Benemérita Universidad Autónoma de Puebla

Facultad de Ciencias Físico Matemáticas

Aprendizaje automático para el desarrollo de procesos en las instituciones financieras

Tesis presentada al

Colegio de Actuaría

como requisito parcial para la obtención del grado de

LICENCIADO EN ACTUARÍA

por

Luis Felipe Heros Cárdenas

Asesorado por

M.C. Edgar Santiago Moyotl Hernández

Puebla Pue.

Mayo 2022

Aprendizaje automático para el desarrollo de procesos en las instituciones financieras

Luis Felipe Heros Cárdenas

Mayo 2022

Título: Aprendizaje automático para el desarrollo de procesos en las instituciones financieras

Estudiante: LUIS FELIPE HEROS CÁRDENAS

COMITÉ

M.I. Mónica Macías PérezPresidente

M.F. Jorge Luis Reyes García
Secretario

M.C. José Hernández Asunción
Vocal

M.C. Edgar Santiago Moyotl Hernández
Asesor

Índice general

Índice de figuras

Índice de tablas

Resumen

El presente trabajo tiene como objetivo exponer dos de las principales áreas del conocimiento actuarial y su relación en la práctica. En este sentido, se muestra la importancia de los procesos, la administración de riesgos y de la toma de decisiones en las instituciones financieras, además se estudian las técnicas de agrupación y clasificación, pertenecientes al aprendizaje automático como herramientas de interés en la automatización y mejora de los propósitos de dichas instituciones. De la misma manera, se explica cómo implementar estos algoritmos, desde el preparar y analizar los datos hasta evaluar el desempeño de los resultados obtenidos. Para concluir se presentan dos aplicaciones en el lenguaje de programación Python, aplicaciones que las instituciones financieras emplean y donde se reafirma la utilidad de los conocimientos previamente mencionados.

Palabras clave: *Aprendizaje automático, Clasificación, Agrupamiento, Riesgo de crédito, Inteligencia de negocios.*

Introducción

Una de las características más interesantes de la ciencia actuarial es su amplia variedad de enfoques. Un actuario tiene la capacidad de desenvolverse en distintos entornos como administración de riesgos, seguros, ciencia de datos, finanzas y economía por mencionar algunos. De igual forma, es cautivador cómo estos campos pueden relacionarse entre ellos, dando lugar a actividades particularmente interesantes.

La *administración de riesgos* ha sido crucial para la estabilidad de la economía mundial, un mal manejo podría concluir en consecuencias desastrosas originando una crisis económica, tal cual la historia lo ha demostrado, provocando una serie de reacciones en cadena en donde se perderían empleos, ahorros de familias e incluso fondos de pensiones de los trabajadores. En la actualidad, los estándares para evitar situaciones no deseadas y realizar un correcto manejo de las posibles contingencias financieras y no financieras han ido en aumento, y las técnicas que dan sustento a las estrategias de gestión son cada vez más sofisticadas. Las empresas no sólo han descubierto cómo administrar correctamente sus vulnerabilidades, sino también han aprendido a ampliar y perfeccionar sus estrategias para impulsar sus negocios. Obtener ganancias es una consecuencia de una correcta gestión, una serie de adecuadas decisiones y el desarrollo de metodologías eficientes. La *inteligencia de negocio* (Vercellis, 2011) consiste principalmente en la implementación de modelos matemáticos para explotar los datos de tal manera que se obtengan conocimientos útiles para la toma de decisiones, buscando claramente elecciones más acertadas y encaminadas a fin de tener mayores rentabilidades.

Para lograr los objetivos previamente mencionados se utiliza el *aprendizaje automático*, el cual es una rama de la *inteligencia artificial* que tiene por objetivo darle sentido a los datos convirtiéndolos en conocimiento mediante algoritmos matemáticos y/o estadísticos que sirven para la agrupación, clasificación, regresión o pronóstico de datos (Raschka, 2015; Torres, 2018). El aprendizaje automático también es de gran utilidad para la automatización de procesos y para la realización de tareas complicadas, por lo que es un excelente apoyo para la inteligencia de negocios y la administración de riesgos.

En el presente trabajo se procede de la siguiente manera. En el Capítulo 1 se realiza un resumen panorámico enfocado en los conceptos y procesos financieros utilizados en capítulos siguientes. A continuación, en el Capítulo 2 se explican determinadas técnicas de agrupación y clasificación que forman parte del aprendizaje automático. A modo de fusión en los Capítulos 3 y 4 se exponen formas de generar buenos modelos mediante tareas fundamentales como son el preprocesamiento de datos, análisis exploratorio de datos y medidas de evaluación, logrando en conjunto exponer cómo las instituciones financieras y empresas en general, utilizan y toman ventajas de los métodos descritos en favor de resolver distintas problemáticas asociadas a la disminución de sus riesgos y aumento de sus ganancias. En el apéndice B se describe el entorno de trabajo que se usó para poder probar los códigos que a lo largo del documento se presentan.

Capítulo 1

Procesos en las instituciones financieras

Para monitorizar, regular y asegurar un correcto funcionamiento, todas las empresas cuentan con procesos que brindan soporte, facilitan y mecanizan las actividades requeridas para sus escalamientos a la gerencia correspondiente. Un *proceso*, de forma resumida, puede definirse como una serie de acciones o tareas que se realizan para llegar a un resultado preestablecido. Gracias a las computadoras, se pueden efectuar procesos complejos de forma eficiente lo cual conlleva a mejores resultados en el desempeño de la institución. A manera de ejemplo, si se tiene la tarea de ubicar un monto de interés en determinados archivos, es considerablemente más eficaz correr una instrucción en la computadora que realizar manualmente la labor. De la misma forma, existen cálculos o análisis estadísticos, matemáticos y financieros por dar algunos ejemplos, que se ejecutan significativamente más rápido con un proceso automatizado que si un equipo entero de trabajadores se dedicaran a hacerlo. Por este motivo, la frecuencia con la que aparecen la tecnología y los lenguajes de programación [1] aplicados a los procesos de las empresas es cada vez mayor.

Un claro ejemplo de una situación en la que la automatización de procesos puede ser fuertemente necesaria en las empresas es en los sucesos inconvenientes como enfermedades o accidentes. Una empresa funciona gracias a la organización de las posiciones profesionales y la correcta distribución entre cada una de ellas. Por lo regular cuando hay una vacante, el equipo u otras personas deben ser capaces de cubrir las necesidades principales mientras la posición se ocupa. Sin embargo, en ocasiones determinadas, como en los sucesos de pandemias, los trabajadores suelen enfermarse dejando numerosas actividades para realizar sin una persona responsable que las haga. Aunque todo esto suceda, ciertos procesos y actividades tienen que realizarse, por lo que otras personas tendrán que encargarse de efectuar el trabajo, lo cual para ellos será sumamente complicado. Al haber procesos automatizados, este problema reduce considerablemente su gravedad, permitiendo que las tareas sean más fáciles de ser ejecutadas tanto para los responsables del ejercicio como para algún delegado que lo tenga que hacer en situaciones de emergencia.

Es importante mencionar, que una gran parte de las labores que las instituciones financieras realizan son regulativas, es decir, son requeridas por organismos que se dedican a regular el correcto funcionamiento de este tipo de empresas. En México, los ejemplos de organismos del sistema financiero pueden ser la Comisión Nacional Bancaria de Valores (CNBV), la Secretaría de Hacienda y Crédito Público (SHCP), o los que aparecen en la figura 1.1.

1.1. Instituciones financieras

En la amplia gama de empresas que existen en la actualidad, hay un tipo de sociedades que se dedican principalmente al área de banca, valores y seguros. Las instituciones financieras ofrecen productos y servicios considerados por la ley como financieros que, explicados de forma práctica, son servicios relacionados con el dinero de los clientes. Las más populares son las pertenecientes al sector bancario, donde sobresalen

[1] Algunos de los lenguajes utilizados en las instituciones bancarias son SAS, SQL, VBA, Python, R, etc.

Figura 1.1: Instituciones pertenecientes al sistema financiero mexicano.

empresas como HSBC, BBVA, Citi y Banorte, pero también existen instituciones financieras no bancarias como las cajas populares de ahorro y crédito, aseguradoras, casas de bolsa, sistemas de ahorros para el retiro, fondos financieros, fideicomisos, entre otras.

Estas instituciones, al tratar con dinero de otras personas, están altamente reguladas para asegurar a los consumidores el buen manejo del mismo, evitando así robos o estafas. Las reglamentaciones a las que están sujetas estas entidades van más allá de sólo la protección del dinero, sino que también protegen los datos que poseen sobre sus usuarios. Es tanto el nivel de atención que se tiene para proteger el dinero de las personas que existe una organización en México llamada Instituto para la protección al ahorro bancario (IPAB) el cual como su nombre lo indica, asegura el dinero que las personas tienen en el banco hasta por 400 mil Unidades de Inversión (UDIs) [2]. Análogamente, es tal la protección de los datos de los usuarios que de romper esta práctica también implica romper la ley y podría traer consecuencias graves para los involucrados. De la misma manera, desempeñan un papel fundamental en prevenir y evitar malas prácticas como lo son el financiamiento al terrorismo o el lavado de dinero.

Las instituciones financieras se dedican a captar recursos y dinero de fuentes externas, ya sean personas, empresas u otras instituciones financieras a cambio de un servicio determinado. Por ejemplo, las casas de bolsa permiten a sus clientes comprar y vender acciones y otros valores en el mercado, mientras que las aseguradoras brindan protección y cobertura monetaria en caso de ocurrencia de siniestros especificados. Los bancos proveen el servicio de guardar el dinero de los clientes y permitirles disponer de él a través de tarjetas de débito o cajeros automáticos. El dinero obtenido de los clientes, suele ser invertido de diversas maneras, la principal es prestando el dinero en forma de créditos, los cuales pueden ser apoyos a inversiones o soluciones para problemas no previstos. Las inversiones pueden ser desde disponer de un pequeño monto de dinero para comprar mercancía que se planea vender en el futuro hasta presupuestos para grandes proyectos inmobiliarios. De la misma forma, los problemas no previstos pueden ir desde disponer de dinero para alguna compra unos días antes de recibir una nómina hasta refinanciar una deuda hipotecaria para no perder un hogar.

Al tratarse de dinero, todos los procedimientos deben llevarse a cabo con mucho cuidado, y cada cumplimiento regulativo tiene que tomarse con la seriedad necesaria. En el mismo sentido, la forma de sacar rentabilidad del negocio no puede ser muy arriesgada, ya que el dinero que se pone en peligro es principalmente el de los usuarios. Para lograr resolver con éxito este problema, las instituciones financieras se han apoyado de metodologías y procedimientos a seguir, estos pueden variar dependiendo de cada área de la empresa. Las secciones en las que estas organizaciones están divididas puede ser muy amplia, ya que se necesitan servicios de leyes, contaduría, informáticos, etcétera. En particular, este trabajo se enfocará en como aprovechar la programación en computadoras para automatizar, mejorar y/o facilitar la ejecución de ciertos procesos de administración de riesgos y toma decisiones.

[2]Una UDI equivale aproximadamente a 7 pesos ($7.18 a finales de febrero 2022) por lo que 400 mil UDIS serían aproximadamente 2,800,000 pesos .

1.2. Administración de riesgos

Podemos asegurar que todas las formas de inversión conllevan un determinado nivel de riesgo, dicho de otra manera, siempre existe la posibilidad de pérdida del dinero invertido. Esto quiere decir que existe mayor riesgo cuando hay una mayor incertidumbre sobre cual será el valor esperado de retorno en una inversión, esta cantidad prevista puede variar tanto para bien como para mal, lo que explica la frase "a mayor riesgo mayor ganancia". Al hablar de empresas, específicamente de instituciones financieras es claro que las inversiones y la toma de riesgos es algo que se ejecuta día con día. La *administración de riesgos* busca eliminar la incertidumbre sobre los flujos de efectivo para que las instituciones continúen cosechando ganancias mientras su capital permanece con el menor peligro posible. Para esto se tienen que identificar todos los riesgos potenciales a los cuales la entidad está expuesta y así poder controlarlos. Es importante resaltar que para las inversiones financieras existen técnicas para controlar el riesgo y reducirlo, pero no se puede erradicar de forma definitiva.

Con el fin de lograr lo anterior, se tiene que saber el apetito de riesgo [3] que se decidió para la institución; tener políticas y sistemas definidos que faciliten el manejo de los procesos. Así mismo, tienen que existir metodologías para la medición del riesgo actual y futuro, tanto en los escenarios con mayor probabilidad de ocurrencia como en escenarios estresados. En otras palabras, es importante saber que el nivel de riesgo no se mantendrá igual a lo largo de la vida de las inversiones, por lo que se tienen que considerar situaciones futuras adversas, como pueden ser crisis o pandemias.

Las instituciones financieras son muy importantes para la economía de un país, si llegaran a quebrar, millones de personas perderían sus fondos, ahorros e inversiones. Con el propósito de que esto no suceda, existen ciertas leyes y regulaciones nacionales e internacionales que sirven para asegurar que las empresas cuenten con los requisitos mínimos para que su probabilidad de quiebra sea muy baja. Por ejemplo, para los bancos, una de las más importantes regulaciones es *El acuerdo de Basilea*.

1.2.1. Basilea

como se explica en (Comité De Basilea, 1999), el acuerdo de Basilea es un convenio donde están marcados los estándares internacionales para la regulación de los bancos. El propósito del acuerdo es asegurarse que los bancos mantengan suficiente capital para afrontar los riesgos a los cuales se enfrentan y así crear un ambiente económico estable y confiable para los consumidores. Este acuerdo se ha renovado llegando a ser ahora Basilea IV. A pesar de que la mayoría de los bancos se siguen regulando por Basilea II, estos se preparan para hacer el cambio en sus procesos y alinearse al nuevo acuerdo.

Un punto de vista bastante interesante, es que si las instituciones como los bancos desean crecer, es decir, inyectar más capital en préstamos o en inversiones, tienen que inyectar también más capital a sus reservas, por lo tanto, los bancos al buscar tener un crecimiento lo hacen de una manera estable y con menor riesgo. Algunos de los tipos de riesgo explicados en este acuerdo son el riesgo de crédito, riesgo de liquidez, riesgo de mercado y riesgo operacional.

1.2.2. Riesgo de mercado

El riesgo de mercado según (Comisión Nacional Bancaria y de Valores, 2021) es la pérdida potencial por cambios en los factores de riesgo que inciden sobre la valuación o sobre los resultados esperados de las operaciones activas, pasivas o causantes de pasivo contingente, tales como tasas de interés, tipos de cambio e índices de precios, entre otros. Por ejemplo, al adquirir una acción en otro país, al momento de vender la acción tendrá cierta variación respecto a su precio de compra, de la misma forma esta acción se liquidará en la moneda correspondiente del país en donde cotiza, por lo que si esta moneda se devaluó o se revalorizó respecto a la moneda local también representará una pérdida o ganancia. Además, para hacer el cambio entre divisas también se cobran comisiones, todo esto sin mencionar los impuestos o los costos operativos.

Existen diferentes formas en las que se lleva a la práctica el control del riesgo de mercado, la más popular actualmente, en gran parte gracias a las *criptodivisas*, es la compra venta a corto plazo de los activos finan-

[3] Riesgo que la institución está dispuesta a exponerse.

cieros. También están los portafolios de inversión en donde se busca tener una cartera rentable, esto se logra teniendo activos diversificados por lo que si uno llega a bajar su precio, otro activo seguramente esté subiendo al mismo tiempo. Al contrario de la compra venta a corto plazo de un solo activo esto trae menores rentabilidades pero definitivamente menores riesgos de pérdida. Es preciso señalar que las criptomonedas no son aceptadas al día de hoy en el sector bancario.

1.2.3. Riesgo operacional

El riesgo operacional es un tipo de riesgo que incluso fue llamado tonto cuando fue introducido en Basilea II, sin embargo, dejaron de llamarlo así cuando los supervisores mostraron numerosas pérdidas operacionales que alcanzaban las 9 cifras. La definición formal en (Comisión Nacional Bancaria y de Valores, 2021) dice que el riesgo operacional es la pérdida potencial por fallas o deficiencias en los controles internos, por errores en el procesamiento y almacenamiento de las operaciones o en la transmisión de información, así como por resoluciones administrativas y judiciales adversas, fraudes o robos. Algunos ejemplos de lo que abarca el riesgo operacional son los fraudes internos o externos, malas prácticas por parte de clientes, mala realización, fallas en los procesos o daños a los activos físicos por causas extremas como el terrorismo. Este tipo de riesgo es muy complicado de medir y de manejar, su principal objetivo es identificar todas las causas posibles de riesgo y cubrirse contra todas esas posibles pérdidas.

1.2.4. Riesgo de liquidez

El riesgo de liquidez es el responsable de la quiebra de muchas instituciones durante la crisis hipotecaria del 2007. De acuerdo con (Comisión Nacional Bancaria y de Valores, 2021) el riesgo de liquidez puede ser definido como la incapacidad para cumplir con las necesidades presentes y futuras de flujos de efectivo afectando la operación diaria o las condiciones financieras de la institución. Para un banco, tener suficiente liquidez significa poder hacer pagos en efectivo cuando sea requerido, el no tener esta capacidad podría tener como consecuencia la venta de los activos a menores precios por lo que significaría pérdidas de dinero. Igualmente en las transacciones o en la inversión de valores, tener liquidez se traduce en poder vender los activos de forma rápida a precio de mercado, esto sucede cuando un activo tiene volúmenes elevados de transacciones por lo que su precio se mantiene más estable. Cabe mencionar que si se tiene liquidez de más, también se está perdiendo la oportunidad de tener ese dinero invertido en activos con mayor rendimiento que traerían mayores ganancias para la empresa.

1.2.5. Riesgo de crédito

En (Comité De Basilea, 1999) el *riesgo de crédito*, de especial interés para este trabajo, se define como la posibilidad de que un prestatario o contra parte no pueda cumplir con sus obligaciones de acuerdo con los términos acordados. En otras palabras, se entiende por riesgo de crédito la exposición a la posible pérdida monetaria que puede ocurrir al prestar dinero, la cual es la actividad más importante para generar riqueza por parte de los bancos. De la misma manera, siguiendo con el objetivo de disminuir las pérdidas, esta actividad se encarga de definir estrategias para maximizar la recuperación de los créditos deteriorados o aparentemente perdidos.

Los créditos son una herramienta financiera sumamente útil, gracias a ellos existe el mundo tal y como lo conocemos. Sería prácticamente imposible comprender la grandeza de algunas empresas o el desarrollo de ciertos países sin la existencia del apalancamiento. No es una coincidencia que algunos de los países más endeudados del mundo sean también de los más desarrollados, esto se debe a que los créditos fueron usados para impulsar el crecimiento económico de estos lugares y de esta forma, ser capaces de solventar su deuda y ganar aún más dinero.

Algo parecido sucede con las empresas o empresarios, cuando recurren a la búsqueda de inversionistas, los cuales están dispuestos a prestar su dinero a otras personas con el fin de desarrollar sus proyectos e ideas. Los inversionistas, por supuesto, ponen ese dinero a disposición con el objetivo de tener un fin lucrativo para ellos, ya que a cambio de ese desembolso, ellos obtendrán parte de las ganancias derivadas del proyecto y/o participación en la empresa donde invirtieron. En el caso de que no se consigan inversionistas,

lo más común es solicitar un crédito bancario el cual permita la realización de las iniciativas de negocio. como contra parte, el banco espera la suma prestada de vuelta más una ganancia preestablecida por anticipar el dinero a su cliente. Análogamente, podemos comentar sobre los créditos más comunes y abundantes aunque de menor cantidad, los préstamos minoristas que se le dan a las personas para poder financiar sus casas, autos, estudios, bienes, etcétera.

Los créditos otorgados por las instituciones financieras se dividen principalmente en dos secciones, la cartera minorista y la mayorista, diferenciadas por los tipos de productos y segmentos de la población a los que están enfocadas. La cartera minorista es probablemente la más conocida, ya que se compone por numerosos préstamos de cantidades relativamente pequeñas otorgadas a personas físicas, se divide principalmente en financiamientos para consumo y vivienda; los préstamos de consumo pueden ser tarjetas de crédito, créditos automotrices, para adquisición de bienes de consumo duradero, entre otros. Por su parte, la cartera mayorista se compone por un número bajo de préstamos de cantidades grandes enfocados a personas morales, puesto que estos créditos son otorgados a empresas, otras entidades financieras y gobiernos.

Así como los créditos pueden ser una herramienta útil para ambas partes del contrato, también pueden traer consecuencias negativas para ambas partes. Este trabajo se enfocará completamente en el punto de vista de las instituciones de crédito. ¿Qué pasa si un prestatario no puede pagar su deuda? Sin adentrarse a elementos como las garantías de los préstamos, condonaciones, castigos, reconstrucciones por mencionar algunos ejemplos, la respuesta es la pérdida de la suma total de los pagos restantes del crédito.

Si bien es cierto que un incumplimiento del pago total de un préstamo no es capaz de llevar a la quiebra a una institución financiera, cierta cantidad de pagos incumplidos si puede ser responsable. Esta es la razón por la que debe existir un manejo del riesgo de crédito, para poder controlar en su mayoría los riesgos que el incumplimiento de las obligaciones de los clientes pueden provocar.

1.2.6. Factores de riesgo de crédito

Es importante mencionar que todas las definiciones utilizadas en esta sección fueron tomadas de (Banco de México, 2005).

- La *probabilidad de incumplimiento* es la medida de qué tan probable es que un acreditado deje de cumplir con sus obligaciones contractuales. Invariablemente, todos los préstamos conllevan cierta cantidad de riesgo, incluso los otorgados a empresas grandes y estables o gobiernos. La probabilidad de incumplimiento es una medida de probabilidad como su nombre lo indica y por lo tanto se mide de 0 a 1 donde 0 significa que no existe posibilidad de que la contraparte no pague su deuda y 1 cuando es completamente seguro que no la pagará. Esta medida tiene que ser vigilada a lo largo de todo el préstamo, principalmente al inicio, para así saber si la persona que solicita el crédito va a pagar.

  La definición de incumplimiento puede llegar a ser subjetiva, y puede variar dependiendo de la institución financiera. Incumplimiento puede definirse como dejar de cumplir con las obligaciones, pero retrasarse en un número determinado de pagos no significa que el cliente ya no pagará. Es por eso que existen otras definiciones más concretas para incumplimiento como un número específico de días de atraso en el pago, esto también puede depender del portafolio o tipo del producto que se esté considerando.

- La *exposición* es lo que debe el deudor cuando ocurre el evento de incumplimiento. No es lo mismo que un deudor no pague la cantidad inicial que se le fue otorgada a que no pague el último pago para liquidar su deuda, claramente la primera opción es la que representaría una mayor pérdida para la institución prestadora.

- Se entiende como *severidad de la pérdida* a lo que pierde el proveedor en caso de incumplimiento del deudor y se mide como una proporción de la exposición. Una vez que ya se haya incurrido a incumplimiento y esté cuantificado la exposición al momento de quebrantar las obligaciones de pago, no todo el dinero está perdido, esto se debe a que existen estrategias por parte de las empresas para poder reducir la mayor cantidad de pérdida posible. Esto puede hacerse mediante quitas, ventas de

cartera o cobros de garantías previamente incluidas en el contrato, como es en su mayoría en los casos de los préstamos hipotecarios.

- La *concentración de cartera* significa que hay mucho crédito en pocas manos, lo cual puede ser riesgoso. Un ejemplo sería una situación en donde una institución de crédito tenga un cliente principal, el cual representa el 70 % del dinero prestado, una empresa fuerte y estable que es capaz de afrontar sus deudas debido a sus ventas constantes en el sector telefónico. Existe la probabilidad del surgimiento de una nueva empresa telefónica que arrase con las ventas de tal forma que lleve a la quiebra a varios de sus competidores, incluyendo la compañía que representa el 70 % de los créditos otorgados de la institución de crédito en cuestión. El resultado del surgimiento de la innovadora empresa telefónica llevaría a la quiebra no sólo a sus competidores sino también al prestador que no supo cómo administrar correctamente su riesgo de concentración. Esta es la razón por la cual, lo ideal para tener buenas prácticas de riesgo es que las empresas diversifiquen correctamente los sectores a los cuales dan préstamos, porque ningún sector ni ninguna región geográfica están exentos de entrar en crisis.

  Vale la pena resaltar que a mayor riesgo, mayor ganancia, por lo cuál existen calificadores de empresas las cuáles otorgan calificaciones crediticias, con el fin de que los bancos sepan que tan riesgoso sería el otorgamiento de un crédito. Claro que esto depende del apetito de riesgo de cada institución financiera, puesto que, mientras menor calificación crediticia tengan, mayor es la tasa de interés cobrada para compensar el riesgo que se está tomando.

- La *pérdida esperada* indica cuánto se puede perder en promedio y normalmente está asociada a la política de reservas preventivas que la institución debe tener contra riesgos crediticios. En otras palabras, es un requisito regulador para las instituciones de crédito calcular un monto estimado del dinero que se perderá por motivo de incumplimiento de los clientes y, en consecuencia, reservar dinero con el objetivo de evitar la pérdida de liquidez y solvencia por parte del banco.

  Las pérdidas esperadas entre portafolios son distintas. Cuando una persona pierde su trabajo o se ve en situaciones complicadas de dinero, es muy probable que deje de pagar su tarjeta de crédito o su carro, pero es muy poco probable que deje de pagar su hipoteca, por lo cual las pérdidas esperadas son distintas entre distintos tipos de créditos.

- Así como existe la pérdida esperada también se encuentra la *pérdida no esperada*, la cual es un monto que refleja una cantidad extrema de las posibles pérdidas.

De manera similar con las reservas existen modelos de recuperación, en donde se estima una cifra que puede retornar en caso de que un crédito no sea pagado. Esto se basa en montos más allá del préstamo, ya que también se tienen que considerar las cantidades gastadas en servicios legales y el tiempo que lleva el proceso.

Si se consideran todos los puntos anteriores se vuelve prácticamente imposible para una institución poder hacer un análisis similar para todos y cada uno de sus clientes, por lo que se toman medidas para poder automatizar cada uno de los procesos correspondientes mientras se apegan a las buenas prácticas del manejo del riesgo de crédito. Un mal manejo de esto podría, no sólo traducirse en la quiebra de una institución financiera, sino también en desagradables consecuencias para sus clientes y, dependiendo del tamaño de la institución, consecuencias para la economía de una determinada región, por lo cual existen varias regulaciones, nacionales, internacionales e internas para poder evitar este tipo de situaciones. Un ejemplo podrían ser los requisitos internacionales IFRS9 (*International Financial Reporting Standard*), para manejar las reservas de las carteras tomando en cuenta su deterioro o su incremento de probabilidad de incumplimiento debido a su comportamiento.

1.2.7. Modelos de puntuación

Los *modelos de scoring*, *credit scoring* o *modelos de puntuación* por su traducción del inglés, son modelos que tienen como objetivo la clasificación de buenos y malos créditos. Existen tanto de originación como de seguimiento y recuperación, los mismos permiten decidir quienes obtendrán el préstamo, hasta qué cantidad pueden disponer y estrategias para monitorizarlos y asegurar la máxima rentabilidad para la empresa

prestadora, esto definiendo los límites de crédito que los clientes tienen. Cabe destacar que este tipo de instituciones no sólo tienen pérdidas cuando su dinero no es completamente pagado, sino también cuando su dinero no es colocado, ya que con cantidades grandes de dinero, un pequeño rendimiento en él es capaz de generar una gran cantidad de riqueza.

Estos modelos de puntuación son comúnmente encontrados en bancos al solicitar un préstamo, y tienen como objetivo comprobar que el nivel de riesgo de otorgar dinero a un cliente es bajo y es negocio. Para esto se toman en cuenta variables como edad, estado civil, género, ingresos, entre otras. Para saber cuáles son variables relevantes a medir se utilizan principalmente modelos estadísticos, los cuales presentan evidencia suficiente para saber a qué variables darles importancia y a cuáles darles aún más peso dentro del modelo. También, los modelos de originación son constantemente evaluados, midiendo su efectividad para la discriminación de buenos y malos consumidores, evitando así pérdidas financieras provocadas por fallas en los modelos.

Los modelos de puntuación llevan detrás bastante matemática y probabilidad, sin embargo comúnmente son simplificados a encuestas o formas fácilmente aplicables y entendibles para el público general. Con estos modelos se empieza el ciclo de vida de un crédito, mejor descrito en la figura 1.2.

Figura 1.2: Ciclo de vida de un crédito

1.3. Inteligencia de negocios

A lo largo de la historia, las empresas siempre han estado en constantes esfuerzos para mantenerse adaptadas a las nuevas tecnologías y poder aprovecharlas. Estas innovaciones han ayudado a las compañías a su crecimiento, por lo que las instituciones están conscientes de todo el potencial que las novedades tecnológicas pueden representar. En los últimos años, la capacidad de almacenar datos y disponer de ellos de forma sencilla ha ido evolucionando, permitiendo así estrategias de almacenamiento y acceso cada vez más sofisticados. De esto, incluso se han derivado numerosas carreras y áreas de crecimiento profesional, ya que escenarios alentadores son considerados cuando se tratan de posibles maneras de aumentar ganancias y disminuir riesgos.

