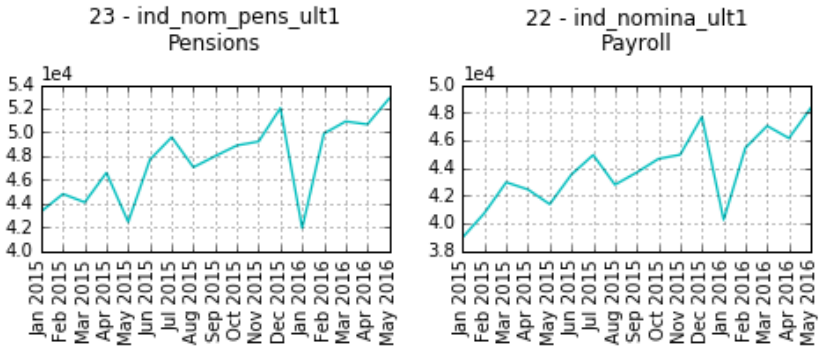
## 5.1 EXPLORACIÓN Y ANÁLISIS DE DATOS

Para poder evaluar el método propuesto se usó el *dataset* que suministra la competencia de Kaggle [11], el cual es de acceso público. Este *dataset* tiene registros de 17 meses, desde enero de 2015 hasta mayo de 2016, del comportamiento de aproximadamente 950.000 clientes del Banco Santander. Es necesario aclarar que la muestra de datos que ofrece la competencia no incluye clientes reales del Banco Santander, así que no son datos representativos de los clientes reales del banco en cuestión. El dataset se compone de un total de 13'647.310 registros y de 24 *features* o características propias

de cada cliente tales como edad, código del cliente, antigüedad, fecha de registro, ingresos mensuales y otros. Además, cada uno de los clientes tiene asociado 24 productos bancarios, que pueden ser cuentas corrientes, tarjetas de créditos, créditos hipotecarios y otros. Para efectos de este proyecto los productos bancarios representan las etiquetas o *labels*, si un cliente tiene un producto se representa con un 1, si pasa lo contrario se indica con un 0.

Según la exploración de datos que se hizo, se pudo observar que entre los 24 *features* de los clientes, hubo dos que representaban ruido para todo el *dataset* y se procedió a eliminar estas características, esto se hizo porque no tenían definidos los valores en un 96%, y no se podían inferir con los datos existentes. Además, se identificó que sólo 6 *features* eran de tipo numérico, mientras que el resto eran atributos categóricos, que posteriormente se cambiaron a tipo numérico. Entre las exploraciones que se hicieron a los datos originales, se puede destacar que entre los diez productos que los clientes más adquieren, hubo dos productos que presentaron comportamientos similares durante el intervalo de tiempo en el que se tienen los datos, como se muestra en la Figura 11, estos productos son los productos relacionados con la nómina de sueldos de los clientes, esto sirvió para entender que aparentemente si un cliente adquiere uno de los dos productos necesariamente adquiere el otro, así que se presenta una relación fuerte entre ellos.

Figure 11. Similitud de productos. En esta gráfica se puede observar que los productos presentan un comportamiento similar.



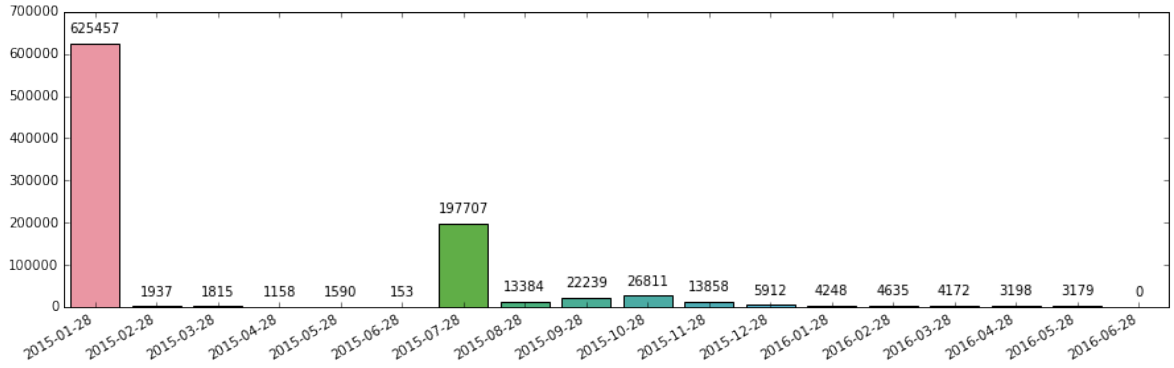

Ahora, en la Tabla 1 se puede ver la relación que hay entre compradores, que son clientes que adquieren al menos un producto cada mes, y los clientes de los que se tienen registros cada mes, la relación está definida por $\frac{compradores}{clientes\ totales}$. No se debe olvidar

que el fin de este proyecto es poder hacer recomendaciones de los productos que un cliente posiblemente vaya a adquirir el próximo mes, por esta razón se hace necesario este análisis, pues se deben tener en cuenta solo los compradores. Así pues, este análisis es necesario para observar la cantidad de compradores que puede haber cada mes, y además poder revisar si se presenta cierta tendencia entre los meses. Como análisis adicional, se puede observar que en cada mes hay un promedio de 3,6% de clientes que compran al menos un producto. También se puede observar en la Figura 12 que no hay una gran variación en la cantidad de nuevos clientes que llegan cada mes, excepto en el mes de junio del 2015 en el que hay un crecimiento importante.

