



Tesis
Automatización de lectura de Curriculum Vitae
para selección de personal en el sector IT

Calonge, Federico Matias
calongefederico@gmail.com

10 de abril de 2022

Resumen

En la Tesis de Ingeniería en Informática que se presenta, se diseña un *sistema de lectura automática de Curriculum Vitae* accesible vía Web. La finalidad del mismo es ayudar al reclutador laboral a elegir a los mejores candidatos para los puestos laborales de IT que tenga disponible. Esta elección se realiza mediante el uso de algoritmos de *machine learning* y basándose, principalmente, en una medición de similitud entre textos: Curriculum Vitae de los candidatos por un lado, y descripciones de los puestos laborales de IT por el otro. El sistema esta desarrollado utilizando el lenguaje de programación Python, permitiendo verificar la teoría desarrollada.

Abstract

This Computer Engineering Thesis introduces an *automatic Curriculum Vitae reading system* accessible via the Web. The purpose of it is to help the job recruiter to choose the best candidates for the available IT job positions. This choice is made through the use of *machine learning* algorithms and based mainly on a measurement of similarity between texts: Curriculum Vitae of the candidates on the one hand, and IT job descriptions on the other hand. The system is developed using the Python programming language allowing to verify the developed theory.

Índice

1. Introducción.	6
1.1. Objetivos del Proyecto.	9
1.1.1. Objetivo general.	9
1.1.2. Objetivos específicos.	9
1.2. Alcance del Proyecto.	10
1.3. Organización.	10
2. Reclutamiento y selección laboral.	12
2.1. Introducción.	12
2.2. Reclutamiento vs selección.	12
2.3. Evolución de los procesos de reclutamiento y selección laboral.	14
2.4. Cribado o screening.	15
2.4.1. Screening manual vs screening automatizado	15
2.5. Sistemas de screening: Estado del arte.	16
2.6. Enfoque del Proyecto.	18
3. Algoritmos de Machine Learning.	19
3.1. Introducción.	19
3.2. Machine Learning (ML).	19
3.2.1. Aprendizaje supervisado y no supervisado.	20
3.2.1.1. Regresión.	21
3.2.1.2. Clasificación.	22
3.2.1.3. Agrupación (clustering).	23
3.2.1.4. Algoritmos más conocidos.	24
3.2.2. Aprendizaje transductivo	25
3.2.3. Separación de los datos.	26
3.2.4. ¿Cómo implementar un modelo de ML?	27
3.2.4.1. Cross Validation.	31
3.2.4.2. Los Problemas de ML: Overfitting y Underfitting.	32
4. Natural Language Processing.	32
4.1. Introducción.	33

4.2.	Preprocesamiento de textos.	34
4.3.	Similitud entre textos.	35
4.4.	Técnicas para medir Similitud entre textos.	36
4.4.1.	Cosine Similarity.	36
4.4.2.	Word Mover's Distance (WMD).	37
4.5.	Algoritmos de vectorización.	38
4.5.1.	TF-IDF.	38
4.5.2.	Word Embeddings.	39
4.5.2.1.	¿Cómo entrenar Word Embeddings?	40
5.	Implementación.	41
5.1.	Obtención del modelo de clasificación.	42
5.1.1.	Introducción.	42
5.1.2.	Esquema.	42
5.1.3.	Obtención de sets de datos.	43
5.1.3.1.	Curriculum Vitae.	43
5.1.3.2.	Descripciones Puestos Laborales.	43
5.1.4.	Preprocesamiento y limpieza de datos.	44
5.1.5.	Cantidad final del set de datos y su uso en las distintas etapas.	45
5.2.	Comparando textos y obteniendo similitudes.	47
5.3.	Armado del modelo de clasificación KNN.	48
5.4.	Clasificación de nuevas muestras y resultados obtenidos.	49
5.5.	Integración al Sistema Web.	50
5.5.1.	Base de datos.	51
5.5.2.	Secciones del sistema	53
5.5.3.	Manejo de los datos.	56
5.5.3.1.	Modelado.	57
5.5.3.2.	Filtrado.	58
5.5.3.3.	Visualización.	59
5.6.	Pipeline Flow final del Sistema.	60
5.7.	Conclusiones.	61
5.8.	Caso de Uso.	62
5.9.	Limitaciones del sistema.	63

6. Próximos pasos.	64
7. Anexos.	65

1 Introducción.

Los procesos de *reclutamiento y selección laboral* se han vuelto cruciales para el manejo de recursos humanos en el mundo moderno. Con las transformaciones digitales de las empresas y del mercado laboral en general, identificar los perfiles más acordes a las necesidades de la empresa se convirtió en uno de los retos más ambiciosos de Recursos Humanos, en especial cuando hablamos del *Sector IT*, donde año tras año se van generando nuevos puestos de trabajo y estos mismos van creciendo en demanda. Este crecimiento de la demanda en distintos puestos del Sector IT lo podemos evidenciar, a modo de resumen, en la figura 1.1.

En estos últimos años se implementaron una gran cantidad de herramientas de Software que utilizan algoritmos inteligentes y que permiten automatizar y gestionar información de los candidatos de una manera mucho más intuitiva [6][7][9][10][11][12][13][14][15][16][17][18][19][20][21]. La *automatización* se transformó en un factor clave e indispensable para que el reclutador pueda conseguir a los candidatos más relevantes para los puestos que ofrece la empresa en un tiempo muy corto. Se espera que para el año 2025 muchas tareas realizadas actualmente por humanos, sean automatizadas por máquinas: esto lo podemos evidenciar en la figura 1.2.