De acuerdo con (Vercellis, 2011) la *Inteligencia de Negocios* también conocida en inglés como *Business Intelligence* o *BI* puede ser definida como un grupo de modelos matemáticos y análisis metodológicos que explo-

tan los datos disponibles para generar información y conocimiento útil para procesos complejos de toma de decisiones. Dicho con otras palabras, la inteligencia de negocios tiene el propósito de tomar decisiones que beneficien a los objetivos de la empresa, para eso, se basan en su información recopilada históricamente sobre su desempeño, ingresos y distintas variables que puedan ser de utilidad. Estas decisiones se toman con ayuda de modelos matemáticos y análisis metodológicos los cuales pueden detectar patrones que la mayoría de las veces el ojo humano no puede identificar.

No es un secreto que la toma de decisiones es crucial para cualquier institución, y la habilidad para tomarlas es imprescindible. Estas tienen que tomarse a tiempo y de forma efectiva, un soporte bastante útil es el apoyo de la información obtenida de los datos, esta información tiene que ser convertida en conocimiento por los analistas para poder obtener un resultado.

Hay numerosas formas de trabajar con los datos en donde las matemáticas cumplen un rol importante, con la creación de modelos o algoritmos útiles y eficientes. Las instituciones grandes, trabajan con volúmenes enormes de datos, con numerosas variables e interacciones entre los datos, por lo que no sólo es importante tener un modelo acertado, sino también una forma de implementación con un costo computacional bajo. Con un bajo coste computacional se refiere a que los procesos se corran de una manera rápida y efectiva, de nada sirve tener una decisión acertada pero tarde ni una resolución a tiempo pero errónea. Se verán algunos ejemplos en el Capítulo siguiente.

El sistema de toma de decisiones consiste en varias etapas.

- La primera se concentra en la obtención y preparación de los datos. Se comienza identificando fuentes de información confiables y de calidad, de esto depende prácticamente todo el proceso ya que, aunque todo se haga correctamente, la sentencia final será deficiente si se basa en información falsa o no confiable.
- Una vez que se cuenta con un buen material, lo siguiente es su tratamiento. La transformación de los datos es requerida por algunos modelos y es de gran utilidad para el siguiente paso.
- Después, la exploración de los datos que sirve para dar una idea de cómo está compuesta la información. Esto se explica a detalle en el capítulo 2.
- Consecuentemente, se procede a aplicar los modelos y algoritmos correspondientes para la obtención de información, esto es una elección importante. De la misma forma, estos algoritmos implementados tienen que ser optimizados para su mejor funcionamiento.
- Una vez que se han completado los pasos anteriores, se puede decir que los datos ya pueden ser convertidos a conocimiento y ser la base para la toma de decisiones.

El proceso se describe en la figura 1.3.

Figura 1.3: Proceso de la toma de decisiones

Los profesionales que se encargan de trabajar en áreas de inteligencia de negocios comúnmente están transformando la información en reportes, gráficos o esquemas fáciles de interpretar para que los trabajadores de alta dirección o los encargados de las tomas de decisiones elijan de forma más acertada. Estos reportes por lo regular son indicadores claros y objetivos de lo que está pasando en los negocios, al observar los datos crudos una persona no puede ser capaz de identificar si las ventas mejoraron o empeoraron, sin embargo, con la creación de estos reportes y resúmenes esta información está representada de forma clara y fácil de entender. Esto se logra mediante *Key Performance Indicators* (KPI's), los cuales traducidos al español significan indicadores claves de desempeño.

1.3.1. Mercadotecnia

La mercadotecnia o habitualmente conocido como *marketing* es, según la Real Academia Española (RAE) (Real Academia Española , 2001), un conjunto de principios y prácticas que buscan aumentar las ventas. Está claro que, la consecuencia de un incremento en las ventas conlleva a mayores entradas de dinero y mayores ganancias, por lo que una estrategia inteligente y efectiva es sumamente deseable para cualquier empresa o institución. La mercadotecnia ha evolucionado con el paso del tiempo y ahora, gracias a las efectivas formas de trabajar con los datos que se mencionaron anteriormente, las estrategias son respaldadas por información y modelos matemáticos.

Una campaña de marketing puede ser altamente efectiva si está correctamente planeada, es ilógico pensar que una campaña de venta de pieles exóticas pueda tener gran éxito si se enfoca en gente amante y protectora de los animales. Dirigir las estrategias a las personas correctas, junto con otros factores, determina el rendimiento de las ventas. Para las instituciones financieras no es la excepción, ya que algunos clientes pueden estar interesados en préstamos grandes para financiar un negocio, mientras que otros prefieren préstamos de menor volumen para comprar bienes como motos o televisiones.

1.3.2. Segmentación de clientes

Como se expuso anteriormente, identificar a los clientes más prometedores para un negocio es una habilidad clave para una compañía. Etiquetar a los clientes de una manera que permita a la empresa tener claridad sobre las características, preferencias, lealtad y rentabilidad de cada cliente es una de las principales estrategias en grandes negocios y es llamada *Customer Relationship Management* en inglés o *gestión de relaciones con el cliente* por su traducción al español.

La segmentación de clientes puede tener distintos enfoques dependiendo de los objetivos de la estrategia o lo que las instituciones estén buscando. Los clientes se pueden segmentar dependiendo de la rentabilidad que han traído a la compañía, de las líneas de negocio en las que han consumido o de sus características, por ejemplo: su ingreso anual, gastos anuales, si son solteros o casados, tipo de escolaridad, etcétera. Los bancos obtienen esta información de sus clientes cuando hacen un contrato con ellos, el cliente accede a dar su información ya que el banco la solicita para calcular el riesgo que puede representar dar un préstamo a esta persona, sin embargo, esta información se puede usar para fines estratégicos.

Esta tarea también puede tener dos planteamientos principales, uno sería etiquetar a los clientes con base a sus distintas características y otro sería la predicción o el pronóstico del comportamiento de los consumidores. Independientemente de cual sea la postura tomada, este procedimiento se hace con el fin de tomar un decisión basada en la información recolectada y procesada para convertirla en conocimiento. Algunos de los algoritmos utilizados para esta tarea serán explicadas en el siguiente Capítulo.

Para finalizar, es importante destacar que el uso de las herramientas computacionales han hecho posible que las tareas expuestas a lo largo de este capítulo sean mucho más fáciles de realizar de forma óptima y eficaz. El proceso de toma de decisiones a través de modelos matemáticos y probabilísticos llevados a cabo a través de lenguajes de programación es una práctica que día con día va aumentando su demanda y uso en distintas empresas. De la misma forma, la utilización de las tecnologías para realizar análisis y modelos para los distintos tipos de riesgo se ha popularizado por lo que el aprendizaje y dominio de estas herramientas se ha convertido en algo primordial para los distintos especialistas en las áreas de riesgos e inteligencia de negocios.

Capítulo 2

Aprendizaje automático

Hoy en día el mundo está lleno de conocimiento e información valiosa escondida en inmensos volúmenes de datos. Gracias a la revolución digital, actualmente, se pueden realizar de forma más sencilla aplicaciones para darle sentido a la información, ya que se cuenta con el apoyo y soporte de la tecnología. El *Machine learning* traducido al español como *aprendizaje automático* (o *aprendizaje de máquina*), es un subcampo de la inteligencia artificial el cual se concentra en el desarrollo e implementación de algoritmos que aprenden a partir de un conjunto de datos (Raschka, 2015; Torres, 2018). Conlleva conocimientos generales de programación, estadística, matemáticas y cualquier otra rama que tenga relación con los datos a tratar, que pueden ir desde negocios hasta medicina. Estos algoritmos están diseñados para llegar a conclusiones tomando como única referencia el *conjunto de datos* proporcionado.

Existen distintos tipos de aprendizaje automático, como son el *aprendizaje supervisado, no supervisado, profundo* y *por refuerzo*. Como se explica en (Raschka, 2015), el aprendizaje supervisado tiene como objetivo obtener una determinada clasificación o regresión de un conjunto de datos mientras que el no supervisado tiene como propósito identificar patrones ocultos en los datos. Según (Torres, 2021) hablamos de aprendizaje por refuerzo cuando el modelo se implementa en forma de un agente que deberá explorar un espacio desconocido y determinar las accionas a llevar a cabo mediante prueba y error, aprendiendo por si mismo gracias a las recompensas y penalizaciones que se obtienen de sus acciones. Mientras que los algoritmos de aprendizaje profundo se basan en redes neuronales artificiales, cuyas estructuras algorítmicas permiten que modelos compuestos por múltiples capas de procesamiento aprendan representaciones de datos con varios niveles de abstracción.

Este trabajo se enfoca principalmente en el aprendizaje supervisado y no supervisado (explicados a detalle más adelante), los cuales pueden considerarse como los enfoques principales. Los alcances van desde aprender cómo agrupar datos con características similares hasta predecir el comportamiento y/o los resultados que se van a obtener dadas ciertas peculiaridades en la información.

En los últimos años, estas habilidades han sido ampliamente requeridas por empresas ya que ayudan a maximizar ganancias, minimizar riesgos, mejorar estrategias de negocios o productividad, detectar enfermedades, etcétera. Para esto, es necesario un determinado procedimiento, el cual consiste básicamente en obtener los datos, preprocesarlos y luego transformarlos en información. Por lo regular el conjunto de datos se divide en dos subconjuntos: *datos de entrenamiento* y *datos de prueba*. Los datos de entrenamiento son los que se usan para que el algoritmo de aprendizaje obtenga los parámetros del modelo, mientras que los datos de prueba se utilizan para la evaluación del modelo. Algunos de los usos del aprendizaje automático se pueden observar en la figura 2.1.

Figura 2.1: Ejemplos de aplicaciones del aprendizaje automático en la actualidad.

2.1. Datos

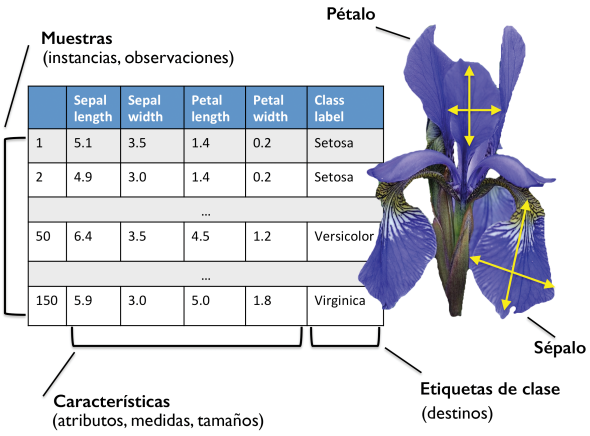
El ingrediente principal para entrenar a los algoritmos de aprendizaje automático son los datos. De ellos se obtiene toda la información cruda que los algoritmos deben procesar para convertirlos en conocimiento (Torres, 2018). Lo anterior puede estar claro, pero es importante resaltar que esta obtención de conocimiento no es una tarea sencilla debido a que no todos los datos son capaces de brindar la información y las respuestas esperadas.

Los datos son la forma en que la evidencia recolectada, ya sea cuantitativa o cualitativa, es transformada en variables numéricas, categóricas y textuales principalmente. Es decir, que los datos recolectados por distintos medios se concentran de tal forma que puedan ser manipulados con distintos objetivos, como su análisis, su procesamiento o la creación de modelos. Los datos numéricos y de texto son tal cual su nombre lo indica y los datos categóricos son números enteros que representan categorías o grupos distintos y pueden o no tener un orden lógico. Por ejemplo, si se quiere representar a la variable género se puede utilizar un dato categórico asignando 0 al género femenino y 1 al masculino. Asimismo, si se requiere de un orden lógico, a la variable que representa la satisfacción de los clientes puede asignarse el 0 de insatisfecho, 1 de neutral y 2 para satisfecho.

En la figura 2.2 publicada en (Raschka, 2015) se puede observar un extracto del conjunto de datos Iris, un ejemplo clásico, en él se encuentran contenidas las medidas de 150 flores iris de tres especies distintas. Cada muestra de flor representa una fila del conjunto de datos y las medidas (largo y ancho de los sépalos y pétalos, en centímetros) se almacenan en columnas. Actualmente, podemos encontrar numerosos datos reales con información verídica relacionada realmente con situaciones de interés. Por ejemplo, en las páginas oficiales del banco de México y del INEGI, o bien, en distintas páginas con licencia pública como Kaggle [1] y Yahoo finanzas [2]. De la misma forma, se puede encontrar información variada y de diversos campos de conocimiento en otros repositorios digitales.

Para decir que un conjunto de datos es de calidad tiene que ser útil para los análisis y contar con valores acertados, es decir, datos realistas. Por ejemplo, en una variable que representa la edad de las personas es completamente ilógico que alguien tenga un número mayor a 200. Así que, en caso de contar con varias fuentes en donde esté disponible la información que se va a utilizar, esta tiene que ser consistente en todas

[1] https://www.kaggle.com
[2] https://finance.yahoo.com



	Sepal length	Sepal width	Petal length	Petal width	Class label
1	5.1	3.5	1.4	0.2	Setosa
2	4.9	3.0	1.4	0.2	Setosa
...					
50	6.4	3.5	4.5	1.2	Versicolor
...					
150	5.9	3.0	5.0	1.8	Virginica

Figura 2.2: Extracto del conjunto de datos Iris.

ellas, mostrando los mismos valores y siendo representadas con las mismas medidas. De igual forma, los datos tienen que estar actualizados, de lo contrario se contaría con conocimiento atrasado. Además, la información no puede ser redundante y tiene que ser de relevancia, considerar la estatura de una persona como determinante para dar un préstamo bancario no tiene mucho sentido. Por otra parte, los valores faltantes son un fenómeno común al recabar información, puesto que no siempre se puede recopilar toda la información, sin embargo, tener muchos valores faltantes puede hacer a la información incompleta y no apta para el análisis.

Los conjuntos de datos de calidad deben pasar por un proceso exhaustivo para poder obtener mejores resultados. Es decir, una vez que se ha mostrado que la información es apta, se tiene que realizar un análisis de los datos de forma estadística y gráfica para deducir el algoritmo óptimo para la obtención de conocimiento a partir de ella. También, de ser necesario se tienen que transformar algunos datos, lo que quiere decir que se deben llevar a otra escala siendo representados por otros valores sin perder la información original, esto se explica a lo largo de este Capítulo.

2.2. Análisis exploratorio de datos

La acción de aplicar un algoritmo de aprendizaje automático a un conjunto de datos recolectado no garantiza la obtención de conocimiento, de hecho, esto puede tener consecuencias desastrosas con resultados aparentemente buenos pero que realmente no lo son. Toda problemática tiene un contexto y un conjunto de consideraciones que tienen que estar siempre presentes para poder extraer todo el saber que estos datos contienen.

Con la existencia abundante y variada de técnicas, no es la mejor idea implementar todas y cada una de ellas buscando cuál es el método que da los mejores resultados. Lo correcto sería identificar cuál es el modelo correcto que mejor se adecúa a las necesidades y contexto del problema a resolver. Para abordar este desafío, se necesita conocer lo mejor posible la información con la que se cuenta o en otras palabras, analizar y explorar los datos. Sin embargo, no existe una fórmula universal para la realización del análisis exploratorio de datos, ya que como se ha expuesto, todo depende de la naturaleza de la información recolectada.

Mediante el análisis exploratorio se busca resaltar las características de la colección de datos y obtener información sobre ella, esto utilizando gráficas y calculando valores estadísticos para identificar relaciones entre los atributos. Esta fase puede dar ideas de las soluciones a las que se puede llegar, también puede ser útil para entender de una forma diferente los resultados que la aplicación del aprendizaje automático arroje. El proceso de análisis incluye tres técnicas principales las cuáles son el análisis univariado, bivariado y multivariado. El primero se adentra en las propiedades de cada variable buscando sus características clave, en el bivariado se busca medir la relación existente entre cada par de variables y por último, en el

análisis multivariado se buscan las relaciones de un grupo de atributos. Los análisis que involucran más de una variable son más complejos debido a las posibles combinaciones resultantes. Por ejemplo, en un análisis entre dos variables, existen las posibilidades de que una de las variables sea numérica y la otra sea categórica, que ambas sean categóricas o que ambas sean numéricas, cuando se hace lo mismo entre múltiples variables se analiza cada par de variable por separado y se representan juntas en un esquema o gráfico que contiene una matriz de resultados.

Como se mencionó anteriormente, el análisis exploratorio de datos es una combinación de distintos tipos de gráficas y distintos valores estadísticos que representan la información de cada atributo. Claramente, no todas las métricas pueden brindar información útil, por lo que dependiendo del proyecto sólo se usan las que son interesantes y tienen capacidad de aportar a la resolución de los objetivos. Por un lado, la tabla 2.1 describe los estadísticos principales del análisis exploratorio de datos.

Media	$\overline{\mu} = \frac{x_1 + x_2 + ... + x_m}{m} = \frac{1}{m}\sum_{i=1}^{m} x_i$
Mediana	$x^{med} = x_{(m+1)/2}$ si es impar y $x^{med} = \frac{x_{m/2} + x_{(m+2)/2}}{2}$ si es par.
Moda	Corresponde al pico más alto de la curva de densidad empírica para un atributo. En otras palabras es el valor más repetido.
Rango	$x^{rango} = x^{max} - x^{min}$
Varianza	$\overline{\sigma}^2 = \frac{1}{m-1}\sum_{i=1}^{m}(x_i - \overline{\mu})^2$
Desviación estándar	$\sqrt{\overline{\sigma}^2}$
Covarianza	$cov(a_j, a_k) = \frac{1}{m-2}\sum_{i=1}^{m}(x_{ij} - \overline{\mu}_j)(x_{ik} - \overline{\mu}_k)$
Correlación	$corr(a_j, a_k) = \frac{cov(a_j, a_k)}{\overline{\sigma}_j \overline{\sigma}_k}$

Tabla 2.1: Estadísticos principales del análisis exploratorio de datos.

Dónde:

m es el número total de observaciones en el atributo;
x_i representa la i-ésima observación;
x^{max} es el valor observado más grande (sólo aplica en variables numéricas);
x^{min} es el valor observado más pequeño (sólo aplica en variables numéricas);
a_j, a_k son un par de variables a comparar.

Por otro lado y para comenzar con los gráficos más importantes para el análisis exploratorio se expone una gráfica bastante popular, sencilla y útil para representar las proporciones y distribución de los datos, la *gráfica de pastel*. Esta gráfica se divide en sectores cuya área es proporcional a los porcentajes de las distintas variables representadas. Por ejemplo, en la figura 2.3 podemos observar la distribución de los

activos financieros en México entre los bancos más importantes del país teniendo una forma visual para comparar la proporción del mercado con la que cada uno de estos bancos cuenta.

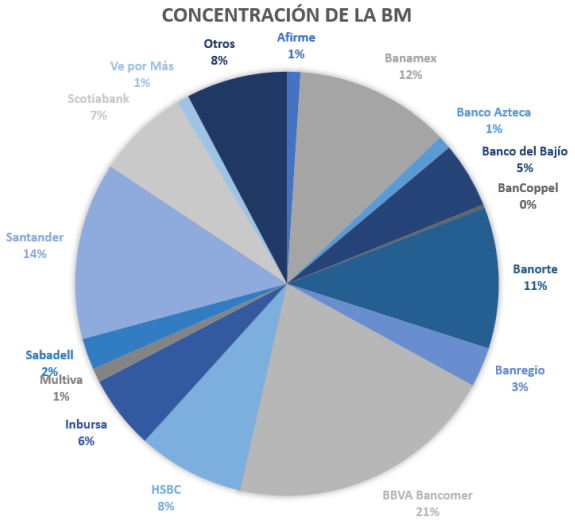


Figura 2.3: Ejemplo de gráfica de pastel.

El *histograma* es otro tipo de gráfica que representa la distribución de los datos, en donde el eje vertical representa la frecuencia de aparición de los valores en el eje horizontal, por lo que es una forma visual de observar el número de ocasiones que un número o un intervalo de números aparecen en un conjunto de datos comparado con otros números o intervalos. Por ejemplo, en la figura 2.4 podemos ver un histograma que representa el número de personas que se tienen en una muestra que cuentan con determinada edad. Al ver los datos de forma global seguramente no podremos concluir nada, sin embargo, con este gráfico podemos decir de forma rápida y segura que en nuestro universo abundan las personas entre los 25 y 30 años mientras que el grupo más pequeño son los que están llegando a los 70 años.

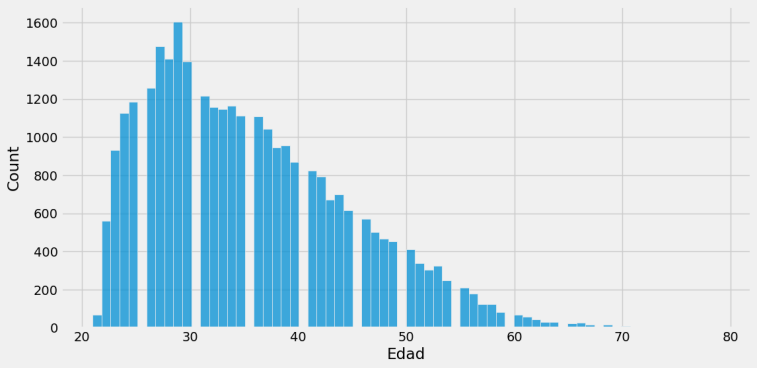


Figura 2.4: Ejemplo de histograma.

Igualmente, el *gráfico de caja* es una forma de representar una serie de datos a través de sus cuartiles [3]. La parte inferior de la caja (o la parte izquierda dependiendo de la orientación de la gráfica) señala el primer cuartil (o el punto en donde se acumulan el 25 % de los datos recolectados), la linea de en medio de la caja indica el segundo cuartil o mediana (donde se acumulan el 50 %), y la parte superior o derecha de la caja indica el tercer cuartil. La altura de la caja es conocida como rango intercuartílico. Esta representación

[3] Los cuartiles son los valores que dividen una muestra en 4 partes iguales.

también es de gran utilidad para detectar valores atípicos, los cuales son todos los valores que se encuentren a 1.5 rangos intercuartílicos de distancia desde el primer y tercer cuartil hacia los extremos.

En la figura 2.5 podemos observar la variable gasto la cual no cuenta con valores atípicos, ya que no se muestra ninguna observación fuera de la caja que estamos viendo. Podemos notar que la mitad de los datos es separado alrededor del valor 50 en gasto, el primer cuartil está antes de 40 y el tercero antes de 80.

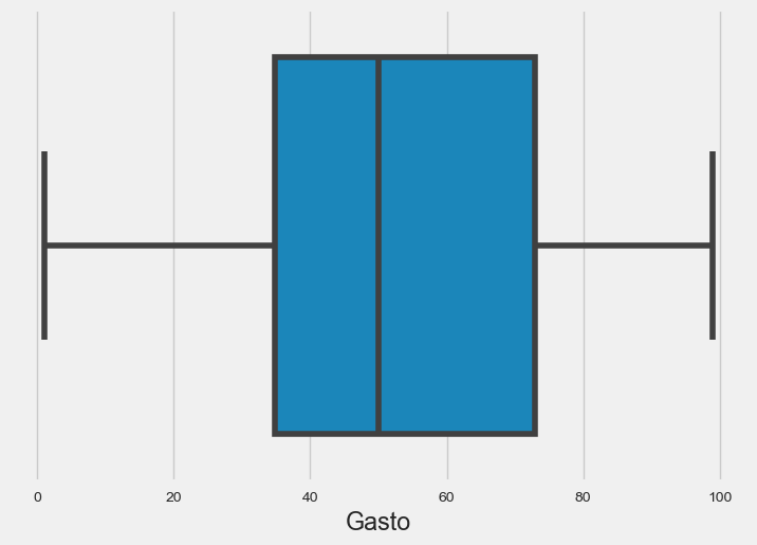


Figura 2.5: Ejemplo de gráfico de caja.

También, la *gráfica de gusanos* es muy útil para visualizar la forma en que los datos están distribuidos y el contraste que tienen respecto a otras variables, por lo que es beneficioso para fines comparativos. En la figura 2.6 se puede ver como los datos correspondientes a los gusanos azul y amarillo están distribuidos entre valores más altos, mientras que los gusanos rojo y morado están conformados por valores pequeños.

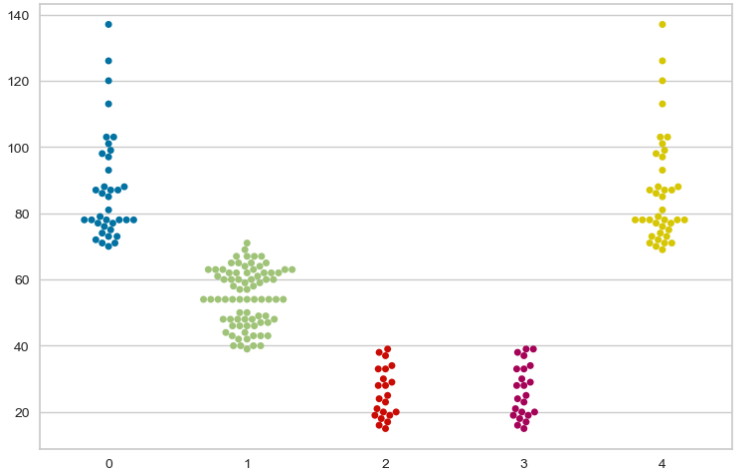


Figura 2.6: Ejemplo de gráfica de gusanos.

Por otra parte, un *gráfico de dispersión* acomoda los datos para facilitar la comparación entre dos y tres variables principalmente. El gráfico de dispersión más común es el de dos variables en donde una de ellas es representada por el eje x, la otra por el eje y y se disponen los valores según sus coordenadas cartesianas; es muy útil para encontrar tendencias o correlaciones entre los atributos. En la figura 2.7 se puede observar

la comparativa entre la variable ingresos y la variable edad. A pesar de que no se ve una tendencia claramente marcada, la gráfica es de utilidad puesto que descarta la correlación entre estas variables porque las observaciones se ven bastante dispersas. Las conclusiones que se podrían tomar de este gráfico es que las personas más jóvenes y las más grandes no tienen ingresos altos ya que estos ingresos se concentran entre las personas de entre 30 y 50 años.

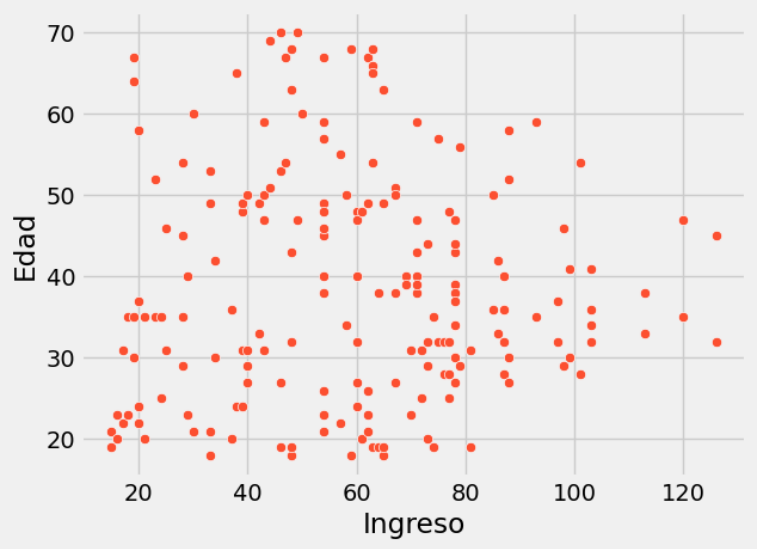


Figura 2.7: Ejemplo de gráfica de dispersión.

Por último, una *serie de tiempo* es una representación gráfica de una variable a lo largo del tiempo. Sirve para poder comparar las tendencias que se tienen y para tratar de pronosticar datos futuros. Las series de tiempo tienen un estudio más amplio y completo ya que están sujetas a muchas transformaciones para su mejor comprensión y entendimiento. En la figura 2.8 de (Instituto Nacional de Estadística y Geografía (INEGI), 2022) se puede observar el comportamiento del Índice Nacional de Precios al Consumidor, en el cual a corto plazo de los últimos años (2020 en adelante) podemos ver una tendencia claramente creciente, sin embargo, si nos enfocamos en el periodo de tiempo del 2018 al 2020 observamos una tendencia a la baja. Esta gráfica también muestra que los valores del IPC oscilan entre el 2 % y el 7 % por lo que también se identifican los valores extremos a lo largo de los años. Otras conclusiones que se pueden tomar es que la tendencia a la alza de los últimos años es demasiado agresiva y alcanzó un nuevo valor máximo, por lo que la fecha debería ser foco de atención y de especial estudio. El caso que se ve es que el IPC subió principalmente por la pandemia COVID-19.

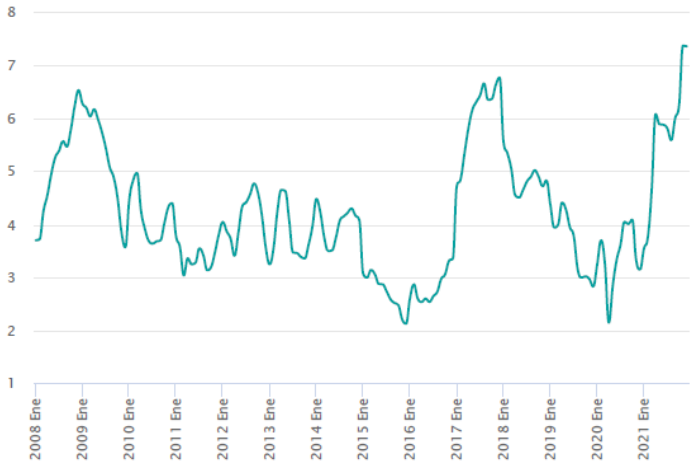


Figura 2.8: Ejemplo de serie de tiempo: Índice Nacional de Precios al Consumidor.

