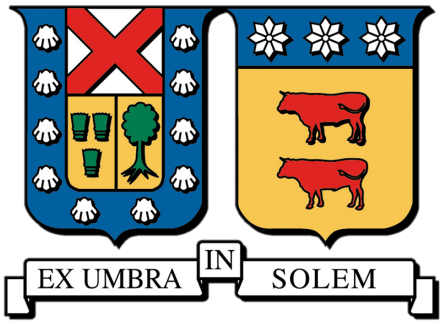
UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA
DEPARTAMENTO DE ELECTRÓNICA
VALPARAÍSO - CHILE



"SISTEMA HÍBRIDO DE RECOMENDACIÓN DE ROLES BASADO EN FILTRADO COLABORATIVO Y MODELOS DE CLASIFICACIÓN"

ENRIQUE EDUARDO ESCALONA VILLARREAL

MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL TELEMÁTICO

PROFESOR GUÍA:	**MAURICIO ARAYA**
PROFESOR CORREFERENTE:	**PATRICIO OLIVARES**

MARZO 2026

CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título; ☐ Tesis de Postgrado;

Título del trabajo: SISTEMA HIBRIDO DE RECOMENDACION DE ROLES BASADO EN FILTRADO COLABORATIVO Y MODELOS DE CLASIFICACION

Nombre del candidato(a): Enrique Eduardo Escalona Villarreal

Carrera / Grado: Ingeniería Civil Telemática

Campus: Casa Central Valparaiso **;** **Departamento**: Electrónica

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Mauricio Araya , en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL

☒ El trabajo **NO contiene información que amerite confidencialidad** y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (embargo) por:

☐ 6 meses; ☐ 12 meses; ☐ 2 años; ☐ 3 años; ☐ 5 años; ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 6/4/26 ; **Firma:** _____

Estudiante o Candidato(a):

Fecha: 6/4/26 ; **Firma:** _____

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.

Agradecimientos

Quiero expresar mi más sincero agradecimiento a todas las personas que formaron parte de este proceso. Sin su ayuda y apoyo, no creo que este logro hubiera sido posible.

A mis padres, Carmen Luz y Luis, gracias por su apoyo incondicional. Mamá, gracias a que dejaste muchas cosas de lado para centrarte en criarnos a mí y a mis hermanas, soy la persona que soy hoy en día. Papá, gracias por siempre esforzarte al máximo por nosotros y por ser mi modelo a seguir. Ver lo contentos y orgullosos que están cuando hablan de mis logros era una de las cosas que más me alegraban y motivaban para seguir esforzándome al máximo durante este proceso. Gran parte de este logro es gracias a ustedes, gracias por ser los mejores padres del mundo.

A mis hermanas, Isidora e Ignacia. A pesar de que me cueste admitirlo en persona y no lo demuestre mucho, su compañía hace que mi día a día sea mucho más alegre y llevadero.

A Javier Martínez, uno de mis mejores amigos y compañero en la gran mayoría de los trabajos que realicé durante esta etapa. Gracias por todas las horas que pasamos jugando y estudiando prácticamente al mismo tiempo, no creo que hubiera sido posible hacer eso y que me fuera igual de bien con otra persona.

A mis amigos y compañeros de carrera: Ignacio, Diego, Javier Zamora, Gabriel, Cristóbal y Benjamín. Su compañía convirtió momentos que deberían haber sido estresantes en recuerdos llenos de alegría. Gracias por su constante apoyo en esta etapa.

A mi profesor guía, Mauricio Araya, por su constante apoyo y orientación, los cuales fueron fundamentales para el desarrollo de este trabajo.

Finalmente agradecer a la empresa CMPC, por levantar este desafío y permitirme realizar este trabajo de titulación con ustedes.

Resumen

En organizaciones de gran envergadura como CMPC, la gestión del control de accesos y la asignación de roles operativos en sistemas complejos como SAP suele ser un proceso manual, reactivo y propenso a generar cuellos de botella. Ante la restricción de interactuar directamente con la plataforma transaccional de la empresa, el presente trabajo propone el diseño y desarrollo de un motor predictivo basado en filtrado colaborativo para automatizar la sugerencia de roles, operando sobre un extracto estático de datos correspondientes a las sucursales de Brasil.

El principal desafío técnico del proyecto radicó en modelar matemáticamente la similitud entre trabajadores. Para ello, se diseñó una arquitectura híbrida en cascada. Inicialmente, se aplicó Análisis de Componentes Principales con Kernel (KPCA) para la extracción de características no lineales y, posteriormente, se combinó esta representación latente con una métrica de similitud para generar una lista de recomendaciones candidatas.

Dado que esta generación introducía una alta tasa de falsos positivos, se implementó una segunda etapa algorítmica consistente en clasificadores de Gradient Boosting basados en árboles de decisión. Esta última capa actuó como un filtro de precisión, logrando discriminar el ruido y reduciendo el volumen de recomendaciones a una escala operativamente viable.

El resultado final de este trabajo es una herramienta que no solo disminuye el número de solicitudes formales, sino que optimiza los tiempos de respuesta del departamento y sienta las bases para una gestión de accesos inteligente y automatizada.

Índice de figuras

Índice de cuadros

Índice general

Capítulo 1

Introducción

CMPC es una empresa chilena con presencia nacional e internacional, enfocada en el negocio forestal y papelero. Debido a su tamaño y complejidad operativa, esta empresa utiliza SAP (*Systems, Applications, and Products in Data Processing*) como plataforma para centralizar el gran flujo de información que genera en sus distintos procesos.

El uso de SAP requiere una rigurosa gestión de los accesos dentro de la plataforma, motivo por el cual existe un equipo dedicado a controlar los accesos de los distintos trabajadores dentro del sistema. Este equipo tiene la responsabilidad de asignar, modificar y eliminar roles de usuarios en SAP, donde cada rol contiene una serie de permisos que permiten a los empleados cumplir con sus funciones laborales. Por este motivo, este departamento tiene que realizar la asignación de estos roles de manera correcta, para así asegurar no solo que los trabajadores cuenten con los accesos necesarios para desempeñar sus labores, sino también protegiendo la integridad de la información y asegurando el correcto funcionamiento de la plataforma SAP de la empresa.

1.1. Problema a resolver

En un entorno con miles de empleados como el de CMPC, es común que algunos trabajadores no cuenten con todos los roles necesarios para realizar sus funciones. Cuando esto ocurre, deben enviar una solicitud al departamento de control de accesos para que se le asig-

nen los permisos necesarios.

Actualmente, este proceso se realiza de forma manual, lo que conlleva una alta carga operativa para el equipo, especialmente considerando la gran cantidad de solicitudes recibidas diariamente. Esta situación genera cuellos de botella lo que provoca que se retrase la habilitación a los usuarios. Por lo tanto, surge la necesidad de automatizar el proceso de asignación de roles con el objetivo de agilizar este proceso y reducir el numero de solicitudes que recibe el departamento.

Al inicio del proyecto, la empresa se encontraba en pleno proceso de migración hacia un nuevo sistema de gestión de solicitudes. Sumado a esto, las políticas de seguridad y la complejidad logística impidieron otorgar credenciales de acceso directo a la plataforma SAP a personal externo a la compañía. Por consiguiente, el departamento optó por proporcionar un extracto estático de la base de datos, el cual contenía la configuración histórica de los trabajadores y sus roles asignados al momento de la extracción.

Ante este contexto restrictivo, se decidió abordar la problemática mediante un enfoque predictivo. Para ello, el equipo de trabajo tomó la decisión conjunta de desarrollar una plataforma externa e independiente de los sistemas de la empresa. La herramienta consiste en una aplicación web diseñada para que los miembros del departamento de gestión de accesos realicen un análisis exhaustivo de los datos. El flujo de esta solución comienza con la ingesta manual de la información estructurada por parte del usuario, una vez procesados los datos, el sistema ejecuta una predicción de los posibles permisos faltantes de los trabajadores y, finalmente, despliega estas recomendaciones en la interfaz para su revisión y validación por parte del equipo.

Si bien la construcción de dicha plataforma interactiva constituyó la totalidad del desafío grupal, el alcance de la presente trabajo se centra exclusivamente en el diseño y desarrollo del motor predictivo. Este núcleo analítico adopta un enfoque basado en la estimación de similitud, por lo que el desafío principal de esta investigación radica en la definición de una métrica de afinidad efectiva. Un agrupamiento básico por jerarquías o equipos directos resulta insuficiente dada la complejidad y especificidad de los roles en SAP. Por consiguiente, la problemática técnica a resolver es cómo modelar matemáticamente la similitud entre usuarios

para detectar patrones latentes de comportamiento, permitiendo inferir permisos necesarios que no son evidentes a través de una simple comparación administrativa.

1.2. Acercamiento a la solución

Para abordar esta problemática, se dispone de un conjunto de datos proveniente de las sucursales de CMPC en Brasil. Esta información estructura el perfil de los trabajadores incluyendo su cargo, departamento y los roles asignados.

La estrategia de solución se fundamenta en dos etapas principales. En primer lugar, en lugar de realizar comparaciones rígidas por cargo, se implementará un análisis de similitud vectorial. Esto permitirá agrupar dinámicamente a los trabajadores que tienen patrones de comportamiento y necesidades de acceso similares. A partir de estos grupos, el sistema inferirá los posibles roles faltantes comparando los roles del usuario objetivo contra los de sus vecinos del grupo generado.

En segundo lugar, para garantizar la seguridad y el cumplimiento de la lógica de negocio, se desarrollará un módulo de filtrado inteligente. Dado que existen roles incompatibles o restringidos para ciertas funciones, no basta con la similitud; por lo tanto, es necesario validar la viabilidad de la asignación. Este filtro aprenderá de los datos históricos de asignaciones para descartar sugerencias inválidas, asegurando que las recomendaciones presentadas al equipo sean de alta calidad y se adapten a la evolución de la empresa.

Finalmente, debido a la naturaleza crítica de la información y los riesgos de seguridad asociados a la plataforma SAP, el sistema se desarrollará como una herramienta de apoyo a la toma de decisiones y no como un sistema de asignaciones automático. El objetivo es generar una recomendación priorizada que facilite la labor del equipo de gestión de accesos, manteniendo siempre la validación final bajo supervisión humana para evitar asignaciones erróneas o violaciones de políticas de seguridad.

1.3. Objetivos

1.3.1. Objetivo General

Optimizar el proceso de asignación de roles y permisos en la plataforma SAP, a través de un sistema de recomendación predictivo que permita anticipar las necesidades de los usuarios y reducir la carga operativa del equipo de Gestión de Accesos.

1.3.2. Objetivos Específicos

- **Objetivo Especifico 1:** Implementar un modelo de similitud que permita identificar relaciones complejas entre los trabajadores, con la finalidad de inferir roles faltantes en los usuarios objetivo.
- **Objetivo Especifico 2:** Desarrollar un filtro inteligente que valide las recomendaciones generadas, asegurando el cumplimiento de la lógica de negocio.
- **Objetivo Especifico 3:** Validar el desempeño del sistema propuesto utilizando métricas de evaluación técnica y pruebas retrospectivas, contrastando los resultados con asignaciones reales.

Capítulo 2

Estado del Arte y de la Técnica

Este capítulo presenta los fundamentos teóricos y la revisión del estado del arte que sustentan el diseño y desarrollo del sistema propuesto. Para abordar la problemática de manera estructurada, la investigación se ha subdividido en tres ejes fundamentales. En primer lugar, se expone el contexto actual de los sistemas de Control de Accesos Basado en Roles (RBAC) y la disciplina de *Role Mining*. En segundo lugar, se analizan los paradigmas y arquitecturas para el modelamiento y construcción de Sistemas de Recomendación. Finalmente, la tercera sección profundiza en las justificaciones matemáticas y algorítmicas necesarias para procesar los datos de entrada, abordando los desafíos específicos que presenta el modelado de variables de naturaleza categórica y altamente dispersa.

2.1. Control de Accesos Basado en Roles (RBAC) y *Role Mining*

El Control de Accesos Basado en Roles (RBAC) es un modelo que controla el acceso asignando roles apropiados a los usuarios. En lugar de asignar los permisos directamente a cada individuo, en RBAC se introduce el concepto de roles para descomponer esta estructura en dos relaciones independientes: la asignación de usuarios a roles específicos y la asignación de una serie de permisos a dichos roles. Esta descomposición facilita la administración de la

seguridad, especialmente en organizaciones que cuentan con miles de usuarios y permisos [1].

A pesar de los beneficios operativos de este modelo, su adopción conlleva el desafío de transformar los permisos existentes de la empresa en un conjunto coherente de roles. El proceso de generar roles se conoce como *Role Engineering* [2]. Esta disciplina es necesaria para desarrollar los diversos componentes de RBAC, como las jerarquías de roles, los permisos y las restricciones. La literatura divide esta practica en dos enfoques *top-down* y *bottom-up* [3]. El enfoque *top-down* requiere contratar expertos para analizar los requisitos del negocio basándose en su experiencia, lo cual exige mucha mano de obra y tiempo. Por el contrario, el enfoque *bottom-up*, también conocido como *Role Mining*, utiliza la minería de datos para generar y asignar roles basándose en listas de control de acceso, registros del sistema e información del negocio.