↗ Increasing demand		↘ Decreasing demand	
1	Data Analysts and Scientists	1	Data Entry Clerks
2	AI and Machine Learning Specialists	2	Administrative and Executive Secretaries
3	Big Data Specialists	3	Accounting, Bookkeeping and Payroll Clerks
4	Digital Marketing and Strategy Specialists	4	Accountants and Auditors
5	Process Automation Specialists	5	Assembly and Factory Workers
6	Business Development Professionals	6	Business Services and Administration Managers
7	Digital Transformation Specialists	7	Client Information and Customer Service Workers
8	Information Security Analysts	8	General and Operations Managers
9	Software and Applications Developers	9	Mechanics and Machinery Repairers
10	Internet of Things Specialists	10	Material-Recording and Stock-Keeping Clerks
11	Project Managers	11	Financial Analysts
12	Business Services and Administration Managers	12	Postal Service Clerks
13	Database and Network Professionals	13	Sales Rep., Wholesale and Manuf., Tech. and Sci.Products
14	Robotics Engineers	14	Relationship Managers
15	Strategic Advisors	15	Bank Tellers and Related Clerks
16	Management and Organization Analysts	16	Door-To-Door Sales, News and Street Vendors
17	FinTech Engineers	17	Electronics and Telecoms Installers and Repairers
18	Mechanics and Machinery Repairers	18	Human Resources Specialists
19	Organizational Development Specialists	19	Training and Development Specialists
20	Risk Management Specialists	20	Construction Laborers

Figura 1.1: Top 20 demanda de roles laborales en aumento y disminución para el año 2020, por participación de las empresas encuestadas por el *Foro Económico Mundial*[1].

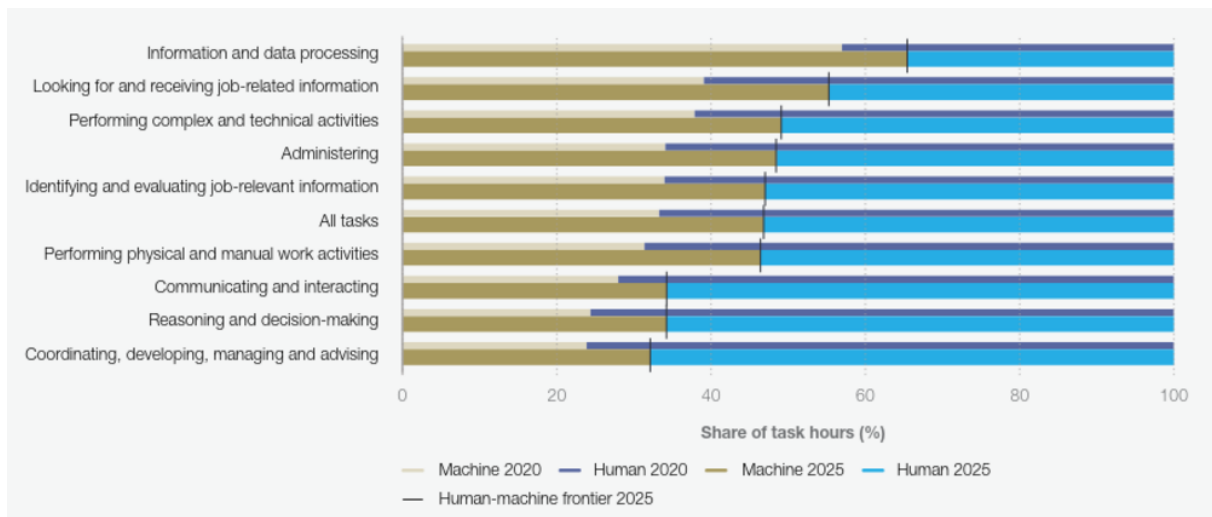


Figura 1.2: Porcentaje de tareas realizadas por humanos frente a máquinas, 2020 y 2025 (previsto), por participación de las empresas encuestadas por el *Foro Económico Mundial*[1].

El tema de este Proyecto de Tesis será desarrollar un *sistema de lectura automática de Curriculum Vitae* accesible vía Web. La finalidad del mismo es ayudar al reclutador laboral a elegir a los mejores candidatos para los puestos laborales de IT que tenga disponible. Esta elección se realiza mediante el uso de algoritmos de Machine Learning y basándose, principalmente, en una medición de similitud entre textos: Curriculum Vitae de los candidatos por un lado, y descripciones de los puestos laborales de IT por el otro.

La medición de similitudes de documentos es uno de los problemas más cruciales del *Procesamiento del Lenguaje Natural (NLP)*. Encontrar similitudes entre documentos se utiliza en varios dominios, tales como recomendación de películas, libros o artículos similares, identificación de documentos plagiados o documentos legales, etc.

Para que las máquinas puedan descubrir esta similitud entre documentos, se necesita definir una forma de medir matemáticamente la similitud, la cual debe ser comparable para que la máquina pueda identificar qué documentos son más similares (o menos). Previamente a esto necesitamos representar el texto de los documentos en una forma cuantificable (que suele ser en forma vectorial), de modo que podamos realizar cálculos de similitud sobre él.

Por lo tanto, los pasos necesarios para que las máquinas puedan medir la similitud entre documentos son:

1. Convertir un documento en un objeto matemático (vector).
2. Definir y emplear una medida de similitud.

Para el primer paso se utilizarán los algoritmos de vectorización **TF-IDF** y **Word Embeddings**; y para el segundo paso se emplearán las técnicas **Cosine Similarity** y **Word Mover's Distance -WMD-**.

Una vez obtenidas estas mediciones de similitud entre los Curriculum Vitae de los candidatos y las descripciones de los puestos laborales de IT, estos valores se utilizarán para alimentar un **algoritmo de clustering K-means** que a su vez, con sus datos de salida (4 clusters), alimentará a un **modelo de clasificación KNN**. Finalmente este modelo KNN nos servirá para lograr, en base a los valores de similitud de nuevos candidatos, clasificar qué tan similares son dichos candidatos con respecto a la descripción de un puesto de IT: similitud escasa, similitud media, similitud alta, similitud muy alta.