2.3. Preprocesamiento de datos

Lo datos son información en bruto capaces de proveer una amplia cantidad de conocimiento a sus portadores aunque todo depende de la tenacidad de los especialistas para aprender de ellos. Por lo regular, los datos se presentan de una forma cruda, tal cual fueron recopilados, lo que a veces complica el trabajo de los analistas. El preprocesamiento de los datos tal vez no sea una parte tan sonada o popular cuando alguien habla del aprendizaje automático, pero es una etapa fundamental por la que todo conjunto de datos tiene que pasar y que todo especialista debe tomar en cuenta.

De acuerdo con lo anterior, es necesario resaltar la importancia que tiene el asegurarse de la calidad de los datos que serán usados como base para la realización de cualquier modelo, así como, examinar si son útiles y confiables. El preparar los datos de la manera más adecuada para su análisis es un tema bastante extenso, existen técnicas variadas y sustentadas que mejoran considerablemente la calidad de los datos. Este trabajo, se enfoca principalmente en tres técnicas, las cuáles podrían considerarse como las más frecuentes y algunas bastante simples, sin embargo, el tratar los *valores duplicados*, los *outliers*[4] y los *valores faltantes* puede mejorar en gran medida los datos y reflejarse significativamente en los resultados.

Al contrario de creencias populares, las personas que se encargan de trabajar con datos ya sea para su análisis o para minería de información, dedican la mayor cantidad de su tiempo en este paso. En los últimos años, existen puestos que se dedican exclusivamente al preprocesamiento de los datos para el posterior análisis por parte de otras áreas.

Indice de Cobertura (B)				
	Banco 1	Banco 2	Banco 3	Banco 4
Cartera total	200.974544	201.946929	213.102568	216.411742
Cartera total	200.974544	201.946929	213.102568	216.411742
Cartera total de créditos comerciales	211.402936	219.34811	227.067873	233.710117
Cartera de empresas	188.631933	198.420656	205.229844	212.130982
Cartera de entidades financieras	4156.14642	4263.00498	4486.71109	4597.94261
Cartera de entidades gubernamentales	n. a.	n. a.	n. a.	n. a.
Cartera de tarjeta de crédito	182.13607	186.104835	194.264094	987.749303

Figura 2.9: Conjunto de datos con problemas de valores faltantes, atípicos y duplicados.

En la figura 2.9 se puede observar un pequeño conjunto de datos con problemas de datos atípicos, faltantes

[4]Valores atípicos en inglés.

y duplicados. En este caso, el valor duplicado sería la variable cartera total, la cual aparece dos veces por lo que que una fila no es necesaria y dejar ambas filas podría traer un impacto negativo al modelo. Por otro lado, en la variable Cartera de entidades gubernamentales se pueden observar los recuadros con N.A.[5], por lo que falta esa información que puede ser valiosa para el modelo. Finalmente, para identificar los datos atípicos se tiene que tener una noción y conocimiento de lo que se está estudiando. El índice de cobertura significa que tan suficiente fue lo que reservó el banco para determinada cartera. Por ejemplo, si el índice de cobertura es 200 % significa que la institución reservó el doble de la cantidad de dinero que perdió por lo que los valores de 4000 % de índice de cobertura que se tiene para la cartera de entidades financieras en la Figura 2.9 claramente está mal.

2.3.1. Valores duplicados

Los casos de empresas de nueva creación que requieren hacer un análisis exploratorio a sus datos, los proyectos de presupuesto limitado para la recolección de información y los proyectos en donde la información es limitada o difícil de conseguir son ejemplos de conjuntos de datos con poca información, es decir, conjuntos de datos pequeños. Por lo tanto, contar con valores duplicados es más peligroso en situaciones como las antes mencionadas, ya que pueden impactar en los resultados de una forma más grave, principalmente cuando el cálculo de probabilidad de ocurrencia de eventos está presente.

Por otra parte, los conjuntos de datos grandes son susceptibles a tener numerosos casos de datos duplicados así que eliminarlos llevaría a tener un conjunto de mayor calidad y más fácil de manipular. Por lo que, corregir el problema de datos duplicados puede resultar en reducción de costos de almacenaje, lo que hace más efectivo y veloz el trabajar con la información, tanto para el análisis como para la implementación de algoritmos.

De la misma forma, si asumimos que los valores duplicados no afectan los modelos y arrojan buenos resultados, todavía existirán problemas por utilizar estos valores dobles. como ejemplo, suponga una situación en donde se planea vender un producto nuevo a clientes ya existentes y se tiene la estrategia de enviar folletos junto con regalos personalizados a los compradores potenciales. Considere que como resultado de un modelo, se obtiene una lista de direcciones para mandar la información y los regalos previamente acordados. Está claro que el tener varios valores duplicados generaría un gasto completamente innecesario para la empresa, lo cual representaría pérdidas. En la actualidad es una tarea sencilla encargarse de este problema mediante instrucciones que identifican rápidamente estos eventos y se encargan de eliminarlos.

2.3.2. Valores atípicos

La forma en que los datos se recaban puede influir considerablemente en los resultados de cualquier modelo de aprendizaje de máquina, ya que a pesar de todos los esfuerzos y filtros de calidad que se puedan llevar a cabo para la recolección de datos, siempre existirán valores que no son confiables en su totalidad. Las entrevistas, encuestas, observaciones, entre otros, son métodos para recopilar datos y todos están sujetos a errores. Por ejemplo, en las entrevistas y encuestas, la condición de los datos depende de la honestidad y calidad de respuesta de las personas entrevistadas, incluso de otros factores que pueden parecer poco relevantes como el clima o la hora del día. De la misma manera, las observaciones tienen que estar sujetas a las mismas condiciones para ser efectivas en el modelo. Por esta razón, existe el tratamiento contra los valores atípicos. De esta forma, si en una encuesta alguien mintió sobre cuantos hermanos tiene diciendo una cantidad grande como 20, se elimina de los datos ya que probablemente es información errónea. Si no fuera un dato falso, igualmente es algo completamente fuera de lo común, por lo que es conveniente no tomarlo en cuenta.

El proceso para tratar valores atípicos comienza por identificar estas indeseables anomalías. La forma más simple de hacerlo está basado en un concepto estadístico llamado *dispersión*. Existen varias medidas de dispersión pero nos enfocaremos en dos, cuartiles y rango intercuartil. Una estrategia es que todos los valores que se encuentren fuera del rango expresado en la ecuación 2.1 sean considerados valores atípicos (Vercellis, 2011).

[5]NA significa No Aplica o por su traducción del ingles Not Available, no disponible.

$$(C1 - 1{,}5 * RI, C3 + 1{,}5 * RI) \tag{2.1}$$

Siendo $C1$ y $C3$ el primer y tercer cuartil, respectivamente, y RI el rango intercuartil.

2.3.3. Valores faltantes

Por lo anteriormente mencionado, se sabe que los datos comúnmente están en un estado crudo, lo que provoca que hayan distintos tipos de inconvenientes. Un impedimento bastante frecuente es el de los valores faltantes, llamados así por la omisión de información en las variables, lo cual provoca una pérdida de información que puede llegar a ser relevante. La acción de ignorar el problema o trabajar con un conjunto de datos con este problema no suele ser una buena idea, por lo que existen distintas técnicas para tratar esta complicación.

Entre las múltiples técnicas que existen para el tratamiento de los valores faltantes la más fácil es eliminar las observaciones (filas) o las características (columnas) que tengan algún valor ausente. Esto claramente puede tener desventajas, principalmente para los conjuntos de datos de menor tamaño ya que se pierde información. Además, se pueden perder características importantes para el modelo u observaciones que dan solidez y confianza a los resultados obtenidos. Es más común eliminar las observaciones que las características, de hecho, es mejor eliminar las características sólo en caso de que el atributo no sea relevante.

Una técnica más efectiva es rellenar los datos faltantes con información coherente obtenida a través de los datos. Esto permite poder conservar la mayor cantidad de información recopilada, tanto de características como de observaciones, lo que facilita la aplicación de los modelos. La pregunta de interés sería, ¿cómo saber qué información puede ser adecuada para rellenar los espacios en blanco? Esto depende mucho del tipo de datos con los que se está trabajando. Por ejemplo, si se tiene una variable numérica continua lo más recomendable sería atribuirle el valor de la media o el promedio de todos los resultados de la variable mencionada. Por el contrario, si contamos con una variable numérica discreta que sólo toma valores enteros, la mejor opción consistiría en optar por la moda.

Existen métodos más complejos como sustituir los valores aplicando regresiones o interpolaciones, estas son técnicas más avanzadas que pueden arrojar datos más precisos, sin embargo, pueden no ser las más eficientes; todo depende de las características del conjunto de datos con el que se esté trabajando, estas técnicas son más utilizadas cuando los valores faltantes son realmente importantes.

2.3.4. Estandarización

La *estandarización* (también conocida como *normalización*) es un conjunto de técnicas de transformación de los datos que ayudan a los modelos de aprendizaje automático a obtener mejores resultados. Las técnicas más populares son los métodos *Min-max* y *Z-index* (Vercellis, 2011). La estandarización sirve para facilitar el análisis entre las variables de un conjunto de datos y se encarga de eliminar diferencias como la escala; trata de evitar que los modelos le den más peso e importancia a los atributos que contengan los valores más altos.

El método Min-max consiste en fijar los valores entre -1 y 1 o entre 0 y 1, es decir, el valor máximo de un atributo se transforma en 1 y el valor mínimo se transforma ya sea en -1 o en 0. Los valores restantes se transformarían de manera que se encuentren dentro del intervalo elegido, esto se logra mediante la siguiente ecuación:

$$x'_{i,j} = \frac{x_{i,j} - x_{min,j}}{x_{max,j} - x_{min,j}}(x'_{max,j} - x'_{min,j}) + x'_{min,j} \tag{2.2}$$

donde:

$x_{min,j} = \min_i x_{i,j}$ es decir, el valor más pequeño correspondiente a la determinada columna j,

$x_{max,j} = \max_i x_{i,j}$ es decir, el valor más grande correspondiente a la determinada columna j,

$x'_{min,j} = -1$ y $x'_{max,j} = 1$, o bien, $x'_{min,j} = 0$ y $x'_{max,j} = 1$

siendo $x'_{i,j}$ los valores ya transformados y $x_{i,j}$ los valores originales.

Por su parte, el método Z-index utiliza la transformación:

$$x'_{i,j} = \frac{x_{i,j} - \overline{\mu}_j}{\overline{\sigma}_j} \tag{2.3}$$

donde $\overline{\mu}_j$ y $\overline{\sigma}_j$ son la media y la desviación estándar muestrales de la columna j-ésima, respectivamente. Si la distribución de los datos es normal, este método produce valores casi seguros en el rango (-3,3).

2.3.5. Reducción de datos

La reducción de los datos es una técnica utilizada para trabajar con grandes volúmenes de datos conservando la eficiencia de los algoritmos de aprendizaje automático y la calidad de la información. Existen ciertos indicios que indican cuando es conveniente reducir el conjunto de datos, uno de ellos es el tiempo que una computadora tarda en trabajar con la información, este tiene que mantenerse lo más corto posible así que, por la complejidad de algunos algoritmos, tener menos datos puede ser una diferencia crucial que ayude a lograr este objetivo. Más aún, tener una correcta selección de las características que se usarán para los modelos permite tener resultados más acertados. Por lo que, la simplicidad es otro factor importante, ya que permite a los analistas entender los criterios implementados, incluso, algunos expertos están dispuestos a intercambiar más simplicidad por resultados un poco menos precisos.

Las ventajas que la reducción de datos puede traer en la implementación de modelos son importantes, sin embargo, estas técnicas de optimización son útiles aunque no se vayan a implementar algoritmos de aprendizaje. Guardar la mayor cantidad de información posible en el menor espacio posible es un objetivo primordial para las empresas que trabajan con volúmenes considerables de datos. Existen servicios que se dedican a almacenar esta información en la nube, pero cobran por el espacio utilizado, por lo que hacer el conjunto de datos menos grande ayuda a prevenir el gasto innecesario de recursos.

Análisis de componentes principales

El *Principal Component Analysis* (PCA) o *Análisis de componentes principales* es una técnica de reducción de atributos por medio de proyecciones. El objetivo de este método es remplazar el número original de atributos por un número más pequeño, esto, mediante combinaciones lineales; para entender este proceso se tienen que tener conocimientos previos de álgebra lineal. La experiencia muestra cómo esta transformación puede guiar a resultados más precisos pero antes de utilizar esta técnica es conveniente hacer una estandarización de los datos.

Como podemos encontrar en (Hull, 2012), el procedimiento PCA se ocupa de expresar la estructura de los datos sobre n variables correlacionadas con un número menor de variables no correlacionadas. Primero se tiene que calcular una matriz a partir de los datos, esta matriz es conocida como matriz de varianzas y covarianzas. El siguiente paso es calcular los valores propios y vectores propios para esta matriz, los vectores propios se eligen para que tengan una longitud 1; el vector propio correspondiente al valor propio más alto es el primer componente principal, el vector propio correspondiente al segundo valor más alto es el segundo componente principal, etcétera. De este modo, el valor propio para el i-ésimo componente principal como porcentaje de la suma de todos los valores propios es el porcentaje de la varianza general explicada por el i-ésimo componente principal y la raíz cuadrada del i-ésimo valor propio es la desviación estándar de la puntuación del i-ésimo componente.

Explicado de diferente manera, lo que hace este proceso es obtener nuevas variables mediante la construcción de una combinación lineal con las variables originales, lo cual es una ecuación que multiplica los

valores de cada variable por un escalar; esta ecuación es normalizada por lo que la suma de todos los escalares es igual a 1. El objetivo es encontrar las nuevas variables con mayor varianza y que no estén correlacionadas entre ellas, la de mayor varianza es el primer componente principal, la segunda con mayor varianza es el segundo componente principal y así sucesivamente.

Al tratar con varianzas, esta técnica es sumamente sensible a los valores atípicos por lo que se recomienda su tratamiento previo. Igualmente, las varianzas se miden en las escalas de las variables, por lo que todas las variables tienen que estar en la misma escala para tener resultados coherentes, por lo que la estandarización también tiene que ser implementada de forma previa. Cualquier software computacional generará los mismos resultados de PCA pero pueden variar por el signo, lo cual no afecta los resultados.

Existen otras técnicas para reducir dimensionalidad como *kPCA* (extensión de PCA que utiliza métodos kernel), *Descomposición de valores singulares* (técnica que permite descomponer una matriz en otras matrices o *Análisis de componentes independientes*. Todas estas técnicas están disponibles en la biblioteca de Scikit-learn para Python. De igual forma, existen herramientas como *LASSO* el cuál es un análisis de regresión que selecciona las variables indicadas para mejorar la exactitud del modelo estadístico.

2.4. Aprendizaje no supervisado

El *aprendizaje no supervisado* según (Raschka, 2015) es una rama del aprendizaje de máquina que tiene el propósito de identificar patrones ocultos en los datos, los cuales, en un conjunto de datos de tamaño considerable, no podrían ser fácilmente detectados por el análisis manual de una persona por mayor experiencia y conocimientos que tenga. En esta categoría el único objetivo es el desarrollar un modelo sólido capaz de reconocer patrones en los datos de forma acertada, por lo tanto, la información con la que se trabaja suele no estar etiquetada de ninguna forma.

Existe una clase de modelos en el aprendizaje no supervisado que responde al nombre de *agrupamiento*, estos modelos buscan obtener grupos de observaciones más parecidas entre los miembros del grupo al que pertenecen, que entre observaciones pertenecientes a otros grupos. Aunque las similitudes están basadas en distancias entre observaciones, existe la posibilidad de agrupar datos mediante variables categóricas.

La importancia del aprendizaje no supervisado como se mencionó anteriormente radica en encontrar patrones de información, un claro ejemplo sería contar con un grupo de clientes de una compañía de los cuales se saben algunas características, a partir de un agrupamiento se podría saber qué clientes son más parecidos a otros y de esta forma, ofrecerles los mismos productos a ellos, también se podría recomendar los productos que varias de las personas de un grupo compraron.

Para realizar la tarea de agrupamiento existen diferentes algoritmos y enfoques, cada una con sus respectivas ventajas y dependiendo de la correcta elección del modelo (según las características de la información a la que se procederá a hacer el análisis) pueden reducirse sus desventajas. De acuerdo con (Badia Contelles, s.f.), los principales enfoques son algoritmos jerárquicos, particionales y basados en densidad. Los jerárquicos consisten en minimizar alguna distancia o maximizar alguna medida de similitud; se dividen en aglomerativos y disociativos. Por su parte, los algoritmos particionales tienen un conocimiento previo del número de grupos a los que tienen que llegar y optimizan algún criterio o función objetivo. Por último, los métodos basados en densidad utilizan distintas técnicas para determinar los grupos las cuales pueden ser grafos, histogramas, kernels, etc.

Algunos de los algoritmos de agrupamiento que existen son *K-Means, DBSCAN, Affinity Propagation, Mean Shift, Spectral Clustering, Hierarchical Clustering*, entre otros (véase la figura 2.10). En este trabajo se usaron los algoritmos de K-Means y DBSCAN que son métodos particionales y de densidad, respectivamente. Los demás algoritmos están disponibles en la librería Scikit-learn para Python, en donde se puede encontrar toda su documentación y ejemplos prácticos de como implementarlos correctamente.

Figura 2.10: Algoritmos de aprendizaje no supervisado.

2.4.1. *K*-means

Es sumamente complicado poder entablar una conversación sobre aprendizaje no supervisado o sobre métodos de agrupamiento sin mencionar al algoritmo *K-means (o K-medias)* presentado en (MacQueen, J. B., 1967), el cual sin lugar a dudas es uno de los algoritmos más populares en estos campos. Su renombre se debe principalmente a su implementación sencilla y su bajo coste computacional, dos características cruciales para ser un procedimiento preferido tanto por novatos como por expertos.

Al formar parte de la familia de métodos de agrupamiento los datos no deben contar con ningún tipo de etiquetado previo, ya que este es el objetivo de *K*-means, el poder agrupar las observaciones según sus características en común y poder diferenciarlos con otros grupos con distintas peculiaridades. Por lo general, tiene un funcionamiento bastante preciso y acertado en su implementación en bases de datos de grandes volúmenes y es usado en variadas áreas de interés. Un ejemplo en particular es la segmentación de clientes que las empresas requieren para llevar a cabo de forma más efectiva sus estrategias de negocio.

K-means pertenece a una categoría de agrupamiento conocida como *prototype-based clustering* lo cual en español se traduce como agrupamiento basado en prototipos, esto significa que un prototipo (un punto dentro de la muestra) es el representante de cada grupo semejante, este prototipo es comúnmente el centro del conjunto; a esto se le debe su nombre, ya que existen *K* centros del conjunto generalmente representados por la media de los datos pertenecientes a cada grupo.

Hay una desventaja clara a resaltar cuando de *K*-means se trata, la necesidad de definir un número específico de grupos que el método tiene que identificar desde el principio; esto puede ser fácilmente resuelto al mapear y visualizar los datos para tener una idea de cuál podría ser una propuesta apropiada, sin embargo, con conjuntos de datos que no están conformados por un número pequeño de variables se pueden presentar problemas de visualización, lo que hará prácticamente imposible poder definir idóneamente el número de grupos de forma visual. Esto nos lleva a tomar en cuenta algunas estrategias que van de la mano con *K*-means como son el *Elbow Method (Método del codo)* y *Silhouette Score (coeficiente de silueta)* los cuales

serán explicados más adelante o el *Principal Component Analysis (Análisis de componentes principales, PCA)*, explicado en la sección anterior. Cabe destacar que a pesar de existir numerosas técnicas ninguna ha sido aprobada como mejor que las demás.

El procedimiento para ejecutar *K*-means consiste inicialmente en elegir un número *K* de *centroides* o puntos de la muestra (puede ser de forma aleatoria o siguiendo algún tipo de estrategia) para proponerlos como centro de los *K* grupos a definir. A continuación, tomando como referencia los centroides propuestos, se calcula la distancia de cada muestra a todos los centros, asignando las observaciones al centro más cercano (la métrica habitualmente usada es la distancia euclidiana cuadrada). Luego, ya que todas las muestras pertenezcan a un grupo se procede a reubicar los centroides colocándolos en el centro de todas las observaciones que forman parte del agrupamiento. Estos pasos se repiten de forma iterativa por lo que hay distintas formas para saber cuando es el mejor momento de detener el proceso; una forma es que los centroides ya no puedan ser reubicados puesto que siempre se acomodan en el mismo lugar; otra forma es fijar un número máximo de repeticiones en la cual el algoritmo tendrá que detenerse.

Por la naturaleza de *K*-means se tiene que tomar en consideración que los resultados de esta implementación no siempre serán iguales, los grupos finales pueden variar dependiendo de la propuesta de ubicación inicial de los centroides por lo que se recomienda implementar una estrategia para determinar qué resultados conservar. En la figura 2.11 se puede observar a la izquierda un conjunto de datos no agrupado y a la derecha el mismo conjunto de datos después de ser agrupado por *k*-medias.

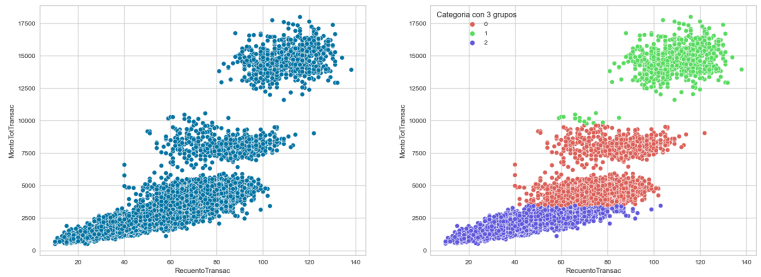


Figura 2.11: Comparativo de datos no agrupados y agrupados por *k*-medias.

Modelo

Una forma más formal, técnica y matemática para definir *K*-means sería la siguiente: escoger los centroides que minimicen la suma de errores cuadrados entre ellos y cada observación dentro del grupo.

$$\sum_{i=0}^{n} \min_{\mu_j \in C} \left(\left\| x_i - \mu_j \right\|^2 \right) \tag{2.4}$$

donde:

x_i se refiere a la i-ésima observación;
μ_j se refiere a la media de las observaciones en el grupo o lo que es equivalente al centroide del grupo;
C represente al grupo formado.

Método del codo

Como se mencionó antes, una de las desventajas del algoritmo *K*-means es la necesidad previa de definir el número de grupos en que los datos serán divididos. Uno de los enfoques para resolver esta problemática es el método del codo que se encarga de señalar el número óptimo de segmentos en que los datos serán separados, de una manera visual.

La técnica consiste en la ejecución del método *K*-medias un número determinado de veces y en cada ocasión se utiliza un número consecutivo diferente de grupos a segmentar. El siguiente paso es calcular la suma al cuadrado de todas las distancias entre los centros de cada grupo y sus puntos pertenecientes; se tiene que considerar que mientras más grupos haya menor será el valor de esta métrica, esto debido a que existirán más grupos y por tanto centros más cercanos a cada observación por lo que elegir el algoritmo con la menor distancia o con el menor error no es la solución buscada; se tiene que encontrar el número de grupos que tenga el cambio más radical entre las distancias para después tener disminuciones más pequeñas.

Cuando los valores de las distancias se grafican la figura resultante se asemeja a un brazo, la parte que corresponde visualmente al codo es la que se elije como el número ideal de agrupaciones para el modelo. Sin embargo, hay situaciones en donde esta visualización no es tan clara a la vista, por lo que también puede obtenerse de forma analítica, en la figura 2.12 se puede observar el punto de interés en el valor 5, por lo que el número óptimo de grupos en este caso sería 5.

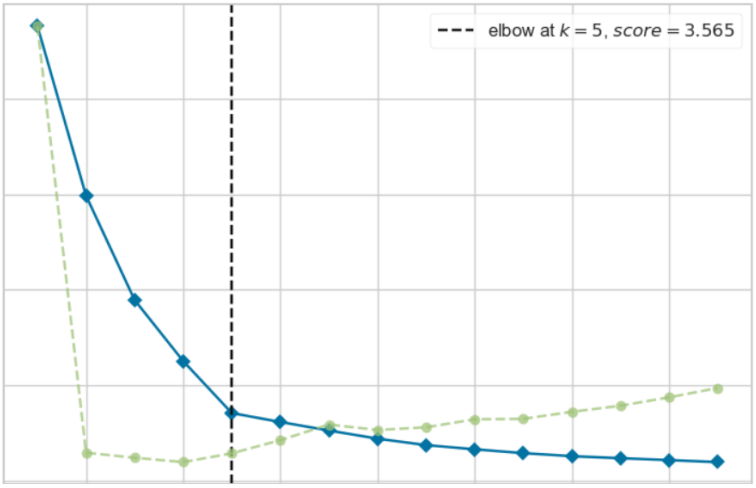


Figura 2.12: Gráfica representativa del método del codo.

2.4.2. DBSCAN

Hoy en día, tal como es el caso de *K*-means, el algoritmo *DBSCAN (Density-Based Spatial Clustering of Applications with Noise)* o Agrupación espacial basada en densidad de aplicaciones con ruido por sus siglas en inglés, presentado por primera vez en (Martin. E., Kriegel, H., Sander, J., Xu, X., 1996), es uno de los algoritmos de agrupación más populares. Tal como su nombre lo indica, su principal característica es que hace los agrupamientos de puntos basándose en la densidad, esto quiere decir que, mientras haya un conjunto de puntos con una determinada proximidad, estos pertenecerán a un grupo, separado de otros grupos debido a la falta de densidad de puntos que existe entre ellos.

Este método presenta ciertas peculiaridades, la principal es que no espera que los grupos tengan cierto tipo de formas o estructuras. Por la naturaleza de algunos modelos, los agrupamientos tienen una forma esférica, sin embargo, las formas que los grupos bajo el algoritmo de DBSCAN pueden tener son infinitas, siempre y cuando mantengan una densidad mínima necesaria. Otro rasgo distinto a lo visto en *K*-means, es que no necesita especificarse el número de grupos que se obtendrán al final; a pesar de no tener que especificarse este valor, el algoritmo DBSCAN requiere la especificación de otros dos parámetros para su correcta realización y obtención de resultados precisos.

El primer parámetro demandado es el de *puntos mínimos*, este se refiere a cuántos puntos tienen que ser vecinos o cuantos puntos tienen que estar suficientemente juntos para ser considerados un grupo; por supuesto, esto también va a depender de las necesidades del conjunto de datos, del problema o de la empresa que solicita este agrupamiento. Ciertamente pueden existir métodos para saber cual es el número óptimo de puntos mínimos a considerar para tener el modelo con mejor desempeño, pero los requerimientos pue-

den tomar ventaja sobre esto. Por ejemplo, una empresa quiere hacer distintos tipos de publicidad para las diferentes clases de clientes con los que se cuentan, por lo tanto, al tener tantos clientes, sólo lanzaría estrategias para los grupos conformados por más de 10 000 clientes, lo que significa que los puntos mínimos para considerar grupos en este caso serían 10 000.

El siguiente parámetro es conocido como *épsilon* y se encarga de definir la distancia máxima que debe haber entre dos puntos para ser considerados del mismo grupo; como se dijo anteriormente, DBSCAN se basa en la densidad que existe entre los puntos, pero el algoritmo no se encarga de decidir cuando los puntos están lo suficientemente juntos para considerarlos del mismo grupo. Este parámetro también necesita un análisis previo o alguna estrategia para definirlo, ya que si este coeficiente es muy pequeño existe la posibilidad de que no exista ningún par de puntos que cumpla la condición y si es muy grande todo el conjunto podría pertenecer al mismo grupo. En la figura 2.13 se puede encontrar un ejemplo de un conjunto de datos agrupado por DBSCAN.

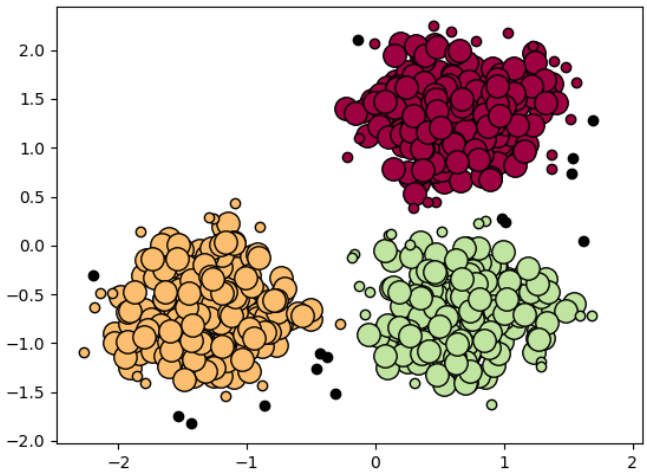


Figura 2.13: Datos agrupados con DBSCAN

Modelo

Como se explica en (Raschka, 2015) la densidad es definida por el número de puntos dentro de un radio específico ε. De acuerdo con el algoritmo DBSCAN, se etiqueta cada punto siguiendo el siguiente criterio:

- se considera *punto central* si tiene un número específico de puntos (*puntosMínimos*) vecinos dentro de un radio específico ε;
- un *punto fronterizo* es un punto que tiene menos puntos (*PuntosMínimos*) vecinos dentro de ε pero se encuentra dentro del ε de un punto central;
- todos los otros puntos son considerados *puntos de ruido*.