Table 1. Relación entre compradores y los clientes que hay cada mes. El intervalo de la tabla inicia en febrero de 2015 porque se necesitan dos meses consecutivos para calcular los compradores que adicionan al menos un producto a los que ya tienen

| **Fecha registro** | **Cantidad clientes** | **Compradores** | **Relación** |
|---|---|---|---|
| Feb 2015 | 617487 | 23515 | 0,03835 |
| Mar 2015 | 620171 | 24936 | 0,04047 |
| Apr 2015 | 622148 | 25286 | 0,04088 |
| May 2015 | 624450 | 21034 | 0,03389 |
| Jun 2015 | 626359 | 33185 | 0,05328 |
| Jul 2015 | 806246 | 25962 | 0,04145 |
| Aug 2015 | 822016 | 20494 | 0,02545 |
| Sep 2015 | 835376 | 26435 | 0,03220 |
| Oct 2015 | 857258 | 28443 | 0,03410 |
| Nov 2015 | 883718 | 24994 | 0,02921 |
| Dec 2015 | 897769 | 31054 | 0,03519 |
| Jan 2016 | 902405 | 24346 | 0,02720 |
| Feb 2016 | 907860 | 33733 | 0,03744 |
| Mar 2016 | 912980 | 25397 | 0,02802 |
| Apr 2016 | 916992 | 24939 | 0,02735 |
| May 2016 | 920194 | 26272 | 0,02869 |

Figure 12. Cantidad de nuevos clientes cada meses



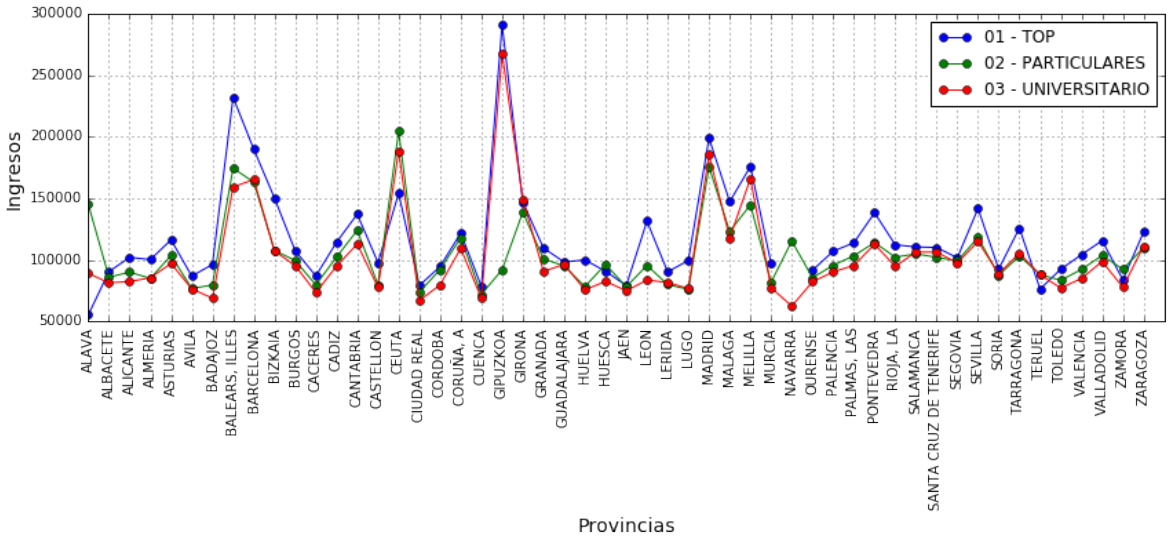

## 5.2 PROCESAMIENTO DE DATOS

Siguiendo el método propuesto, la fase que sigue es la limpieza y procesamiento de datos, y posteriormente poder construir los *datasets* necesarios para el entrenamiento de los modelos predictivos. En el análisis exploratorio realizado, se evidenció que un poco más del 20% del *feature* de *renta* no tenía datos definidos, así que se procede a hacer el análisis respectivo para definir si estos datos faltantes se pueden inferir con base a los datos existentes. Después de hacer este análisis se pudo determinar que estos datos faltantes del *feature* de *renta* se pueden inferir a partir de dos *features* que están plenamente definidos que son *segmento* y *nomprov*. A continuación, se hace una decripción de los *features* que se han mencionado hasta el momento para dar contexto al análisis hecho. El *feature* de *renta*, es de tipo númerico e indica los ingresos mensuales que tiene el cliente; el *feature* de *segmento* es una etiqueta que tiene el banco para sus clientes con el que pueden hacer una segmentación básica, como es un atributo categórico tiene tres valores posibles que son TOP, PARTICULAR y UNIVERSITARIO[1]; y el *feature* de *nomprov* indica el nombre de la provincia en la que reside el cliente, la mayoría de los clientes residen en España, evidentemente también es un atributo categórico que tiene tantos valores posibles como provincias hayan en España. Para inferir los datos faltantes de *renta* se asume que los clientes, quienes no tienen definido este atributo, reciben unos ingresos similares a la media de ingresos según el *segmento* en el que estén clasificados y su provincia de residencia. El fin de completar estos datos es para darle más forma y sentido a los datos, además, como estos datos son sintéticos