Recientemente, la investigación en *Role Mining* se ha diversificado para abordar las limitaciones operativas y semánticas de las aproximaciones clásicas, expandiéndose hacia áreas como la inferencia probabilística, el manejo de restricciones, la reconfiguración de sistemas y la interpretabilidad.

En primer lugar, algunos enfoques critican que los algoritmos tradicionales traten la minería de roles como un mero problema de compresión de datos con pérdida [4]. Al intentar reducir la matriz de accesos a un conjunto menor de roles, estos métodos agrupan permisos basándose exclusivamente en la eficiencia matemática, ignorando el contexto del negocio. Como consecuencia, el algoritmo tiende a sobreajustar los datos, memorizando como válidas las asignaciones excepcionales o erróneas del pasado. Esto genera roles artificiales, fragmentados y poco intuitivos para los administradores. Para solucionar este problema, se ha propuesto replantear el *Role Mining* como un problema de inferencia, utilizando modelos probabilísticos para aprender la configuración RBAC que con mayor probabilidad generó la matriz de accesos original. Esta metodología generativa logra extraer roles que generalizan de forma mucho más efectiva frente a escenarios con usuarios no observados en el modelo inicial.

Por otro lado, la literatura ha destacado la necesidad de incorporar políticas de seguridad

reales durante la fase de minería. En este ámbito, se ha demostrado que las técnicas convencionales, como la *Boolean Matrix Decomposition* (BMD) [5], son ineficaces al no considerar reglas críticas del negocio, como la Separación de Funciones. Como solución, se desarrolló la *Extended Boolean Matrix Decomposition* (EBMD) [6], la cual introduce el concepto de "permisos negativos" en los roles o en las asignaciones de usuarios. Este *constraint-aware role mining problem* (CRM) no solo requiere menos roles para representar las asignaciones, sino que también permite descubrir automáticamente las restricciones subyacentes ocultas en las políticas de la organización.

Además de la creación de roles desde cero, existe el desafío de mantener sistemas RBAC ya desplegados que, debido a la evolución corporativa y el traspaso de empleados, se vuelven redundantes o ineficientes. Para resolver esto, se han propuesto enfoques de reconfiguración jerárquica orientados a optimizar un sistema existente [7]. Estos métodos evalúan la calidad de los roles actuales y reestructuran las asignaciones buscando reducir la complejidad estructural, pero aplicando un principio estricto de "mínima perturbación". De esta manera, se asegura que la nueva arquitectura se mantenga lo más cercana posible a la original, reduciendo significativamente el impacto operativo sobre los usuarios durante la transición.

Finalmente, en cuanto a la interpretabilidad de los roles, destacan algoritmos recientes orientados explícitamente a extraer agrupaciones que tengan un significado lógico para las organizaciones. Por ejemplo, el algoritmo IRMAOC [8] evalúa la interpretabilidad de un rol basándose en la similitud de los usuarios, la cual se calcula a partir de sus permisos y atributos de forma conjunta. Para lograr esto, primero se genera un grafo de asociación de usuarios y se aplican técnicas de *clustering* para definir roles candidatos basados en dicho grafo. Esta metodología de agrupar usuarios en función de su similitud directa asegura que los roles resultantes mantengan una alta interpretabilidad y aplicabilidad práctica en el entorno empresarial.

2.2. Sistemas de Recomendación y Cálculo de Similitud de Usuarios

Los sistemas de recomendación son herramientas analíticas que utilizan el comportamiento histórico y los datos de un grupo de usuarios para predecir preferencias y sugerir elementos relevantes. Si bien su uso tradicional se enfoca en el *e-commerce* y el entretenimiento, su aplicación se puede extender a la optimización de recursos corporativos, como la sugerencia automatizada de asignaciones de roles o permisos en sistemas de control de acceso.

En este contexto, una de las técnicas más robustas y utilizadas es el Filtrado Colaborativo (*Collaborative Filtering*, CF) [9]. Esta aproximación destaca en escenarios donde el contenido o las características intrínsecas de los elementos a recomendar son difíciles de describir mediante sus metadatos. El CF opera construyendo una matriz de interacciones usuario-ítem, la cual modela el estado actual de las asignaciones. A partir de esta matriz, el algoritmo identifica "vecindarios" de usuarios con perfiles similares para inferir que los elementos presentes en el perfil de los vecinos, pero ausentes en el del usuario objetivo, constituyen recomendaciones de alto valor.

La literatura clasifica el Filtrado Colaborativo en dos grandes familias de algoritmos: *Memory-based* y *Model-based* [10, 11]. Los algoritmos *Memory-based* operan directamente sobre la matriz de datos original y se subdividen en enfoques basados en usuarios (*User-based*) y basados en ítems (*Item-based*). El enfoque *User-based* se centra en cuantificar la similitud entre individuos comparando sus perfiles. El rendimiento de estos algoritmos recae en que las métricas de similitud se basan en ítems que tengan usuarios en común, por lo que cuando el universo de objetos presenta una estructura dispersa, estas recomendaciones se vuelven poco confiables [11].

Una alternativa eficaz para mitigar este problema es la inclusión de los atributos inherentes de los usuarios en la determinación de la similitud [12]. Por ejemplo, en [13] se diseña un sistema centrado en la recomendación de restaurantes; originalmente, las sugerencias se generaban basándose únicamente en las clasificaciones explícitas de los clientes, pero los

autores demostraron que al incorporar información contextual y demográfica del usuario (como el género, la edad, la educación y la información laboral), la capacidad y precisión de recomendación de la aplicación mejoró sustancialmente.

Otro enfoque más moderno para mitigar el problema de datos dispersos es el uso de los algoritmos *Model-based*. Estos métodos emplean técnicas de *Machine Learning* y minería de datos para descubrir patrones latentes y generar un modelo predictivo pre-entrenado. Entre las técnicas más destacadas para este propósito se encuentran los métodos de reducción de dimensionalidad, como la Descomposición en Valores Singulares (SVD) [14] o el Análisis de Componentes Principales (PCA) [15]. Al proyectar los perfiles de los usuarios desde una matriz dispersa hacia un espacio vectorial continuo, denso y de menor dimensionalidad, es posible capturar correlaciones semánticas ocultas, mejorando de esta forma las recomendaciones generadas.

Más allá de la estrategia de generación de candidatos utilizada, la calidad final de las recomendaciones es un factor crítico para la viabilidad de cualquier sistema. Las deficiencias en el modelo predictivo pueden traducirse en dos tipos de errores fundamentales: falsos negativos (recomendaciones relevantes que el usuario necesita, pero que el sistema omite) y falsos positivos (recomendaciones irrelevantes que se presentan al usuario final).

En la gran mayoría de los sistemas de recomendación, una alta tasa de falsos positivos resulta particularmente problemática. La generación constante de sugerencias sin valor satura al usuario y degrada su confianza en la efectividad de la herramienta [16]. Para mitigar esta problemática, una alternativa es la implementación de una arquitectura híbrida en cascada, añadiendo una segunda etapa encargada exclusivamente de filtrar y depurar los candidatos iniciales.

Un ejemplo de esta metodología se presenta en [17], donde los autores aplican una arquitectura de dos etapas en el contexto de una plataforma de citas *online*. En la primera fase, el sistema utiliza Filtrado Colaborativo para generar un conjunto de recomendaciones potenciales. Posteriormente, en la segunda fase, se emplea un algoritmo supervisado basado en árboles de decisión para reevaluar y reordenar los candidatos entregados por el módulo anterior. Los resultados de dicho estudio demostraron que la integración de esta capa de clasificación final

incrementa la precisión y calidad de las sugerencias, superando al rendimiento del sistema basado únicamente en el Filtrado Colaborativo tradicional.

2.3. Manejo de variables categóricas dispersas

Dada la naturaleza categórica y dispersa de los datos utilizados, es necesario estudiar la forma correcta de procesarlos para poder desarrollar el sistema de recomendación el cual se acerque al óptimo.

En términos generales, los datos pueden clasificarse fundamentalmente en dos tipos: numéricos y categóricos. La principal diferencia radica en que las variables numéricas son representadas por valores que tienen un significado matemático directo, es decir, están sujetos a relaciones y operaciones aritméticas. Por otra parte, las variables categóricas existen como un medio para representar cualidades, etiquetas o atributos que no cumplen con esta condición.

En consecuencia, el análisis cuantitativo de este tipo de información requiere de técnicas de transformación específicas que habiliten su modelado computacional [18].

En el contexto de los sistemas de recomendación, comúnmente se recomienda la transformación de estas variables mediante técnicas de binarización, tales como *One-Hot Encoding* o *Label Encoding* [19]. Al aplicar estas técnicas, una variable categórica con múltiples categorías se transforma en un vector de valores binarios. En la literatura existen diversas métricas para estimar la similitud entre un par de estos vectores. Por ejemplo, el estudio [20] presenta un análisis exhaustivo de ocho coeficientes de similitud distintos aplicados a matrices binarias. El objetivo de dicho trabajo fue comparar las estructuras de los clústeres generados por estas ocho técnicas para entender las diferencias matemáticas entre cada una de ellas.

Si bien los resultados demostraron que varias métricas logran rendimientos comparables, el estudio destaca la robustez del Coeficiente de Jaccard. La principal ventaja matemática de Jaccard radica en que excluye las “co-ocurrencias negativas” de su expresión. Es decir, ignora los ceros compartidos entre dos vectores. En el contexto de este proyecto, esta característica es fundamental: el hecho de que dos usuarios no posean un permiso específico (un doble cero)

no implica que sus perfiles sean similares. Por lo tanto, el uso de coeficientes que no incluyen las co-ocurrencias negativas, como Jaccard, está ampliamente justificado, ya que enfocan el cálculo exclusivamente en los atributos que efectivamente poseen en común, facilitando además su interpretación.

No obstante, como se expuso en la sección anterior, el filtrado colaborativo que opera sobre el espacio original explícito presenta limitaciones para descubrir relaciones semánticas u ocultas entre usuarios. Para optimizar la calidad de las recomendaciones, se requiere extraer las representaciones latentes de los perfiles. Clásicamente, este descubrimiento se aborda mediante técnicas de reducción de dimensionalidad como el Análisis de Componentes Principales (PCA). Sin embargo, PCA asume que la varianza máxima de los datos se explica mediante combinaciones lineales, una premisa ineficaz frente a la topología de datos binarios y altamente dispersos [21].

Para superar esta limitación, se adopta *Kernel Principal Component Analysis* (KPCA). KPCA actúa como una generalización no lineal que, mediante el denominado *kernel trick*, mapea los datos originales hacia un espacio de características de mayor dimensión. En este nuevo hiperespacio, las estructuras latentes y no lineales se tornan linealmente separables, permitiendo finalmente proyectar la información hacia un espacio denso y de menor dimensionalidad [22]. Múltiples investigaciones corroboran que la integración de funciones *kernel* permite modelar de forma significativamente superior las interacciones subyacentes en conjuntos de datos complejos, superando con creces a las aproximaciones estrictamente lineales [23, 24, 25].

Si bien la implementación de KPCA resuelve de manera eficaz la generación inicial de candidatos, el sistema aún debe enfrentarse al desafío de controlar la cantidad de falsos positivos generados. Una posible solución a esto, como ya se mencionó en la sección anterior, es el uso de una arquitectura híbrida de dos etapas que mitiga este problema mediante el uso de un clasificador como filtro posterior a la obtención inicial de candidatos. Sin embargo, antes de implementar esta arquitectura, es necesario identificar clasificadores que hayan demostrado eficacia en el manejo de variables categóricas dispersas.

En este contexto, la investigación de [26] presenta un estudio comparativo entre CatBoost

y XGBoost para la predicción de fraude en el sector médico. Los resultados muestran que, para esta tarea de clasificación, CatBoost presenta una capacidad de predicción superior a XGBoost. El estudio destaca, además, que el manejo por defecto de variables categóricas de CatBoost es efectivo para problemas de alta cardinalidad.

Por otro lado, en [27] se evalúa la capacidad de predicción del abandono y el éxito académico estudiantil, operando en un escenario caracterizado por atributos de naturaleza categórica y un fuerte desbalance entre las clases a predecir. Los autores comparan el desempeño de múltiples algoritmos tradicionales, tales como Árboles de Decisión, *Support Vector Machines* (SVM) y *Random Forest*, frente a técnicas avanzadas de *Gradient Boosting* como *Extreme Gradient Boosting* (XGBoost), CatBoost y *Light Gradient Boosting Machine* (LightGBM). Los resultados evidencian que los métodos basados en *boosting*, especialmente LightGBM y CatBoost, entregan los mejores resultados generales, superando el rendimiento de los clasificadores más tradicionales.

A modo de síntesis, los contenidos revisados en esta sección del estado del arte evidencian las técnicas necesarias a tener en cuenta para el desarrollo de un sistema de recomendación basado en Filtrado Colaborativo. Específicamente, fundamentan las metodologías para la asignación de roles en entornos de alta dispersión categórica, como el que se abordará en el presente trabajo.