Después de etiquetar los puntos DBSCAN puede ser resumido en dos pasos:

1. Formar diferentes grupos para cada punto central o grupos de puntos centrales conectados (los grupos centrales están conectados sino están más lejos de ε).
2. Asignar cada punto fronterizo al grupo de su correspondiente punto central.

2.4.3. Otros modelos de aprendizaje no supervisado

A continuación se expondrá una breve explicación de otros modelos de aprendizaje no supervisado.

Affinity Propagation (*Scikit*, 2021) es un modelo de aprendizaje no supervisado el cual crea grupos a partir de dos matrices, una de responsabilidad y otra de disponibilidad. La primera es la encargada de determinar que tan responsable es cada observación de nuestro conjunto de datos y la siguiente matriz está encargada de determinar la cantidad de puntos que tiene una observación alrededor. No necesita especificarse cuantas agrupaciones serán las resultantes.

Mean Shift (D. Comaniciu, P. Meer, 2002) está basado en ventanas que intentan encontrar áreas densas de observaciones. Está basado en el centroide por lo que tiene como objetivo identificar el punto central ideal de cada agrupamiento. La forma en que obtiene los centroides es actualizando los puntos centrales de tal forma que éstos sean la media dentro de la ventana que se está evaluando. No es necesario especificar el número de grupos.

2.4.4. Coeficientes de evaluación

Los coeficientes de evaluación desempeñan un papel importante en el desarrollo de modelos de aprendizaje automático. Con apoyo en ellos, podemos tener conocimiento sobre el rendimiento y la eficacia de los modelos implementados sabiendo si son útiles o no. Existen muchas medidas de evaluación que pueden utilizarse para calificar los algoritmos de aprendizaje de máquina, en particular para el aprendizaje no supervisado los coeficientes más populares y más utilizados son el *Coeficiente de silueta*, el *Coeficiente de Calinski* y el *Coeficiente de Davies*.

Coeficiente de silueta

Este coeficiente toma valor entre el rango de -1 a 1 en donde -1 significa que los agrupamientos son incorrectos y 1 que son correctos. La forma de obtener el coeficiente de silueta se obtuvo de (Rousseeuw, 1987).

$$s = \frac{b - a}{max(a, b)} \tag{2.5}$$

Dónde:

a: se refiere a la distancia promedio entre una observación y las demás observaciones de su grupo;

b: se refiere a la distancia promedio entre una observación y las demás observaciones del grupo vecino más cercano.

Coeficiente de Calinski

El coeficiente de Calinski nos indica que el modelo tiene una mejor calidad de agrupamiento mientras el coeficiente sea más grande. La forma de obtener el coeficiente de Calinski se obtuvo de (Caliński y Harabasz, 1974).

$$s = \frac{tr(B_k)}{tr(W_k)} \times \frac{n_E - k}{k - 1} \tag{2.6}$$

Dónde:

$$W_k = \sum_{q=1}^{k} \sum_{x \in C_q} (x - c_q)(x - c_q)^T \tag{2.7}$$

$$B_k = \sum_{q=1}^{k} n_q (c_q - c_E)(c_q - c_E)^T \tag{2.8}$$

E: conjunto de datos;

n_E: número de muestras del conjunto de datos;

k: número de grupos en los que las observaciones fueron agrupadas;

B_k: matriz de dispersión entre grupos;

W_k: matriz de dispersión dentro de un determinado grupo;

$tr()$: traza de la matriz, es decir, la suma de los elementos de la diagonal;

q: un grupo determinado;

C_q: conjunto de puntos dentro de un grupo determinado;

c_q: centro de un grupo determinado;

c_E: centro del conjunto de datos E;

n_q: número de puntos en un grupo determinado.

Coeficiente de Davies

El número de grupos óptimo es aquel que minimice el valor del coeficiente de Davies. La forma de obtener el coeficiente de Davies se obtuvo de (Davies y Bouldin, 1979). El coeficiente de Davies es la similitud promedio entre cada grupo C_i para $i = 1, 2, ..., k$ y su grupo más similar C_j. La similitud es definida como R_{ij}. El coeficiente de Davies-Bouldin es definido como:

$$DB = \frac{1}{k} \sum_{i=1}^{k} \max_{i \neq j} R_{ij} \tag{2.9}$$

Dónde:

$$R_{ij} = \frac{s_i + s_j}{d_{ij}} \tag{2.10}$$

s_i: distancia promedio entre cada punto del grupo i y su centro;

d_{ij}: la distancia entre los centros de los grupos i y j.

2.5. Aprendizaje supervisado

El *aprendizaje supervisado* según (Raschka, 2015) es una rama del aprendizaje automático el cual tiene como objetivo tomar ventaja de la información proporcionada sobre los datos para generar un modelo capaz de recibir un conjunto de características y deducir una determinada clasificación o regresión para dicho conjunto. El conjunto de datos suele tener una variable objetivo la cual está etiquetada, y dicha etiqueta es la que se tiene como objetivo para asignar a nuevos datos no etiquetados. Esto se hace mediante un entrenamiento, usando la mayoría o la totalidad del conjunto de datos (esto depende del tamaño de la muestra con la que se cuente).

En otras palabras, el aprendizaje supervisado es un conjunto de técnicas que se utilizan para aprender de la información que se tiene, estudiando el comportamiento de sus variables en relación con una variable en específico, la cuál es la variable objetivo. La esencia radica en utilizar esta información con datos futuros, para poder predecir el comportamiento de la variable de interés ante distintos comportamientos de las variables restantes. En aplicaciones como predicciones de incumplimiento de pagos, este tipo de aprendizaje puede ser de gran utilidad y representar mayores ganancias para la institución.

La computadora aprende de los datos para poder predecir nuestra variable de interés la cual puede ser de distintos tipos, como nominal, binaria, numérica e incluso texto; dependiendo de como sea la variable objetivo se busca un modelo que se aplique mejor a las necesidades. En la actualidad, existen diversos tipos de aprendizajes supervisados, este trabajo se enfocará en dos de los más comunes y efectivos de ellos, *Regresión logística* y *Árboles de decisión*, sin embargo, en la Figura 2.14 se pueden ver otros algoritmos de aprendizaje supervisado.

Evidentemente, al necesitarse un entrenamiento previo para poder lograr etiquetar el posible comportamiento de los datos nuevos, es imprescindible contar con la mejor calidad de los datos y por supuesto, conforme pasa el tiempo, ir monitorizando los resultados y continuar alimentando de conocimiento al modelo, para que esté mejor entrenado y con el paso del tiempo pueda concluir en mejores resultados.

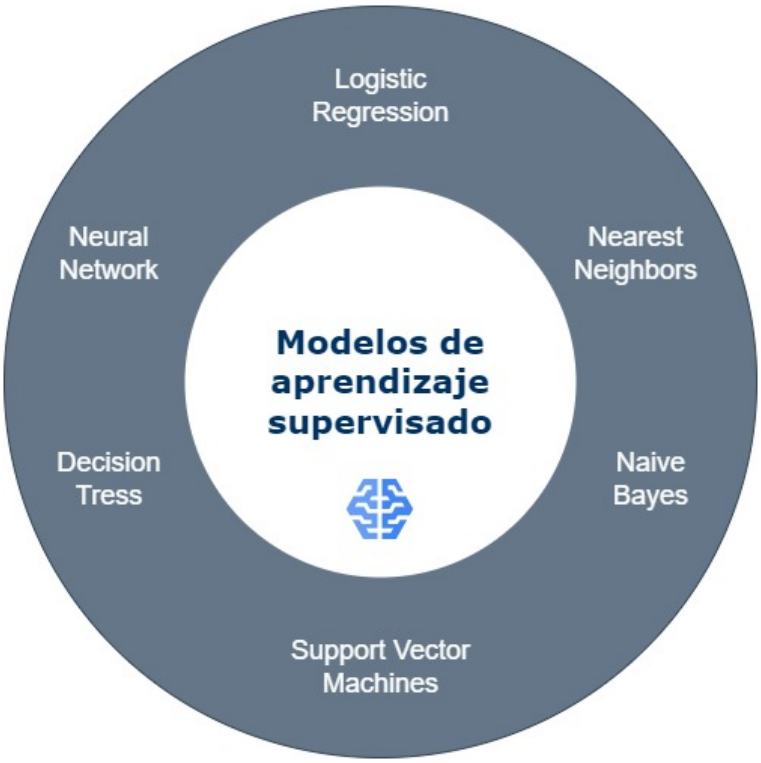


Figura 2.14: Algoritmos de aprendizaje supervisado.

2.5.1. Conjuntos de entrenamiento y de prueba

Una vez hecho un correcto preprocesamiento de los datos y un efectivo análisis exploratorio, se puede empezar a seleccionar las variables independientes que servirán como base para explicar el comportamiento de la variable objetivo. Un punto a seguir bastante relevante y de aplicación casi obligatoria es dividir nuestro conjunto de datos en dos grupos, un conjunto de entrenamiento y un conjunto de prueba. La apreciación cualitativa del analista desempeña un papel muy importante en esta etapa, ya que dependiendo de la característica de los datos es la forma en que el conjunto debe ser dividido.

Por ejemplo, al contar con un conjunto de datos enorme, la idea más adecuada sería hacer la división con una relación de 90/10 o 99/1 utilizando el 90 % o el 99 % para el conjunto de entrenamiento y el porcentaje restante para la evaluación, esto pasa ya que al haber muchos datos, se tendrá un número suficiente de evaluación con un pequeño porcentaje de la información y se pueden aprovechar la mayor cantidad de datos para hacer un entrenamiento más eficaz. Es claro que el contar con un conjunto de datos enorme es algo subjetivo, pero esto puede ser una buena idea cuando hablamos de un número de observaciones con 6 cifras o más. Cuando los datos son vastos, el entrenamiento puede considerar y reconocer incluso casos no

muy comunes, casos que se perderían completamente al hacer el conjunto de prueba más grande.

Por otro lado, el tener un conjunto de datos muy pequeño, podría llevar a tomar como consideración dividir el conjunto en 100/0 ya que lo más razonable sería utilizar todos los datos disponibles para entrenamiento. En esta situación, el dividir de una forma más equitativa podría traer falsas buenas evaluaciones o falsas malas evaluaciones dependiendo de los datos escogidos para el entrenamiento, lo cual no sería muy confiable. Continuando con el mismo razonamiento, un conjunto de datos pequeño es algo subjetivo y depende de cada analista, comúnmente sucede con información que las empresas recolectan de forma mensual, trimestral o anual y el tiempo que llevan recopilándola es corto.

Lo más común es que se utilicen divisiones de 80/20 o 70/30, pero como se expuso anteriormente, la decisión más acertada depende de los datos y del análisis cualitativo de la persona encargada. Algunas personas también consideran una tercera división, que corresponde a los datos de validación, esta no será tomada en cuenta en este trabajo ni en las aplicaciones, pero vale la pena mencionar su existencia.

Es importante precisar el concepto de *sobre ajuste* de un modelo, que se produce cuando el modelo obtenido se ajusta tanto a los ejemplos etiquetados de entrada que no puede realizar las predicciones correctas en aquellos ejemplos de datos que no han sido utilizado con anterioridad.

2.5.2. Regresión lineal

Probablemente la técnica más conocida y utilizada del aprendizaje automático supervisado es la *regresión lineal*, la cual expresa una relación lineal entre características y un determinado resultado o etiqueta (Torres, 2018). En la fase de entrenamiento de un modelo se aprenden los valores ideales para los parámetros del modelo y en el aprendizaje supervisado, la manera de conseguirlo es aplicar un algoritmo que obtenga el valor de estos parámetros examinando muchos ejemplos etiquetados e intentar determinar los valores para estos parámetros del modelo que minimicen lo que llamamos la variable error.

Podríamos decir que los algoritmos de regresión modelan la relación entre distintas variables de entrada utilizando una medida de error, que se intentará minimizar en un proceso iterativo para poder realizar predicciones lo más acertadas posibles. Hablaremos de dos tipos: regresión lineal y regresión logística. La diferencia principal entre regresión logística y lineal es en el tipo de salida de los modelos; cuando nuestra salida sea discreta hablamos de regresión logística, y cuando la salida sea continua hablamos de regresión lineal.

Modelo

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k + \epsilon \tag{2.11}$$

Donde:

y: es la etiqueta, la variable dependiente o el resultado que se quiere conocer.

x_i: son las características que describen a la etiqueta o las variables independientes.

β_i con $i = 1, 2, ..., k$: es la pendiente de la recta y que en general le llamaremos peso y es uno de los dos parámetros que se tienen que aprender del modelo durante el proceso de entrenamiento para poder usarlo luego para inferencia.

ϵ: es el punto de intersección de la recta en el eje también conocido como el error.

2.5.3. Regresión logística

La regresión logística es un algoritmo de clasificación, parte de la familia de aprendizaje supervisado. A pesar de que pueda parecer diferente, es un modelo lineal que tiene como objetivo clasificar de forma binaria un problema utilizando como referencias variables predictoras que serán parte de una ecuación

similar a la regresión lineal, en la cual se tiene que encontrar el mejor ajuste a la realidad; esto se puede extender a un problema de clasificación multiclase.

Hacer una buena implementación de una regresión logística definitivamente representa un reto amplio que excede los alcances de este trabajo, sin embargo, se mencionarán algunas consideraciones que son vitales para tener en cuenta y tener resultados satisfactorios. Para una mejor comprensión, se necesitan tener conocimientos sólidos sobre conceptos matemáticos y estadísticos.

También conocida como modelo logit, toma como base la probabilidad de cada observación para pertenecer a determinada categoría, dicha probabilidad es condicionada por los valores de las variables predictoras, dependiendo de la probabilidad obtenida se clasifican las observaciones.

Modelo

$$f(z) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_k X_k)}}$$

2.5.4. Árbol de decisión

Es notoria la fama y popularidad que tienen los modelos de *Árboles de decisión* divididos entre *Árboles de clasificación* y *Árboles de regresión*, incluso tienen mucho renombre los *Bosques de clasificación* y los *Bosques de regresión*, los cuales son una extensión y modificación de los primeros modelos. El índice de aprobación de estos algoritmos puede deberse en gran medida a la facilidad visual que representa para los analistas saber qué es lo que está pasando de una forma muy fácil e intuitiva de entender, incluso es amigable para comprender de forma gráfica.

A diferencia de otros algoritmos, los árboles de decisión no requieren un preprocesamiento exhaustivo al conjunto de datos, sin embargo, al ser sensible con cantidades grandes, sí es indispensable hacerlo. Adicionalmente, existe una amplia variedad de mecanismos para llevar a cabo este método, por ejemplo, uno llamado *Top-down induction of decision trees* el cual inicia asignando cada observación como nodo raíz del árbol, los cuales se incluyen en la lista L de nodos activos.

2.5.5. Matriz de confusión

La matriz de confusión es una herramienta del aprendizaje automático la cual nos ayuda a medir el desempeño de los modelos que serán evaluados. En resumen, en la matriz podremos ver la comparación de los resultados obtenidos contra las predicciones hechas por el modelo a evaluar, siendo las columnas en donde se reflejan los pronósticos y en las filas los valores reales.

Esta herramienta nos representa 4 instancias, verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos. Los valores verdaderos positivos son aquellos que se pronosticaron que serían verdaderos y efectivamente son verdaderos. Los verdaderos negativos son los valores pronosticados negativos que efectivamente son negativos. Los valores falsos positivos fueron los que se pronosticaron positivos pero realmente son negativos y los valores falsos negativos fueron los pronosticados negativos y realmente son positivos. Por ejemplo, si un banco hace un modelo en donde espera pronosticar que clientes harán su pago correspondiente del mes y quienes no; un verdadero positivo sería los que el modelo predijo que pagarían y efectivamente pagaron; un verdadero negativo serían los que el modelo acertó que no pagarían; un falso positivo sería en dónde el modelo se equivocó diciendo que el cliente pagaría y realmente no pagó y un falso negativo en donde el modelo se equivocó y el cliente si pagó.

Las métricas indicadas por la matriz de confusión por sí mismas son muy útiles, sin embargo, esta herramienta puede utilizarse para obtener otras métricas que son valiosas para la evaluación de modelos de aprendizaje automático. De forma resumida, estas métricas son las siguientes:

- Exactitud: Mide los pronósticos que fueron acertados, es decir, la suma de los valores verdaderos positivos y verdaderos negativos entre el total de los pronósticos.

- Precisión: Mide los pronósticos positivos que fueron acertados, es decir, los verdaderos positivos entre la suma de verdaderos y falsos positivos.
- Sensibilidad: Mide la cantidad de casos positivos correctamente identificados, es decir, verdaderos positivos entre el total de todos los casos positivos (verdaderos positivos y falsos negativos). También puede ser la probabilidad de que un caso positivo realmente sea positivo.
- Especificidad: Análogamente a la sensibilidad pero con casos negativos, es decir, verdaderos negativos ente el total de todos los casos negativos (verdaderos negativos y falsos positivos). También puede ser la probabilidad de que un caso negativo realmente sea negativo.
- F1 score: Concentra la precisión y la sensibilidad en una sola métrica. Su fórmula es $2 * (Sensibilidad * Precisin)/(Sensibilidad + Precisin)$. Un modelo aceptable tiene un F1 score de 80.

Con estas métricas en mente un analista puede saber qué es lo que más le interesa obtener de su modelo para que con ayuda de estas medidas sepa si va por el camino correcto y está consiguiendo los resultados esperados.

Capítulo 3

Aplicación de aprendizaje no supervisado

3.1. Caso de estudio de aprendizaje no supervisado

Un buen ejemplo del aprendizaje no supervisado en el mundo de las finanzas es la intención o necesidad de algunas empresas en segmentar a sus clientes. La correcta realización de este ejercicio se traduce en beneficios económicos para las compañías y proyectos de mayor efectividad, como pueden ser campañas de marketing con un público objetivo más preciso, creación de productos con características más afines a los clientes o incluso, estrategias de cobranza más apropiadas. El objetivo de esta aplicación es desarrollar una buena agrupación de los clientes de un negocio utilizando los datos: edad, género, ingreso anual en miles de dólares y un puntaje de gasto por parte de los clientes. La base de datos que se utilizó se obtuvo de la plataforma web Kaggle [1].

Primero se definieron los módulos que sirvieron de apoyo para realizar el trabajo, el uso de ellos hace la realización considerablemente más cómoda y menos complicada, en ellos ya vienen definidos algoritmos eficientes que son de suma utilidad al momento de la implementación.

[1]:
```
#Manipulación de datos
import pandas as pd
import numpy as np
#Visualización
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from yellowbrick.cluster import KElbowVisualizer
from yellowbrick.cluster import SilhouetteVisualizer
#Preprocesamiento
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.preprocessing import MinMaxScaler
#Aprendizaje no supervisado
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_score
from sklearn import metrics
from sklearn.cluster import DBSCAN
```

[1] https://www.kaggle.com/vjchoudhary7/customer-segmentation-tutorial-in-python

Los módulos utilizados para esta aplicación, sus versiones y sus propósitos son explicados en la Tabla 3.1.

Pandas 1.3.1	Manipulación y tratamiento de datos.
Numpy 1.20.3	Cálculo numérico y análisis de datos.
Matplotlib.pyplot 3.4.2	Creación y personalización de gráficos en dos dimensiones.
Seaborn 0.11.1	Creación y personalización de gráficos en dos dimensiones.
Plotly.express 5.1.0	Creación y personalización de gráficas interactivas.
Yellowbrick.cluster 1.3	Análisis visual de algoritmos de aprendizaje automático.
Scikit learn 0.24.2	Amplio rango de algoritmos de aprendizaje automático.

Tabla 3.1: Bibliotecas utilizadas para la aplicación de aprendizaje no supervisado

3.2. Características generales

Luego se observaron las características principales del conjunto de datos con el fin de obtener información que pueda ser relevante para así tener una idea de si se tiene que modificar el conjunto de datos y cómo hacerlo para un análisis más efectivo; esto permite identificar posibles inconvenientes en la información como son los valores atípicos, datos duplicados o valores nulos.

Las particularidades observadas en los datos mostraron que está construida por 5 variables descriptivas las cuales son `CustomerID`, `Gender`, `Age`, `Annual Income (k$)` y Spending Score (1-100). En la misma dirección, la base está constituida por 200 observaciones que en este caso, corresponden a datos de clientes.

Todas las variables son numéricas con excepción de la variable `Gender` la cual es una variable binaria que indica si la persona es hombre o mujer. Gracias al resumen aplicado se contempla que si se escoge un cliente al azar seguramente se obtendría una mujer de 39 años, con ingreso anual de 60 mil dólares y un puntaje de gasto de 50. Aunado a esto, no se encontró ningún tipo de valor nulo o duplicado por lo que fue seguro continuar trabajando con la evidencia conseguida.

[2]:
```
datos = pd.read_csv('Mall_Customers.csv') #Lee la base de datos

print(datos.dtypes, "\n") #Muestra de qué tipo son los datos
print(datos.describe(), "\n") #Describe características principales

#Aplica un conteo a los distintos grupos en la variable "Género"
print(datos.groupby('Gender').size())
```

```
CustomerID                int64
Gender                   object
Age                       int64
Annual Income (k$)        int64
Spending Score (1-100)    int64
dtype: object
```

```
       CustomerID         Age  Annual Income (k$)  Spending Score (1-100)
count  200.000000  200.000000          200.000000              200.000000
mean   100.500000   38.850000           60.560000               50.200000
std     57.879185   13.969007           26.264721               25.823522
min      1.000000   18.000000           15.000000                1.000000
25%     50.750000   28.750000           41.500000               34.750000
50%    100.500000   36.000000           61.500000               50.000000
75%    150.250000   49.000000           78.000000               73.000000
max    200.000000   70.000000          137.000000               99.000000
```

```
Gender
Female    112
Male       88
dtype: int64
```

[3]:
```
#Detecta la cantidad de valores duplicados en la base
duplicados = datos[datos.duplicated()]
print("Número de renglones duplicados: ", duplicados.shape, "\n")
print("Número de valores nulos por variable:")
#Detecta la cantidad de variables sin información en la base
print(datos.isnull().sum())
```

```
Número de renglones duplicados:  (0, 5)

Número de valores nulos por variable:
CustomerID                0
Gender                    0
Age                       0
Annual Income (k$)        0
Spending Score (1-100)    0
dtype: int64
```

Para fines prácticos se eliminó la variable `CustomerID` ya que no nos proporciona ninguna información relevante para la segmentación. A continuación, se cambió el nombre a las variables para una manipulación más fácil y entendible siendo el nombre de las nuevas variables `Genero`, `Edad`, `Ingreso` y `Gasto`.

[4]:
```
datos = datos.drop(['CustomerID'], axis = 1) #Elimina la variable "Customer ID"
datos.columns = ['Genero', 'Edad', 'Ingreso', 'Gasto'] #Renombra las columnas
#Reemplaza los valores existentes por nuevos valores
datos = datos.replace({"Male": "Hombres", "Female": "Mujeres"})
datos.head(5) #Muestra las primeras 5 observaciones de la base de datos
```

[4]:

	Genero	Edad	Ingreso	Gasto
0	Hombres	19	15	39
1	Hombres	21	15	81
2	Mujeres	20	16	6
3	Mujeres	23	16	77
4	Mujeres	31	17	40

3.3. Análisis exploratorio de datos

Después, se procedió a hacer un análisis más específico de los datos tanto de forma analítica como gráfica. De esta manera se pueden identificar las variables más importantes para el agrupamiento o si hay alguna variable irrelevante para el objetivo del trabajo. En primer lugar, se revisó la correlación de las variables buscando alguna relación significativa y suficientemente grande para poder conservar sólo una de las variables involucradas. Se observó que las variables `Gasto` y `Edad` tienen una correlación negativa suficientemente alta para tomar en consideración la decisión de eliminar una de ellas (véanse las Figuras 3.1 y 3.2), sin embargo, esta decisión es subjetiva y depende del analista y sus objetivos. En este caso las dos variables se conservarán para observar los resultados.

En segunda instancia, se examinó la variable `Genero`, explorando su utilidad en la segmentación. De forma numérica se tiene que las mujeres tienen un mejor puntaje de gasto que los hombres y que los hombres

tienen un salario mayor que el de ellas. Si bien es cierto que hay un género dominante en cada área, la diferencia es demasiado pequeña siendo 3 puntos la ventaja de puntaje que tienen las mujeres y 3 mil dólares anuales la ventaja de salario que presentan los varones, lo cual hace que la discrepancia sea irrelevante, esto se comprueba de manera gráfica (véase la Figura 3.3). Finalmente, se aplicó la misma comparación con las demás variables y los resultados se observan en la Figura 3.4.

[5]:
```
plt.style.use('fivethirtyeight') #Establece el tema de los gráficos

plt.figure(figsize = (4,3)) #Fija el tamaño de la imagen

#Produce un mapa de calor de los resultados de la tabla de correlación
sns.heatmap(datos.corr(), annot = True)
plt.show()

fig = plt.figure(figsize=(20,4))

plt.subplot(131) #Permite imprimir más de una gráfica a la vez y las ordena

#Imprime una gráfica de puntos con una regresión lineal
sns.regplot(x = 'Ingreso', y = 'Gasto', data = datos)

plt.subplot(132)
sns.regplot(x = 'Ingreso', y = 'Edad', data = datos)

plt.subplot(133)
sns.regplot(x = 'Edad', y = 'Gasto', data = datos)

plt.show() #Imprime los resultados
```

Figura 3.1: Mapa de calor de la tabla de correlación.

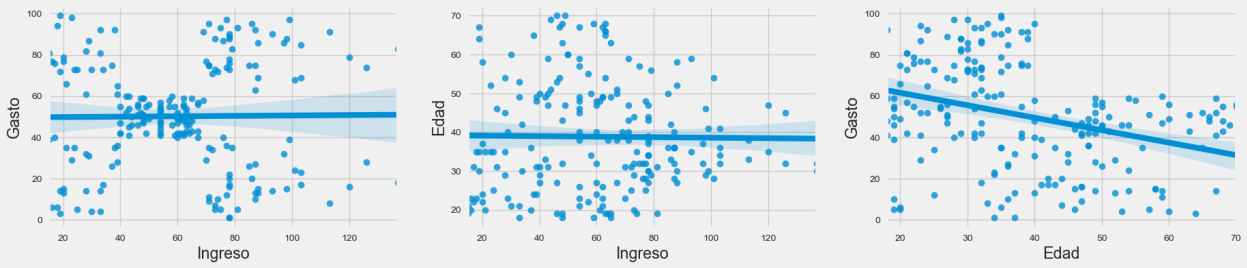


Figura 3.2: Línea de correlación y gráfica de dispersión.

[6]:
```
print("Género dividido por gasto")
print(datos.groupby('Genero')['Gasto'].mean(), "\n")

print("Género dividido por ingreso")
print(datos.groupby('Genero')['Ingreso'].mean())
```

```
Género dividido por gasto
Genero
Hombres    48.511364
Mujeres    51.526786
Name: Gasto, dtype: float64

Género dividido por ingreso
Genero
Hombres    62.227273
Mujeres    59.250000
Name: Ingreso, dtype: float64
```

[7]:
```
#Produce una conjunto de gráficas analizando la relación entre variables
sns.pairplot(datos, hue='Genero', aspect=1.5)

#Separa observaciones en los grupos distrintos de la variable "Género"
plt.show()
```

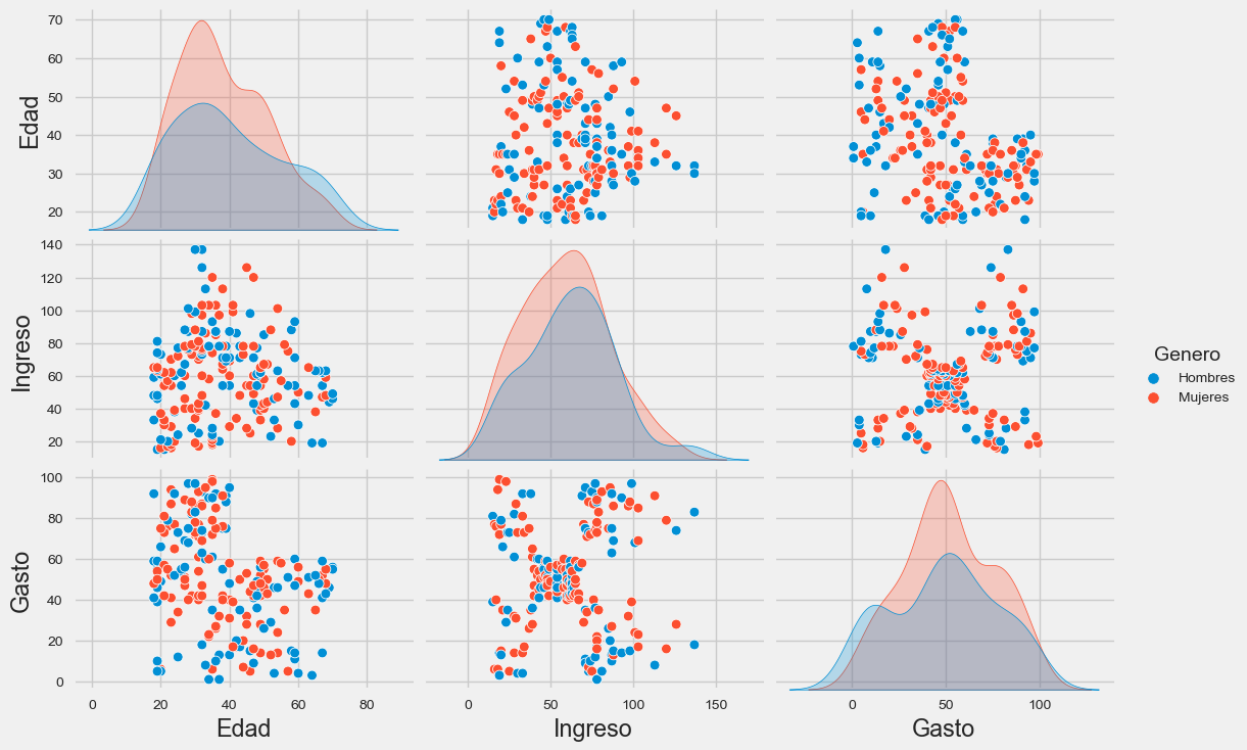


Figura 3.3: Tabla múltiple.