[1] La competencia de Kaggle [11], quien brinda los datos, no tiene una descripción explícita para cada valor

deben ser coherentes con el contexto del *dataset* original. Inicialmente se había pensado en completar estos datos solamente con la *renta* media de la provincia correspondiente, sin embargo, cuando se hace un análisis más profundo se puede observar que en cada provincia viven clientes que tienen valores del *feature* de *segmento* diferentes, así que no se puede completar los ingresos de esta manera, por esto, se recurre a inferir la *renta* teniendo en cuenta el lugar de residencia y el *segmento* del cliente. Un ejemplo claro para entender los diferentes ingresos que hay en cada provincia se puede ver en la Figura 13, donde un cliente categorizado como PARTICULAR que viva en Navarra tiene en promedio mayores ingresos que un cliente UNIVERSITARIO que resida en la misma provincia, cabe aclarar que esto es según el análisis del *dataset* suministrado. En la Figura 13 se puede ilustrar las diferencias de ingresos que tienen los clientes en cada provincia y según el segmento en el que se encuentren clasificados.

Figure 13. *Renta* promedio según la provincia y el segmento del cliente




Después de inferir los datos de *renta* faltantes, queda un poco menos del 2% del *dataset* sin definir, esta cantidad de datos es depreciable en un conjunto de datos que tiene un poco más de trece millones de registros, así que se pueden eliminar, además no se encontraron estrategias para inferir esos datos faltantes.

Para la realización de los experimentos que se describen más adelante se construyeron dos *datasets* base, que se describen a continuación:

- El primer *dataset* es similar al original, lo que lo diferencia es que se la hecho un procesamiento de datos que incluye la eliminación de ruido, conversión de tipos de *features* e inferencia de los datos faltantes de renta. Con respecto a su dimensión tiene 13'387.956 filas y un total de 28 *features*, se agregaron 6 atributos de más cuando se hizo la conversión de tipo categórico a numérico en los atributos de fecha. Para efectos prácticos y de nomenclatura, este *dataset* tendrá como nombre, **DATASET ORIGINAL**.

- El segundo *dataset* se basa en el anterior, este está compuesto sólo por los clientes que han comprado al menos un producto en el intervalo de tiempo desde enero 2015 a mayo 2016, esto hace que la cantidad de registros disminuya considerablemente a 420.025 con 28 *features*. Igualmente que en el punto anterior, para efectos de practicidad el nombre de este *dataset* será **DATASET PURCHASERS**, haciendo referencia a los clientes compradores.

## 5.3 EXPERIMENTOS Y RESULTADOS

Antes de describir los experimentos que se llevaron a cabo en este proyecto, es necesario fijar ciertos criterios que se deben tener en cuenta, como se dijo anteriormente se usarán los dos *datasets* base, a ambos *datasets* se les aplicarán las mismas técnicas de preprocesamiento y de muestreo, previo al entrenamiento de los modelos predictivos. La metodología para elegir el mejor modelo predictivo tiene dos fases, en la primera fase se construyen diferentes modelos predictivos que se validan con datos de un sólo mes. En la segunda fase se eligen los tres mejores resultados que se obtengan de esta primera fase para ser validados con datos de seis meses diferentes y poder ver si los modelos construidos previamente son consistentes en sus resultados con diferentes datos de prueba. Como se mencionó en el método propuesto, para la creación de los modelos predictivos se usarán los algoritmos de clasificación de árboles de decisión, o *decision trees*, y bosques aleatorios, o *random forest.*

**5.3.1 Primera fase de experimentación** Esta fase es la más extensa, se hicieron en total 8 experimentos, por cada uno de estos se construyeron dos modelos predictivos con los algoritmos de clasificación previamente mencionados, y cada uno de estos modelos fueron entrenados con los dos *datasets* base, teniendo como resultado

32 modelos predictivos validados. Es necesario recordar que la validación de los experimentos de esta primera fase se hizo con los datos de un solo mes, que en esta fase fue junio de 2016. A continuación, se describen los experimentos que se llevaron a cabo:

- El primer experimento consistió en entrenar los modelos predictivos con todos los datos previos al mes de test, esto con el fin de hacer un primer acercamiento con los modelos predictivos y generar una línea de referencia base en el *score* de precisión.

- En el segundo experimento sólo se usaron los datos del mes de mayo de 2016, mes previo al de test, previendo que con la información del mes inmediatamente anterior fuese suficiente para realizar una buena predicción.