1.1 Objetivos del Proyecto.

1.1.1 Objetivo general.

El objetivo de este Proyecto de Tesis es lograr un desarrollo, tanto teórico como práctico, de un *sistema de lectura automática de Curriculum Vitae* accesible vía Web. La finalidad del mismo es ayudar al reclutador laboral a elegir a los mejores candidatos para los puestos laborales de IT que tenga disponible. Esta elección se realiza mediante el uso de algoritmos de Machine Learning y basándose, principalmente, en una medición de similitud entre textos:

- los Curriculum Vitae de los candidatos por un lado,
- y descripciones de los puestos laborales de IT por el otro.

1.1.2 Objetivos específicos.

Los objetivos específicos de este Proyecto de Tesis son:

- Describir el estado del arte actual de los Sistemas de lectura y análisis de Curriculum Vitae en las fases de reclutamiento y selección laboral.
- Implementar un Sistema de lectura automática de Curriculum Vitae basado en la comparación y medición de similitudes entre textos, para finalmente obtener una visualización de los mejores candidatos para un puesto laboral de IT determinado.
- Aprender los conceptos y técnicas principales utilizadas dentro del procesamiento de lenguaje natural (NLP) aplicando técnicas de preprocesamiento y limpieza de textos.
- Implementar diferentes técnicas para medir similitudes entre los textos (Cosine Similarity y Word Mover's Distance -WMD-) y diferentes algoritmos de vectorización (TF-IDF y Word Embeddings), analizando su funcionamiento tanto teórica como matemáticamente, ventajas y desventajas.
- Conocer, implementar e integrar el algoritmo de clustering K-means junto al algoritmo de clasificación KNN para obtener un modelo de clasificación de candidatos en base a las medidas de similitud entre los textos.
- Evaluar los Frameworks disponibles para tener una UI ¹ accesible vía web e integrar el mismo al Sistema.
- Almacenar datos de candidatos, reclutadores y puestos laborales en una base de datos.

¹La interfaz de usuario o user interface (UI) de una página web refiere a todo aquello tangible con lo que los usuarios interactúan de forma directa en la misma.

1.2 Alcance del Proyecto.

El alcance de esta Tesis de Grado de Ingeniería incluye el desarrollo de conceptos de análisis de datos y machine learning, procesamiento de lenguaje natural, técnicas de preprocesamiento y limpieza de los datos, algoritmos de vectorización y técnicas para medir la similitud entre textos, algoritmos de clasificación y clustering, integración con frameworks, visualización de datos, y gestión de Base de Datos, de acuerdo a lo enunciado en los objetivos específicos.

1.3 Organización.

Este Proyecto de Tesis fue organizado para trabajarlo en tres secciones:

1. Análisis e investigación inicial.

Esta sección abarca principalmente la parte teórica del trabajo, haciendo hincapié en el análisis e investigación de:

- El estado del arte (actual y pasado) de los sistemas de lectura y análisis de Curriculum Vitae.
- Técnicas usadas para el procesamiento del lenguaje natural (NLP).
- Técnicas para medir similitudes entre textos: Cosine Similarity y WMD.
- Algoritmos de Vectorización: TF-IDF y Word Embeddings.
- Algoritmos de Machine Learning para tareas de clasificación (KNN) y clustering (K-means).

Esta primera sección abarca los capítulos *Reclutamiento y selección laboral*, *Algoritmos de Machine Learning* y *Natural Language Processing* de este Informe de Tesis.

2. Investigación e implementación de distintas técnicas y algoritmos para la obtención del modelo de clasificación KNN.

Esta sección hace referencia a la aplicación práctica dentro del marco teórico desarrollado en la primera sección, mediante la realización de una serie de análisis en documentos de Jupyter Notebook ² utilizando Python ³, para la obtención final de nuestro modelo de clasificación KNN capaz de clasificar, en base a los valores de similitud de nuevos candidatos, qué tan similares son dichos candidatos con respecto a la descripción de un puesto de IT: similitud escasa, similitud media, similitud alta, similitud muy alta.

²Aplicación cliente-servidor que permite crear documentos web en formato JSON que siguen un esquema versionado y una lista ordenada de celdas de entrada y de salida. Estas celdas albergan, entre otras cosas, código, texto (en formato Markdown), fórmulas y ecuaciones matemáticas. Estos documentos que se generan funcionan en cualquier navegador estándar.

³Lenguaje de programación interpretado y multiplataforma de código abierto, popularizado en los últimos años por su facilidad para trabajar con inteligencia artificial, big data, machine learning y data science, entre muchos otros campos en auge.

Los items que abarca esta sección son:

- Obtención de sets de datos: curriculums vitae y descripciones laborales.
- Preprocesamiento de los textos.
- Comparación entre textos y obtención de similitudes entre los mismos mediante el uso de las técnicas para medir distancias y obtener dichas similitudes (WMD y Cosine Similarity) y los algoritmos de vectorización (TF-IDF y Word Embeddings).
- Obtención del modelo de clasificación KNN utilizando como datos de entrada los clusters devueltos por el algoritmo K-means obtenidos en base a las mediciones de similitud previamente realizadas.
- Análisis y primeras visualizaciones de los resultados.

Esta sección abarca el capítulo *Implementación* (desde *Obtención del modelo de clasificación* hasta *Clasificación de nuevas muestras y resultados obtenidos*) de este Informe de Tesis.