[8]:
```
#Genera una gráfica 3D, divide por colores respecto a "Género"
fig = px.scatter_3d(x = datos['Ingreso'],y = datos['Gasto'],
                    z=datos['Edad'], color=datos['Genero'],
                    title='Género: Ingresos vs Gasto vs Edad')
fig.show()
```

[9]:
```
fig = plt.figure(figsize=(20,4))
plt.subplot(131)
#Elabora una gráfica estilo scatter
sns.scatterplot(data=datos, x='Ingreso', y='Gasto', hue='Edad')
plt.title('Ingreso vs Gasto dividido por Edad')

plt.subplot(132)
sns.scatterplot(data=datos, x='Edad', y='Gasto', hue='Ingreso')
plt.title('Edad vs Gasto dividido por Ingreso')

plt.subplot(133)
sns.scatterplot(data=datos, x='Edad', y='Ingreso', hue='Gasto')
plt.title('Edad vs Ingreso dividido por Gasto')
plt.show()
```

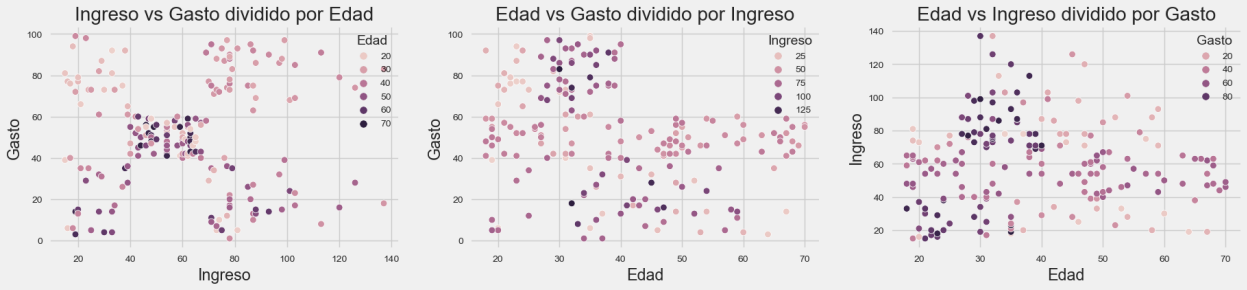


Figura 3.4: Comparación entre dos variables divididos por la variable restante.

3.4. Preprocesamiento de datos

Al llevar a cabo el preprocesamiento se busca obtener resultados más precisos en los modelos. Por esta razón se crearon distintos conjuntos de datos con el objetivo de probar distintos esquemas para comparar los resultados de las segmentaciones. Como los algoritmos de agrupación se basan principalmente en distancias se transformó la variable `Género` convirtiendo los valores `Hombres` y `Mujeres` a `1` y `0` respectivamente. De eso se desprende la primera base **Modelo 0**. Para obtener los otros modelos primero se corrige el problema de *outliers*. Por la naturaleza de las variables, la única que podría tener outliers sería la que describe el ingreso anual (véase la Figura 3.5), por lo que se revisó, detectó y corrigió el problema con la creación de una función para borrar los outliers. A continuación, la descripción y comentarios sobre los otros conjuntos de datos creados (considerando casi todas las combinaciones posibles):

Modelo 1: Se aplicó una reducción de dimensionalidad, ya que al tener 4 variables se complica la visualización de los resultados, de la misma forma es más fácil para el modelo hacer un agrupamiento con menos variables. Se usó PCA como método de reducción. Antes de implementar PCA se hizo una estandarización de los datos ya que el algoritmo es bastante sensible respecto a las varianzas de las variables iniciales.

Modelo 2: Se trabajó sin la variable `Género` ya que por los análisis realizados se concluyó que era la variable menos relevante. Se hizo una transformación de los datos para que todas las variables quedaran entre el intervalo $[0, 1]$, esto debido a lo anteriormente comentado sobre la importancia de la distancia en los algoritmos de agrupación.

Modelo 3: Se consideraron sólo las variables `Ingreso` y `Gasto`, las cuales se observaron como las dos mejores variables para hacer el agrupamiento, también se pasaron al intervalo $[0, 1]$.

Modelo 4: Se operó con las variables `Gasto` y `Edad` transformadas.

Modelo 5: Se utilizaron las variables `Ingreso` y `Edad` transformadas.

[10]:
```
datos = datos.replace({"Hombres": 1, "Mujeres": 0})

modelo0 = datos.copy() #Hace una copia exacta del conjunto de datos
```

[11]:
```
sns.boxplot(x = datos['Gasto']) #Diseña una gráfica de caja
plt.show()
```

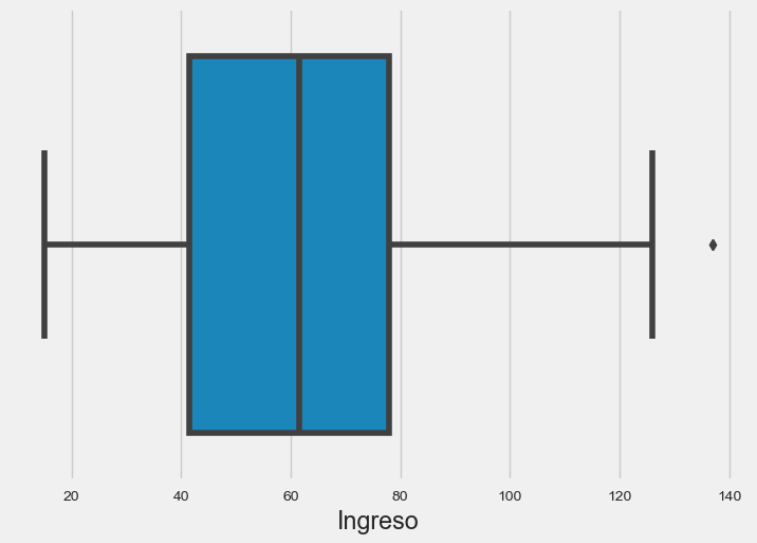


Figura 3.5: Gráfica de caja con valor atípicos.

[12]:
```
#Función para borrar outliers
def borrarOutliers(data, col):
        Q1 = data[col].quantile(0.25) #Establece lo que valdrá la variable "Q1"
        Q3 = data[col].quantile(0.75)
        IQR = Q3-Q1
        #Elimina datos que cumplan con las características descritas
        data = data.drop( data[ (data[col] < (Q1 - 1.5*IQR)) |
                                (data[col] > (Q3 + 1.5*IQR)) ].index )
return data

datos = borrarOutliers(datos, 'Ingreso')
```

[13]:
```
%datosoriginales = datos.copy()

#Se estandarizan los datos
scale = StandardScaler()
data = scale.fit_transform(datos)

#Se aplica PCA
pca = PCA(n_components = 2)

modelo1 = pca.fit_transform(data)
%modeloPCA = pca.fit_transform(data)

modelo2 = datos.drop(['Genero'], axis = 1)
%base2 = modelo2
escala = MinMaxScaler()
modelo2 = escala.fit_transform(modelo2)

modelo3 = datos.drop(['Genero', 'Edad'], axis = 1)
%base3 = modelo3
escala = MinMaxScaler()
modelo3 = escala.fit_transform(modelo3)
```

```
modelo4 = datos.drop(['Genero', 'Ingreso'], axis = 1)
%base4 = modelo4
modelo4 = escala.fit_transform(modelo4)

modelo5 = datos.drop(['Genero', 'Gasto'], axis = 1)
%base5 = modelo5
modelo5 = escala.fit_transform(modelo5)
```

3.5. *K*-medias

Una vez creados los modelos propuestos, se ejecutó el algoritmo *K*-medias para todos ellos con el fin de observar cual obtiene los mejores resultados en la segmentación. Para fines prácticos, sólo se revisará el modelo con los mejores resultados siendo este el **Modelo 3**, después se comparará con los resultados de los otros modelos explicando por qué este modelo fue el mejor.

Modelo 3: Se empezó aplicando el algoritmo de *K*-medias tomando como el número de clústers todas las opciones entre 1 y 16, esto con la finalidad de realizar el *método del codo* y conseguir el número óptimo de grupos a segmentar. En el código del primer bloque se fijaron parámetros que tienen el propósito de hacer el modelo más acertado. Con `init = k-means++` se evita que el algoritmo fije los primeros centroides de forma aleatoria para así evitar convergencias más lentas. Con `n_init = 20` especificamos que el proceso se repita 20 veces, conservando el que tenga la menor suma de errores cuadrados. Con `max_iter = 1000` se determina el número máximo de iteraciones aplicadas del procedimiento, es decir, si nuestro modelo no converge se detendrá en las 1000 iteraciones. Con `tol = 0.0001` lidiamos con los problemas que puede traer `max_iter`; `tol` se refiere a la tolerancia entre la cual se miden los cambios entre los valores de la suma de errores cuadrados para declarar convergencia entre los modelos. Con valores grandes de tolerancia es más fácil que el modelo converja. Después de aplicar este proceso se observó que el número óptimo de grupos para la base de datos era 5 (véase la Figura 3.6).

[14]:
```
#Ejecuta el algoritmo k-medias
km = KMeans(init = 'k-means++', n_init = 20, max_iter = 1000, tol = .0001)
#Visualización del método del codo
elbow = KElbowVisualizer(km, k = (1,16))
elbow.fit(modelo3)
elbow.show()
```

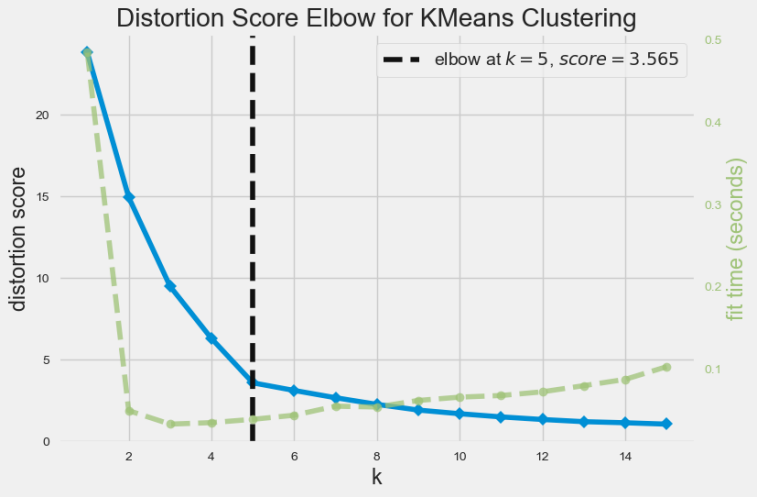


Figura 3.6: Gráfico del codo con *k* óptimo de 5.

Luego, se procedió a ejecutar nuevamente el algoritmo de *K*-medias para el número de grupos que fue indicado por el método del codo. Después, en nuestro conjunto de datos añadimos para cada observación el grupo al que pertenecen para poder así representarlo en una gráfica. En la Figura 3.7 se puede analizar de forma visual la forma en que el modelo organizó a los clientes y las características principales de estos grupos. Por ejemplo, el grupo 0 o color rojo tienen un ingreso bajo y un puntaje de gasto alto mientras que el grupo 3 o azul tienen un ingreso alto y un puntaje de gasto bajo.

[15]:
```
#Produce un vector con los valores de agrupamiento de cada observación
kmeans3 = KMeans(n_clusters = 5, init = 'k-means++', n_init = 20,
                 max_iter = 1000, tol = .0001)

kdata3 = kmeans3.fit_predict(modelo3)

#Se crea una base de datos exactamente igual pero se le agrega una columna
#que indica a que número de grupo corresponde la observación
concat = "Categoria con {} grupos".format(len(kmeans3.cluster_centers_))
modelo3 = pd.DataFrame(modelo3, columns = base3.columns)
modelo3[concat] = kmeans3.labels_
base3[concat] = kmeans3.labels_

sns.scatterplot(data = modelo3, x = 'Ingreso', y = 'Gasto',
                hue = modelo3.iloc[:,-1].name, palette=sns.color_palette('hls', 5))
plt.title('Modelo3, Ingresos vs Gasto')
plt.show()
```

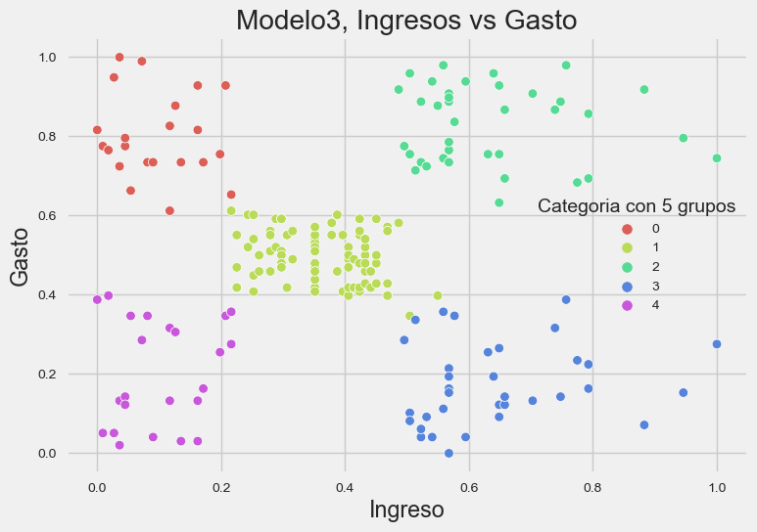


Figura 3.7: Datos del Modelo 3 agrupados con *K*-medias.

[16]:
```
fig = plt.figure(figsize = (30,6))
ax = fig.add_subplot(121)

#Produce una gráfica de gusanos
sns.swarmplot(x=modelo3.iloc[:,-1].name, y='Ingreso', data=modelo3, ax=ax)
ax.set_title('Categoría según ingresos')

ax = fig.add_subplot(122)
sns.swarmplot(x=modelo3.iloc[:,-1].name, y='Gasto', data=modelo3, ax=ax)
ax.set_title('Categoría según gastos')

plt.show()
```

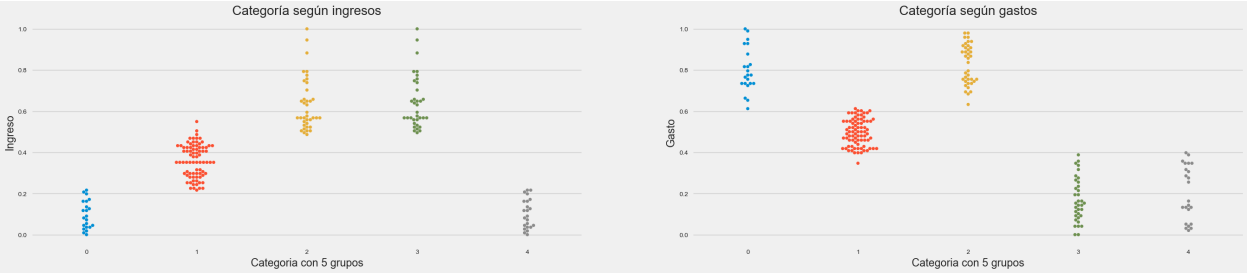


Figura 3.8: Gráfico de gusano.

Los coeficientes de Silueta, Calinski y Davies son coeficientes que califican la eficacia del modelo, con el fin de obtener evaluaciones más confiables se crearon 3 funciones que ejecutan el algoritmo *K*-medias un determinado número de veces evaluando los coeficientes en cada una de ellas, para después obtener un promedio. como se expuso anteriormente, el coeficiente de silueta es mejor mientras sea más cercano a 1, el de Calinski mientras más grande sea y el de Davies mientras más pequeño sea. Después de obtener los resultados se graficó el coeficiente de silueta para poder visualizar los resultados (véase la Figura 3.9).

[17]:
```
def siluetaprom(data, n, nclusters):
    silueta = []
    #Se ejecuta un ciclo para realizar un número determinado de veces
    #el mismo algoritmo e ir guardando los resultados
    for x in range(n):
        kmeans = KMeans(n_clusters = nclusters, init = 'k-means++',
                        n_init = 20, max_iter = 1000, tol = .0001)
        kdata = kmeans.fit_predict(data)
        #Obtiene y almacena el coeficiente de silueta
        silueta.append(silhouette_score(data,kdata))

    #Se regresa el promedio de los resultados obtenidos
    return sum(silueta)/len(silueta)

def calinskiprom(data, n, nclusters):
    calinski = []
    for x in range(n):
        kmeans = KMeans(n_clusters = nclusters, init = 'k-means++',
                        n_init = 20, max_iter = 1000, tol = .0001)
        kdata = kmeans.fit_predict(data)
        #Obtiene y almacena el coeficiente de Calinski
        calinski.append(metrics.calinski_harabasz_score(data, kdata))

    return sum(calinski)/len(calinski)

def daviesprom(data, n, nclusters):
    davies = []
    for x in range(n):
        kmeans = KMeans(n_clusters = nclusters, init = 'k-means++',
                        n_init = 20, max_iter = 1000, tol = .0001)
        kdata = kmeans.fit_predict(data)
        #Obtiene y almacena el coeficiente de Davies
        davies.append(metrics.davies_bouldin_score(data, kdata))

    return sum(davies)/len(davies)
```

[18]:
```
modelo3 = modelo3.drop([modelo3.iloc[:,-1].name], axis = 1)

#Obtiene los coeficientes promedio al ejecutar k-means
silueta3 = siluetaprom(modelo3, 50, 5)
calinski3 = calinskiprom(modelo3,50,4)
davies3 = daviesprom(modelo3,50,4)

#Imprime los coeficientes promedio
print(silueta3)
print(calinski3)
print(davies3)
```

```
0.5640959164678206
180.19492868891936
0.7005684534112605
```

[19]:
```
#Grafica el coeficiente de silueta
graficasilueta = SilhouetteVisualizer(kmeans3, colors='yellowbrick')

graficasilueta.fit(modelo3)
graficasilueta.show()
```

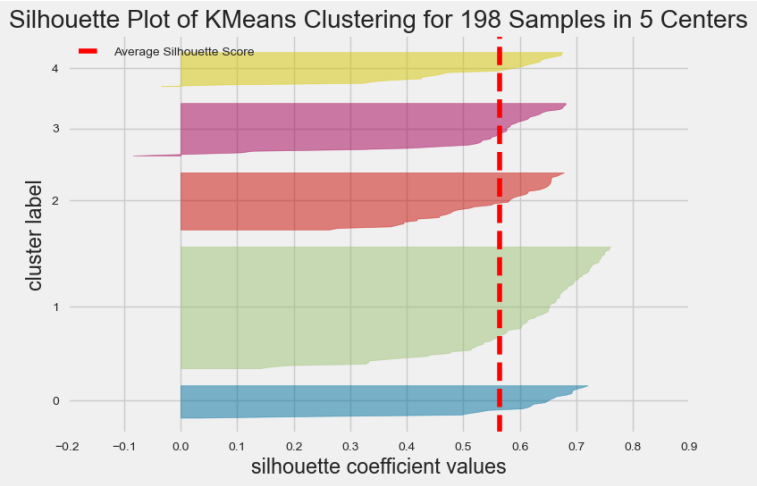


Figura 3.9: Gráfico del coeficiente de silueta.

Comparación de modelos

Después de aplicar el método de *k*-medias con los otros conjuntos de datos se ordenaron los coeficientes obtenidos de cada modelo, con esto se pudo confirmar que el **Modelo 3** fue el que mejor desempeño tuvo para hacer la segmentación de clientes con el método de *k*-means.

[20]:
```
#Modelo 0
kmeans0 = KMeans(n_clusters = 5, init = 'k-means++', n_init = 20,
                 max_iter = 1000, tol = .0001)
kdata0 = kmeans0.fit_predict(modelo0)
model0 = pd.DataFrame(modelo0)
concat = "Categoria con {} grupos".format(len(kmeans0.cluster_centers_))
modelo0[concat] = kmeans0.labels_
Base0 = modelo0
modelo0 = modelo0.drop([modelo0.iloc[:,-1].name],axis = 1)
silueta0 = siluetaprom(modelo0, 50, 5)
calinski0 = calinskiprom(modelo0,50,5)
davies0 = daviesprom(modelo0,50,5)

#Modelo 1
kmeans1 = KMeans(n_clusters = 4, init = 'k-means++', n_init = 20,
                 max_iter = 1000, tol = .0001)
kdata1 = kmeans1.fit_predict(modeloPCA)
modeloPCA = pd.DataFrame(modeloPCA)
concat = "Categoria con {} grupos".format(len(kmeans1.cluster_centers_))
modeloPCA[concat] = kmeans1.labels_
modelo1 = datos
modelo1[modeloPCA.iloc[:,-1].name] = kmeans1.labels_
```

```
modelo1.head(5)
modeloPCA = modeloPCA.drop([modeloPCA.iloc[:,-1].name], axis = 1)
Base1 = modelo1
modelo1 = modelo1.drop([modelo1.iloc[:,-1].name], axis = 1)
silueta1 = siluetaprom(modeloPCA, 50, 4)
calinski1 = calinskiprom(modeloPCA,50,4)
davies1 = daviesprom(modeloPCA,50,4)

#Modelo2
kmeans2 = KMeans(n_clusters = 4, init = 'k-means++', n_init = 20,
                 max_iter = 1000, tol = .0001)
kdata2 = kmeans2.fit_predict(modelo2)
concat = "Categoria con {} grupos".format(len(kmeans2.cluster_centers_))
modelo2 = pd.DataFrame(modelo2, columns = base2.columns)
modelo2[concat] = kmeans2.labels_
base2[concat] = kmeans2.labels_
modelo2 = modelo2.drop([modelo2.iloc[:,-1].name], axis = 1)
silueta2 = siluetaprom(modelo2, 50, 6)
calinski2 = calinskiprom(modelo2,50,4)
davies2 = daviesprom(modelo2,50,4)

#Modelo 4
kmeans4 = KMeans(n_clusters = 4, init = 'k-means++', n_init = 20,
                 max_iter = 1000, tol = .0001)
kdata4 = kmeans4.fit_predict(modelo4)
concat = "Categoria con {} grupos".format(len(kmeans4.cluster_centers_))
modelo4 = pd.DataFrame(modelo4, columns = base4.columns)
base4[concat] = kmeans4.labels_
silueta4 = siluetaprom(modelo4, 50, 4)
calinski4 = calinskiprom(modelo4,50,4)
davies4 = daviesprom(modelo4,50,4)

#Modelo5
kmeans5 = KMeans(n_clusters = 3, init = 'k-means++', n_init = 20,
                 max_iter = 1000, tol = .0001)
kdata5 = kmeans5.fit_predict(modelo5)
concat = "Categoria con {} grupos".format(len(kmeans5.cluster_centers_))
modelo5 = pd.DataFrame(modelo5, columns = base5.columns)
base5[concat] = kmeans5.labels_
silueta5 = siluetaprom(modelo5, 50, 3)
calinski5 = calinskiprom(modelo5,50,3)
davies5 = daviesprom(modelo5,50,3)
```

Organización de modelos

[21]:
```
Siluetas = pd.DataFrame([silueta0, silueta1, silueta2, silueta3,silueta4,
                         silueta5], columns = ['Siluetas'])
Calinskis = pd.DataFrame([calinski0, calinski1, calinski2, calinski3,calinski4,
                          calinski5], columns = ['Calinskis'])
Daviess = pd.DataFrame([davies0, davies1, davies2, davies3,davies4,
                        davies5], columns = ['Daviess'])
Modelos = pd.DataFrame(['Modelo0','Modelo1', 'Modelo2', 'Modelo3','Modelo4',
                        'Modelo5'], columns = ['Modelos'])
```

```
ScoresS = pd.concat([Modelos, Siluetas],axis = 1)
ScoresC = pd.concat([Modelos, Calinskis],axis = 1)
ScoresD = pd.concat([Modelos, Daviess],axis = 1)

#Ordena los modelos dependiendo de los valores obtenidos
print(ScoresD.sort_values(by = "Daviess", ascending = True))
print(ScoresC.sort_values(by = "Calinskis", ascending=False))
print(ScoresS.sort_values(by = "Siluetas", ascending=False))
```

```
   Modelos   Daviess
3  Modelo3  0.700568
5  Modelo5  0.772998
0  Modelo0  0.822078
4  Modelo4  0.853330
1  Modelo1  0.856543
2  Modelo2  0.950755

   Modelos   Calinskis
4  Modelo4  222.613804
5  Modelo5  207.587722
3  Modelo3  180.194929
1  Modelo1  178.353148
0  Modelo0  150.925798
2  Modelo2  126.190088

   Modelos  Siluetas
3  Modelo3  0.564096
5  Modelo5  0.445610
0  Modelo0  0.444045
2  Modelo2  0.430517
4  Modelo4  0.428857
1  Modelo1  0.397187
```

Para terminar con este Capítulo, en los bloques siguientes se pueden observar las características que tiene los grupos hechos con el Modelo 3 por K-means, el cual fue el mejor modelo. La gráfica 3.10 y los resultados nos indican que una persona promedio del grupo 0 tiene un ingreso de 55 mil dólares, calificación de gasto de 49, una edad de 42 años y probablemente sea mujer. De la misma manera, con la información mostrada se pueden obtener las características principales de cada grupo.

[22]:
```
datosoriginales[concat] = kmeans3.labels_
```

[23]:
```
datosoriginales
```

[23]:

	Genero	Edad	Ingreso	Gasto	Categoria con 5 grupos
0	1	19	15	39	3
1	1	21	15	81	1
2	0	20	16	6	3
3	0	23	16	77	1
4	0	31	17	40	3
..	...	...	...	...	...
193	0	38	113	91	4
194	0	47	120	16	2
195	0	35	120	79	4

```
196         0     45         126       28                          2
197         1     32         126       74                          4

[198 rows x 5 columns]
```

[24]:
```
sns.pairplot(datosoriginales, hue='Categoria con 5 grupos', aspect=1.5)
plt.show()
```

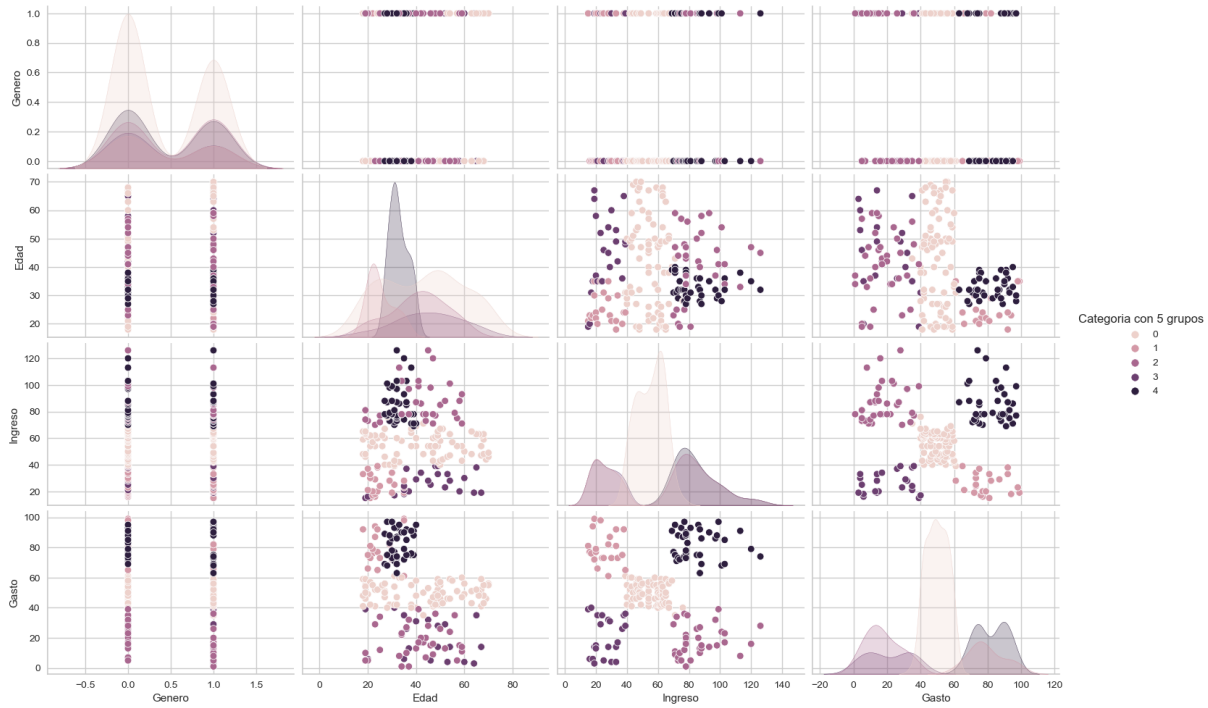


Figura 3.10: Tabla múltiple 2.