Capítulo 3

Desarrollo del sistema

Este capitulo presenta el desarrollo del sistema de recomendación de roles. La exposición se estructura en tres partes fundamentales que abarcan desde la definición de necesidades hasta la implementación técnica:

Requisitos del sistema: Donde se establecen los límites funcionales, operativos y de entorno del proyecto.

Arquitectura general: Que describe la organización de alto nivel y la interacción entre los componentes.

Descripción detallada de los módulos: Donde se profundiza en la lógica interna y los algoritmos específicos que componen la solución.

3.1. Requisitos del sistema

En esta sección se presentan los requisitos del proyecto, clasificados en tres categorías principales: funcionales, no funcionales y de ambiente. En primer lugar, los requisitos funcionales definen las capacidades específicas que el sistema debe implementar para satisfacer las necesidades operativas del equipo de Gestión de Accesos. Por otra parte, los requisitos no funcionales establecen los atributos de calidad y las restricciones técnicas que la solución

debe cumplir para operar correctamente en un entorno productivo. Finalmente, los requisitos de ambiente describen las especificaciones de infraestructura necesarias para el correcto despliegue del sistema, asegurando su viabilidad operativa y la reproducibilidad de los resultados.

3.1.1. Requisitos Funcionales

El levantamiento de los requerimientos mostrados a continuación es el resultado de un proceso de análisis realizado en conjunto con el equipo de Gestión de Accesos. A través de reuniones técnicas, se profundizó en el funcionamiento del flujo actual de asignación de roles y se identificaron las principales problemáticas operativas, tales como la sobrecarga manual y la reactividad del proceso. A partir de este diagnóstico, se establecieron los siguientes requisitos:

RF1 - Detección Proactiva: El sistema debe ser capaz de inferir y detectar los roles faltantes de los trabajadores, permitiendo identificar necesidades de acceso antes de que el usuario objetivo genere una solicitud formal.

RF2 - Validación de Lógica de Negocio: Las recomendaciones generadas por el sistema deben someterse a un proceso de filtrado que garantice su alineación con las reglas de asignación y restricciones vigentes de la empresa.

RF3 - Adaptabilidad: El sistema debe poseer la capacidad de actualizar sus criterios de decisión basándose en los nuevos datos históricos, adaptándose así a la evolución natural en la lógica del negocio de la empresa.

3.1.2. Requisitos No Funcionales

La definición de los siguientes requisitos no funcionales fue planteada considerando la infraestructura tecnológica y el volumen operativo de CMPC.

RNF1 - Eficiencia en el Procesamiento de Datos: Dado el gran volumen de información, el sistema debe ser capaz de procesar y analizar estos datos de manera eficiente, optimizando el uso de recursos computacionales para permitir la ejecución de los algoritmos de similitud y clasificación en tiempos viables para la operación.

RNF2 - Interpretabilidad de Resultados: Dado que el sistema funciona como apoyo a la toma de decisiones, los resultados entregados deben ser presentados en un formato claro que permita a los miembros del equipo de gestión de accesos entender la justificación de la recomendación.

3.1.3. Requisitos de Ambiente

Infraestructura de Hardware

Para la ejecución del sistema, se requiere disponer de un equipo, ya sea alojado en infraestructura local o en la nube, que cuente con capacidad de cómputo suficiente para procesar los volúmenes de datos históricos de la empresa.

Es importante destacar que, debido a la naturaleza de los algoritmos seleccionados, el sistema hace un uso intensivo de la CPU y la memoria RAM del equipo. Por consiguiente, no es estrictamente necesario contar con una GPU dedicada, ya que la carga computacional está optimizada para arquitecturas de procesadores estándar, permitiendo su implementación en servidores convencionales sin incurrir en costos de hardware especializado.

Entorno de Software

El desarrollo e implementación del sistema se fundamenta en el lenguaje de programación `Python 3.X`, seleccionado por su robusto ecosistema para ciencia de datos. Para garantizar el funcionamiento de los módulos de procesamiento y predicción, el entorno debe contar con las siguientes librerías base:

Manipulación de Datos: Se requiere el uso de bibliotecas como `pandas` y `numpy` para la gestión eficiente de estructuras de datos vectoriales.

Modelado y Aprendizaje Automático: El entorno debe soportar `scikit-learn` para los procesos de reducción de dimensionalidad y bibliotecas de *Gradient Boosting* (como `CatBoost` o `LightGBM`) para la clasificación de roles.

Control de Versiones: Se utiliza `Git` para la gestión del código fuente, asegurando la trazabilidad de los cambios y la colaboración en el desarrollo.

3.2. Diseño de Arquitectura del Sistema

3.2.1. Diagrama de contexto

Para definir el alcance y los límites del sistema, se presenta el diagrama de contexto en la Figura 3.1. Este modelo ilustra la interacción del "Modelo de recomendación de roles" con su entorno, el cual consta únicamente del "Equipo de gestión de Accesos".

Como se aprecia en el diagrama, el flujo de información es bidireccional. En este proceso, el equipo da inicio a la operación extrayendo los datos de los trabajadores desde la plataforma SAP y suministrándolos a la herramienta, para posteriormente obtener un conjunto de sugerencias de asignación de roles priorizadas.

Esta representación refuerza la naturaleza del sistema como una herramienta de apoyo a la decisión, donde el sistema actúa como una "caja negra" que procesa datos operativos para entregar inteligencia accionable al equipo humano, sin intervenir directamente en la base de datos de SAP.

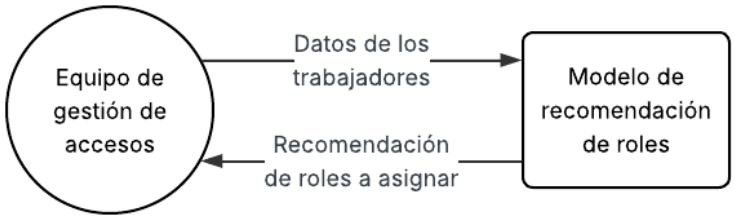


Figura 3.1: Diagrama de contexto del sistema.

3.2.2. Diagrama de arquitectura

Para facilitar la comprensión del diseño técnico, la arquitectura del sistema se presenta en dos niveles de abstracción. En primera instancia, se expone una vista de alto nivel que ilustra el flujo secuencial de la información a través de los subsistemas principales. Posteriormente, se presenta una vista detallada que descompone estos subsistemas en módulos funcionales específicos.

Vista de Alto Nivel

Como se aprecia en la Figura 3.2, la solución se estructura en tres subsistemas que operan en cadena:

- **Procesamiento de Datos:** Encargado de la ingesta, limpieza y transformación de los datos crudos provenientes de SAP.
- **Extractor de Potenciales Recomendaciones:** Subsistema que genera los grupos de usuarios y detecta, mediante inferencia, los roles potenciales a asignar.
- **Clasificador de Asignación de Roles:** Etapa final de validación donde un modelo supervisado evalúa la confianza de las asignaciones sugeridas, filtrando aquellas que no superan el umbral de probabilidad establecido.

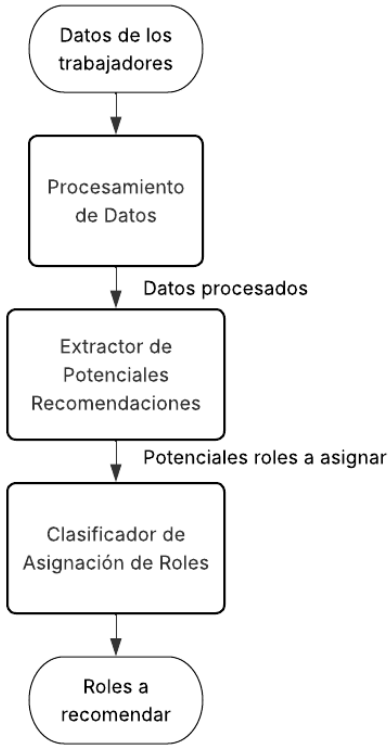


Figura 3.2: Diagrama de arquitectura de alto nivel del sistema.

Vista Detallada y Modular

Por otra parte, la Figura 3.3 despliega la arquitectura interna de los componentes anteriores. En este diagrama se evidencia el enfoque híbrido del sistema, caracterizado por una bifurcación tras la etapa de extracción de atributos:

- Una rama se orienta al aprendizaje no supervisado (Cálculo de Similitud entre trabajadores) para la generación de candidatos.
- La otra rama se dedica a la construcción y entrenamiento del modelo supervisado. Cabe destacar que este flujo de entrenamiento se ejecuta condicionalmente, únicamente cuando no existe un modelo previo o se requiere un reentrenamiento por actualización de datos.

- Finalmente, ambos flujos convergen en el Motor de Recomendación Híbrido, unificando la inferencia con la validación.

A continuación, se describe la composición de cada subsistema:

El subsistema **"Procesamiento de datos"** se compone del módulo **Pre-procesamiento y Agregación de Entidades (M1)**, responsable de la limpieza de los datos de entrada, y del módulo **Parsing y Extracción de Atributos del Rol (M2)**, encargado de descomponer la codificación de los roles para conservar únicamente los componentes relevantes.

Por otra parte, el subsistema **"Extractor de Potenciales Recomendaciones"** se subdivide en los módulos: **Generación de Embeddings de Usuario (M3)**, cuya función es transformar los datos categóricos de los usuarios a una representación vectorial numérica densa; **Reducción de Dimensionalidad (M4)**, responsable de reducir la alta dimensionalidad generada en la etapa anterior y de descubrir relaciones latentes en la estructura de los datos; **Cálculo de Similitud (M5)**, destinado a calcular la métrica de similitud entre los trabajadores para identificar agrupaciones; e **Inferencia de Permisos Candidatos (M6)**, encargado de identificar permisos que poseen los "vecinos" del grupo pero que el usuario objetivo aún no tiene asignados.

Finalmente, el subsistema **"Clasificador de Asignación de Roles"** integra los módulos: **Construcción de Dataset Supervisado (M7)**, destinado a generar los datos para entrenar el modelo; **Entrenamiento del Modelo Predictivo (M8)**, responsable del aprendizaje del clasificador; y por último, el **Motor de Recomendación Híbrido (M9)**, encargado de la integración final que toma los "permisos candidatos" y los filtra mediante la validación del clasificador, entregando así la lista definitiva.

La Tabla 3.1 resume la definición y ubicación técnica de cada uno de estos módulos en el documento.

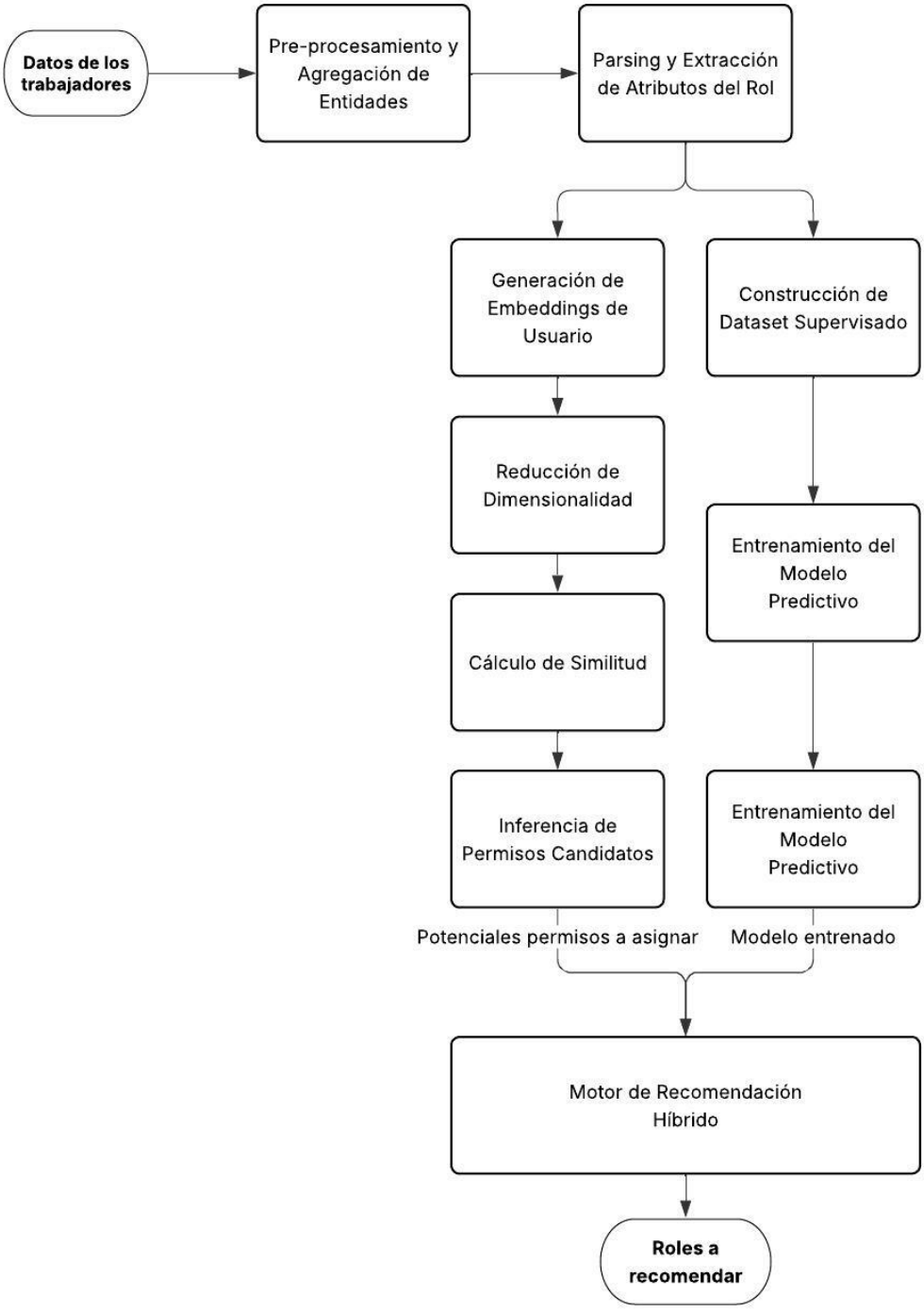


Figura 3.3: Diagrama detallado de arquitectura.