- En el tercer experimento sólo se usaron los datos del mes de junio de 2015, el mes del año anterior al de test, asumiendo que los meses iguales, junio de 2015 y 2016, tienden a tener un mismo comportamiento.

- Los dos siguientes experimentos se basaron en el segundo experimento, previamente descrito, en los que se usaron sólo los datos del mes de mayo de 2016 con un preprocesamiento de PCA de 10 y 15 componentes respectivamente.

- En el sexto experimento que se planteó se usaron los datos del mes de mayo 2016 añadiendo como *features* los productos que los clientes tenían el mes inmediatamente anterior al mes de entrenamiento. Esto con el fin de añadir información sobre el cliente del mes anterior. En la Figura 14 se puede apreciar una pequeña vista de cómo es la estructura del *dataset*, en la que se puede ver que primero se encuentran los *features* o características del cliente, y al final se encuentran los productos con valores de 1 y 0 que indican si el cliente tiene o no ese producto.

Figure 14. Estructura de datos para el experimento 6

| fecha_dato | ncodpers | age | ••• | ind_ahorr_ult | ind_tjcr_ult |
|---|---|---|---|---|---|
| 2016-05-28 | 15889 | 45 | ··· | 0 | 0 |
| 2016-05-28 | 15890 | 28 | ··· | 1 | 0 |

- El séptimo experimento consistió en el entrenamiento de los algoritmos de clasificación con todos los datos disponibles de entrenamiento añadiendo tres *features*

que indican el cambio de algunas características del cliente durante el intervalo de tiempo del presente *dataset*. Los tres *features* adicionales que tienen un cambio son *renta*, *segmento* y el *feature* de *antigüedad*, este último es un atributo numérico que indica el tiempo en meses que un cliente ha pertenecido al banco. En la Figura 15 se ilustra los tres *features* delta que se añaden al final, si el valor que se encuentra allí es mayor a uno indica que hay un cambio positivo, si es menor que uno indica que hubo un cambio negativo y si el valor es uno no hubo ningún cambio en ese *feature*.

Figure 15. Estructura de datos para el experimento 7

| fecha_dato | ncodpers | age | ••• | renta_delta | antigueda_delta | segmento_delta |
|---|---|---|---|---|---|---|
| 2016-05-28 | 15889 | 45 | ... | 0.3 | 1.5 | 0.1 |
| 2016-05-28 | 15890 | 28 | ... | 1 | 1.9 | 1.5 |

- El último experimento consistió en añadir como *features* adicionales la cantidad de veces que un cliente ha tenido cada producto hasta el mes anterior del mes de entrenamiento, en este experimento se usaron sólo los datos del mes de mayo 2016. La adición de estos *features* se hace para aumentar la información histórica del cliente y tener un *dataset* que caracterice mejor a cada uno de los clientes. En la siguiente Figura 16 se ilustra un pequeño ejemplo de los *features* que se añadieron.

Figure 16. Estructura de datos para el experimento 8

| fecha_dato | ncodpers | age | ••• | ind_ahorr_ult | ind_tjcr_ult |
|---|---|---|---|---|---|
| 2016-05-28 | 15889 | 45 | ... | 15 | 2 |
| 2016-05-28 | 15890 | 28 | ... | 15 | 8 |

### 5.3.2 Resultados de la primera fase de experimentación

En la Tabla 2 se muestran los resultados que se obtuvieron de los experimentos previamente mencionado con cada uno de los *datasets* base. Es necesario mencionar que el *score* ideal para estas pruebas es de 0,031 aproximadamente, en este caso el *score* ideal se tomó del ganador de la competencia de Kaggle [11] en la que se basa este proyecto, así pues, se puede calcular una proporción de precisión con respecto a este *score*. Como

dato adicional, se debe indicar que el tiempo promedio de la realización de cada experimento fue de dos minutos y medio, a excepción del primer y séptimo experimento donde la cantidad de datos era considerable y marcaron un tiempo promedio de treinta minutos.

Table 2. Resultados de los experimentos de la primera fase. Para efectos prácticos se tienen ciertas convenciones en la tabla descritas así: El algoritmo Random Forest está descrito como RF, y el Decision Tree como DT. En el caso de los *datasets* base se tiene que ORIGINAL son todos los datos del dataset inicial y PURCHASERS son los datos de solamente los clientes que han comprado al menos un producto