[25]:
```
# Se calcula el promedio de ingreso en cada agrupación hecha por el modelo
datosoriginales.groupby('Categoria con 5 grupos')['Ingreso'].mean()
```

[25]:
```
Categoria con 5 grupos
0    55.087500
1    25.727273
2    86.342857
3    26.304348
4    85.210526
Name: Ingreso, dtype: float64
```

[26]:
```
datosoriginales.groupby('Categoria con 5 grupos')['Gasto'].mean()
```

[26]:
```
Categoria con 5 grupos
0    49.712500
1    79.363636
2    17.571429
3    20.913043
4    82.105263
```

```
Name: Gasto, dtype: float64
```

[27]:
```
datosoriginales.groupby('Categoria con 5 grupos')['Edad'].mean()
```

[27]:
```
Categoria con 5 grupos
0    42.937500
1    25.272727
2    40.914286
3    45.217391
4    32.763158
Name: Edad, dtype: float64
```

[28]:
```
datosoriginales.groupby('Categoria con 5 grupos').Genero.value_counts(normalize = True)
```

[28]:
```
Categoria con 5 grupos  Genero
0                       0         0.587500
                        1         0.412500
1                       0         0.590909
                        1         0.409091
2                       1         0.514286
                        0         0.485714
3                       0         0.608696
                        1         0.391304
4                       0         0.552632
                        1         0.447368
Name: Genero, dtype: float64
```

3.6. DBSCAN

Con el objetivo de no prolongar innecesariamente esta sección, se mostrarán los mejores resultados obtenidos con el algoritmo de DBSCAN. De la misma forma que con *K*-medias, el tercer modelo fue el que obtuvo los mejores resultados. Para poder implementar el cálculo, se tuvieron que definir los dos parámetros de DBSCAN, `eps` y `min_samples`. Como se vio en la sección correspondiente, `eps` se refiere a la distancia máxima entre los puntos de un mismo grupo, y `min_samples` se refiere al número mínimo de observaciones que tienen que haber en un grupo para que efectivamente sea reconocido como tal. Una vez fijados estos parámetros se obtuvo el etiquetado de los datos para después obtener sus respectivos coeficientes de evaluación.

Después de consolidar los grupos, se graficaron los resultados obteniendo una gráfica bastante similar a la alcanzada por *K*-means (véase la Figura 3.11). El decidir con cuál modelo quedarse dependerá de análisis posteriores y la adecuación a las necesidades de la problemática de la empresa. Una vez terminada la implementación, se eliminan las etiquetas por si se requieren hacer pruebas posteriores con otros algoritmos de agrupación.

[22]:
```
db3 = DBSCAN(eps = .091, min_samples =4) #Algoritmo DBSCAN
dbdata3 = db3.fit_predict(modelo3) #Vector de resultados de agrupamiento

#Obtiene los coeficientes al ejecutar DBSCAN
print(silhouette_score(modelo3,dbdata3)) #Guarda el coeficiente de silueta
print(metrics.calinski_harabasz_score(modelo3, dbdata3)) #Coeficiente de calinski
print(metrics.davies_bouldin_score(modelo3, dbdata3)) #Coeficiente de davies
```

```
0.4715267951703447
105.70180020189271
1.7787886553928354
```

[23]:
```
#Agrega los grupos correspondientes a cada observación
concat = "Categoria con {} grupos".format(len(set(db3.labels_)))
modelo3[concat] = db3.labels_

#Imprime la gráfica
sns.scatterplot(data = modelo3, x = 'Ingreso', y = 'Gasto',
                hue = modelo3.iloc[:,-1].name,
                palette=sns.color_palette('hls', len(set(db3.labels_))))
plt.title('Modelo3, Ingresos vs Gasto')
plt.show()
```

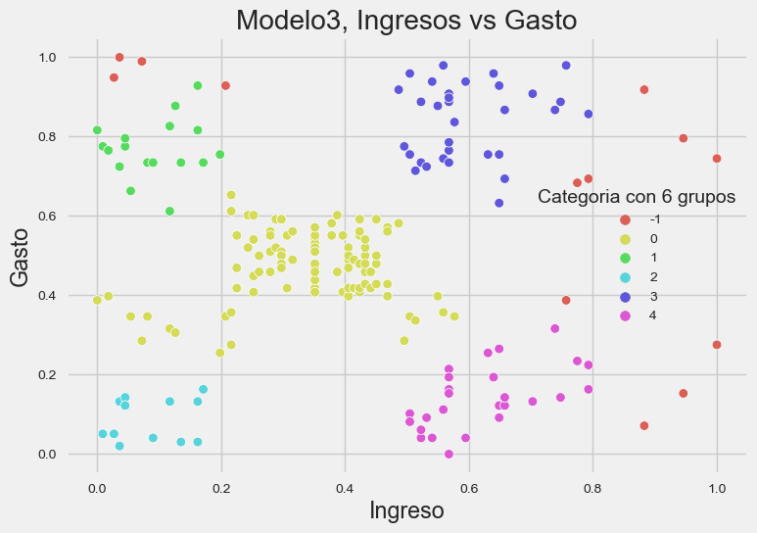


Figura 3.11: Datos del Modelo 3 agrupados con DBSCAN.

En los bloques siguientes se pueden observar las características que tienen los grupos hechos por el algoritmo DBSCAN. La gráfica 3.12 y las métricas nos indican que una persona promedio del grupo 0 tiene un ingreso de 52 mil dólares anuales, calificación de gasto de 47, una edad de 42 años y probablemente sea mujer. De forma similar, se pueden obtener las características principales de cada grupo. Se puede observar que estas características son muy similares a las obtenidas con el modelo K-Means.

[24]:
```
# Generación de la gráfica
sns.pairplot(datosoriginales, hue='Categoria con 6 grupos', aspect=1.5)
plt.show()
```

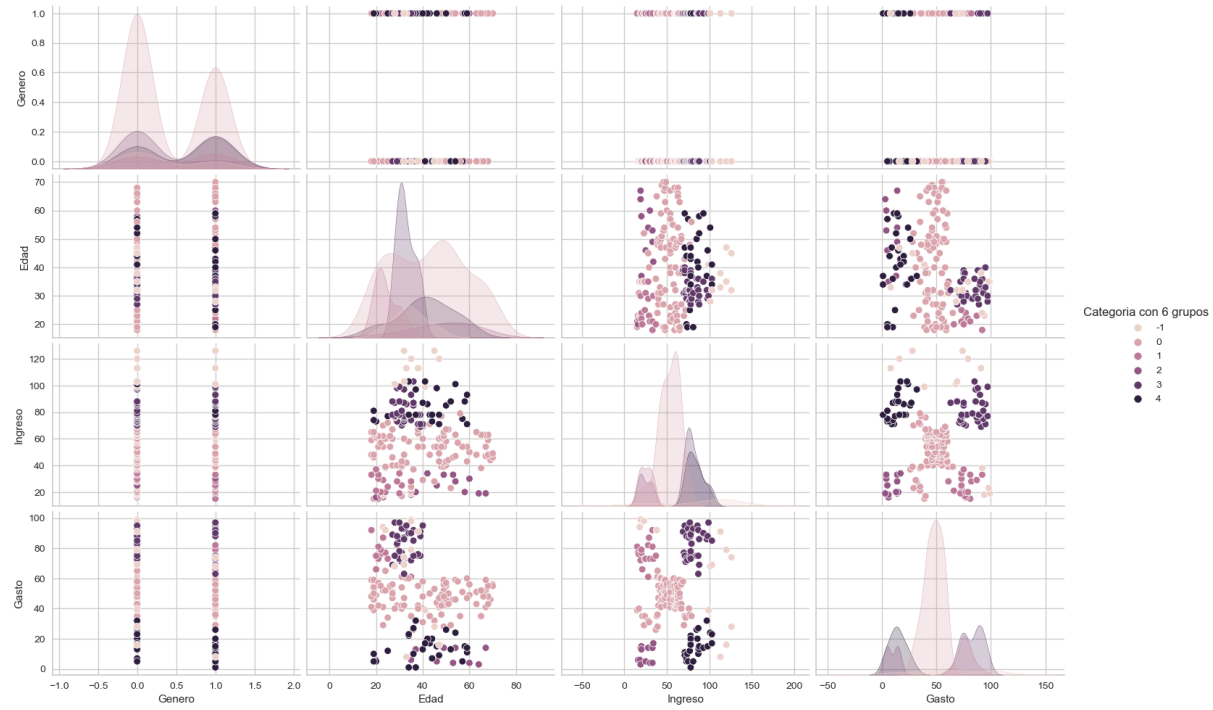


Figura 3.12: Tabla múltiple 3.

[25]:
```
# Calcula el promedio del ingreso anual para cada agrupación hecha por el modelo
datosoriginales.groupby('Categoria con 6 grupos')['Ingreso'].mean()
```

[25]:
```
Categoria con 6 grupos
-1    86.076923
 0    52.645833
 1    25.235294
 2    24.583333
 3    81.060606
 4    83.925926
Name: Ingreso, dtype: float64
```

[26]:
```
datosoriginales.groupby('Categoria con 6 grupos')['Gasto'].mean()
```

[26]:
```
Categoria con 6 grupos
-1    65.769231
 0    47.312500
 1    76.352941
 2     9.583333
 3    83.000000
 4    14.444444
Name: Gasto, dtype: float64
```

[27]:
```
datosoriginales.groupby('Categoria con 6 grupos')['Edad'].mean()
```

[27]:
```
Categoria con 6 grupos
-1    34.461538
```

```
0    42.354167
1    24.411765
2    48.750000
3    32.727273
4    41.259259
Name: Edad, dtype: float64
```

[28]:
```
datosoriginales.groupby('Categoria con 6 grupos').Genero.value_counts(normalize = True)
```

[28]:
```
Categoria con 6 grupos  Genero
-1                      0         0.692308
                        1         0.307692
 0                      0         0.604167
                        1         0.395833
 1                      0         0.529412
                        1         0.470588
 2                      0         0.583333
                        1         0.416667
 3                      0         0.545455
                        1         0.454545
 4                      1         0.592593
                        0         0.407407
Name: Genero, dtype: float64
```

3.7. Resultados

De este Capítulo se pueden enfatizar varios puntos. Primero se puede observar la importancia del análisis exploratorio previo para tener una idea de las variables o las estrategias que pueden dar mejores resultados, en la práctica esto puede ahorrar tiempo al evitar la implementación de modelos innecesarios y al mismo tiempo mantener la coherencia y el sentido de los resultados con los modelos. En esta ocasión en particular, el concluir visualmente y analíticamente que las variables Ingreso y `Gasto` eran las que podrían ser más eficientes resultó una hipótesis acertada.

También se puede agradecer en gran medida la importancia de las herramientas computacionales utilizadas ya que la implementación de este análisis resultó sumamente sencilla en comparación con haberla hecho a mano.

Por último, en este Capítulo se nota cómo unos resultados que parecen buenos, como los obtenidos mediante DBSCAN, pueden no ser realmente los mejores, lo que indica la importancia de elegir el mejor modelo para los datos puesto que es bastante relevante para la obtención de los mayores beneficios posibles para las empresas. La segmentación de los clientes trae como consecuencia una mejor implementación de estrategias de cobranza, originación de créditos y/o estrategias de mercadotecnia lo cual conlleva a mejores resultados en las finanzas de las instituciones.

Capítulo 4

Aplicación de aprendizaje supervisado

4.1. Caso de estudio de aprendizaje supervisado

La búsqueda constante de tener un negocio rentable y sostenible a través del tiempo ha llevado a las empresas a la utilización e implementación de algoritmos de aprendizaje supervisado. Lo más común es ver como las corporaciones exploran activamente distintas formas de hacer dinero, pero para algunas instituciones financieras es igualmente importante tomar medidas efectivas de como evitar perderlo. El objetivo de esta aplicación es implementar un modelo con capacidad para identificar a los prestadores con mayor probabilidad de incumplimiento a los pagos restantes de su deuda, esto se logra analizando el comportamiento de los clientes e identificando los patrones comunes entre los buenos y malos pagadores.

Primero se definieron las bibliotecas que sirvieron de apoyo para realizar el trabajo, el uso de ellas hace la realización considerablemente más cómoda y menos complicada, en ellas ya vienen definidos algoritmos eficientes que son de suma utilidad al momento de la implementación.

[1]:
```
#Manipulación de datos
import pandas as pd
import numpy as np
#Automatización
import pandas_profiling as pdp
import dabl
#Visualización
import matplotlib.pyplot as plt
import seaborn as sns
import graphviz
#Preprocesamiento
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
#Aprendizaje supervisado
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
from sklearn.ensemble import RandomForestClassifier
import statsmodels.api as sm
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
from sklearn.metrics import roc_auc_score
```

Las bibliotecas utilizadas para esta aplicación junto con sus versiones y propósitos son explicados en la tabla 4.1.

Pandas 1.3.1	Manipulación y tratamiento de datos.
Numpy 1.20.3	Cálculo numérico y análisis de datos.
Matplotlib.pyplot 3.4.2	Creación y personalización de gráficos en dos dimensiones.
Seaborn 0.11.1	Creación y personalización de gráficos en dos dimensiones.
Graphviz 2.38	Representación visual de diagramas.
Pandas_profiling 3.0.0	Análisis exploratorio de datos automático.
Dabl 0.2.2	Implementación automatizada de aprendizaje automático.
Scikit learn 0.24.2	Amplio rango de algoritmos de aprendizaje automático.
Statsmodels.api 0.12.2	Modelos estadísticos

Tabla 4.1: Bibliotecas utilizadas para aprendizaje supervisado

4.2. Características generales

Enseguida se cargaron y exploraron los datos, el conjunto de datos para esta aplicación se obtuvo de la plataforma Kaggle [1], dicho conjunto está constituido por 30,000 observaciones correspondientes a distintos clientes de un banco. Los resultados indican que todas las variables son numéricas, sin embargo, algunas variables corresponden realmente a variables categóricas pero diferenciadas con números, indicando que pertenecen a un distinto grupo de personas con una característica particular, por lo que después se aplicarán los ajustes necesarios.

[2]:
```
#Carga de datos
datos = pd.read_csv("UCI_Credit_Card.csv")

#Impresión de tipos de datos y datos estadísticos
print(datos.dtypes, "\n")
print(datos.describe(), "\n")
```

```
ID                              int64
LIMIT_BAL                     float64
SEX                             int64
EDUCATION                       int64
MARRIAGE                        int64
AGE                             int64
PAY_0                           int64
PAY_2                           int64
PAY_3                           int64
PAY_4                           int64
PAY_5                           int64
PAY_6                           int64
BILL_AMT1                     float64
BILL_AMT2                     float64
BILL_AMT3                     float64
BILL_AMT4                     float64
BILL_AMT5                     float64
BILL_AMT6                     float64
PAY_AMT1                      float64
PAY_AMT2                      float64
PAY_AMT3                      float64
```

[1] https://www.kaggle.com/uciml/default-of-credit-card-clients-dataset

```
PAY_AMT4                     float64
PAY_AMT5                     float64
PAY_AMT6                     float64
default.payment.next.month     int64
dtype: object
```

	ID	LIMIT_BAL	SEX	EDUCATION	MARRIAGE
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000
mean	15000.500000	167484.322667	1.603733	1.853133	1.551867
std	8660.398374	129747.661567	0.489129	0.790349	0.521970
min	1.000000	10000.000000	1.000000	0.000000	0.000000
25%	7500.750000	50000.000000	1.000000	1.000000	1.000000
50%	15000.500000	140000.000000	2.000000	2.000000	2.000000
75%	22500.250000	240000.000000	2.000000	2.000000	2.000000
max	30000.000000	1000000.000000	2.000000	6.000000	3.000000

	AGE	PAY_0	PAY_2	PAY_3	PAY_4
count	30000.000000	30000.000000	30000.000000	30000.000000	30000.000000
mean	35.485500	-0.016700	-0.133767	-0.166200	-0.220667
std	9.217904	1.123802	1.197186	1.196868	1.169139
min	21.000000	-2.000000	-2.000000	-2.000000	-2.000000
25%	28.000000	-1.000000	-1.000000	-1.000000	-1.000000
50%	34.000000	0.000000	0.000000	0.000000	0.000000
75%	41.000000	0.000000	0.000000	0.000000	0.000000
max	79.000000	8.000000	8.000000	8.000000	8.000000

	...	BILL_AMT4	BILL_AMT5	BILL_AMT6	PAY_AMT1
count	...	30000.000000	30000.000000	30000.000000	30000.000000
mean	...	43262.948967	40311.400967	38871.760400	5663.580500
std	...	64332.856134	60797.155770	59554.107537	16563.280354
min	...	-170000.000000	-81334.000000	-339603.000000	0.000000
25%	...	2326.750000	1763.000000	1256.000000	1000.000000
50%	...	19052.000000	18104.500000	17071.000000	2100.000000
75%	...	54506.000000	50190.500000	49198.250000	5006.000000
max	...	891586.000000	927171.000000	961664.000000	873552.000000

	PAY_AMT2	PAY_AMT3	PAY_AMT4	PAY_AMT5
count	3.000000e+04	30000.00000	30000.000000	30000.000000
mean	5.921163e+03	5225.68150	4826.076867	4799.387633
std	2.304087e+04	17606.96147	15666.159744	15278.305679
min	0.000000e+00	0.00000	0.000000	0.000000
25%	8.330000e+02	390.00000	296.000000	252.500000
50%	2.009000e+03	1800.00000	1500.000000	1500.000000
75%	5.000000e+03	4505.00000	4013.250000	4031.500000
max	1.684259e+06	896040.00000	621000.000000	426529.000000

	PAY_AMT6	default.payment.next.month
count	30000.000000	30000.000000
mean	5215.502567	0.221200
std	17777.465775	0.415062
min	0.000000	0.000000
25%	117.750000	0.000000
50%	1500.000000	0.000000
75%	4000.000000	0.000000

```
max    528666.000000                    1.000000

[8 rows x 25 columns]
```

Las variables con las que se cuenta son las siguientes:

- `LIMIT_BAL`. Cantidad de crédito disponible (en dólares NT). Incluye el crédito disponible tanto del cliente como del monto complementario a familiares.
- `SEX`. Género: correspondiendo 1 a hombres y 2 a mujeres.
- `EDUCATION`. Educación: correspondiendo 1 a graduados, 2 a universitarios y 3 a nivel preparatoria; 0, 4, 5 y 6 corresponden a otros.
- `MARRIAGE`. Estado civil: correspondiendo 1 a casados, 2 a solteros y 3 a divorciados; 0 corresponde a otros.
- `AGE`. Edad en años.
- `PAY_0 - PAY_6`. Historial de pagos desde abril del 2005 hasta septiembre del 2005 empezando con septiembre: -2 corresponde a no consumo, -1 a totalmente pagado, 0 uso del crédito, 1 es pago atrasado por un mes, 2 es pago atrasado por dos meses y así sucesivamente.
- `BILL_AMT1 - BILL_AMT6`. Monto del Estado de Cuenta (en dólares NT). Corresponde desde abril del 2005 hasta septiembre del 2005 empezando con septiembre.
- `PAY_AMT1 - PAY_AMT6`. Monto previo de pago (en dólares NT). Corresponde desde abril del 2005 hasta septiembre del 2005 empezando con septiembre.
- `default.payment.next.month`. Comportamiento de los clientes en octubre del 2005: 0 corresponde a pago y 1 corresponde a incumplimiento del pago.

La variable de interés corresponde a `default.payment.next.month` en donde la descripción muestra que los pagadores que han cumplido con sus pagos representan la gran mayoría de los clientes.

[2]:
```
#Agrupación por la columna "default.payment.next.month"
print(datos.groupby('default.payment.next.month').size())
```

```
default.payment.next.month
0    23364
1     6636
dtype: int64
```

Luego se observaron las características principales del conjunto de datos para identificar posibles inconvenientes en los datos como son los outliers, datos duplicados o valores nulos. Los resultados muestran que no existe ningún tipo de valor nulo o duplicado, por lo que fue seguro continuar trabajando con la evidencia recolectada.

[3]:
```
#Obtener lista de datos duplicados
duplicados = datos[datos.duplicated()]
print("Número de renglones duplicados: ", duplicados.shape, "\n")

#Obtener cantidad de datos nulos
print("Número de valores nulos por variable:")
print(datos.isnull().sum())
```

```
Número de renglones duplicados:  (0, 25)

Número de valores nulos por variable:
ID                          0
```

```
LIMIT_BAL                     0
SEX                           0
EDUCATION                     0
MARRIAGE                      0
AGE                           0
PAY_0                         0
PAY_2                         0
PAY_3                         0
PAY_4                         0
PAY_5                         0
PAY_6                         0
BILL_AMT1                     0
BILL_AMT2                     0
BILL_AMT3                     0
BILL_AMT4                     0
BILL_AMT5                     0
BILL_AMT6                     0
PAY_AMT1                      0
PAY_AMT2                      0
PAY_AMT3                      0
PAY_AMT4                      0
PAY_AMT5                      0
PAY_AMT6                      0
default.payment.next.month    0
dtype: int64
```

Para fines prácticos se eliminó la variable `ID` ya que no proporciona ninguna información relevante para la segmentación. Además, para una manipulación más fácil y entendible se cambió el nombre de todas las variables. Finalmente, los valores asociados a `Educacion`, `Genero` y `EstadoCivil` se agruparon para tener grupos correctamente identificados. Por el momento estos valores no son numéricos, pero al momento de comenzar a implementar el modelo estos datos serán modificados.

[4]:
```
#Modificar tabla
datos = datos.drop(['ID'],axis = 1)

datos.columns = ['LimiteCredito', 'Genero',
                 'Educacion', 'EstadoCivil',
                 'Edad', 'StatusSept', 'StatusAg', 'StatusJul', 'StatusJun',
                 'StatusMay', 'StatusAbr', 'EstadoCuentaSept',
                 'EstadoCuentaAg', 'EstadoCuentaJul', 'EstadoCuentaJun',
                 'EstadoCuentaMay', 'EstadoCuentaAbr', 'PagoAntSept',
                 'PagoAntAg', 'PagoAntJul', 'PagoAntJun', 'PagoAntMay',
                 'PagoAntAbr', 'Incumplimiento']

#Convierte la variable a tipo int64
datos['LimiteCredito'] = datos['LimiteCredito'].astype(np.int64)
#Junta todas las categorías desconocidas en "Educación"
datos['Educacion'] = datos['Educacion'].replace({4: 0, 5: 0, 6:0})

datos['Genero'] = datos['Genero'].replace({1: "Hombres", 2: "Mujeres"})

datos['Educacion'] = datos['Educacion'].replace({0: 'Desconocido', 1: 'Graduado',
                                                 2: 'Universidad', 3: 'Prepa'})
datos['EstadoCivil'] = datos['EstadoCivil'].replace({0: 'Desconocido', 1: 'Casado',
```

```
                                                    2: 'Soltero', 3: 'Divorciado'})
#Se estable el número máximo de columnas a mostrar
pd.set_option('display.max_columns', 30)
datos.head(5)
```

[4]:

	LimiteCredito	Genero	Educacion	EstadoCivil	Edad	StatusSept	StatusAg	StatusJul	StatusJun	StatusMay	StatusAbr	EstadoCuentaSept	EstadoCuentaAg	EstadoCuentaJul	EstadoCuentaJun	EstadoCuentaMay	EstadoCuentaAbr	PagoAntSept	PagoAntAg	PagoAntJul	PagoAntJun	PagoAntMay	PagoAntAbr	Incumplimiento
0	20000	Mujeres	Universidad	Casado	24	2	2	-1	-1	-2	-2	3913.0	3102.0	689.0	0.0	0.0	0.0	0.0	689.0	0.0	0.0	0.0	0.0	1
1	120000	Mujeres	Universidad	Soltero	26	-1	2	0	0	0	2	2682.0	1725.0	2682.0	3272.0	3455.0	3261.0	0.0	1000.0	1000.0	1000.0	0.0	2000.0	1
2	90000	Mujeres	Universidad	Soltero	34	0	0	0	0	0	0	29239.0	14027.0	13559.0	14331.0	14948.0	15549.0	1518.0	1500.0	1000.0	1000.0	1000.0	5000.0	0
3	50000	Mujeres	Universidad	Casado	37	0	0	0	0	0	0	46990.0	48233.0	49291.0	28314.0	28959.0	29547.0	2000.0	2019.0	1200.0	1100.0	1069.0	1000.0	0
4	50000	Hombres	Universidad	Casado	57	-1	0	-1	0	0	0	8617.0	5670.0	35835.0	20940.0	19146.0	19131.0	2000.0	36681.0	10000.0	9000.0	689.0	679.0	0

4.3. Análisis exploratorio de datos

Después, se procedió a realizar un análisis más específico de los datos tanto de forma analítica como visual. De esta manera se pueden identificar las variables más importantes para la clasificación o si hay alguna variable irrelevante para el objetivo del trabajo. Se revisó la correlación de las variables buscando alguna relación significativa y suficientemente grande para poder conservar sólo una de las variables involucradas. Se prestó especial atención a la correlación que la variable de interés `Incumplimiento` tenía con las otras variables, ya que estas se consideraron como los mejores prospectos para los modelos de clasificación (véase la Figura 4.1). Se observó que las variables `Status`, `EstadoCuenta` y `PagoAnt` son las que tenían mayor correlación entre ellas, esto debido a que estas variables son series de tiempo, lo que hace que sus valores actuales sean considerablemente condicionados por sus valores anteriores. Esto tiene que ser considerado de forma especial para incluirlo al modelo.

[5]:
```
#Se calcula la correlación entre variables
plt.figure(figsize = (20,20))
sns.heatmap(datos.corr(), annot = True)
plt.show()
```

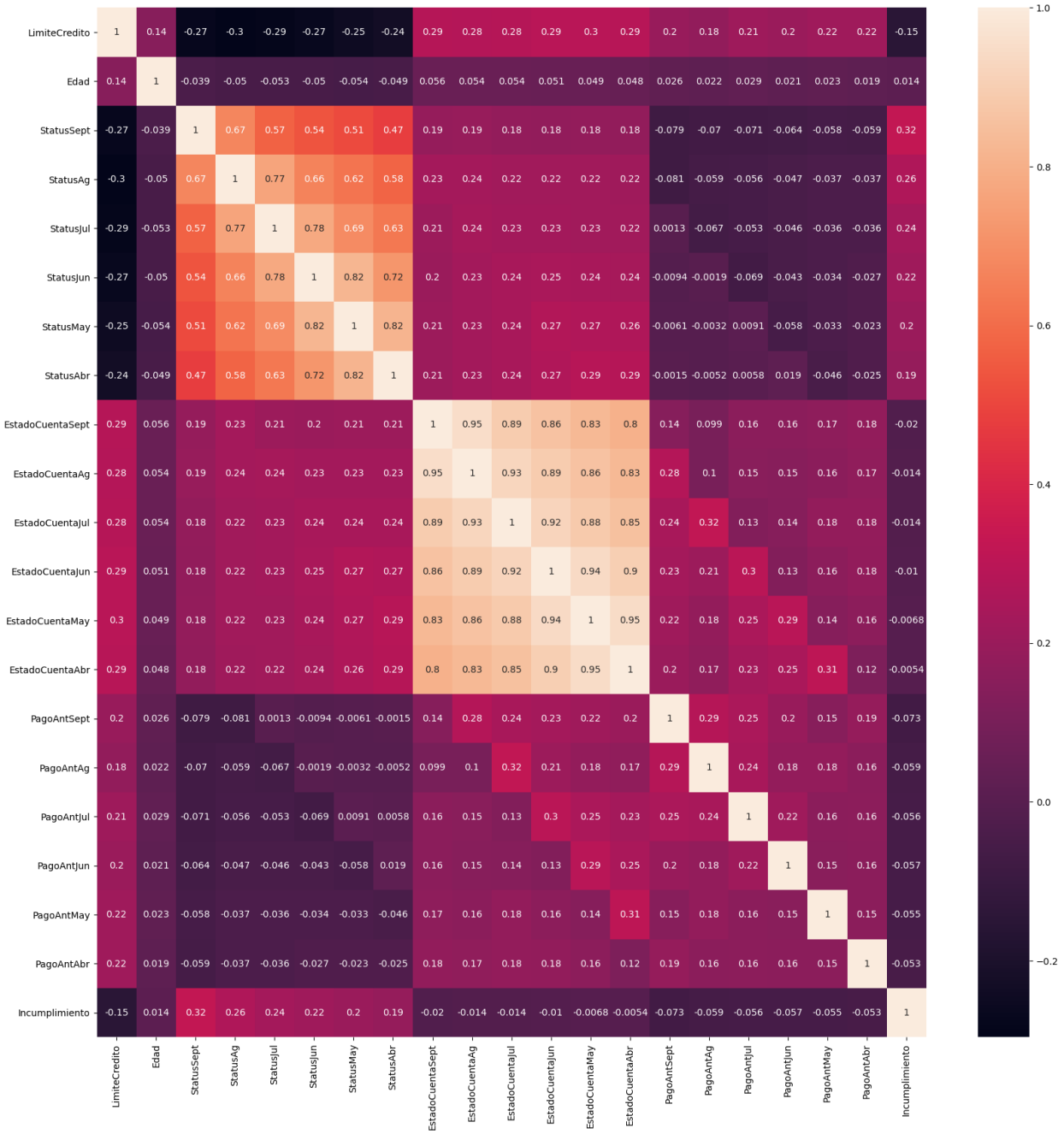


Figura 4.1: Mapa de calor de la tabla de correlación de la base de datos.