Cuadro 3.1: Módulos de la arquitectura del sistema

Módulo	**Propósito**	**Sección**
Pre-procesamiento y Agregación de Entidades (M1)	Módulo encargado de la ingesta y limpieza de datos crudos. Realiza el filtrado segmentado por sucursal operativa y consolida los atributos categóricos y la lista de permisos históricos por cada identidad única de trabajador.	3.3.1
Parsing y Extracción de Atributos del Rol (M2)	Módulo responsable de descomponer la cadena de texto original de los roles. Su función es aislar y conservar únicamente los componentes semánticos relevantes, descartando los segmentos de codificación interna sin valor predictivo.	3.3.2
Generación de Embeddings de Usuario (M3)	Módulo encargado de codificar la información de los usuarios (texto y categorías) transformándola en una representación vectorial numérica densa que capture las características latentes del trabajador.	3.3.3
Reducción de Dimensionalidad (M4)	Módulo encargado de aplicar técnicas de reducción de dimensionalidad sobre los embeddings generados, comprimiendo la información para optimizar el cálculo de distancias y reducir el ruido.	3.3.4
Cálculo de Similitud (M5)	Módulo encargado de calcular la métrica de similitud entre los trabajadores para identificar agrupaciones y vecinos cercanos.	3.3.5

Continúa en la siguiente página...

Cuadro 3.1 – Continuación de la página anterior

Módulo	**Propósito**	**Sección**
Inferencia de Permisos Candidatos (M6)	Módulo encargado de reconstruir los permisos potenciales basándose en filtrado colaborativo: identifica permisos que poseen los ”vecinos cercanos”pero que el usuario objetivo aún no tiene asignados.	3.3.6
Construcción de Dataset Supervisado (M7)	Módulo encargado de generar los pares de entrenamiento (usuario, permiso) etiquetados, incluyendo tanto ejemplos positivos (permisos reales) como negativos para el aprendizaje del clasificador.	3.3.7
Entrenamiento del Modelo Predictivo (M8)	Módulo encargado de entrenar un clasificador para determinar la probabilidad de asignación de un permiso específico a un usuario, basándose en patrones históricos aprendidos.	3.3.8
Motor de Recomendación Híbrido (M9)	Módulo de inferencia final que integra ambas ramas. Toma los ”permisos candidatos”sugeridos por la similitud y los filtra mediante la validación del clasificador entrenado, entregando la recomendación final.	3.3.9

3.3. Implementación de Módulos del Sistema

En esta sección se presenta la implementación técnica de cada uno de los módulos que componen el sistema.

3.3.1. Pre-procesamiento y Agregación de Entidades (M1)

El primer paso crítico para asegurar la calidad de las recomendaciones es la correcta selección, limpieza y estructuración de los datos de entrada.

El alcance del estudio se delimitó a las sucursales 504 y 514, correspondientes a las operaciones de CMPC en Brasil. La selección estratégica de estas unidades por parte del equipo de gestión de acceso responde a que estas son las sucursales que representan una gran parte de su carga laboral.

De los múltiples archivos recibidos, se identificaron como relevantes para el análisis los siguientes:

- **Maestro de Trabajadores:** Contiene atributos demográficos y organizacionales (Departamento, Función, Fecha de Ingreso, Sucursal de origen).
- **Asignaciones Históricas:** Contiene el registro de permisos (roles) activos para cada usuario.

La implementación de este módulo se realizó utilizando la librería `pandas` [28] en Python. La operación central consiste en un *merge* de ambas fuentes utilizando el identificador único del usuario como llave primaria. El objetivo es transformar los registros transaccionales en una estructura centrada en la entidad "Usuario".

Como resultado de este procesamiento, se obtiene un `DataFrame` unificado donde cada fila representa a un trabajador único, conteniendo sus atributos categóricos (Departamento, Función) junto con un vector o lista con la totalidad de los roles que tiene asignados actualmente. La Tabla 3.2 resume el volumen de la información consolidada y, adicionalmente, a modo de ejemplo, la Tabla 3.3 ilustra la estructura final del conjunto de datos procesado, donde se observa la consolidación de atributos y la lista de roles por usuario.

Cuadro 3.2: Cardinalidad de las variables categóricas principales

Variable	**Cantidad de Valores Únicos**
Total de Trabajadores	1039
Departamentos	201
Funciones	452
Roles	3091

Cuadro 3.3: Muestra de la estructura de datos unificada (Datos ficticios)

User ID	**Depto**	**Función**	**Roles Asignados (Lista)**
U-1024	Operaciones	Supervisor	[ZD_ALCOPC0:0504, ZD_LOGIST:0504]
U-1025	Finanzas	Analista Jr	[FI_GL_USER:0504]

3.3.2. Parsing y Extracción de Atributos del Rol (M2)

El análisis exploratorio preliminar reveló que los identificadores de roles seguían una estructura técnica compleja (por ejemplo: `ZD_ALCOPC0-001-03-001:0504`). Ante esta estructura, se procedió a validar con el equipo de Gestión de Accesos la categorización interna de estos códigos para entender cuáles aportaban un valor semántico real al modelo.

Como resultado de esta validación, se determinó que los segmentos centrales (ej. `001-03-001`) corresponden a códigos internos de versiones sin valor semántico para la recomendación. Por el contrario, la información relevante se encuentra al inicio y final de este código, correspondiendo al identificador funcional del rol (ej. `ZD_ALCOPC0`) y a la sucursal de aplicación (ej. `:0504`) respectivamente.

A partir de esta definición, se implementó una rutina de procesamiento de texto (*parsing*) diseñada para extraer y conservar únicamente estos componentes relevantes.

Paralelamente, se realizó un estudio sobre la distribución de roles entre sucursales. Los resultados mostraron que solo un **16 %** de los usuarios poseía permisos válidos para sucursales distintas a su base de operaciones. Dado este bajo porcentaje y el riesgo de introducir

ruido en el modelo de similitud, se tomó la decisión de diseño de acotar el alcance de la recomendación exclusivamente a la sucursal base del trabajador.

La implementación conjunta de este filtrado por sucursal y la eliminación de versiones duplicadas tras el *parsing* generó un impacto significativo en la eficiencia del sistema: se logró reducir la cardinalidad del espacio de roles en un **88 %**, pasando de 3,091 identificadores crudos a un total de **361 roles únicos** y semánticamente densos.

3.3.3. Generación de Embeddings de Usuario (M3)

Para estimar la similitud de los trabajadores, se optó por utilizar un diseño arquitectónico similar al propuesto en [8], en donde la información de los atributos organizacionales y la de los permisos se procesan de manera independiente. Dada la naturaleza puramente categórica de los datos originales, fue necesario transformar estos perfiles a un espacio vectorial para permitir el cálculo matemático de afinidad.

En concreto, la vectorización se dividió en dos procesos: por un lado, se aplicó *One-Hot Encoding* a los atributos del trabajador (Departamento y Función), garantizando la equidistancia matemática entre las distintas áreas de la empresa; por otro lado, se utilizó *Multi-Hot Encoding* para procesar la lista de roles, generando un vector binario que representa los múltiples permisos de un mismo usuario. Como resultado de esta etapa, la entidad del trabajador deja de ser un registro transaccional y pasa a estar representada por estructuras vectoriales independientes, preparadas para las siguientes fases de reducción de dimensionalidad y cálculo de similitud.

3.3.4. Reducción de Dimensionalidad (M4)

Dada la naturaleza dispersa y la alta dimensionalidad de los vectores generados en la etapa anterior, sumado al objetivo de descubrir relaciones latentes en la estructura de los datos, se optó por implementar una reducción de dimensionalidad mediante el Análisis de Componentes Principales con Kernel (*Kernel PCA* o KPCA). Para la ejecución de este proceso, se utilizó la clase `KernelPCA` provista por la librería `scikit-learn` [29].

Esta transformación se aplicó exclusivamente sobre los permisos (roles), aislando los atributos organizacionales (Departamento y Función). Esta decisión radica en que es necesario preservar estos atributos estructurales debido a que la coincidencia de estos campos debe ser exacta. Por el contrario, la asignación de roles obedece a comportamientos subyacentes y relaciones latentes donde la reducción de ruido sí aporta un valor significativo.

Adicionalmente, para la configuración matemática de esta transformación, las funciones *kernel* seleccionadas corresponden a RBF (*Radial Basis Function*) y Cosine.

3.3.5. Cálculo de Similitud (M5)

Con el objetivo de respaldar y cuantificar el impacto de las decisiones técnicas adoptadas en los módulos anteriores, se implementaron distintos enfoques para el cálculo de similitud entre los trabajadores. Esta metodología comparativa permite evaluar el rendimiento del sistema propuesto frente a escenarios tradicionales.

El primer enfoque corresponde al caso base o manual, el cual emula el proceso de búsqueda empírica que actualmente realiza el equipo de Gestión de Accesos. En este escenario, la afinidad se determina exclusivamente mediante los atributos organizacionales del usuario (compañeros directos). Para su implementación, se estableció una comparación directa y estricta de estos campos, asignando una ponderación equitativa de 0.5 puntos a la coincidencia exacta de Departamento y 0.5 puntos a la coincidencia de Función, alcanzando un máximo de 1.

Los siguientes enfoques se fundamentan en la decisión arquitectónica de procesar los atributos estructurales y los permisos de forma independiente. El segundo enfoque evalúa la similitud con datos crudos, operando sobre el espacio vectorial original. Dada la naturaleza binaria y altamente dispersa (*sparse*) de la matriz de asignaciones, se seleccionó el Coeficiente de Jaccard para medir la superposición de roles entre dos usuarios (A y B):

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{3.1}$$

La elección de esta métrica por sobre alternativas como la distancia de Hamming radica

en la distribución del dominio: en la gran mayoría de los casos, un trabajador posee menos del 10 % del universo total de roles disponibles. El coeficiente de Jaccard se centra exclusivamente en la intersección de los permisos que los usuarios sí tienen asignados, evitando que la inmensa cantidad de roles ausentes en ambos perfiles infle artificialmente el valor de similitud.

El tercer enfoque representa el diseño propuesto en este trabajo, calculando la similitud sobre el espacio latente reducido (producto del módulo M4). Al estar en un espacio vectorial continuo, denso y de menor dimensionalidad, métricas de conjuntos como Jaccard pierden aplicabilidad. Por consiguiente, se implementó la Similitud del Coseno, métrica estándar y de alto rendimiento para evaluar la orientación y cercanía de vectores continuos:

$$\text{Similitud}(u, v) = \frac{u \cdot v}{\|u\|\|v\|} \tag{3.2}$$

donde u y v representan los vectores de características continuas (*embeddings* reducidos) correspondientes a dos trabajadores distintos.

Finalmente, para los enfoques que separan atributos y permisos, la evaluación de los atributos organizacionales mantiene la misma lógica del caso base. El resultado definitivo del algoritmo se formula como una similitud compuesta, calculada a través de una combinación ponderada lineal de ambas componentes:

$$Sim_{total}(u, v) = w_a \cdot Sim_{atributos}(u, v) + w_p \cdot Sim_{permisos}(u, v) \tag{3.3}$$

donde $Sim_{total}(u, v)$ es la similitud final calculada entre el trabajador u y el trabajador v; $Sim_{atributos}$ y $Sim_{permisos}$ corresponden a los puntajes de similitud parciales obtenidos de los campos estructurales y de roles, respectivamente. Finalmente, w_a y w_p son hiperparámetros que representan el peso relativo asignado a cada métrica. Para garantizar que la similitud final se mantenga correctamente normalizada, estos coeficientes deben estar acotados en el intervalo $[0, 1]$ y cumplir con la restricción matemática de que su suma sea exactamente 1.

3.3.6. Inferencia de Permisos Candidatos (M6)

Una vez cuantificada la similitud entre los trabajadores, el sistema procede a inferir el conjunto de roles potenciales. La implementación de esta etapa se basa en un enfoque de Filtrado Colaborativo, cuyo flujo lógico se detalla en la Figura 3.4.