| **Datos de entrenamiento** | **Algoritmo** | **ORIGINAL** | | **PURCHASERS** | |
|---|---|---|---|---|---|
| | | *Score* | Proporción | *Score* | Proporción |
| Todos los datos previos | RF | 0,01635 | 52,06% | 0,013831 | 44,03% |
| | DT | 0,00969 | 30,88% | 0,013449 | 44,82% |
| Datos de Mayo 2016 | RF | 0,01589 | 50,61% | 0,01411 | 44,92% |
| | DT | 0,01422 | 45,27% | 0,01364 | 43,42% |
| Datos de Junio 2015 | RF | 0,01451 | 46,19% | 0,01394 | 44,37% |
| | DT | 0,01195 | 38,04% | 0,01285 | 40,92% |
| Datos de Mayo 2016 y | RF | 0,01359 | 43,27% | 0,01379 | 43,91% |
| PCA de 10 componentes | DT | 0,01416 | 45,07% | 0,01276 | 40,61% |
| Datos de Mayo 2016 y | RF | 0,01324 | 42,15% | 0,01356 | 43,18% |
| PCA de 15 componentes | DT | 0,01422 | 45,26% | 0,01267 | 40,35% |
| Mayo/16 con productos | RF | 0,01454 | 46,31% | 0,0128 | 40,74% |
| del mes anterior | DT | 0,01452 | 46,22% | 0,01233 | 39,26% |
| Todos los datos con | RF | 0,01341 | 42,7% | 0,01351 | 43,02% |
| *features* de cambio | DT | 0,01541 | 49,07% | 0,01524 | 48,51% |
| Datos de Mayo/16 con | RF | 0,01578 | 50,23% | 0,02047 | 65,19% |
| cantidad de productos | DT | 0,01568 | 49,93% | 0,01888 | 60,11% |

En términos generales, se puede observar en los resultados que, el mejor algoritmo es el de bosques aleatorios, o *random forest*, esto se puede dar porque en principio este algoritmo es un modelo donde se ensamblan varios árboles de decisión, y por esto puede llegar a ser más robusto que un sólo árbol, aunque hay algunos experimentos donde son superiores los árboles de decisión, pero son casos particulares. Además, la mejor configuración que resultó en los experimentos fue aquella en la que se usan como datos de entrenamiento el DATASET PURCHASERS, es decir, los datos de los compradores,

junto con los *features* adicionales que indican la cantidad de cada producto que el cliente ha tenido previamente.

**5.3.3 Segunda fase de experimentación** Dicho lo anterior, ahora se puede hacer la selección de los mejores modelos predictivos para iniciar la segunda fase de experimentación. Como se observó en la Tabla 2, los mejores rendimientos se dan en el primer experimento, donde se entrenan los modelos con todos los datos previos; el segundo experimento, en el que se usa los datos de mayo 2016; y el octavo experimento, en el que se usan los datos de mayo 2016 con los *features* adicionales que representan la cantidad de veces que cada cliente ha tenido cada producto. Es necesario notar que entre estos tres experimentos el mejor algoritmo fue el de *random forest*, por esta razón, en las siguientes pruebas se usará el solo el algoritmo mencionado. Y por último, cabe recordar que en esta segunda fase se hace la validación de cada uno de los modelos elegidos con datos de seis meses diferentes, por lo tanto, en la Tabla 3 se puede observar cuáles serán los *scores* ideales para cada uno de los meses de prueba. El cálculo de estos *scores* se obtuvo de la relación entre clientes y compradores que se muestra en a Tabla 1, pues se debe tener en cuenta que la cantidad de predicciones que se hagan correctamente debe ser la misma cantidad de clientes que compran al menos un producto. Estos valores se usarán como referencia para saber el rendimiento que tiene cada uno de los modelos predictivos en esta fase de experimentación.

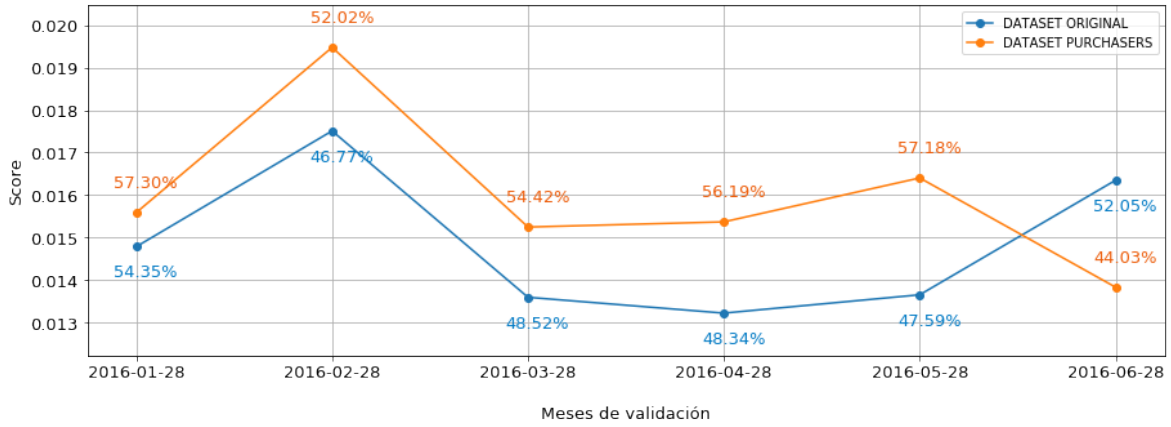
Table 3. Score ideal para cada mes de prueba

| Mes de prueba | Score ideal |
|---:|---|
| Enero 2016 | 0,02720 |
| Febrero 2016 | 0,03744 |
| Marzo 2016 | 0,02802 |
| Abril 2016 | 0,02735 |
| Mayo 2016 | 0,02869 |
| Junio 2016 | 0,03141 |