También es claro que las variables referentes a `Status` son las más vinculadas con el posible incumplimiento de los pagos.

[6]:
```
#Ordena los valores dependiendo de los resultados calculados de su correlación
#con la variable "Incumplimiento"
pd.DataFrame(datos.corr(method = 'pearson')
             ['Incumplimiento']).sort_values(by = "Incumplimiento",
                                             ascending=False)
```

[6]:

	Incumplimiento
Incumplimiento	1.000000
StatusSept	0.324794
StatusAg	0.263551
StatusJul	0.235253
StatusJun	0.216614
StatusMay	0.204149
StatusAbr	0.186866
Edad	0.013890
EstadoCuentaAbr	-0.005372
EstadoCuentaMay	-0.006760
EstadoCuentaJun	-0.010156
EstadoCuentaJul	-0.014076
EstadoCuentaAg	-0.014193
EstadoCuentaSept	-0.019644
PagoAntAbr	-0.053183
PagoAntMay	-0.055124
PagoAntJul	-0.056250
PagoAntJun	-0.056827
PagoAntAg	-0.058579
PagoAntSept	-0.072929
LimiteCredito	-0.153520

Gracias al análisis exploratorio se sabe que la proporción de clientes que recurren a impago es del 22.12 % (véase la Figura 4.2). Las observaciones a resaltar son las diferencias considerables que existen entre los pagadores e incumplidores dentro de los valores de `LimiteCredito`, `Stauts` y `PagoAnt`. En las otras variables también existen diferencias pero en su mayoría son poco significativas e intrascendentes. Los datos también indican que la probabilidad de incumplimiento baja entre los clientes que se encuentran dentro de grupos desconocidos. Todo esto se tomó en cuenta al momento de realizar el modelo.

[7]:
```
#Gráfica de pastel
plt.style.use('fivethirtyeight')
datos.Incumplimiento.value_counts(normalize = True).plot.pie(shadow =
                                          True, autopct = '%1.2f%%')
plt.show()
```

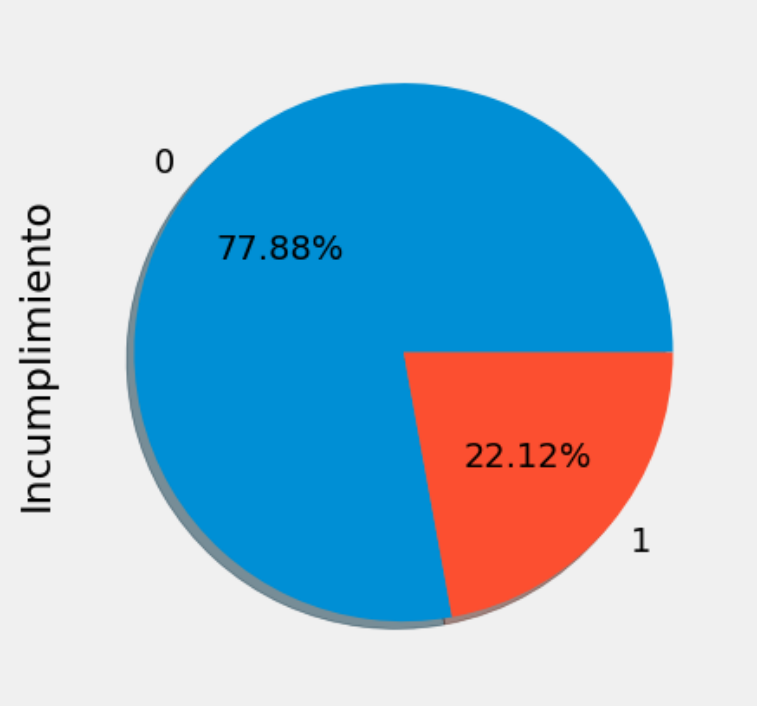


Figura 4.2: Gráfica de pastel que indica la proporción de incumplimiento entre los clientes.

En las métricas calculadas se pueden observar algunas diferencias entre los clientes que presentan incumplimiento y los clientes que no. Por ejemplo, hablando de límite de crédito las personas que sí pagan presentan un límite de crédito más alto que aquellos que no, lo cual significa que la institución financiera hace un buen trabajo distribuyendo los montos dispuestos a prestar entre sus clientes. En cuanto a educación vemos que las personas ya graduadas de la universidad presentan una probabilidad más baja de incumplimiento que aquellas que siguen en la universidad o están en la preparatoria, sin embargo, esta diferencia no es del todo significativa. Por otra parte, al analizar el estado civil se observa que las personas divorciadas tienen mayor tendencia a no pagar que las personas casadas, pero igual la diferencia no es del todo notoria.

[8]:
```
#Describe únicamente los datos pero divididos como se indica
datos.groupby('Incumplimiento')['LimiteCredito'].describe()
```

[8]:
```
                   count           mean            std      min      25%  \
Incumplimiento
0                23364.0  178099.726074  131628.359660  10000.0  70000.0
1                 6636.0  130109.656420  115378.540571  10000.0  50000.0

                    50%       75%        max
Incumplimiento
0                150000.0  250000.0  1000000.0
1                 90000.0  200000.0   740000.0
```

[9]:
```
#Se muestra de forma total y porcentual el número de personas
#pertenecientes a cada grupo
a = datos.groupby('Educacion')['Incumplimiento'].value_counts(normalize =
                                                              True) #Porcentual
b = datos.groupby('Educacion')['Incumplimiento'].value_counts()
```

```
pd.concat([a,b], axis = 1)
```

[9]:

		Incumplimiento	Incumplimiento
Educacion	Incumplimiento		
Desconocido	0	0.929487	435
	1	0.070513	33
Graduado	0	0.807652	8549
	1	0.192348	2036
Preparatoria	0	0.748424	3680
	1	0.251576	1237
Universidad	0	0.762651	10700
	1	0.237349	3330

[10]:
```
a = datos.groupby('EstadoCivil')['Incumplimiento'].value_counts(normalize = True)
b = datos.groupby('EstadoCivil')['Incumplimiento'].value_counts()

pd.concat([a,b], axis = 1)
```

[10]:

		Incumplimiento	Incumplimiento
EstadoCivil	Incumplimiento		
Casado	0	0.765283	10453
	1	0.234717	3206
Desconocido	0	0.907407	49
	1	0.092593	5
Divorciado	0	0.739938	239
	1	0.260062	84
Soltero	0	0.790717	12623
	1	0.209283	3341

La serie de tiempo 4.3 muestra cómo las personas que no presentan incumplimiento tienen un promedio negativo en cuanto a su estatus de pago, recordando que los valores negativos representan a no consumo, pago total o consumo y pago al corriente, mientras que las personas que incumplieron presentaban determinados meses de atraso.

[11]:
```
 #Calcula el promedio de los valores especificados
a = datos.groupby('Incumplimiento')['StatusSept'].mean()
b = datos.groupby('Incumplimiento')['StatusAg'].mean()
c = datos.groupby('Incumplimiento')['StatusJul'].mean()
d = datos.groupby('Incumplimiento')['StatusJun'].mean()
e = datos.groupby('Incumplimiento')['StatusMay'].mean()
f = datos.groupby('Incumplimiento')['StatusAbr'].mean()

#Ordena los resultados en un DataFrame
serie1 = pd.concat([a,b,c,d,e,f], axis = 1)
Status0 = pd.DataFrame({"Date": [6,5,4,3,2,1],
                        "Status": [serie1.iloc[0,0] ,serie1.iloc[0,1],
                        serie1.iloc[0,2], serie1.iloc[0,3], serie1.iloc[0,4],
                        serie1.iloc[0,5]]})
Status1 = pd.DataFrame({"Date": [6,5,4,3,2,1], "Status": [serie1.iloc[1,0] ,
                        serie1.iloc[1,1] ,serie1.iloc[1,2], serie1.iloc[1,3],
                        serie1.iloc[1,4], serie1.iloc[1,5]]})

#Grafica un par de series tiempo
```

```
sns.lineplot(x = 'Date', y = 'Status', data = Status0, marker = 'o', markersize = 10,
             label = 'No incumplimiento')
sns.lineplot(x = 'Date', y = 'Status', data = Status1, label = 'Incumplimiento',
             marker = 'o', markersize = 10)

#Rellena el espacio entre las series
plt.fill_between(Status0['Date'], Status0['Status'], Status1['Status'],
                 interpolate = True, alpha = 0.1, label = 'Diferencia',
                 where =(Status1['Status'] != Status0['Status']))

plt.legend()
plt.show()
```

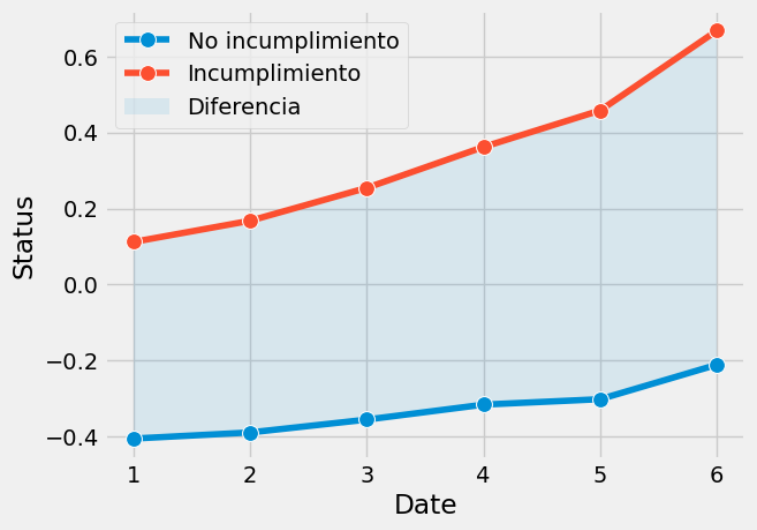


Figura 4.3: Serie de tiempo que indica el comportamiento del estatus de deuda.

De forma análoga, la serie de tiempo 4.4 muestra el pago promedio entre las personas que incumplieron y las que sí pagaron. Claramente se ve la diferencia y cómo las personas que no incumplieron aportaban más al pago de sus deudas.

[12]:
```
a = datos.groupby('Incumplimiento')['PagoAntSept'].mean()
b = datos.groupby('Incumplimiento')['PagoAntAg'].mean()
c = datos.groupby('Incumplimiento')['PagoAntJul'].mean()
d = datos.groupby('Incumplimiento')['PagoAntJun'].mean()
e = datos.groupby('Incumplimiento')['PagoAntMay'].mean()
f = datos.groupby('Incumplimiento')['PagoAntAbr'].mean()

serie3 = pd.concat([a,b,c,d,e,f], axis = 1)
Status0 = pd.DataFrame({"Date": [6,5,4,3,2,1], "PagoAnt": [serie3.iloc[0,0],
                        serie3.iloc[0,1] ,serie3.iloc[0,2], serie3.iloc[0,3],
                        serie3.iloc[0,4], serie3.iloc[0,5]]})
Status1 = pd.DataFrame({"Date": [6,5,4,3,2,1], "PagoAnt": [serie3.iloc[1,0],
                        serie3.iloc[1,1] ,serie3.iloc[1,2], serie3.iloc[1,3],
                        serie3.iloc[1,4], serie3.iloc[1,5]]})
sns.lineplot(x = 'Date', y = 'PagoAnt', data = Status0, marker = 'o', markersize = 10,
             label = 'No incumplimiento')
```

```
sns.lineplot(x = 'Date', y = 'PagoAnt', data = Status1, label = 'Incumplimiento',
             marker = 'o', markersize = 10)

plt.fill_between(Status0['Date'], Status0['PagoAnt'], Status1['PagoAnt'],
                 interpolate = True, alpha = 0.1, label = 'Diferencia',
                 where =(Status1['PagoAnt'] != Status0['PagoAnt']))

plt.legend()
plt.show()
```

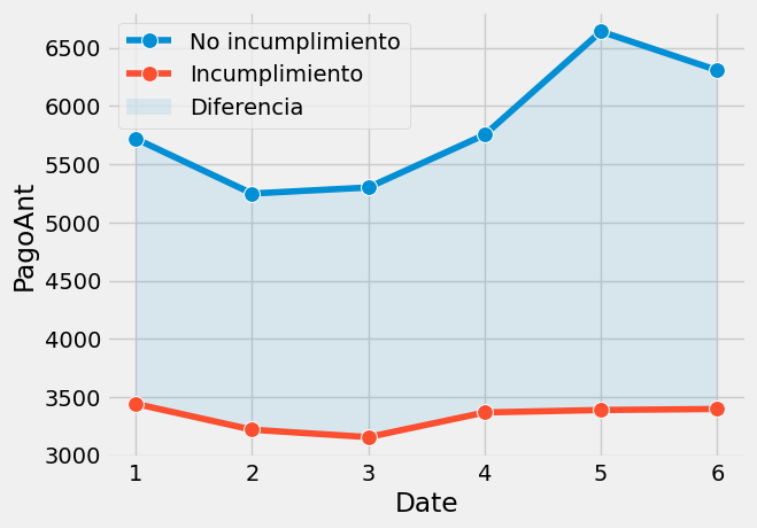


Figura 4.4: Serie de tiempo que indica el comportamiento de los pagos anticipados.

4.4. Preprocesamiento de datos

Incluir una gran cantidad de variables en el modelo puede ser contraproductivo por varias razones, como la *multicolinearidad*, por lo que es mejor conservar pocas variables con gran impacto sobre Incumplimiento. Por practicidad, sólo se propondrá un modelo para los algoritmos de aprendizaje supervisado ya que, con tantas opciones de variables y posibles transformaciones que estas pueden tener, sería innecesario exponer todas las alternativas. La selección de variables escogidas para el modelo fueron una transformación, que consiste en restar el pago de abril al pago de septiembre.

De esta forma los regresores serían la nueva variable `PagoAnt` y `StatusSept`. Los regresores y la variable dependiente se escalan para que no haya problemas con la diferencia en los valores de las categorías.

[13]:
```
datos_original = datos.copy() #Respaldamos la base de datos original
```

[16]:
```
datos['PagoAnt'] = ((datos.iloc[:,22].values) - (datos.iloc[:,17].values))
datos = datos[['StatusSept', 'PagoAnt','Incumplimiento']]

escala = MinMaxScaler()
data = escala.fit_transform(datos)
datos = pd.DataFrame(data, columns = datos.columns)
```

4.5. Conjuntos de entrenamiento y de prueba

Como se expuso en la sección correspondiente, la distribución de los datos entre el conjunto de entrenamiento y el conjunto de prueba depende del conjunto de datos con el que se trabaja. En este caso, se aplicó una división del 80 % para entrenamiento y 20 % para evaluación ya que, aunque se cuenta con un número suficientemente grande de muestras (30 000), se tiene un número escaso de incumplimientos, que es justamente lo que se quiere pronosticar.

En la función `train_test_split` se fija el parámetro `random_state = 8`, este es una semilla que permite que los resultados sean reproducibles; por su parte, `Suffle = True` revuelve de forma aleatoria el orden de los clientes en el conjunto de datos.

Continuando con el proceso, se crearon las variables `ytrain, xtrain, ytest` y `xtest`, mismas que representan a la variable dependiente *y* y a las variables independientes *x* en el conjunto de entrenamiento y en el de evaluación (*train* y *test*).

[17]:
```
#Divide el conjunto de datos en dos partes, una para entrenamiemnto y otra para␣
 ↪evaluación
datos_ent, datos_test = train_test_split(datos, test_size=.2, random_state=8,
shuffle=True)

ytrain = datos_ent['Incumplimiento']
xtrain = datos_ent.drop(['Incumplimiento'], axis = 1)
ytest = datos_test['Incumplimiento']
xtest = datos_test.drop(['Incumplimiento'], axis = 1)
```

Luego, comprobamos si la variable de interés está bien distribuida entre las dos particiones hechas.

[17]:
```
# Se obtiene porcentaje de incumplimiento en ambos conjuntos con el fin de
comprobar que se tenga la misma proporción de observaciones de interés.
a = datos_ent.Incumplimiento.value_counts(normalize = True)
b = datos_test.Incumplimiento.value_counts(normalize = True)

c = pd.concat([a,b], axis = 1)
c.columns = (['% Entrenamiento', '% Test'])
print(c)
```

```
     % Entrenamiento    % Test
0.0         0.778833  0.778667
1.0         0.221167  0.221333
```

4.6. Regresión logística

La función `LogisticRegression` es la encargada de ejecutar el modelo de regresión logística. Con el comando `fit` se ajustaron a las variables dependientes e independiente del conjunto de entrenamiento para después, con el comando `predict` obtener los resultados propuestos por el modelo para el conjunto de prueba. El comando `predict_proba` tiene una función similar pero a diferencia de la línea anterior, su objetivo no es predecir el incumplimiento sino arrojar cual es la probabilidad de infringir el pago, dando como resultado un vector de probabilidades.

[18]:
```
#Realiza la regresión
reglog = LogisticRegression( max_iter=1000).fit(xtrain, ytrain)

#Obtiene los resultados de las predicciones hechas por el modelo
```

```
ypredict = reglog.predict(xtest)
#Obtiene las probabilidades obtenidas de incumplimiento hechas por el modelo
predictproba = reglog.predict_proba(xtest)
```

Se pueden observar los resultados en la matriz de confusión en donde se aprecia que se predijo correctamente a 434 personas que iban a incumplir y a 4 506 que iban a pagar (véase la Figura 4.5). Sin embargo, 894 personas que se pronosticó que no incumplirían no hicieron su pago correspondiente y 166 que se pronosticaron que incumplirían sí pagaron.

[19]:
```
#Matriz de confusión aplicada al modelo de regresión logística
cm = pd.crosstab(ytest, ypredict, rownames=['Actual'], colnames=['Predicted'])
fig, (ax1) = plt.subplots(ncols=1, figsize=(5,4))
sns.heatmap(cm,
            xticklabels=['Not Default', 'Default'],
            yticklabels=['Not Default', 'Default'],
            annot=True,ax=ax1, fmt = 'g',
            linewidths=.1,linecolor="Darkblue", cmap="Blues")

plt.title('Confusion Matrix', fontsize=14)
plt.show()
```

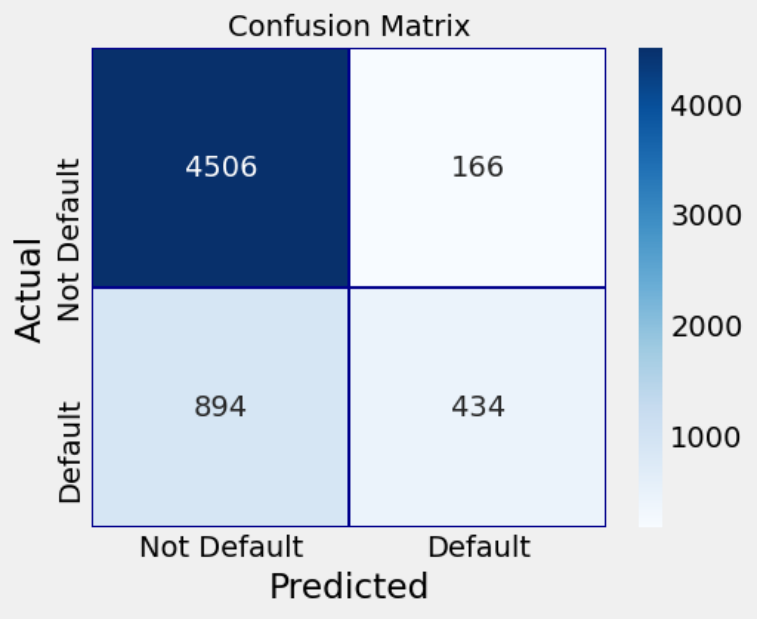


Figura 4.5: Matriz de confusión que explica los resultados de la regresión logística.

Con el siguiente código se calculan las métricas de precisión, sensibilidad y el $f1$-score que brindan información sobre el desempeño del modelo.

[20]:
```
#Despliega un reporte mostrando coeficientes de evaluación para la regresión
print(classification_report(ytest, ypredict))
```

```
              precision    recall  f1-score   support

         0.0       0.83      0.96      0.89      4672
         1.0       0.72      0.33      0.45      1328
```

```
    accuracy                           0.82      6000
   macro avg       0.78      0.65      0.67      6000
weighted avg       0.81      0.82      0.80      6000
```

A continuación, se utiliza una alternativa para la regresión logística.

[21]:
```
reg = sm.Logit(ytrain, xtrain).fit() #Forma alternativa de ejecutar la regresión
print(reg.summary()) #Presenta una tabla de resultados estadísticos de la regresión
```

```
Optimization terminated successfully.
         Current function value: 0.477093
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:         Incumplimiento   No. Observations:                24000
Model:                          Logit   Df Residuals:                    23998
Method:                           MLE   Df Model:                            1
Date:                Mon, 31 Jan 2022   Pseudo R-squ.:                 0.09707
Time:                        10:23:57   Log-Likelihood:                -11450.
converged:                       True   LL-Null:                       -12681.
Covariance Type:            nonrobust   LLR p-value:                     0.000
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
StatusSept     7.2568      0.159     45.715      0.000       6.946       7.568
PagoAnt       -4.9214      0.070    -69.872      0.000      -5.059      -4.783
==============================================================================
```

4.7. Decision Tree

En este algoritmo, primero se fijan los regresores del modelo y se ejecuta con el código `DecisionTreeClassifier`. De forma similar al método de regresión logística el comando `fit` se ajustó a las variables dependientes e independiente del conjunto de entrenamiento para después, con el comando `predict` obtener los resultados propuestos por el modelo para el conjunto creado para evaluación.

[22]:
```
regresores = ['StatusSept', 'PagoAnt'] #Define los regresores
```

[23]:
```
#Establece las especificaciones del algoritmo
arbol = DecisionTreeClassifier(class_weight='balanced', max_depth=5)
#Ejecuta el algoritmo con las variables especificadas
arbol = arbol.fit(datos_ent[regresores], datos_ent['Incumplimiento'])
```

[24]:
```
#Devuelve las predicciones hechas por el modelo
preds = arbol.predict(datos_test[regresores])
```

Después se observan los resultados en la matriz de confusión donde se pronosticaron correctamente 796 personas que incumplirían y 3 629 que pagarían (véase la Figura 4.6).

[25]:
```
#Matriz de confusión aplicada al modelo de árboles de decisión
cm2 = pd.crosstab(datos_test['Incumplimiento'], preds, rownames=['Actual'],
                  colnames=['Predicted'])
fig, (ax1) = plt.subplots(ncols=1, figsize=(5,4))
```

```
sns.heatmap(cm2,
            xticklabels=['Not Default', 'Default'],
            yticklabels=['Not Default', 'Default'],
            annot=True,ax=ax1, fmt = 'g',
            linewidths=.1,linecolor="Darkblue", cmap="Blues")

plt.title('Confusion Matrix', fontsize=14)
plt.show()
```

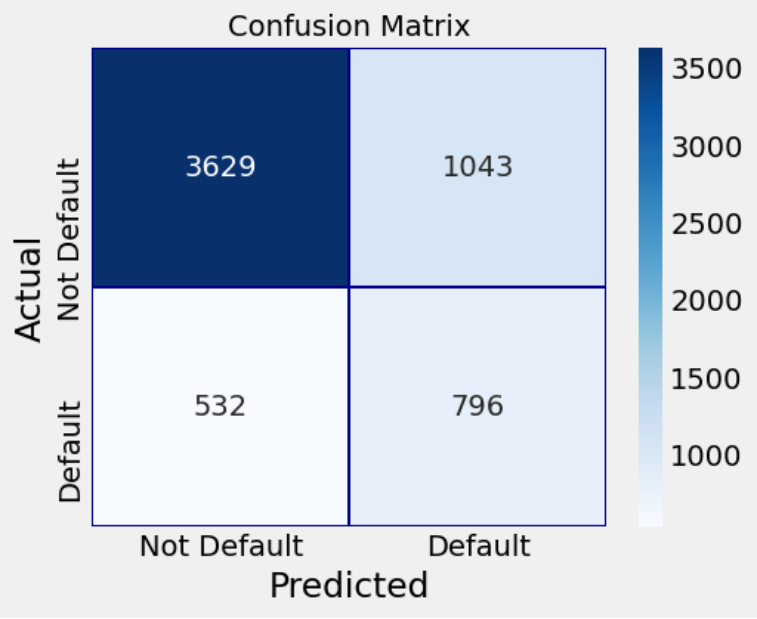


Figura 4.6: Matriz de confusión que explica los resultados del árbol de decisión

Los resultados del bloque de código que sigue se pueden observar en el anexo A en las figuras A.1, A.2, A.3 y A.4.

[ ]:
```
#Imprime la representación gráfica del árbol de decisión
dot_data = tree.export_graphviz(arbol, out_file =None, filled = True, rounded = True,
                                class_names = ['Paga', 'No Paga'],
                                feature_names = datos_ent[regresores].columns)
graph = graphviz.Source(dot_data)
graph
```

Como complemento, la siguiente línea de código es capaz de proveer una ejecución de varios modelos de aprendizaje supervisado mostrando los resultados obtenidos para así, de una manera rápida, obtener una idea de qué modelos podrían ser mejores. Al correr el código dio como resultado que el mejor modelo utilizando todos los regresores de la base de datos `datos_original` era el de regresión logística.

[28]:
```
#Crear varios modelos de aprendizaje automático
Incumplimiento = dabl.SimpleClassifier().fit(datos_original,
                                target_col = "Incumplimiento")
```

```
Running DummyClassifier()
```

```
accuracy: 0.779 average_precision: 0.221 roc_auc: 0.500 recall_macro: 0.500
f1_macro: 0.438
=== new best DummyClassifier() (using recall_macro):
accuracy: 0.779 average_precision: 0.221 roc_auc: 0.500 recall_macro: 0.500
f1_macro: 0.438

Running GaussianNB()
accuracy: 0.288 average_precision: 0.239 roc_auc: 0.546 recall_macro: 0.528
f1_macro: 0.274
=== new best GaussianNB() (using recall_macro):
accuracy: 0.288 average_precision: 0.239 roc_auc: 0.546 recall_macro: 0.528
f1_macro: 0.274

Running MultinomialNB()
accuracy: 0.802 average_precision: 0.513 roc_auc: 0.751 recall_macro: 0.675
f1_macro: 0.688
=== new best MultinomialNB() (using recall_macro):
accuracy: 0.802 average_precision: 0.513 roc_auc: 0.751 recall_macro: 0.675
f1_macro: 0.688

Running DecisionTreeClassifier(class_weight='balanced', max_depth=1)
accuracy: 0.780 average_precision: 0.367 roc_auc: 0.686 recall_macro: 0.686
f1_macro: 0.684
=== new best DecisionTreeClassifier(class_weight='balanced', max_depth=1) (using
recall_macro):
accuracy: 0.780 average_precision: 0.367 roc_auc: 0.686 recall_macro: 0.686
f1_macro: 0.684

Running DecisionTreeClassifier(class_weight='balanced', max_depth=5)
accuracy: 0.754 average_precision: 0.514 roc_auc: 0.759 recall_macro: 0.697
f1_macro: 0.676
=== new best DecisionTreeClassifier(class_weight='balanced', max_depth=5) (using
recall_macro):
accuracy: 0.754 average_precision: 0.514 roc_auc: 0.759 recall_macro: 0.697
f1_macro: 0.676

Running DecisionTreeClassifier(class_weight='balanced',
min_impurity_decrease=0.01)
accuracy: 0.780 average_precision: 0.367 roc_auc: 0.686 recall_macro: 0.686
f1_macro: 0.684

Running LogisticRegression(C=0.1, class_weight='balanced', max_iter=1000)
accuracy: 0.771 average_precision: 0.538 roc_auc: 0.769 recall_macro: 0.706
f1_macro: 0.690
=== new best LogisticRegression(C=0.1, class_weight='balanced', max_iter=1000)
(using recall_macro):
accuracy: 0.771 average_precision: 0.538 roc_auc: 0.769 recall_macro: 0.706
f1_macro: 0.690

Running LogisticRegression(class_weight='balanced', max_iter=1000)
accuracy: 0.769 average_precision: 0.537 roc_auc: 0.767 recall_macro: 0.705
f1_macro: 0.689

Best model:
```

```
LogisticRegression(C=0.1, class_weight='balanced', max_iter=1000)
Best Scores:
accuracy: 0.771 average_precision: 0.538 roc_auc: 0.769 recall_macro: 0.706
f1_macro: 0.690
```

4.8. Análisis

Para finalizar queda responder la pregunta, ¿cómo saber si este modelo puede traer beneficios para la compañía? Para ilustrar los resultados se necesita tener en cuenta varios factores que se expondrán a continuación. Para esta comparación se tomarán como supuestos que los 30 mil clientes deben exactamente la misma cantidad ($10 000) y que en caso de caer en incumplimiento se perderá todo el monto que ellos deben, es decir, no existirá un monto de recuperación. Esto por supuesto es diferente en la práctica, ya que el monto que se pierde es una fracción correspondiente a la deuda y a pesar de caer en incumplimiento se toma en cuenta un posible porcentaje de recuperación. Otro supuesto que se tomará en cuenta será una matriz de gastos, lo que explicará el costo perdido o ganado gracias a cada pronóstico acertado o errado. El último supuesto consiste en omitir los gastos de operación que el cobro de estos créditos puede suponer.