El proceso toma como entrada la Matriz de Similitud generada en el módulo anterior (M5) y opera de manera iterativa sobre cada trabajador objetivo (U_i). Para definir el "vecindario" relevante, el algoritmo busca los k usuarios más afines (Top-k) una vez establecido este grupo de pares, se procede a contrastar los perfiles de acceso; mediante una operación de diferencia de conjuntos ($Roles_{vecinos} - Roles_{usuario}$), el algoritmo aísla aquellos roles que poseen los vecinos pero que el trabajador objetivo aún no tiene asignados.

El producto de esta iteración es una lista bruta de candidatos que consolida todas las posibles asignaciones a recomendar. No obstante, dado el elevado volumen de recomendaciones generadas (como se evidencia en el capítulo de Resultados) y la estricta sensibilidad de la información en la gestión de accesos, esta lista preliminar requiere un refinamiento adicional. Por consiguiente, estos candidatos servirán como entrada para el clasificador supervisado (Módulo 9), el cual se encargará de validar y filtrar las sugerencias para asegurar la calidad de la recomendación final entregada al equipo de Gestión de Accesos.

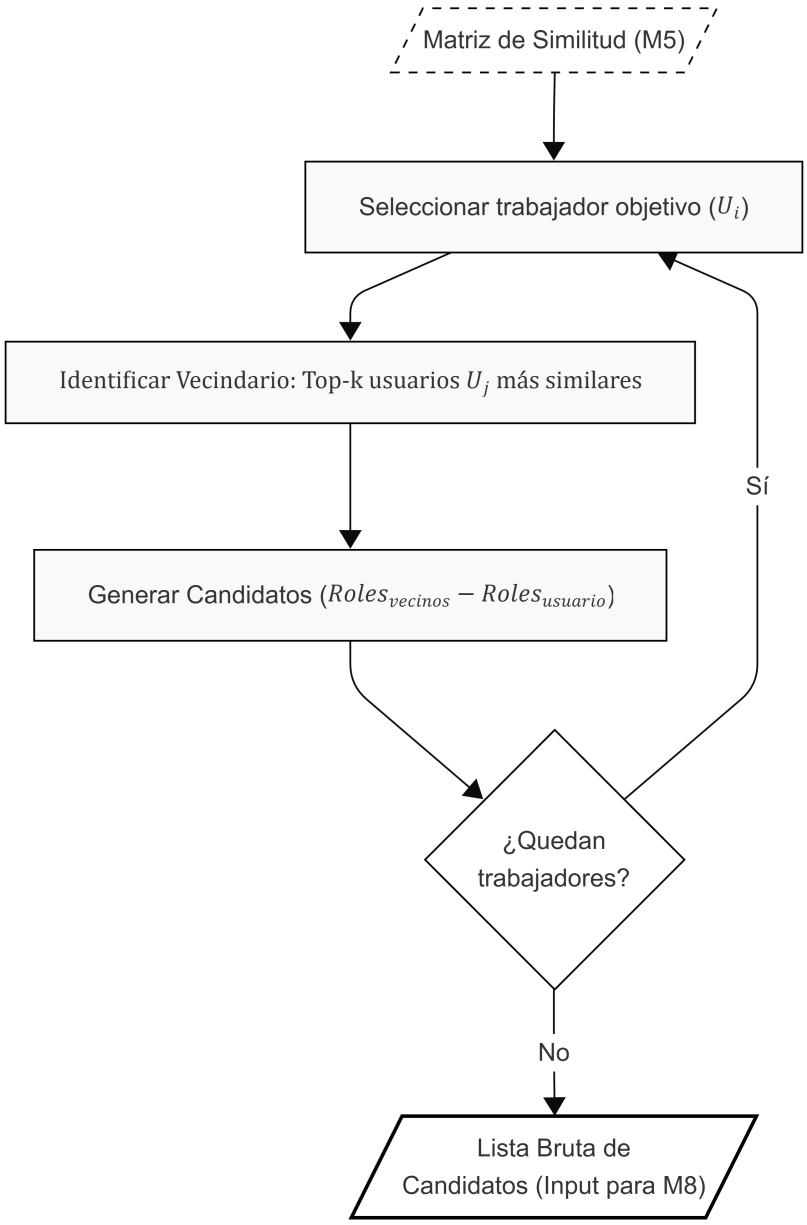


Figura 3.4: Flujo lógico para la inferencia de permisos candidatos.

3.3.7. Construcción de Dataset Supervisado (M7)

Para el entrenamiento del modelo de clasificación, fue necesario transformar los registros históricos de asignaciones en un problema de aprendizaje supervisado. El objetivo fundamental de este módulo es construir un conjunto de datos estructurado que permita al algoritmo

discriminar entre una asignación válida y una errónea basándose en los atributos del perfil.

El *dataset* se construyó generando pares (tuplas) de la forma Usuario-Rol, donde las variables predictoras combinan los atributos del trabajador (Departamento y Función) con el identificador del rol candidato. La variable objetivo (*target*) se definió de manera binaria: un valor de **1** indica una asignación legítima, mientras que un **0** representa una asignación inexistente o inválida.

La generación de estas clases presentó un desafío particular debido a la naturaleza de los datos. La **clase positiva (1)** se extrajo directamente de las asignaciones vigentes en el sistema SAP. Sin embargo, para la **clase negativa (0)**, dado que no existen registros explícitos de "rechazos", fue necesario generar ejemplos sintéticos mediante una técnica de *Negative Sampling*. Esto implica seleccionar aleatoriamente roles que el usuario *no* posee para enseñar al modelo qué constituye una "no-asignación".

Finalmente, para mitigar el desbalance extremo natural de los datos (donde la cantidad de roles no asignados es inmensamente superior a los asignados), se aplicó una estrategia de balanceo controlada. Se estableció un ratio de muestreo de **1:3**; es decir, por cada muestra positiva real, se generaron tres muestras negativas sintéticas. Esta proporción permite que el modelo aprenda la frontera de decisión sin sesgarse excesivamente hacia la clase mayoritaria, preservando la noción de que las asignaciones válidas son eventos específicos y menos frecuentes.

3.3.8. Entrenamiento del Modelo Predictivo (M8)

En relación con la caracterización de los datos realizada en las etapas previas, donde predominan atributos de naturaleza puramente categórica y alta cardinalidad, la estrategia de modelado se orientó hacia algoritmos con soporte nativo y eficiente para estas estructuras. Se descartaron enfoques tradicionales que requieren transformaciones expansivas en favor de arquitecturas de *Gradient Boosting* optimizadas.

Bajo este criterio, se diseñó un experimento comparativo entre dos de los exponentes más robustos del estado del arte: **LightGBM** y **CatBoost**, utilizando para ello las implementacio-

nes oficiales disponibles en las librerías de Python `lightgbm` y `catboost` respectivamente. La selección de estos modelos específicos responde a su capacidad para manejar variables categóricas directamente dentro de la estructura del árbol de decisión, lo cual reduce significativamente el consumo de memoria y el tiempo de entrenamiento frente a la dispersión de los datos.

Ambos algoritmos fueron sometidos a un protocolo de evaluación estandarizado utilizando el mismo conjunto de datos supervisado generado en el módulo anterior (M7). Para cuantificar y comparar su capacidad de generalización, se monitorearon métricas de rendimiento clave: **Precisión** y **Recall** para evaluar la exactitud puntual, junto con el **ROC AUC** y **PR AUC** para medir la robustez del discriminador ante distintos umbrales de decisión. Los resultados cuantitativos de esta comparativa, así como la fundamentación de la elección del modelo final, se detallan en el capítulo de Resultados.

3.3.9. Motor de Recomendación Híbrido (M9)

Este módulo constituye la etapa de consolidación de la arquitectura propuesta. Tal como se fundamentó en el módulo de Inferencia de Candidatos (M6), la lista preliminar obtenida, si bien es exhaustiva, requiere una depuración rigurosa debido al volumen de datos y a la **sensibilidad de la información**.

El objetivo central de este componente es orquestar la integración entre los módulos previos. Para ello, el sistema toma como entrada la lista bruta de permisos potenciales generada en el M6 y la somete a una revisión del modelo predictivo entrenado y validado en el M8. Operativamente, el clasificador evalúa cada par Usuario-Rol candidato, asignándole un **puntaje de probabilidad** (confianza) que cuantifica la certeza técnica de dicha asignación.

Finalmente, el motor aplica un filtro de corte basado en un umbral de decisión predefinido. Únicamente aquellos roles cuya probabilidad calculada supere este umbral conformarán la lista definitiva de sugerencias. Este proceso asegura que el entregable final presentado al equipo de Gestión de Accesos mantenga un estándar de alta precisión, reduciendo significativamente la presencia de falsos positivos.

Capítulo 4

Resultados

Este capítulo expone los resultados empíricos obtenidos tras la implementación del sistema propuesto, con el objetivo de respaldar y cuantificar la eficacia de las decisiones arquitectónicas adoptadas. El contenido de este apartado se divide estructuradamente en tres fases de evaluación:

1. **Optimización del Espacio Vectorial:** Se analizan los resultados de la reducción de dimensionalidad aplicada sobre la matriz de permisos.

2. **Evaluación del Modelo de Clasificación Supervisada:** Se presenta el rendimiento y las métricas de validación del entrenamiento de los clasificadores predictivos.

3. **Rendimiento del Sistema de Recomendación:** Se evalúa el modelo híbrido en su totalidad utilizando datos de asignaciones futuras para medir su capacidad predictiva en un escenario real de negocio.

4.1. Optimización del Espacio Vectorial

Tal como se fundamentó en la sección 3.3.4, se seleccionó el Análisis de Componentes Principales con *Kernel* (KPCA) como técnica de reducción dimensional. Para validar empíricamente la viabilidad de esta compresión, es necesario asegurar que la transformación matemática preserve la estructura e integridad de los datos originales. La métrica estándar para

evaluar este comportamiento es la razón de varianza explicada (*Explained Variance Ratio*) por cada componente proyectada, la cual cuantifica el porcentaje de información retenida en función del número de dimensiones.

En las Figuras 4.1 y 4.2 se presentan la curva de varianza explicada individual y la curva de varianza explicada acumulada, respectivamente. Al analizar ambos gráficos, se evidencia el impacto de la dispersión extrema inherente a este dominio de datos: concentrar la variabilidad en unas pocas dimensiones resulta complejo, observándose que la componente principal más significativa logra capturar únicamente un aproximado de 12 % de la varianza total para el caso del *kernel* RBF, y un 16 % para el caso del *kernel* Cosine.

No obstante, la técnica demuestra su eficacia al evaluar el comportamiento acumulado. El sistema logra retener una vasta mayoría de la información latente utilizando una fracción del hiperespacio de entrada. Un resumen de estos resultados se presenta en la Tabla 4.1, la cual detalla el número de componentes requeridos por cada *kernel* para alcanzar distintos umbrales de retención de información.

Cuadro 4.1: Número de componentes requeridos para alcanzar distintos porcentajes de varianza explicada acumulada, según el tipo de *kernel*.

Varianza Explicada Acumulada	**Componentes Requeridos (*Kernel* Coseno)**	**Componentes Requeridos (*Kernel* RBF)**
50 %	8	15
60 %	13	28
70 %	21	50
80 %	35	91
90 %	64	178

A partir de los datos expuestos en la Tabla 4.1, se aprecia claramente que ambas funciones logran reducir de manera significativa la dimensionalidad original del problema. Sin embargo, destaca el desempeño del *kernel* Cosine, el cual logra este objetivo utilizando un número significativamente menor de componentes en comparación con el *kernel* RBF.

A partir de este análisis, se podría establecer un rango de candidatos viables para la dimensionalidad del espacio latente. Sin embargo, dado que el objetivo primordial del sistema no es solamente la compresión matemática, sino maximizar las sugerencias correctas, la decisión final sobre el número de componentes no puede tomarse de forma aislada. Por lo tanto, la justificación y selección definitiva de este hiperparámetro se posponen y se detallan en la Sección 4.3 (Rendimiento del Sistema de Recomendación), donde se evaluará el impacto directo de distintas configuraciones de dimensionalidad sobre la métrica de *Recall* utilizando datos de asignaciones futuras realizadas por el equipo de Gestión de Accesos.