3. Investigación e integración al sistema web.

Esta última sección hace referencia a la reutilización de las funciones que contenían la lógica de los distintos algoritmos utilizados junto con el modelo de clasificación KNN obtenidos previamente en la sección 2, para integrar todo esto en el sistema Web que, a su vez, está integrado a una base de datos relacional. De esta manera, el sistema cuenta con una interfaz gráfica permitiendo interactuar entre candidatos y reclutadores y, principalmente, permitiendo que el reclutador sea capaz de obtener un listado con los N candidatos más similares a un puesto determinado, y ordenados de mayor a menor de acuerdo a esta *similitud*. Dicha *similitud* representa el resultado obtenido de la clasificación por nuestro modelo KNN.

Los items principales de esta sección son:

- Definición de los usuarios que accederán al sistema.
- Definición de los datos que se almacenarán.
- Integración de frameworks y bases de datos.
- Modelado, filtrado y visualización de los datos.
- Reutilización e integración al sistema de los algoritmos y del modelo KNN utilizados en la fase previa.
- Evaluación del funcionamiento de todo el Sistema integrado.

Esta última sección abarca el capítulo *Implementación* (desde *Integración al Sistema Web* hasta el final del capítulo) de este Informe de Tesis.

2 Reclutamiento y selección laboral.

En este capítulo se va a realizar una introducción a los procesos de reclutamiento y selección laboral, sus diferencias y diferentes tareas involucradas. Por último, se llevará a cabo un análisis del Estado de Arte actual de los sistemas de cribado (o más conocidos como sistemas de *screening*) y se detallará el enfoque utilizado para este Proyecto.

2.1 Introducción.

Los procesos de *reclutamiento y selección laboral* se han vuelto cruciales para el manejo de recursos humanos en el mundo moderno. Con las transformaciones digitales de las empresas y del mercado laboral en general, identificar los perfiles más acordes a las necesidades de la empresa se convirtió en uno de los retos más ambiciosos de Recursos Humanos, en especial cuando hablamos del *Sector IT*, donde año tras año se van generando nuevos puestos de trabajo y estos mismos van creciendo en demanda.

Los *reclutadores*, dentro de un departamento de recursos humanos, son los encargados de llevar a cabo los procesos de *reclutamiento y selección* laboral. Uno de sus objetivos principales es buscar talento humano para cubrir los puestos de trabajo vacantes que tenga la empresa.

2.2 Reclutamiento vs selección.

El **reclutamiento** es el proceso de atracción, búsqueda, recolección e identificación de candidatos que encajan con la oferta de trabajo y, en definitiva, con la empresa. El objetivo del reclutamiento es *atraer, buscar e identificar* a los candidatos más adecuados y mejor calificados para el puesto disponible, según las necesidades de la empresa[3].

El proceso de reclutamiento incluye las siguientes actividades:

- Identificación de las necesidades del puesto a cubrir.
- Análisis de la descripción y especificaciones del puesto.
- Identificación de posibles fuentes de candidatos cualificados para el puesto.
- Publicación del puesto vacante en dichas fuentes.
- Atracción de candidatos para aplicar al puesto.
- Manejo apropiado en las respuestas y en los escrutinios a las postulaciones.

En cambio, la **selección** es un proceso posterior al reclutamiento, donde se evalúa más detalladamente y se entrevista a los candidatos para el trabajo en particular. El objetivo de la selección es *elegir y hacer efectiva la contratación* del candidato más adecuado y mejor calificado para el puesto disponible, según las necesidades de la empresa. Este procedimiento particiona a los candidatos en dos secciones: a los que se les ofrecerán el trabajo, y a los que se descartarán[3].

El proceso de selección incluye las siguientes actividades:

- Recepción de la aplicación al puesto.
- Cribado o Screening de los candidatos, lo que permite avanzar con los candidatos adecuados y descartar a los no adecuados para el puesto.
- Entrevistas a los candidatos.
- Manejo de tests a los candidatos, tales como tests médicos o psicológicos.
- Manejo de exámenes a los candidatos, tales como exámenes técnicos, de aptitud, inteligencia, performance, etc.
- Evaluación de las referencias de los candidatos.
- Decisión final acerca de la contratación o no contratación del candidato.

Como conclusión, podemos decir que en la fase de **reclutamiento** se trata de encontrar muchos candidatos que cumplan con los requisitos de la oferta; mientras que en la etapa de **selección** se debe elegir al mejor candidato para las necesidades de la empresa.

2.3 Evolución de los procesos de reclutamiento y selección laboral.

Los procesos de reclutamiento y selección laboral fueron evolucionando a lo largo del tiempo[16]:

En los modelos de reclutamiento y selección de **primera generación**, las empresas anunciaban sus vacantes de puestos laborales en diarios, revistas, radio y en televisión. Los candidatos enviaban sus currículums por correo postal y los mismos se clasificaban manualmente: algo muy tedioso y que llevaba mucho tiempo en realizar. Una vez preseleccionados los candidatos, los reclutadores llamaban a los mismos para realizar las rondas de entrevistas.

Luego pasamos a la **segunda generación**. En esta época las empresas comenzaron a crecer y también lo hicieron las necesidades de reclutamiento y selección. Las empresas empezaron a subcontratar sus procesos de reclutamiento y selección, naciendo de esta manera las consultoras o agencias de contratación. Estas consultoras requerían que los candidatos cargaran sus currículums en sus sitios web en formatos particulares. Luego, las consultoras revisaban los datos de los candidatos y preseleccionaban a los mismos para la empresa. El gran inconveniente de este proceso fue que habían numerosas consultoras y cada una tenía su propia y única forma de selección, no era un proceso uniforme.