**5.3.4 Resultados de la segunda fase de experimentación** En la Figura 17 están los resultados del primer experimento con todos los seis meses de validación, en el que se entrena el algoritmo de bosques aleatorios con todos los datos disponibles

antes del mes de prueba. Como se puede ver, el *dataset* con sólo los compradores es el que mejor resultados da, a excepción del último mes, esto quizá se deba a que esa validación está hecha en el ambiente de pruebas de Kaggle [11], donde usa sólo un 70% de los datos para probar, mientra que en el resto de meses se utiliza en la validación el 100% de los datos de prueba. En cada una de las siguientes figuras que ilustran los resultados de esta fase de experimentación, se puede observar que están los porcentajes de rendimiento para cada una de las pruebas realizadas, este rendimiento se calcula con respecto a cada *score* ideal que se menciona en la Tabla 3.

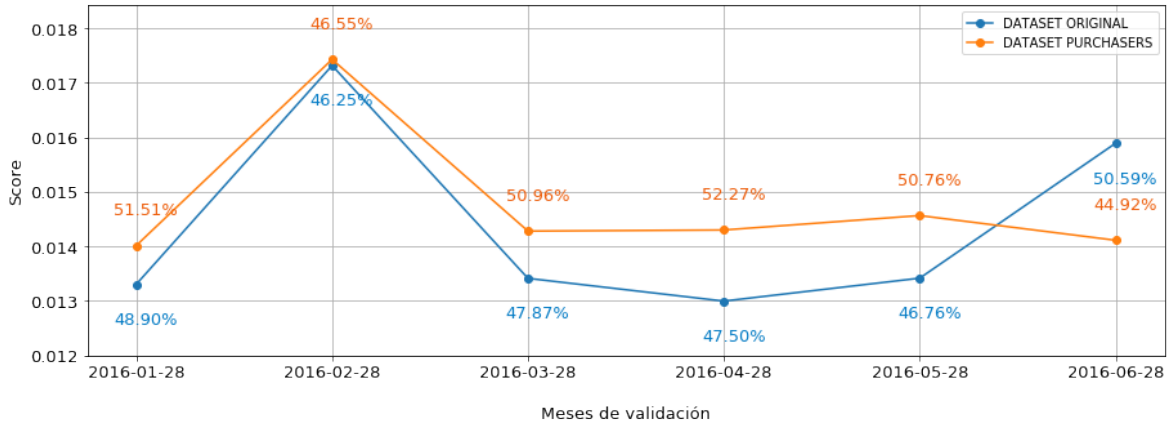
Figure 17. Resultados del experimento 1 con seis meses de prueba




Los resultados del segundo experimento, en el que se entrena el algoritmo de bosques aleatorios con sólo los datos del mes previo al de prueba, están reflejados en la Figura 18, en esta se puede ver que la brecha que separa los *scores* de cada prueba es mucho menor que en la del experimento anterior. También se puede apreciar que en la mayoría de pruebas el *dataset* base de los compradores es el que tiene un mejor rendimiento con respecto al *dataset* original, sin embargo en el último mes hay un cambio, esto podría ocurrir, como se mencionó antes, por la cantidad de datos con la que se hizo esa validación.

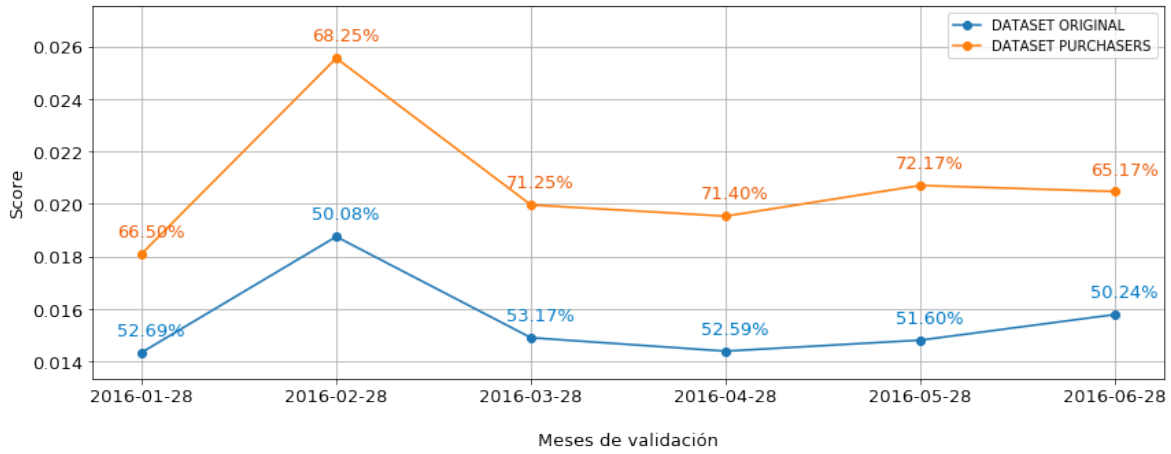
Y por último, en los resultados del octavo experimento, ilustrados en la Figura 19, siguiendo con la numeración de los experimentos que se tenía en la primera fase de experimentación, se tiene una mejora considerable frente a los dos experimentos anteriores en cuanto a rendimiento, además se ve que tiene la misma tendencia de los anteriores resultados donde el *dataset* de compradores tiene mejor rendimiento que el otro conjunto

Figure 18. Resultados del experimento 2 con seis meses de prueba




de datos. Hay que recordar que este experimento consiste en entrenar el algoritmo de bosques aleatorios con los datos del mes previo al mes de prueba y además añadiéndole *features* que indican la cantidad de veces que un cliente ha tenido cada producto que ofrece el banco, la estructura básica de estos datos está en la Figura 16.