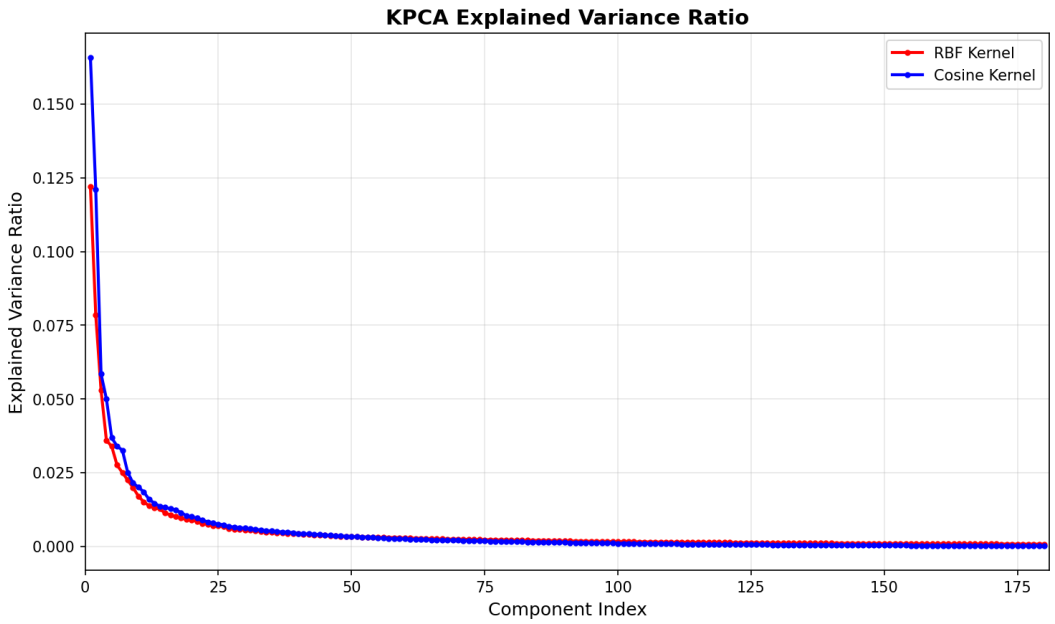


Figura 4.1: Razón de varianza explicada por cada componente principal.

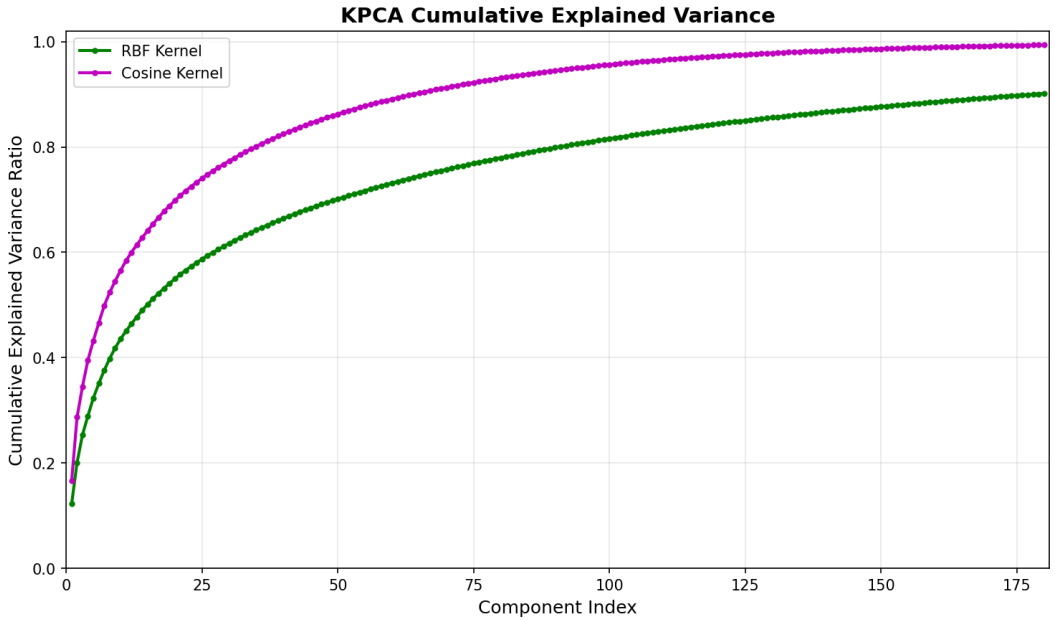


Figura 4.2: Varianza explicada acumulada en función del número de componentes.

4.2. Evaluación del Modelo de Clasificación Supervisada

Tal como se detalló en la fase de entrenamiento (Módulo 8), se realizó un análisis comparativo entre dos de los algoritmos de *Gradient Boosting* más robustos para el manejo de variables categóricas: LightGBM y CatBoost. El objetivo de esta etapa fue determinar qué modelo logra discriminar con mayor exactitud las asignaciones de roles válidas de las inválidas.

Al analizar las métricas obtenidas, se observa un rendimiento sobresaliente por parte de ambos algoritmos; sin embargo, CatBoost supera consistentemente a LightGBM en todos los indicadores evaluados. Específicamente, CatBoost alcanza una precisión de 0.881 y un *Recall* de 0.844, superando los márgenes de 0.835 y 0.737 obtenidos por LightGBM, respectivamente. Aún más importante, analizando el valor de la métrica PR AUC (Área bajo la curva *Precision-Recall*), CatBoost alcanzó un valor de 0.928 frente a los 0.881 de su contraparte. Esto demuestra una clara superioridad en la identificación precisa de las asignaciones válidas.

No obstante, al igual que en la etapa de reducción dimensional, el éxito en las métricas aisladas de entrenamiento no garantiza necesariamente una mejora tangible en la utilidad del sistema en producción. Por consiguiente, el impacto real de este clasificador como filtro refinador se evaluará empíricamente en la Sección 4.3, contrastándolo con el escenario base mediante datos de asignaciones futuras.

4.3. Rendimiento del Sistema de Recomendación

En los apartados anteriores se ha comprobado teóricamente que los módulos implementados cumplen con los criterios matemáticos y de rendimiento de manera aislada. El paso definitivo es validar que el sistema en su conjunto cumple con el objetivo principal de este trabajo: recomendar asignaciones de roles de manera certera y eficiente, demostrando su utilidad como herramienta de apoyo para el equipo de Gestión de Accesos.

Para comprobar esta funcionalidad en un escenario real, se utilizará un conjunto de datos de asignaciones de roles reales que fueron concedidas en un periodo posterior a la fecha de corte de los datos base (entrenamiento). En concreto, este conjunto de validación cuenta con un total de 5.547 asignaciones reales, repartidas entre 271 usuarios (lo que representa un 26 % del total de la plantilla analizada), con un promedio de 20 nuevas asignaciones por trabajador.

El objetivo central de esta prueba es doble: por un lado, cuantificar qué porcentaje de estas asignaciones futuras es capaz de predecir el sistema de recomendación (métrica de Recall); y por otro lado, medir el número promedio de recomendaciones que el sistema necesita generar por usuario para alcanzar dicho nivel de acierto.

Tal como se definió en la metodología del Módulo de Cálculo de Similitud (M5), se evaluarán tres enfoques distintos para medir el impacto individual de cada decisión de diseño. El primer enfoque corresponde al caso base, el cual emula el proceso actual de los trabajadores estimando la similitud de pares únicamente mediante los atributos organizacionales. El segundo enfoque calcula la similitud operando sobre la matriz dispersa de datos crudos (utilizando el Coeficiente de Jaccard). Finalmente, el tercer enfoque evalúa el sistema propuesto operando sobre el espacio reducido (utilizando la Similitud del Coseno).

Antes de realizar la comparación directa de los tres casos mencionados, dado que los dos últimos enfoques utilizan una combinación lineal ponderada entre la similitud basada en atributos y la basada en permisos, es necesario realizar un análisis de sensibilidad. Este análisis consiste en probar distintos valores para los pesos asociados a cada una de estas métricas (w_a y w_p), con el fin de identificar la configuración que maximice el rendimiento del sistema.

En la Figura 4.3 se muestran los resultados obtenidos para el análisis de sensibilidad.

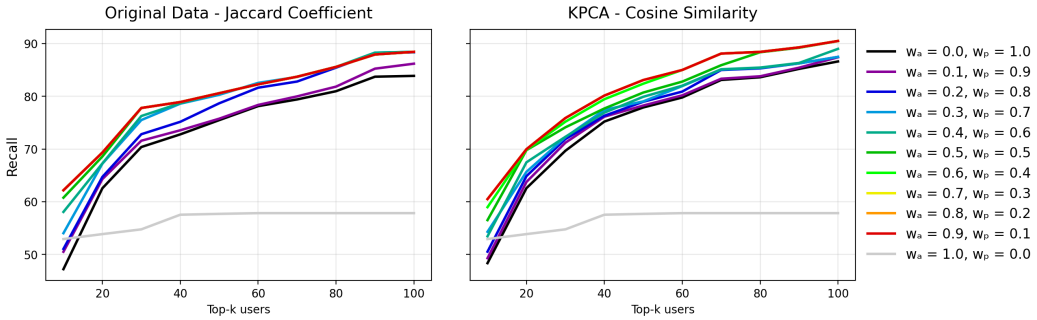


Figura 4.3: Rendimiento del sistema en el análisis de sensibilidad.

Al analizar ambos gráficos, destaca una caída abrupta en el rendimiento para el caso de $w_a = 1$ (similitud evaluada exclusivamente por atributos), especialmente al compararlo con el valor de $w_a = 0{,}9$, el cual presenta, de manera contraintuitiva, los resultados más altos. Este comportamiento se explica por la naturaleza de los datos: en la implementación inicial, enfocada solo en atributos, los valores posibles eran estrictamente discretos (0, 0.5 y 1). Para optimizar el cálculo y evitar procesar usuarios sin afinidad, el sistema se diseñó para descartar automáticamente las similitudes iguales a 0. Sin embargo, al evolucionar hacia una métrica compuesta que incorpora una variable continua (la similitud por permisos), la distancia resultante rara vez equivale a un cero absoluto. En consecuencia, al mantener la regla de descarte original, los casos que teóricamente deberían estar dominados por los atributos (w_p cercano a 1) se ven severamente afectados por el ruido de fondo de los permisos, a pesar de que el peso algebraico de estos últimos sea marginal.

Debido a este comportamiento del sistema actualizado, fue necesario analizar e implementar distintos umbrales de corte para la similitud mínima aceptada, permitiendo filtrar de

manera correcta esta métrica compuesta. La Figuras 4.4 presenta los resultados obtenidos utilizando tres umbrales representativos.

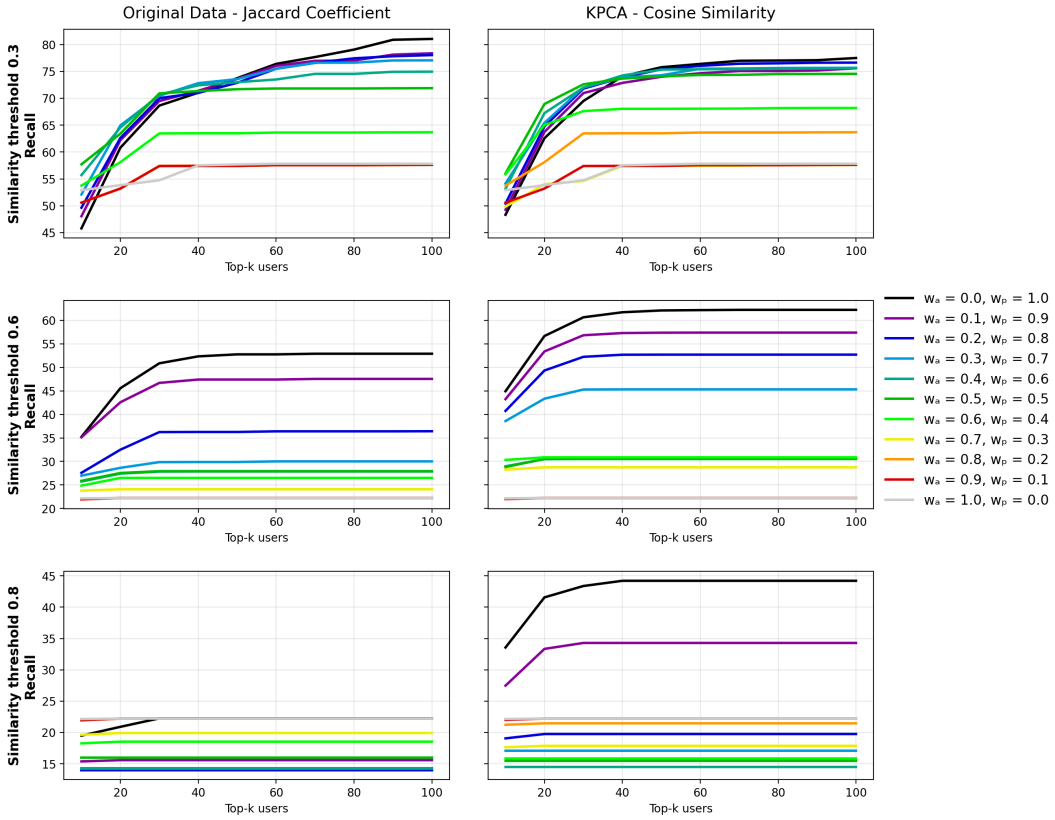


Figura 4.4: Resultados del sistema aplicando distintos umbrales de corte.

Partiendo con el umbral más bajo (0.3), se comienza a notar que priorizar los permisos mejora el rendimiento; no obstante, la falta de un patrón estable deja en evidencia que el ruido de fondo mencionado anteriormente sigue presente. Al transicionar a un umbral medio (0.6), el sistema se estabiliza y muestra una tendencia clara, demostrando la dominancia de la similitud basada en permisos en el contexto de la recomendación. Finalmente, para el filtro más exigente (0.8), el sistema colapsa debido a que alcanzar dichos valores requiere una coincidencia prácticamente perfecta tanto en atributos como en permisos, una condición sumamente infrecuente entre los usuarios, lo que explica el desplome en el desempeño de este intervalo.