Teniendo en cuenta la matriz de confusión hipotética de la Figura 4.7 se tiene lo siguiente: primero, en la columna de la izquierda se ven todas las personas que el modelo pronosticó que pagarían. Las personas que efectivamente pagaron provocan una pérdida de 0 para la empresa, mientras que las personas que no pagaron provocan una pérdida de $10 000. En la segunda columna están las personas que el modelo proyectó que incumplirían su pago. Al ver que incumplirían la institución tomó la decisión de una estrategia para prevenir el incumplimiento la cual consistía en la condonación del 10 por ciento de su deuda por lo que las personas pronosticadas a incumplimiento que efectivamente no iban a pagar se espera que tenga un comportamiento diferente provocando que el 90 por ciento de estas personas continúen con sus pagos. Esto significa que el 10 por ciento a pesar del descuento de su crédito seguiría faltando a su pago por lo que se tiene una pérdida esperada adicional del 10 por ciento que corresponderían ahora a $900 pesos, concluyendo una pérdida de $1900 pesos por cada persona pronosticada a no pagar que efectivamente no iba a pagar. Para terminar las personas que sí cumplieron pagaron $1000 menos provocando una pérdida por dicho monto.

[29]:
```
#Matriz de confusión hipotética
costos = np.array([[0, -1000], [-10000, -1900]])
fig, (ax1) = plt.subplots(ncols=1, figsize=(5,4))
sns.heatmap(costos,
            xticklabels=['Not Default', 'Default'],
            yticklabels=['Not Default', 'Default'],
            annot=True,ax=ax1, fmt = 'g',
            linewidths=.1,linecolor="Darkblue", cmap="Blues")

plt.title('Matriz de costos', fontsize=14)
plt.show()
```

Como se puede observar, por 6 mil personas con una deuda de $10 000 pesos se espera perder el 22 por ciento correspondiente a $13 200 000 pesos. Si se aplica la estrategia de prevención utilizando el modelo de regresión logística o el de árbol de decisión la pérdida bajaría a $9 930 600 o a $7 953 000 representando ahora el 16.6 o el 13.3 por ciento, respectivamente. Suponiendo que la estrategia fuera efectiva, por cada 6 mil personas se estaría ahorrando de 3 a 5 millones de pesos.

[30]:
```
# Se multiplica el préstamos promedio por el número de personas en el universo
-10000*6000*.22
```

[30]:
```
-13200000.0
```

[31]:
```
# Se obtiene cuanto sería la pérdida esperada al multiplicar los valores obtenidos
de la matriz de confusión de la regresión logística por la matriz de costos
costosreglog = cm * costos
costosreglog.sum().sum()
```

[31]: -9930600

[32]:
```
# Se obtiene cuanto sería la pérdida esperada al multiplicar los valores obtenidos
de la matriz de confusión del árbol de decisión por la matriz de costos
costosdectree = cm2 * costos
costosdectree.sum().sum()
```

[32]: -7875400

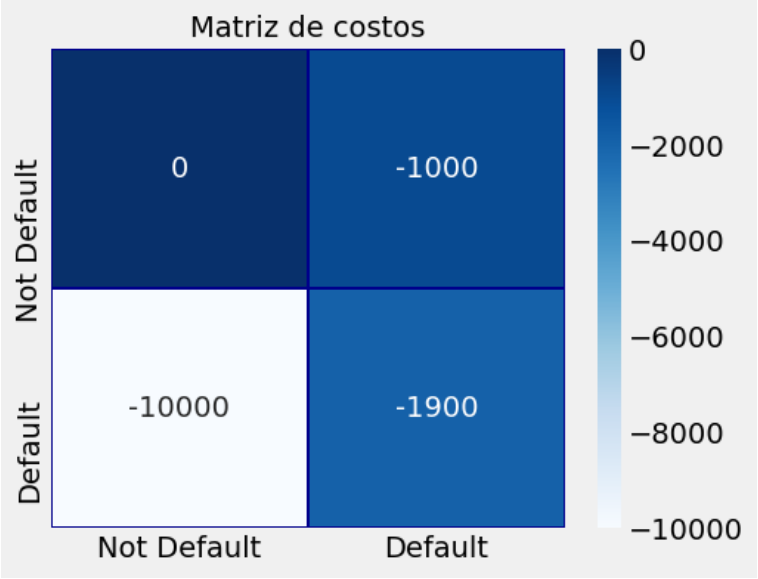


Figura 4.7: Matriz hipotética que explica los costos de la compañía por cada persona que fue correcta o incorrectamente pronosticada.

4.9. Resultados

En resumen, se pueden destacar varios puntos. Análogo al Capítulo 3, se puede observar la importancia del análisis exploratorio previo para tener una idea de las variables o las estrategias que puedan dar mejores resultados, en la práctica esto puede ahorrar tiempo en la implementación de modelos innecesarios para mantener la coherencia y el sentido de los resultados. En esta ocasión en particular, se contaban con numerosas variables y gracias al análisis se pudieron descartar la mayoría de ellas.

De igual modo que en el Capítulo 3 se puede agradecer en gran medida la existencia de las herramientas computacionales utilizadas ya que la implementación de este análisis resultó sumamente sencilla en comparación de haberla hecho sin este apoyo.

Por último, cabe resaltar que a pesar de haber construido un modelo relativamente simple, los resultados pueden representar mejoras en millones de pesos, esto a gran escala definitivamente puede hacer la diferencia entra una institución financieramente saludable y otra que no lo es. También se puede apreciar que a pesar de que estadísticamente no pueda parecer el mejor modelo posible, se tiene que comprender el contexto y la traducción económica que la implementación pueda tener en los resultados, si un ahorro

de varios millones se logró con un modelo modesto, vale la pena preguntarse ¿qué se podrá lograr con un modelo más complejo?

Conclusiones

En un principio, se inició con el conocimiento de la utilización de la administración de riesgos en las instituciones financieras y del uso de la toma de decisiones en estas mismas empresas. Al final de este trabajo es posible tener una conciencia más profunda sobre la importancia de dichas actividades, no sólo siendo relevantes para las instituciones, sino también para la sociedad. Podemos decir con toda seguridad que la implementación de las nuevas estrategias de mitigación de riesgos son acertadas ya que ahora toman en cuenta tanto el pasado histórico como el pronóstico de posibles escenarios adversos que puedan existir. De igual forma la mayor cantidad de dinero que se tiene que reservar para hacer frente a las obligaciones financieras ha provocado una mayor estabilidad en la economía.

Así mismo, podemos asegurar que la forma de obtener conocimiento y de realizar procesos computacionales ha mejorado de forma gratamente impactante en los últimos años. Sin embargo, queda claro que esto no es una tarea sencilla ya que está formada de una series de pasos y análisis que deben ser tomados en cuenta para mejores resultados y, en caso de usarse correctamente, los frutos pueden llegar a ser muy satisfactorios. Principalmente, se puede resaltar la relevancia que tienen los pasos previos a los modelos para tener resultados con sentido y útiles ya que si no se tiene buena calidad de los datos y un buen análisis que les dé propósito, los modelos de aprendizaje automático pueden llegar a ser intrascendentes. Por lo tanto, la detección de valores faltantes, valores duplicados, valores atípicos así como la implementación de la estandarización de los datos y su reducción se convierten en los temas principales para unos resultados acertados.

El objetivo de este trabajo, fue exponer cómo el aprendizaje automático es utilizado en los procesos de las instituciones financieras en donde se pudo observar cómo una de las técnicas más comunes del aprendizaje automático, la regresión logística, es una pieza fundamental para las empresas de este tipo y cómo para ciertos casos otras técnicas pueden ser más acertadas, como fue el árbol de decisión. También se presentaron técnicas de agrupamiento, específicamente K-Medias y DBSCAN las cuáles a pesar de no parecer tan cruciales traen consigo mejoras económicas notables para el negocio. Se concluye que tomar ventaja de estos modelos trae beneficios tanto operacionales como monetarios ya que permiten que la parte operativa en la obtención de conocimientos sea más objetiva que el de un analista, para que así, esta persona pueda concentrar todos sus esfuerzos en analizar los resultados y ejecutar las decisiones.

Como conclusión adicional, este trabajo me permitió observar uno de los numerosos ejemplos que existen entre las combinaciones de conocimientos que la carrera de actuaría puede brindar, fusionando habilidades como matemáticas, estadística, programación y de riesgos, dando como resultado un tema sumamente interesante y útil para el correcto funcionamiento de la economía nacional e internacional y, a mi particular punto de vista, uno de los campos más interesantes para aprender. Hay veces en que especialistas en determinados temas son requeridos y comúnmente son vistos como personas capaces de hacer aportaciones inmensas para las empresas, pero una persona con conocimientos generales en distintos temas es igual de capaz para crear y generar valor en las distintas prácticas actuariales.

Apéndice A

Árbol de decisión

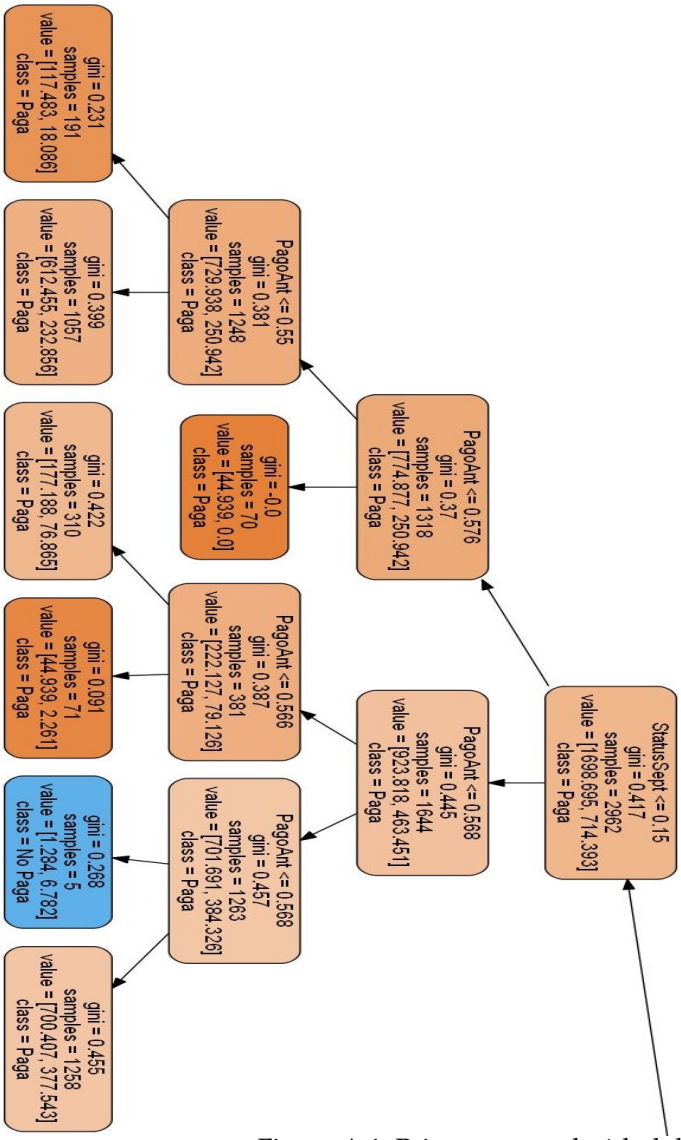


Figura A.1: Primera parte de árbol de decisión

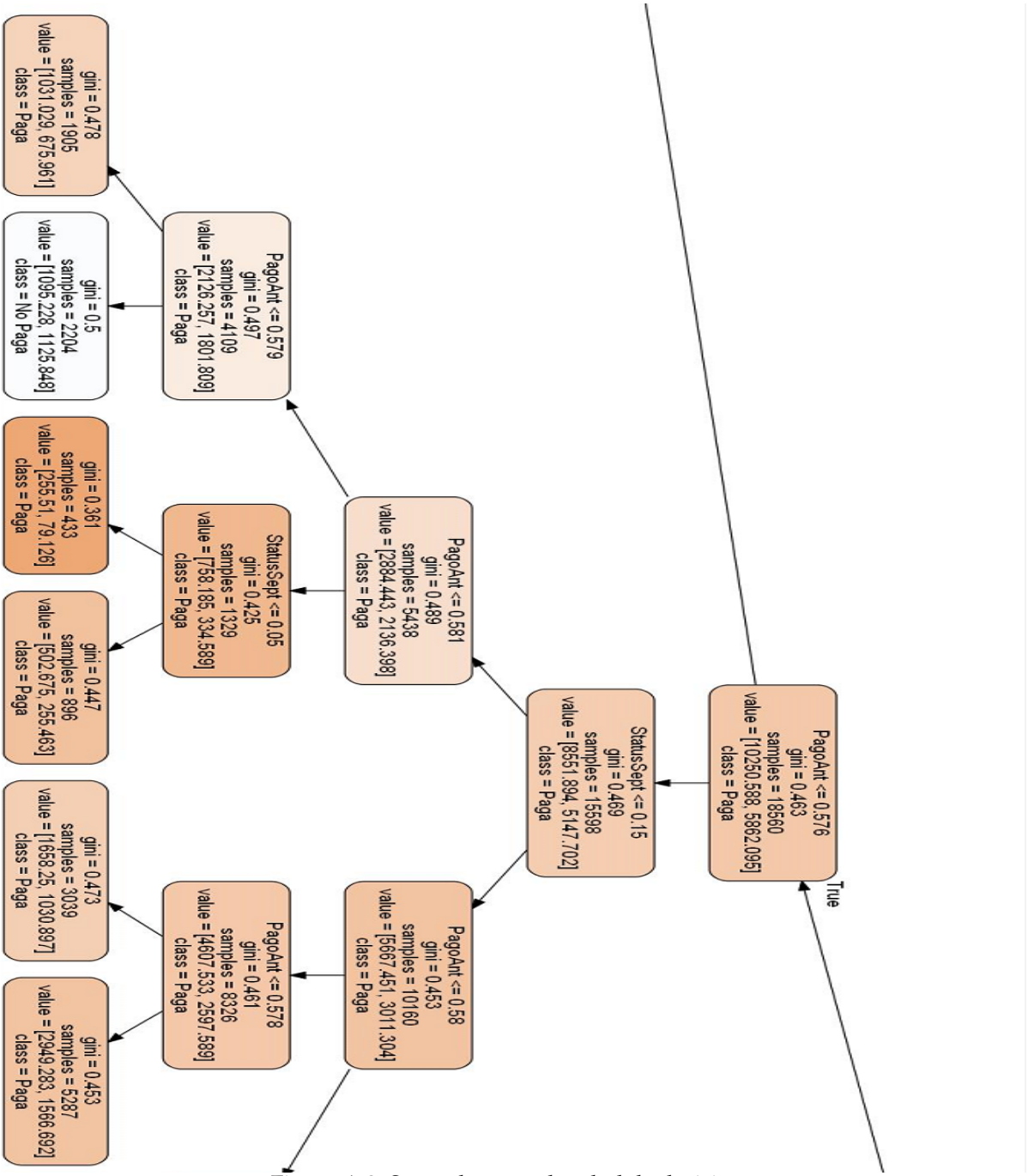


Figura A.2: Segunda parte de árbol de decisión

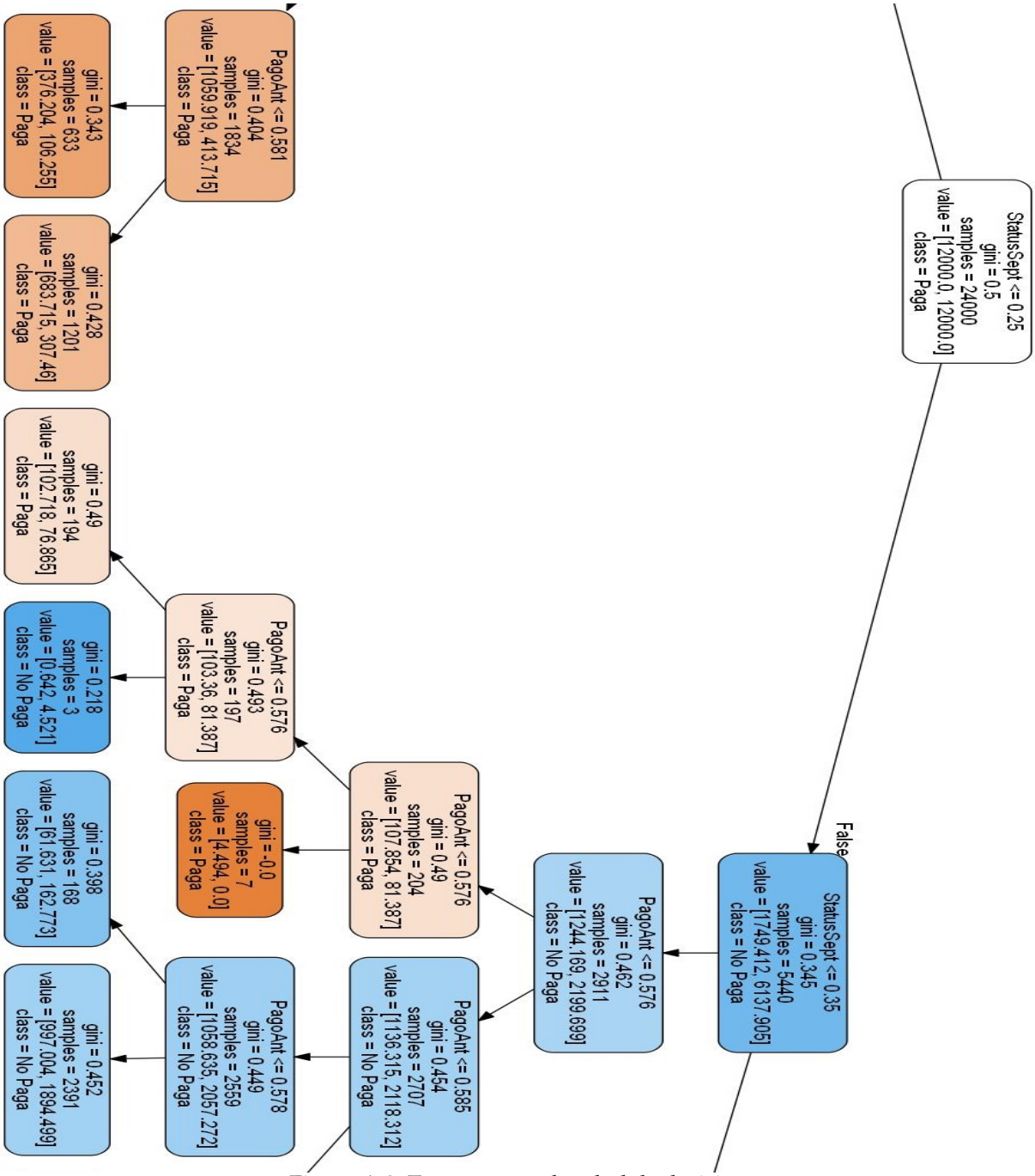


Figura A.3: Tercera parte de árbol de decisión

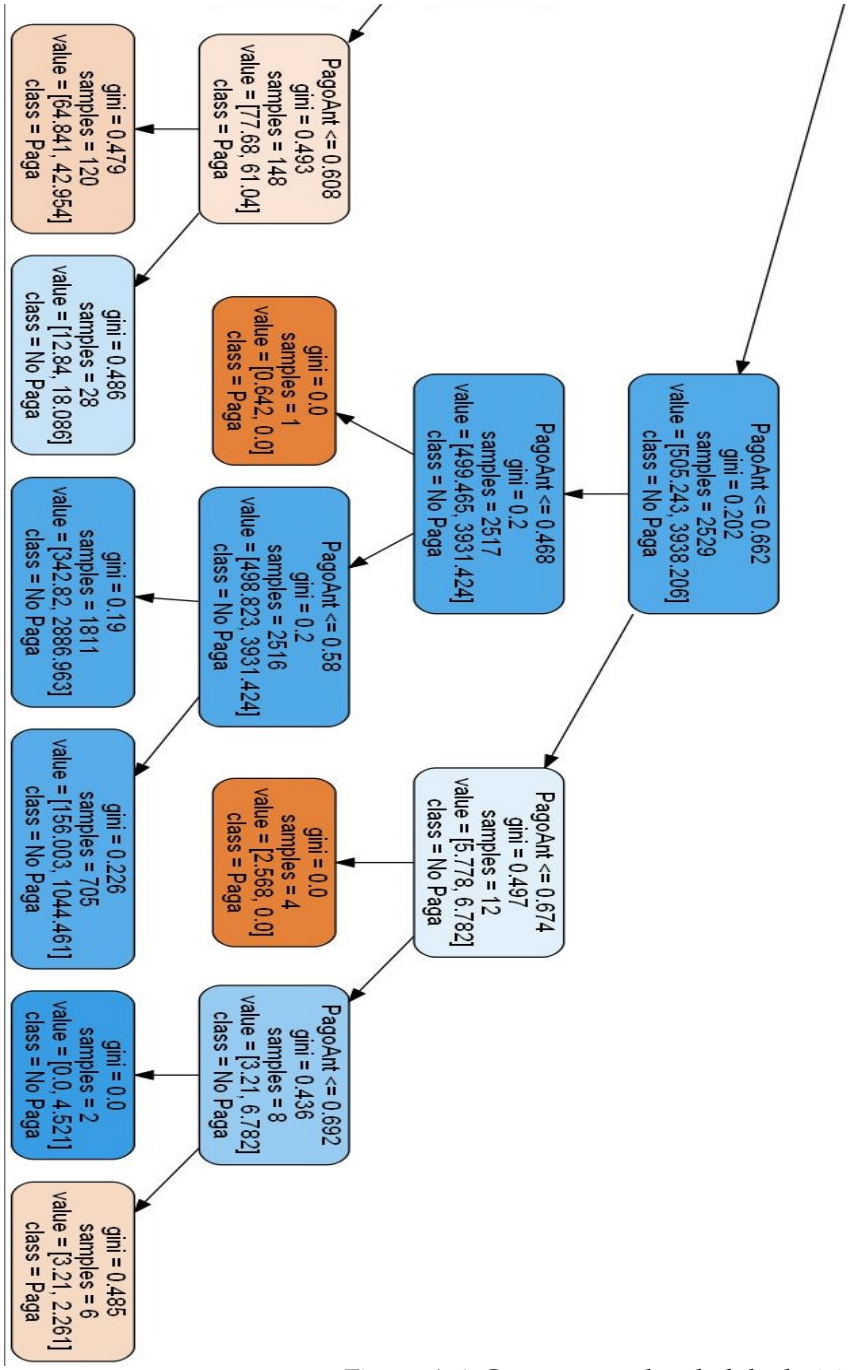


Figura A.4: Cuarta parte de árbol de decisión

Apéndice B

Entorno de trabajo

La presente tesis desarrolló código de programación en lenguaje Python[1], un lenguaje potente y a la vez muy accesible, el más utilizado dentro del aprendizaje automático y el análisis de datos debido a la gran cantidad de bibliotecas y tecnologías que tiene disponibles. En este trabajo se utilizó la versión 3.8.8 de la rama de Python 3.

La instalación de Python se hizo con Anaconda[2], antes Continuum Analytics, Anaconda es una distribución gratuita de Python multiplataforma que incluye todos los paquetes esenciales para aprendizaje automático y ciencia de datos. Para instalar Anaconda basta con acceder a la página web y descargar el ejecutable correspondiente (ver Figura B.1).

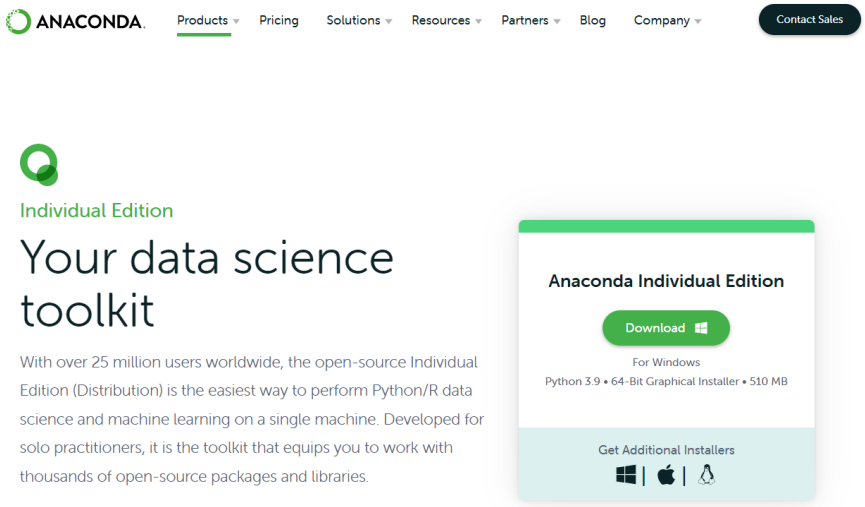


Figura B.1: Página de descarga de Anaconda.

Cuando se ejecuta la aplicación despliega una ventana similar a la que se muestra en la Figura B.2.

[1] Véase https://www.python.org [Consultado: 01/11/2021]
[2] Véase https://www.anaconda.com [Consultado:01/11/2021]

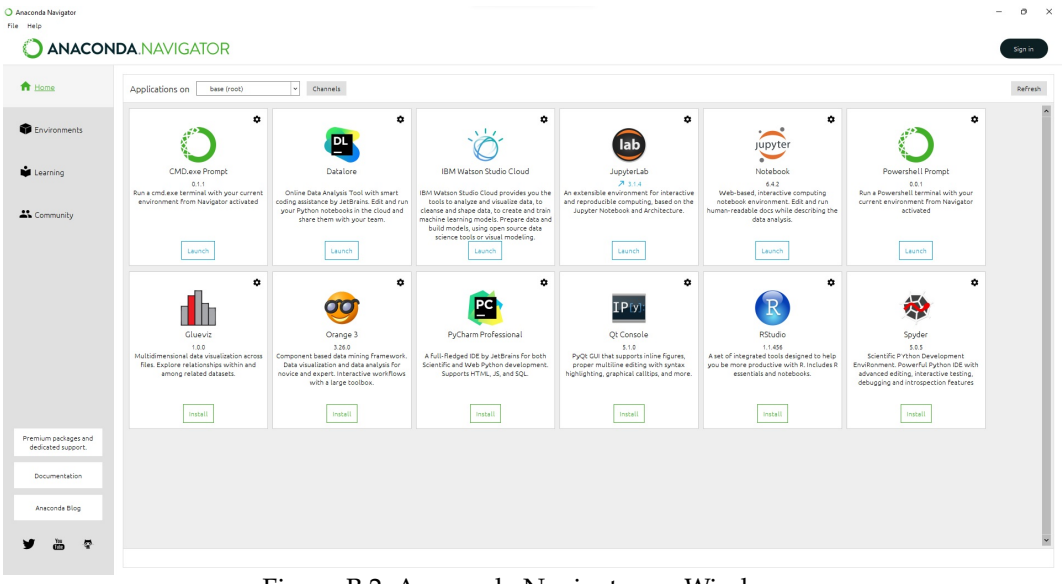


Figura B.2: Anaconda Navigator en Windows.

Por otro lado, la ejecución de los códigos propuestos en este trabajo fue a través de Jupyter Notebook[3], uno de los módulos que por defecto instala Anaconda. Esta aplicación puede ejecutar el código paso a paso y tener todas las salidas resultantes -incluyendo gráficas- junto con el código. El entorno de Jupyter se muestra en la figura B.3.

Figura B.3: Página principal del entorno Jupyter.

Para garantizar que los ejemplos de código en esta tesis funcionan correctamente, la versión de los paquetes debe ser igual o superior a las versiones listadas en las tablas 3.1 y 4.1.

[3]Véase https://jupyter.org [Consultado 01/11/2021]

Referencias

Badia Contelles, F. e. Q. B. R. J. e. B. C. J. M., José Manuel (ed.) Pla Bañón. (s.f.). *Métodos informativos avanzados*.

Banco de México. (2005). Definiciones básicas de riesgos. *México: Banco de México*.

Caliński, T., y Harabasz, J. (1974). A dendrite method for cluster analysis. *Communications in Statistics-theory and Methods*, *3*(1), 1–27.

Comisión Nacional Bancaria y de Valores. (2021). Disposiciones de carácter general aplicables a las instituciones de crédito.

Comité De Basilea. (1999). Principios para la administración del riesgo de crédito. *Washington DC*.

D. Comaniciu, P. Meer. (2002). Mean shift: A robust approach toward feature space analysis.

Davies, D. L., y Bouldin, D. W. (1979). A cluster separation measure. *IEEE transactions on pattern analysis and machine intelligence*(2), 224–227.

Hull, J. (2012). *Risk management and financial institutions,+ web site* (Vol. 733). John Wiley & Sons.

Instituto Nacional de Estadística y Geografía (INEGI). (2022). *Índice nacional de precios al consumidor (inpc).* Descargado de https://www.inegi.org.mx/temas/inpc/

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations.

Martin. E., Kriegel, H., Sander, J., Xu, X. (1996). A density based algorithm for discovering clusters in large spatial databases with noise.

Raschka, S. (2015). *Python machine learning*. Packt publishing ltd.

Real Academia Española . (2001). *Diccionario de la lengua española*. Real academia española Madrid.

Rousseeuw, P. J. (1987). Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, *20*, 53–65.

Scikit. (2021). Descargado de https://scikit-learn.org/stable/

Torres, J. (2018). *Deep learning introducción práctica con keras*. Independently published.

Torres, J. (2021). *Deep reinforcement learning explained*. Independently published.

Vercellis, C. (2011). *Business intelligence: data mining and optimization for decision making*. John Wiley & Sons.