Figure 19. Resultados del experimento 8 con seis meses de prueba




En la Tabla 4 se encuentran resumidos los resultados de los tres experimentos previos, en donde se puede ver de una manera más clara el rendimiento promedio de cada una de los experimentos realizados.

Table 4. Rendimiento promedio de la segunda fase de experimentación

| Prueba | Rendimiento promedio |
|---|---|
| Experimento 1 con DATASET ORIGINAL | 49,60% |
| Experimento 1 con DATASET PURCHASERS | 53,52% |
| Experimento 2 con DATASET ORIGINAL | 47,98% |
| Experimento 2 con DATASET PURCHASERS | 49,50% |
| Experimento 8 con DATASET ORIGINAL | 51,73% |
| Experimento 8 con DATASET PURCHASERS | 69,12% |

Finalmente, se aprecia que el rendimiento promedio de los dos primeros experimentos es cercano al 50%, lo cual indica que la dimensión en filas de los *datasets* no influye demasiado para generar un cambio significativo en la precisión de los modelos, mientras que en el experimento 8, que es el que muestra mejor rendimiento, se puede inferir que el *dataset* con solo los compradores y con la información histórica del cliente que se añadió, sirvió para ajustar mucho más el modelo al rendimiento deseado, e incluso, se puede ver que con respecto a los dos experimentos anteriores, el rendimiento mejora en cada uno de los meses, lo que indica que es un modelo predictivo robusto y fiable.

# *Capítulo 6*
# *CONCLUSIONES Y PERSPECTIVAS*

En este proyecto se presentó un método para hacer recomendaciones de productos bancarios a través de técnicas de *machine learning*. El método propuesto se realizó en diferentes etapas iniciando en la exploración y procesamiento de datos, la construcción de diferentes modelos predictivos, hasta la validación de estos modelos con diferentes conjuntos de datos. Finalmente, entre todos los 32 experimentos iniciales, se pudo elegir un modelo predictivo que alcanzó un rendimiento promedio del 69,12%, este resultado se logró, porque se aplicaron ciertas técnicas de *feature engineering* al *dataset* inicial, haciendo que se redujera su dimensión y aumentara su información de tal manera que el algoritmo de clasificación se ajustara al rendimiento deseado. Las ventajas que ofrece este modelo, es que puede generar resultados aceptables con pocos datos de entrenamiento, como los registros de un solo mes, y evitando el problema de *cold start*, además el tiempo de ejecución de este modelo tiene un promedio de 3 minutos para generar las recomendaciones de todos los clientes. Por último, este trabajo, presenta un enfoque diferente en los sistemas de recomendación del sector bancario y financiero que se mencionaron como estado del arte, pues estas recomendaciones se generan usando solamente técnicas de *machine learning*, lo que puede ser una oportunidad para seguir profundizando en el tema, y mejorar las técnicas que se usaron. Para posibles trabajos derivados, se puede hacer un sistema híbrido con las técnicas tradicionales que existen en sistemas de recomendación y las técnicas usadas en este proyecto.

# *REFERENCIAS BIBLIOGRÁFICAS*


[1] Ricci, F., Rokach, L., and Shapira, B. Introduction to recommender systems handbook. In *Recommender Systems Handbook*. Spring, Boston, MA, 2011, ch. 1, pp. 1–39.

[2] Leskovec, J., Rajaraman, A., and Ullman, J. Recommendation systems. In *Mining of Massive Datasets*. June 2011, pp. 287–320.

[3] Burke, R. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction* (2002), 331–370.

[4] Su, X., and Khoshgoftaar, T. M. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence* (August 2009).

[5] Burke, R. Hybrid web recommender systems. *The Adaptive Web* (2007).

[6] Gallego, D., and Huecas, G. An empirical case of a context-aware mobile recommender system in a banking environment. *2012 Third FTRA International Conference on Mobile, Ubiquitous, and Intelligent Computing* (2012).

[7] Abdollahpouri, H., and Abdollahpouri, A. An approach for personalization of banking services in multi-channel environment using memory-based collaborative filtering. In *The 5th Conference on Information and Knowledge Technology* (May 2013), pp. 208–213.

[8] Gigli, A., Lillo, F., and Regoli, D. Recommender systems for banking and financial services. In *RecSys 2017 Poster Proceedings* (August 2017).

[9] Asosheh, A., Bagherpour, S., and Yahyapour, N. Extended acceptance models for recommender system adaption, case of retail and banking service in iran. *WSEAS Transactions on Business and Economics 5* (May 2008).