En conclusión, el punto óptimo de operación del sistema se ubica en el umbral medio

(0.6), ya que es el nivel que logra limpiar el ruido de las conexiones triviales sin llegar a perjudicar la retención de las recomendaciones potenciales descubiertas.

Establecido el umbral óptimo de similitud en 0.6, el siguiente paso consiste en determinar la configuración del *kernel* que maximice el rendimiento del enfoque basado en reducción de dimensionalidad (KPCA). La Figura 4.5 expone la comparativa de *Recall* utilizando los *kernels Cosine* y *Radial Basis Function* (RBF) bajo distintos porcentajes de varianza explicada.

Al analizar las curvas, se observa que el *kernel Cosine* logra un rendimiento superior al *kernel* RBF. Esta superioridad tiene un sólido respaldo teórico: mientras que RBF se fundamenta en la distancia euclidiana, la cual se degrada severamente en espacios de alta dimensionalidad debido a la dispersión de los datos, el *kernel Cosine* evalúa el ángulo entre los vectores proyectados. Esto lo hace naturalmente robusto frente a la inmensa cantidad de ceros presentes en la matriz original, logrando capturar de mejor manera las estructuras latentes.

Por otro lado, la gráfica revela un comportamiento fundamental respecto al impacto de la varianza explicada en ambos *kernels*: a medida que se exige al modelo retener un mayor porcentaje de la varianza, el *Recall* del sistema disminuye progresivamente. Aunque de forma intuitiva podría asumirse que retener más información produce mejores resultados, en el contexto de datos de accesos altamente dispersos, exigir una alta varianza explicada obliga al modelo a capturar patrones residuales.

Es decir, superado el umbral del 50 % al 60 %, los componentes principales adicionales dejan de modelar las relaciones latentes generales y comienzan a memorizar asignaciones o permisos altamente específicos. En consecuencia, la inclusión de esta varianza introduce ruido en el hiperespacio, provocando un sobreajuste en las representaciones latentes y degradando la capacidad de generalización del recomendador. Finalmente, se concluye que retener entre un 50 % y un 60 % de la varianza utilizando un *kernel Cosine* constituye la parametrización óptima, logrando el equilibrio ideal entre la extracción de relaciones útiles y la filtración del ruido.

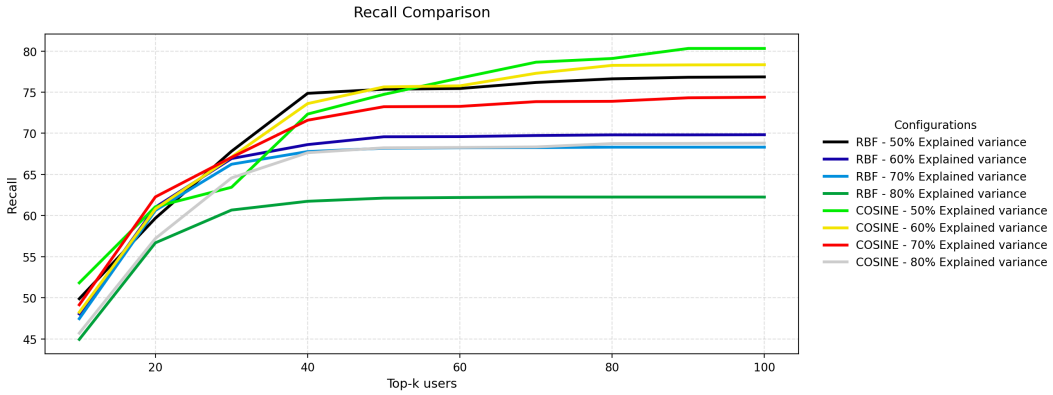


Figura 4.5: Comparativa de rendimiento entre *kernels Cosine* y *Radial Basis Function* (RBF) bajo distintos porcentajes de varianza explicada .

Una vez comprendido el comportamiento de los pesos en las métricas compuestas y seleccionada la configuración de kernel óptima, es posible realizar el análisis comparativo definitivo entre los tres enfoques de recomendación planteados.

La Figura 4.6 y la Figura 4.7 exponen los resultados obtenidos para cada uno de los escenarios.

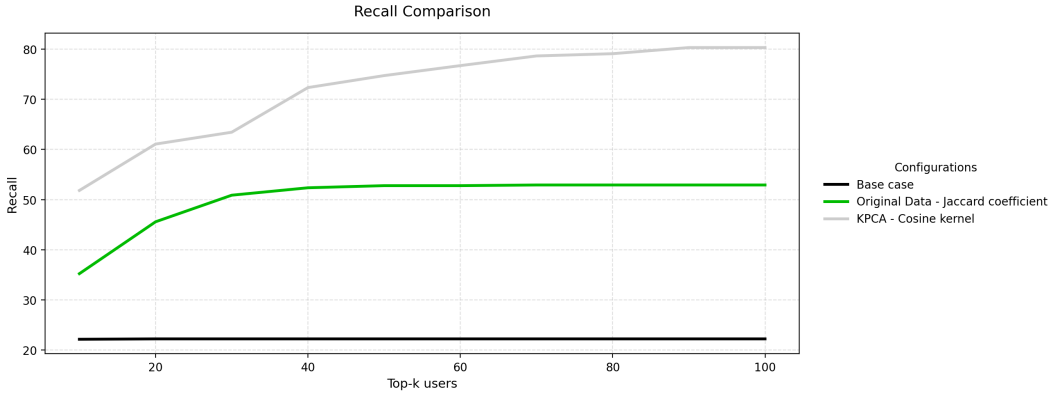


Figura 4.6: Comparativa de rendimiento entre enfoques de recomendación (Métrica de Recall).

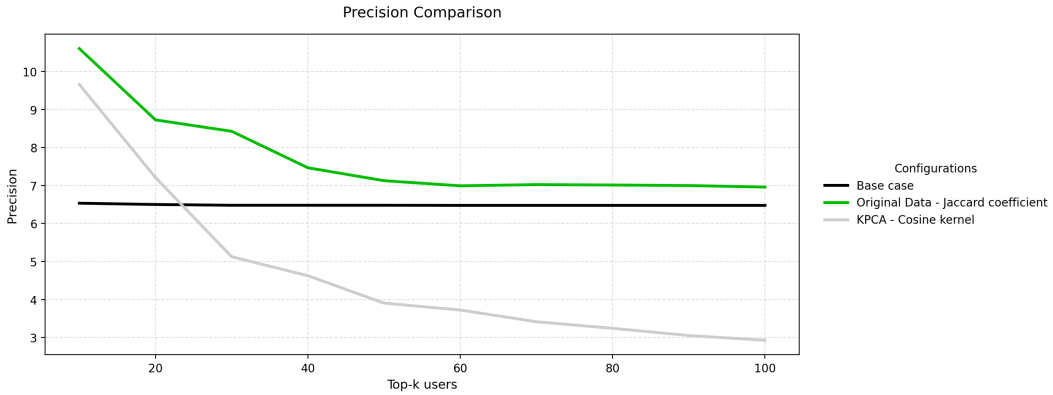


Figura 4.7: Comparativa de enfoques detallando el volumen de recomendaciones.

Como se evidencia claramente en la figura 4.6, los enfoques basados en similitud compuesta cumplen con el objetivo esperado, superando con creces el rendimiento del caso base. Adicionalmente, se confirma que la implementación de KPCA como técnica de reducción de dimensionalidad cumple exitosamente su propósito: el espacio latente logra capturar relaciones complejas que no están presentes de forma evidente en la matriz dispersa original, generando recomendaciones superiores a las obtenidas operando únicamente sobre los datos crudos.

Ahora bien, el análisis anterior se ha centrado de manera exclusiva en la métrica de *Recall* (la capacidad de encontrar los permisos correctos). Sin embargo, al evaluar la segunda métrica de control presente en la figura 4.7 se revela una limitación operativa importante. El sistema emite un número excesivamente elevado de sugerencias por cada trabajador. En términos prácticos, aunque el sistema logra recuperar la gran mayoría de los permisos reales del conjunto de pruebas, lo hace a expensas de una baja precisión, arrojando una lista inmanejable de falsos positivos (recomendaciones irrelevantes) que el usuario final tendría que depurar manualmente.

Tal como se anticipó en el diseño arquitectónico (M6), esta problemática de sobre-recomendación justifica de manera empírica la necesidad de implementar el sistema de clasificación supervisada (M7, M8 y M9). Este clasificador asume el rol de un filtro refinador de alta precisión, operando en cascada tras el filtro de similitud mínima, con el objetivo de descartar los falsos positivos y mejorar la calidad final de las sugerencias.

Para validar esta hipótesis, se replica la prueba de validación temporal detallada anteriormente, pero en esta ocasión evaluando el **modelo de recomendación híbrido**, el cual utiliza los clasificadores entrenados como última capa de decisión.

Las Figuras 4.8 y 4.9 exponen el rendimiento del sistema al implementar los clasificadores de *Gradient Boosting* como capa final de filtrado.

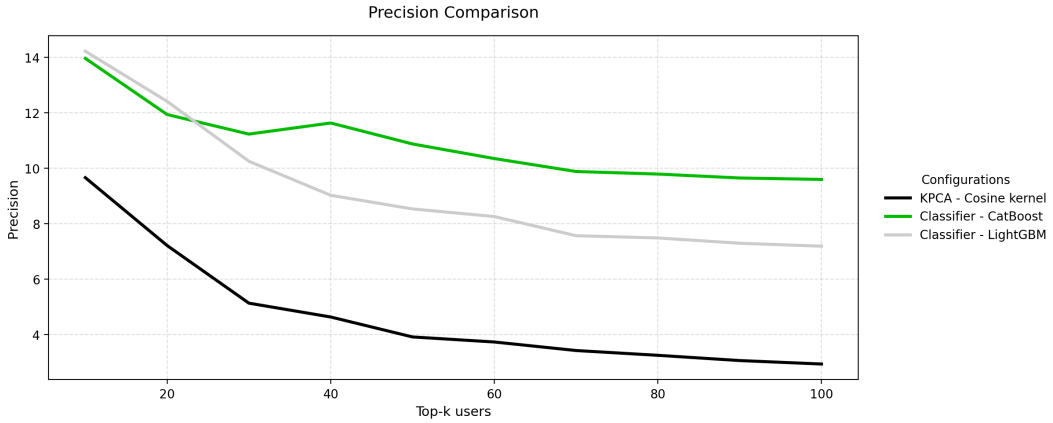


Figura 4.8: Comparación de la métrica de Precisión al implementar clasificadores supervisados.

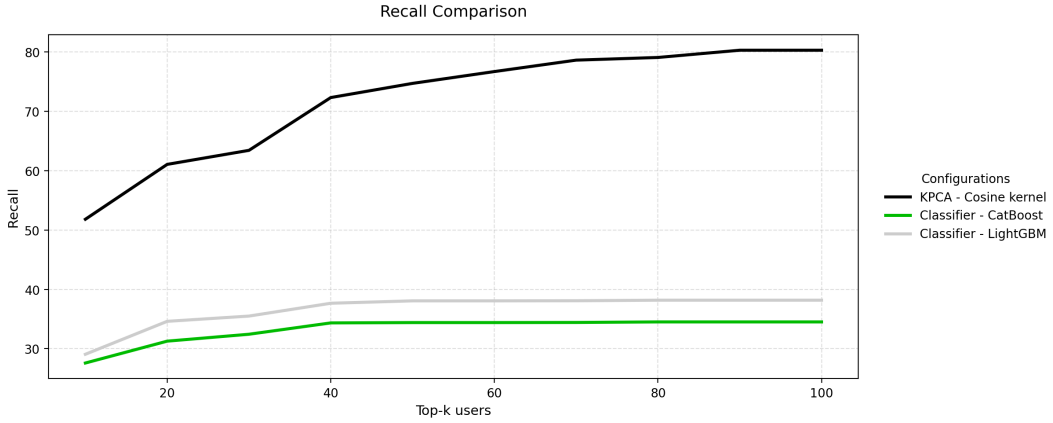


Figura 4.9: Comparación de la métrica de Recall al implementar clasificadores supervisados.

Al comparar los resultados, se evidencia que la arquitectura híbrida logra su objetivo primordial de incrementar drásticamente la precisión del sistema. Considerando el escenario con los 100 usuarios más similares, la precisión pasa de un 3 % (en el modelo base sin clasificador) hasta un 9.6 % utilizando CatBoost y un 7.1 % con LightGBM. Sin embargo, como es

propio en los sistemas de recomendación, este incremento tiene el costo de penalizar el *Recall*, el cual experimenta una caída significativa pasando del 80 % a un 34.5 % con CatBoost y un 38.1 % con LightGBM.