Para intentar superar los problemas anteriores, se llegó a una **tercera generación**, en la que estamos actualmente. En esta generación se crearon, y siguen creándose, una gran cantidad de herramientas de Software que utilizan algoritmos inteligentes y permiten automatizar y gestionar información de los candidatos de una manera mucho más intuitiva. Estos sistemas ayudan a los reclutadores dentro de las empresas y consultoras a analizar la información de cualquier Curriculum Vitae y clasificarlos o listarlos en función de los puestos disponibles. De esta manera, cuando el reclutador publica una oferta de trabajo, estos sistemas clasifican o listan a los currículums basándose en distintas métricas (por ejemplo palabras clave) mostrando así los candidatos más relevantes para la empresa o consultora.

2.4 Cribado o screening.

Como vimos anteriormente en *Reclutamiento vs selección*, el screening (o también conocido como cribado) es una etapa del proceso de selección. En esta etapa los reclutadores revisan los Currículums Vitae que fueron recibiendo por parte de los candidatos, y preseleccionan a los que mejor se adapten a los requisitos de la oferta de empleo de la empresa. Este proceso es muy importante dentro del proceso de selección, ya que permite valorar en ese momento si el candidato es apto para continuar en el proceso de selección o, en caso contrario, si el mismo se descarta por no considerarlo adecuado para el puesto.

2.4.1 Screening manual vs screening automatizado

Si el proceso de screening sigue el modelo tradicional y se realiza manualmente, es un proceso muy tedioso y que lleva mucho tiempo por parte del reclutador, el cual tiene que evaluar una gran cantidad de Currículums Vitae. Existe un estudio[5] realizado en 2018 por Ladders Inc., empresa líder en sitios de carrera, donde se estudió cuánto tiempo en promedio tarda un reclutador en dar una primera vista rápida a un Curriculum Vitae de un candidato. Se descubrió que este tiempo es, en promedio, de 7.4 segundos. Este es el tiempo en el que el reclutador decide inicialmente si el candidato sigue (o no) con el proceso de selección.

Sin embargo, este tiempo es engañoso: ya que este estudio fue realizado con la aplicación de muchos Currículums vitae que no cumplían con los criterios mínimos para calificar a los puestos; es por esto que los reclutadores realizaban una primer vista rápida y los descartaban (o no) a los pocos segundos. Si las aplicaciones provinieran de Currículums Vitae que cumplan con los requisitos mínimos del puesto, este tiempo de observación sería mucho mayor, ya que el reclutador examinaría con mayor detalle los curriculums.

Adicionalmente, este estudio tampoco tiene en cuenta el tiempo consumido por el reclutador al comparar el Curriculum Vitae del candidato con la descripción o requisitos del puesto laboral, por lo que el tiempo en realizar un screening manual se incrementaría significativamente.

Otra de las desventajas al considerar el screening como un proceso manual, es que muchos son los factores que pueden influir al reclutador en el momento de tomar la decisión de descarte o selección del candidato, ya sea cansancio por el volumen de los curriculums ya revisados, la estructura, elementos discriminatorios o información incompleta dentro de los mismos, etc. Es por esto que las probabilidades de descartar prematuramente a un candidato válido son muy altas.

Frente a estas desventajas del modelo tradicional y poco eficiente del screening manual existe una solución: la *automatización*. Actualmente existen sistemas automatizados y basados en Inteligencia Artificial que permiten realizar el screening de un volumen importante de curriculums en unos pocos segundos (Ver *Sistemas de screening: Estado del arte*). De esta manera, el proceso de screening no estaría afectado a factores externos que puedan influir en la decisión del reclutador, y además la herramienta sería capaz de entregar un listado justificado de los mejores candidatos para la siguiente fase del proceso en un tiempo relativamente corto.

Uno de los beneficios potenciales de utilizar sistemas automatizados e inteligentes en los procesos de screening es el acortamiento de los ciclos de contratación, lo que hace que la organización responda mejor a los solicitantes y sea más capaz de competir con otras organizaciones por los mejores candidatos[2].

Muchos gerentes de recursos humanos depositan sus esperanzas en la tecnología y las herramientas automatizadas e inteligentes: desde el aumento de la eficiencia y la reducción de costos hasta el aumento de las aplicaciones de candidatos y la estandarización de todos sus sistemas de selección[2].

2.5 Sistemas de screening: Estado del arte.

Tal como mencionamos en la sección anterior, un sistema de screening automatizado e inteligente nos permite entregar al reclutador una shortlist con los candidatos que mejor se adaptan a los requisitos de la oferta de empleo de la empresa. De esta manera, al realizar dicha preselección de candidatos, el reclutador podrá seguir con las etapas posteriores de selección o podrá realizar otro screening manual para descartar a otros candidatos.

A continuación detallaremos una serie de trabajos de relevancia (tanto de implementación como de investigación) donde se utilizaron distintos enfoques para desarrollar o sugerir un desarrollo de un sistema de screening de candidatos.

En [6] utilizaron el *algoritmo KNN* para clasificar los curriculums de los candidatos en diferentes categorías, y luego utilizaron la métrica de similitud *Cosine Similarity* para averiguar qué tan cerca está el curriculum del candidato con respecto a la descripción de los puestos, y realizar un ranking acorde a estos resultados.

En [7] utilizaron una herramienta inteligente llamada *EXPERT mapping-based candidate screening* que utiliza *ontology mapping*⁴ para crear un sistema automatizado para el screening inteligente de candidatos.

En [9] y [10] también se utilizan sistemas basados en *ontologías* para extraer datos de curriculums y realizar macheos con los puestos disponibles.

En [11] sugieren un método de matching basado en un *modelo probabilístico* para ser utilizado en la selección y recomendación de candidatos.