[10] Moro, S., Cortez, P., and Rita, P. A data-driven approach to predict the success of bank telemarketing. *Decision Support Systems 62* (2014), 22 – 31.

[11] Santander Product Recommendation. Competencia de Kaggle, 2017. https://www.kaggle.com/c/santander-product-recommendation.

[12] Paradarami, T. K., Bastian, N. D., and Wightman, J. L. A hybrid recommender system using artificial neural networks. *Expert Syst. Appl. 83*, C (Oct. 2017), 300–313.

[13] Amatriain, X., Jaimes*, A., Oliver, N., and Pujol, J. M. *Data Mining Methods for Recommender Systems*. Springer US, Boston, MA, 2011, pp. 39–71.

[14] Bowles, M. Understand the problem by understanding the data. In *Machine Learning in Python. Essential techniques for predictive analysis*. Wiley, 2015, pp. 23–73.

[15] Jolliffe, I. Principal component analysis. Springer, 2002.

[16] Nilson, N. J. *Introduction to Machine Learning*. November 1998.

[17] Rokach, L., and Maimon, O. *Data Mining with Decision Trees. Theory and Applications*, vol. 69. World Scientific Publishing Co., 2008.

[18] Breiman, L. Random forests. *Machine Learning 45*, 1 (2001), 5–32.

[19] Zygmunt, Z. What you wanted to know about Mean Average Precision, 2009. http://fastml.com/what-you-wanted-to-know-about-mean-average-precision Tomado el 2018-07-26.

[20] Sawtelle, S. Mean Average, Precision (MAP) for Recommender Systems, 2016. sdsawtelle.github.io/blog/output/mean-average-precision-MAP-for-recommender-systems.html Tomado el 2018-07-26.

# *BIBLIOGRAFIA*


ABDOLLAHPOURI, Himan; ABDOLLAHPOURI, Alireza. An approach for personalization of banking services in multi-channel environment using memory-based collaborative filtering. The 5th Conference on Information and Knowledge Technology, 2013. p. 208 – 213.

AMATRIAIN, Xavier, et al. Data Mining Methods for Recommeder Systems. Recommender Systems Handbook, Boston: Spring, 2011. p. 39 – 71.

ASOSHEH, Abbas, et al. Extended acceptance models for recommender system adaption, case of retail and banking service in Iran. WSEAS Transactions on Business and Economics 5, 2008.

BOWLES, Michael. Understand the problem by understanding the data. Machine Learning in Python. Essential techniques for predictive analysis. Willey, 2015. p. 23 – 73.

BREIMAN, Leo. Random Forest. Machine Learning. Vol. 45. 2001. p. 5 – 32.

BURKE, Robin. Hybrid Recommender Systems: Survey and Experiments. User Modeling and User-Adapted Interaction, 2002. p. 331-370.

BURKE, Robin. Hybrid web recommender systems. The Adaptive Web, 2007.

GALLEGO, Daniel; HUECAS, Gabriel. An empirical case of a context-aware mobile recommender system in banking environment. 2012 Third FTRA International Conference on Mobile, Ubiquitous, and Intelligent Computing, 2012.

GIGLI, Andrea, et al. Recommender Systems for Banking and Financial Services. RecSys 2017 Poster Proceedings, 2017.

JOLLIFFE, Ian T. Principal Component Analysis. Ed. 2, Springer, 2002.

LESKOVEC, Jure, et al. Recommendation systems. Mining of Massive Datasets. Cambridge University Press, 2011. p. 287 – 320.

MORO, Sérgio, et al. A data-driven approach to predict the success of bank telemarketing. Decision Support Systems 62, 2014. p. 22 – 31.

NILSON, Nils J. Introduction to Machine Learning. 1998.

PARADARAMI, Tulasi K, et al. A hybrid recommender system using artificial neural networks. Expert Syst. Appl. 83, C. 2017. p. 300 – 313.

RICCI, Francesco, et al. Introduction to recommender systems handbook. Recommender Systems Handbook. Boston: Spring, 2011. p. 1-39

ROKACH, Lior; MAIMON, Oded Z. Data Mining with Decision Trees. Theory and Applications. Vol. 69, World Scientific Publishing Co., 2008.

SANTANER PRODUCT RECOMMENDATION [en línea]. Competencia de Kaggle, 2017. Disponible en: https://www.kaggle.com/c/santander-product-recommendation

SAWTELLE, Sonia. Mean Average Precision (MAP) for Recommender Systems [en línea], 2016. (Recuperado en 26 de Julio de 2017) Disponible en: http://sdsawtelle.github.io/blog/output/mean-average-precision-MAP-for-recommender-systems.html

SU, Xiaoyuan; KHOSHGOFTAAR, Taghi M. A survey of collaborative filtering techniques. Advances in Artificial Intelligence, 2002. 19 p.

ZYGMUNT, Z. What you wanted to know about Mean Average Precision [en línea], 2009. (Recuperado en 26 de Julio de 2017) Disponible en: http://fastml.com/what-you-wanted-to-know-about-mean-average-precision/