Para evaluar objetivamente si el *trade-off* entre la ganancia en precisión y la pérdida en *Recall* representa una verdadera mejora para el sistema, se analizó el *F1-Score*, métrica que cuantifica el equilibrio armónico entre ambos indicadores.

Los resultados de esta prueba, presentados en la Figura 4.10, evidencian una mejora sustancial en el *F1-Score* de la arquitectura híbrida. En el escenario *Top-100*, el valor base pasa de un 4.8 % a un 11.4 % utilizando CatBoost y a un 9.7 % con LightGBM. Esto confirma matemáticamente que la implementación del filtro mejora el rendimiento global del recomendador, al maximizar la proporción de predicciones correctas en relación directa con la cantidad de sugerencias emitidas.

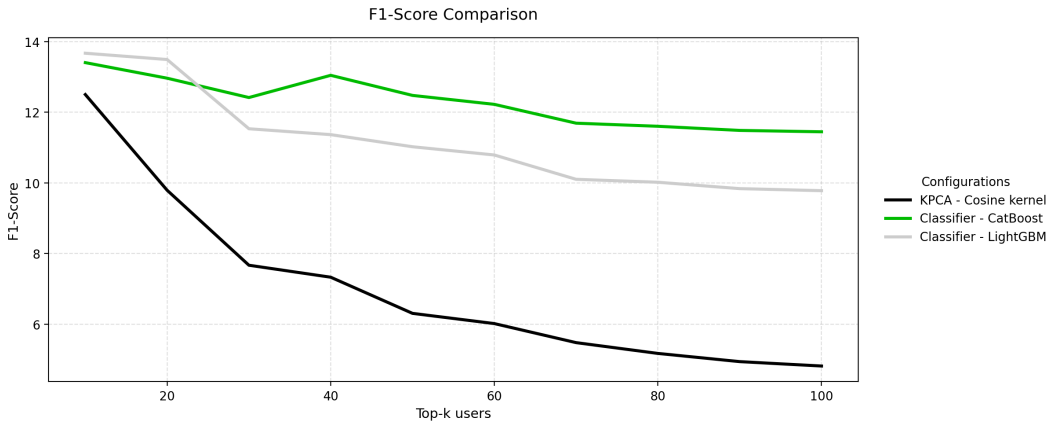


Figura 4.10: Comparación de la métrica F1-Score, evidenciando el equilibrio entre Precisión y Recall.

Al comparar el desempeño individual de ambos algoritmos, los resultados mantienen la misma tendencia observada durante la fase de entrenamiento: CatBoost exhibe un rendimiento general ligeramente superior frente a LightGBM. Si bien LightGBM logra retener un porcentaje marginalmente mayor de *Recall*, la superioridad de CatBoost en la métrica de precisión le permite alcanzar un *F1-Score* más alto, consolidándolo como el modelo de clasificación óptimo para este sistema.

Más allá del análisis matemático, el verdadero impacto operativo de la incorporación de

los clasificadores se ve reflejado en el volumen total de recomendaciones generadas. Mientras que el modelo de filtrado colaborativo puro generaba un promedio inmanejable de 108 sugerencias por trabajador, la capa de clasificación reduce esta cifra a tan solo 11 recomendaciones en el caso de CatBoost y 15 para LightGBM. Esta reducción de aproximadamente un 90 % en el volumen de salida limpia el ruido del sistema, transformando una lista sobredimensionada de recomendaciones en una herramienta de asistencia ágil, funcional y de alto valor estratégico para el equipo de Gestión de Accesos, cumpliendo así con el objetivo principal de este trabajo.

Capítulo 5

Conclusión

El objetivo principal del trabajo desarrollado consistió en crear una herramienta de apoyo para el equipo de Gestión de Accesos, diseñada para facilitar el mantenimiento del complejo sistema basado en roles (RBAC) implementado en la empresa CMPC. Dado que no se dispuso de acceso directo a la plataforma para desarrollar una solución de forma nativa, se optó por un enfoque predictivo. De este modo, se construyó un sistema de recomendación de asignación de roles capaz de sugerir proactivamente los permisos que un usuario requiere, disminuyendo así la necesidad de generar solicitudes manuales y asistiendo directamente en la toma de decisiones del departamento.

Durante las primeras instancias de implementación, se evidenció la importancia del correcto procesamiento de los datos corporativos base, lo que guio decisiones de diseño específicas. Una de las más relevantes fue la reducción del espacio dimensional de las variables, con el propósito de descubrir correlaciones ocultas en los perfiles de los trabajadores. La implementación de KPCA resultó ser un acierto técnico fundamental: no solo demostró capacidad para comprimir la información en un número significativamente menor de dimensiones, sino que cumplió exitosamente con el objetivo de revelar relaciones latentes entre los usuarios.

Si bien esta primera etapa otorgó al sistema la capacidad para generar recomendaciones correctas, la herramienta aún presentaba deficiencias operativas. Las sugerencias válidas se encontraban ocultas por una gran cantidad de “ruido”, producto de un elevado volumen de re-

comendaciones irrelevantes por usuario. Esta sobre-recomendación anulaba el valor práctico de la herramienta para la toma de decisiones. Para solucionar esta problemática, se evolucionó hacia una arquitectura de dos etapas: se conservó el modelo de KPCA como generador inicial de candidatos y se incorporó un filtro adicional mediante clasificadores basados en árboles de decisión. La integración de esta segunda capa logró el objetivo esperado, reduciendo drásticamente la cantidad de sugerencias a un volumen manejable y transformando el sistema en una herramienta verdaderamente funcional para el equipo de Gestión de Accesos.

A pesar de que los resultados demuestran el correcto funcionamiento de la herramienta desarrollada, el sistema presenta ciertas limitaciones inherentes a su diseño y alcance.

En primer lugar, destaca la fuerte simplificación aplicada en el módulo de *Parsing* y Extracción de Atributos del Rol. Esta decisión de diseño está sesgada hacia la estructura de datos específica de las sucursales de Brasil. En consecuencia, al intentar escalar el sistema a otras filiales de la empresa, es altamente probable que el módulo requiera ajustes para conservar una mayor cantidad de metadatos de los roles, permitiendo así capturar las particularidades locales y generar recomendaciones más específicas.

En segundo lugar, y estrechamente relacionado con el punto anterior, el alcance geográfico del modelo predictivo constituye una limitación para su generalización global. Al haber sido entrenado exclusivamente con datos de Brasil, no se puede garantizar que el rendimiento del algoritmo se mantenga estable al incorporar información de otros países con dinámicas organizacionales o nomenclaturas distintas. Esto sugiere que la herramienta podría requerir implementaciones y entrenamientos independientes por país o región. Como efecto colateral, esta segmentación por sucursales dificultaría la capacidad del sistema para descubrir y recomendar roles o permisos que queden fuera del entorno operativo del trabajador.

Finalmente, el sistema no está diseñado para manejar el problema de *Cold Start* (arranque en frío), el cual se presenta ante la incorporación de nuevos trabajadores o la creación de roles inéditos que carecen de asignaciones previas. Dado que el modelo genera la inferencia de permisos basándose fundamentalmente en la similitud entre usuarios, tendrá dificultades para emitir recomendaciones precisas hasta que estos nuevos elementos acumulen un historial suficiente de interacciones dentro del sistema.

Por estos motivos, en próximas iteraciones del proyecto se buscará desplegar este sistema en un entorno de producción para evaluar su rendimiento operativo real y probarlo progresivamente con datos de otras sucursales. Esta fase de pruebas permitirá analizar la escalabilidad de los algoritmos utilizados. En caso de evidenciarse una degradación en el rendimiento al incorporar nuevas filiales de la empresa, se contempla la exploración de nuevas variables, incorporando atributos adicionales de los trabajadores, e incluso la evaluación de algoritmos alternativos tanto para la reducción de dimensionalidad como para el filtrado final de las recomendaciones.

Por otra parte, queda como trabajo futuro la implementación de un bucle de retroalimentación. Este mecanismo cumplirá la función crítica de reentrenar periódicamente el clasificador utilizado como filtro, nutriéndose de las decisiones de aceptación o rechazo tomadas por el equipo de Gestión de Accesos. De esta forma, se garantizará que el modelo predictivo se adapte de manera continua y autónoma a la lógica cambiante del negocio y a la evolución natural de la estructura de roles de la empresa.

Bibliografía

[1] David F. Ferraiolo, Ravi Sandhu, Serban Gavrila, D. Richard Kuhn, and Ramaswamy Chandramouli. Proposed nist standard for role-based access control. *ACM Trans. Inf. Syst. Secur.*, 4(3):224–274, August 2001.

[2] Pete Epstein and Ravi Sandhu. Towards a uml based approach to role engineering. In *Proceedings of the fourth ACM workshop on Role-based access control*, pages 135–143, 1999.

[3] Alexander Fuchs and Günther Pernul. Hydro–hybrid development of roles. In *Information Systems Security*, pages 344–358. Springer, 2008.

[4] Mario Frank, Joachim M Buhmann, and David Basin. Role mining with probabilistic models. *ACM Transactions on Information and System Security (TISSEC)*, 15(4):1–28, 2013.

[5] Haibing Lu, Jaideep Vaidya, and Vijayalakshmi Atluri. Optimal boolean matrix decomposition: Application to role engineering. In *2008 IEEE 24th International Conference on Data Engineering*, pages 297–306, 2008.

[6] Hongyu Lu, Jaideep Vaidya, and Vijayalakshmi Atluri. Constraint-aware role mining via extended boolean matrix decomposition. *IEEE Transactions on Dependable and Secure Computing*, 9(5):655–669, 2012.

[7] Jian Guo, Ji Wang, Zhoujun Li, and Richard Harrison. An approach for hierarchical rbac reconfiguration with minimal perturbation. *IEEE Transactions on Dependable and Secure Computing*, 16(1):141–154, 2017.

[8] Yaqi Yang, Jun'e Li, Tao Zhang, Lu Chen, Guirong Huang, and Zhuo Lv. Irmaoc: an interpretable role mining algorithm based on overlapping clustering. *Cybersecurity*, 8(1):54, 2025.

[9] Fatemeh Alyari and Nima Jafari Navimipour. A survey of recommendation systems: Recommendation models, techniques, and application fields. *Electronics*, 11(1):141, 2022.

[10] F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh. Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, 16(3):261–273, 2015.

[11] Xiaoyuan Su and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, 2009:421425, 2009.

[12] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6):734–749, 2005.

[13] Michael J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13(5):393–408, 1999.

[14] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Application of dimensionality reduction in recommender system - a case study. In *ACM WebKDD Workshop*, 2000.

[15] C.-S. Hwang and Y.-C. Su. An improved collaborative movie recommendation system using computational intelligence. *Journal of Visual Languages & Computing*, 25(6):764–774, 2014.

[16] Yoon Ho Cho, Jae Kyeong Kim, and Soung Hie Kim. A personalized recommender system based on web usage mining and decision tree induction. *Expert Systems with Applications*, 23(3):329–342, 2002.

[17] Alfred Krzywicki, Wayne Wobcke, Xiongcai Cai, Ashesh Mahidadia, Paul Compton, and Yang Sok Kim. Collaborative filtering for people-to-people recommendation in online dating: Data analysis and user trial. *Knowledge-Based Systems*, 73:50–66, 2015.

[18] Hugo Hernandez. Quantitative analysis of categorical variables. *ForsChem Research Reports*, 6:1–25, 2021.

[19] Furkat Bolikulov, Rashid Nasimov, Akbar Rashidov, Farkhod Akhmedov, and Young-Im Cho. Effective methods of categorical data encoding for artificial intelligence algorithms. *Mathematics*, 12(16):2553, 2024.

[20] Andréia da Silva Meyer, Antonio Augusto Franco Garcia, Anete Pereira de Souza, and Cláudio Lopes de Souza Jr. Comparison of similarity coefficients used for cluster analysis with dominant markers in maize (zea mays l). *Genetics and Molecular Biology*, 27:83–91, 2004.

[21] Soheil Feizi and David Tse. Maximally correlated principal component analysis. *arXiv preprint arXiv:1702.05471*, 2017.

[22] Bernhard Schölkopf, Alexander Smola, and Klaus-Robert Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural computation*, 10(5):1299–1319, 1998.

[23] Qingzhen Li, Jiufen Zhao, and Xiaoping Zhu. A kernel pca radial basis function neural networks and application. *2006 9th International Conference on Control, Automation, Robotics and Vision*, 2000.

[24] Jiazhi Jiao et al. Kernel pca for out-of-distribution detection. *arXiv preprint arXiv:2402.02949*, 2024.

[25] M. Al-Sheikh et al. Performance investigation of principal component analysis for intrusion detection system using different support vector machine kernels. *Electronics*, 11(21):3571, 2022.

[26] John T Hancock and Taghi M Khoshgoftaar. Medicare fraud detection using catboost. In *2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science (IRI)*, pages 97–103. IEEE, 2020.

[27] Alice Villar and Carolina Andrade. Supervised machine learning algorithms for predicting student dropout and academic success: a comparative study. *Discover Artificial Intelligence*, 4, 01 2024.

[28] The pandas development team. pandas-dev/pandas: Pandas, February 2020.

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.