En [12] se propuso un sistema automatizado de screening de currículums que usa *Vector Space Model*⁵ para machear cada curriculum con la descripción del puesto correspondiente y *cosine similarity* como medida de similitud.

⁴La *ontología (ontology)* permite la especificación explícita de un dominio de discurso, que permite acceder y razonar sobre el conocimiento de un agente. Las ontologías elevan el nivel de especificación del conocimiento, incorporando semántica a los datos. El *mapeo de ontologías (ontology mapping)* es el proceso mediante el cual dos ontologías se relacionan semánticamente a nivel conceptual y las instancias de ontología de origen se transforman en entidades de ontología de destino de acuerdo con esas relaciones semánticas.[8]

⁵Modelo algebraico para representar documentos de textos en lenguaje natural de una manera formal mediante el uso de vectores de identificadores. Es utilizado para la recuperación, filtrado, indexado y cálculo de relevancia de información.

En [13] y [14] se propuso un sistema para el screening de candidatos, para el cual se utiliza *cosine similarity*, *KNN* y un *sistema de recomendación basado en contenido* (*Content Based Filtering, CBF*) para buscar los curriculums más cercanos a la descripción del puesto.

En [15] se propuso un sistema para el ordenamiento y clasificación de curriculums utilizando el concepto de inteligencia artificial (sin especificar qué algoritmos o técnicas específicas se deberían usar). Lo que se propuso fue un esquema de trabajo para luego seguir con la implementación del sistema; el cual permitiría clasificar a todos los curriculums de acuerdo con los requisitos de la empresa y los enviaría a Recursos Humanos para su posterior consideración.

En [16] se diseñó un sistema que ayuda a los reclutadores a seleccionar los curriculums basados en la descripción del trabajo. Ayuda en un proceso de contratación fácil y eficiente al extraer los requisitos automáticamente. Este sistema utiliza un *modelo NER* (*Named entity recognition*)⁶.

En [17] y [18] se implementan sistemas basados en *semantic annotation*⁷ y *ontologías*, permitiendo realizar macheos entre las ofertas de trabajo y los curriculums de los candidatos.

En [19] se utiliza *relevance feedback*⁸ para la implementación de un sistemas que intenta encontrar a los mejores candidatos para las distintas ofertas de trabajo.

Paul Resnick y *Hal R. Varian* fueron pioneros en los *sistemas de recomendación*⁹[22]. En [20] y [21] se utilizaron dichos *sistemas de recomendación* junto a otros algoritmos para implementar aplicaciones de screening y clasificación de candidatos. Esto es debido a que, dado que el perfil del candidato es necesario para un puesto en particular, la forma en que se comparan los curriculums de los candidatos es muy similar a un sistema de recomendación.

⁶Subtarea de *information extraction (IE)* que permite buscar y categorizar entidades específicas en un cuerpo o cuerpos de textos.

⁷Proceso de etiquetado de documentos con conceptos relevantes. Los documentos se enriquecen con metadatos que permiten vincular el contenido a los conceptos, descritos en un gráfico de conocimiento. Esto hace que el contenido no estructurado sea más fácil de encontrar, interpretar y reutilizar.

⁸Característica de algunos sistemas de *information retrieval (IR)*. La idea de *relevance feedback* es tomar los resultados que se devuelven inicialmente de una consulta/query determinada, recopilar los comentarios de los usuarios y usar información para evaluar si esos resultados son relevantes o no para realizar una nueva consulta/query.

⁹Un *sistema de recomendación* es una subclase de los sistemas de *Information filtering (IF)* que busca predecir la calificación o la preferencia que un usuario le puede dar a un artículo. En palabras simples, es un algoritmo que sugiere artículos relevantes para los usuarios.

2.6 Enfoque del Proyecto.

Como vimos en la sección anterior, hay cientos de enfoques y combinaciones posibles para realizar un sistema de screening automático e inteligente.

El enfoque elegido para este Proyecto de Tesis fue realizar un sistema híbrido que utilice técnicas de mediciones de similitud entre los curriculums de los candidatos y las descripciones de los puestos disponibles, junto con técnicas de machine learning (de clustering y clasificación).

Como métricas de medición de similitud se utilizó una de las más comúnmente usadas, *similitud del coseno*, y una métrica que **no se usó en ninguno de los trabajos anteriores** pero, que sin embargo, es muy reciente y prometedora, y es la principal técnica utilizada para la medición semántica de la distancia entre textos, *Word Mover's Distance (WMD)*. Estas métricas serán explicadas más en detalle en las secciones posteriores (Ver Técnicas para medir Similitud entre textos).

Una vez utilizadas estas mediciones de similitud se utilizaron algoritmos de machine learning: un algoritmo de clustering (K-means), y luego un modelo de clasificación KNN.

Además del uso de *Word Mover's Distance*, otro aspecto distintivo de este sistema es que se realizó una **integración a una interfaz web**, la cual la mayoría de los sistemas descriptos anteriormente no poseen. De esta manera, mediante la interfaz web se logra una interacción entre usuarios candidatos y reclutadores y, principalmente, permite que el reclutador sea capaz de visualizar un listado con los N candidatos más similares a un puesto determinado.

Por último, cabe destacar que los trabajos anteriormente mencionados no tienen un fácil acceso a los datasets ni al código que utilizaron para dichos sistemas, por lo que seguir el trabajo de ellos aplicando las mejoras que mencionan en sus artículos es una tarea casi imposible. En cambio, **este sistema será de código abierto**: los datasets, modelos y códigos utilizados estarán disponibles en Github y Git LFS¹⁰.

¹⁰https://github.com/FedericoCalonge/automatic_reading_of_CVs_using_text_similarity

3 Algoritmos de Machine Learning.

3.1 Introducción.

Continuando con el marco teórico de esta Tesis, es necesario explicar qué es Machine Learning y cómo se pueden clasificar a los distintos algoritmos según su tipo de aprendizaje: haciendo énfasis en los algoritmos K-Nearest Neighbor (KNN) y K-means.

Además, detallaremos la importancia de la separación de los datos y mencionaremos a grandes rasgos cuáles son los pasos a seguir para implementar un modelo de ML teniendo en cuenta algunos de los problemas clásicos que nos pueden ocurrir.

3.2 Machine Learning (ML).

La **Inteligencia Artificial (IA)** es una disciplina dentro de las ciencias de la computación que consiste en el desarrollo de algoritmos¹¹ que imiten el razonamiento humano, teniendo la capacidad de solucionar problemas que comúnmente resuelve la inteligencia natural pero de la manera más eficiente posible. Esencialmente, la IA permite que las máquinas puedan actuar e implementar distintas tareas que están fuera del alcance de los humanos [23].

Machine Learning (ML) o **Aprendizaje Automático** es un subconjunto de la Inteligencia Artificial que se encarga de construir algoritmos que aprenden a hacer algo útil a partir de los datos. Como ML es un subconjunto de IA, esto implica que todos los algoritmos de ML son técnicas de inteligencia artificial, pero no todos los métodos de inteligencia artificial califican como algoritmos de ML.

El objetivo principal de ML es permitir que las computadoras aprendan sin intervención o asistencia humana. Esencialmente, los algoritmos de ML aprenden *patrones* ocultos basados en datos históricos de entrada, los cuales posteriormente utilizan para aprender a clasificar la información o realizar predicciones relacionadas con el problema que se quiera resolver.

Los algoritmos de ML hoy en día se usan en todo tipo de aplicaciones: en Amazon para recomendarnos qué productos comprar, en Twitter para recomendarnos qué usuarios seguir, en Google para predecir qué páginas son las más relevantes para una consulta, en Facebook para reconocer qué fotos contienen rostros de personas, en el mercado inmobiliario para predecir los precios de las propiedades, en el mercado de valores para predecir el precio de las acciones, etc. Son tantas las aplicaciones de ML en la industria que se ha producido una verdadera explosión del tema en los últimos años[4].

¹¹Un algoritmo informático es un conjunto de instrucciones definidas, ordenadas y acotadas para resolver un problema, realizar un cálculo o desarrollar una tarea.

3.2.1 Aprendizaje supervisado y no supervisado.

Podemos dividir a ML en dos grandes categorías: aprendizaje supervisado y aprendizaje no supervisado. Resumidamente, la diferencia entre estas dos categorías es la existencia de etiquetas (labels) en el subconjunto de datos de entrenamiento (Ver *Separación de los datos*).

- En el **aprendizaje supervisado**, el modelo es entrenado utilizando datos que están “etiquetados”. Esto quiere decir que el set de datos que es utilizado para enseñar al algoritmo tiene una etiqueta (o label) que define la respuesta/salida correcta.

Analizándolo matemáticamente, este proceso puede ser representado por una variable de entrada x y una conocida variable de salida y , donde se usa un algoritmo f para que aprenda a predecir resultados para aquellos nuevos datos que no tienen etiqueta.

$$y = f(x)$$

Los algoritmos de aprendizaje supervisado se utilizan para resolver problemas de **clasificación** y de **regresión**.

- El **aprendizaje no supervisado**, en cambio, se refiere al proceso de entrenamiento de un modelo de Machine Learning sin un set de datos etiquetado. Este tipo de aprendizaje es una rama muy importante en Data Science ya que, al trabajar solo con datos sin necesidad de tener un “label” para los mismos, únicamente necesitamos los datos en crudo y estos en general son mucho más fáciles de conseguir que datos ya previamente clasificados. Por ejemplo, las aplicaciones de redes sociales tienen grandes cantidades de datos sin etiquetar.

El objetivo de este método es experimentar con el conjunto de datos (los cuales contienen muchas características) y luego aprender propiedades útiles de la estructura de dicho conjunto de datos. Matemáticamente se cuenta con un dataset de entrenamiento de variables de entrada x pero no se conoce su salida.

$$¿? = f(x)$$

Los algoritmos de aprendizaje no supervisado se utilizan para resolver problemas de **clustering**.

En la Figura 3.1 se detalla un resumen de la división en las categorías de ML y su “subdivisión” en modelos de clasificación, regresión y agrupación (clustering).

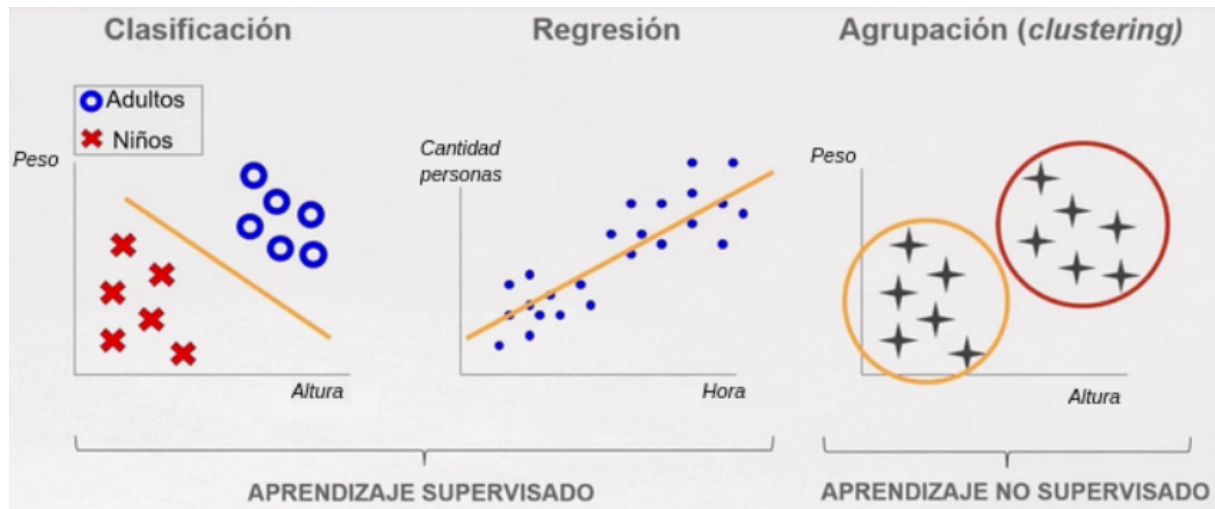


Figura 3.1: Tipos de aprendizaje en ML

3.2.1.1 Regresión.

En un problema de regresión se intenta predecir el valor de una variable numérica y continua a partir de un cierto conjunto de datos.

En general contamos con un set de entrenamiento en el cual conocemos el valor de la variable que queremos predecir. El objetivo es entonces construir un modelo que nos permita predecir el valor de nuestra variable de salida a partir de datos nuevos. Dicho de otra forma, lo que se hace es buscar una función que represente los datos de entrenamiento y que permita generalizar correctamente¹².

El caso más simple es la regresión lineal (observado previamente en la Figura 3.1), en el cual nuestro modelo es una recta, la recta que mejor se ajusta a los puntos de nuestro set de entrenamiento. En este ejemplo se intenta predecir la cantidad de personas en un lugar a una hora determinada.

Los problemas de regresión pueden usarse para realizar distintas predicciones, ya sea el valor de las acciones en el mercado de valores, predecir el costo de una propiedad, estimar las ganancias de un negocio, etc.

Las claves para identificar un problema de regresión[4] son las siguientes:

- Queremos predecir una variable que es numérica y continua.
- Contamos con un set de entrenamiento para el cual conocemos el valor de dicha variable.

¹²Generalizar es la capacidad de nuestro modelo de ML de obtener buenos resultados con datos nuevos reconociendo patrones generales de los datos de entrenamiento en lugar de reconocer particularidades específicas.

3.2.1.2 Clasificación.

En los problemas de clasificación la variable que se intenta predecir no es continua sino discreta, frecuentemente tiene pocos valores posibles y en muchos casos los valores posibles son solo dos: clasificación binaria. La idea es la misma que antes, contamos con un set de entrenamiento en el cual para cada dato conocemos la clase a la cual pertenece el mismo, queremos construir un modelo que nos permita clasificar automáticamente datos nuevos cuya clase desconocemos.

Un caso típico de la clasificación binaria es un problema de análisis de sentimiento, donde queremos saber si un cierto texto es positivo o negativo, es decir si habla bien o mal de un cierto tema. Como set de entrenamiento entonces deberíamos contar con textos para los cuales ya conocemos su sentimiento.

Por ejemplo, en la Figura 3.1, observamos que tenemos una serie de personas, de las cuales contamos con su peso y altura, y queremos clasificarlas en adultos o niños. Lo que va a hacer un modelo de clasificación es aprender dónde están estos puntos y crear un clasificador que, para nuevos datos de entrada, consiga segmentarlos correctamente en adultos o niños.

Otras aplicaciones por ejemplo podrían ser analizar si los reviews de un producto son buenos o malos, medir la actitud del público en general ante determinadas noticias por los comentarios que existen en redes sociales, clasificar si un email es spam o no lo es, etc.

Las claves para reconocer un problema de clasificación[4] son:

- Queremos determinar la clase / grupo a la que pertenece cada dato.
- La clase es una variable discreta con un set de valores posibles limitado y definido.
- Contamos con un set de entrenamiento para el cual conocemos los datos y a qué clase pertenecen.

3.2.1.3 Agrupación (clustering).

En un problema de clustering contamos con datos que queremos dividir en grupos de forma automática. Estos datos no tienen etiquetas, no sabemos a qué grupo pertenecen. Lo que hace un algoritmo de clustering es intentar buscar agrupaciones en los datos, creando de esta forma clusters¹³ con características similares. En algunos casos la cantidad de clusters la debemos indicar previamente y en otros el algoritmo es capaz de determinarla por sí mismo.

En el ejemplo de la Figura 3.1, podemos ver que se agrupan los puntos de la misma forma que en el problema de clasificación. La diferencia es que en este caso no sabemos si se trata de adultos o niños, sino que es el propio algoritmo el que va a identificar que hay dos grupos, y nosotros somos los que tenemos que interpretar los resultados.

Otros ejemplos de clustering podrían ser agrupar películas automáticamente de forma que queden juntas las que son de un mismo género; o la detección de comunidades en una red social, el cual es un típico problema de clustering en donde los puntos son los usuarios y queremos agruparlos automáticamente en comunidades. De esta forma podemos descubrir grupos de usuarios que tienen un cierto interés en común aun sin saber exactamente cuál es dicho interés.

Las claves para reconocer un problema de clustering[4] son:

- Contamos con un set de datos y queremos agruparlos en clusters/grupos.
- No son necesarios labels.

¹³Cluster, o grupo, es un conjunto de objetos que son "similares" entre ellos y "diferentes" de los objetos que pertenecen a los otros grupos. La palabra cluster viene del inglés y significa agrupación. Desde un punto de vista general, el cluster puede considerarse como la búsqueda automática de una estructura o de una clasificación en una colección de datos no etiquetados.

3.2.1.4 Algoritmos más conocidos.

Resumidamente, en la Figura 3.2 se detallan los algoritmos más conocidos para las distintas clasificaciones de Machine Learning previamente explicadas.

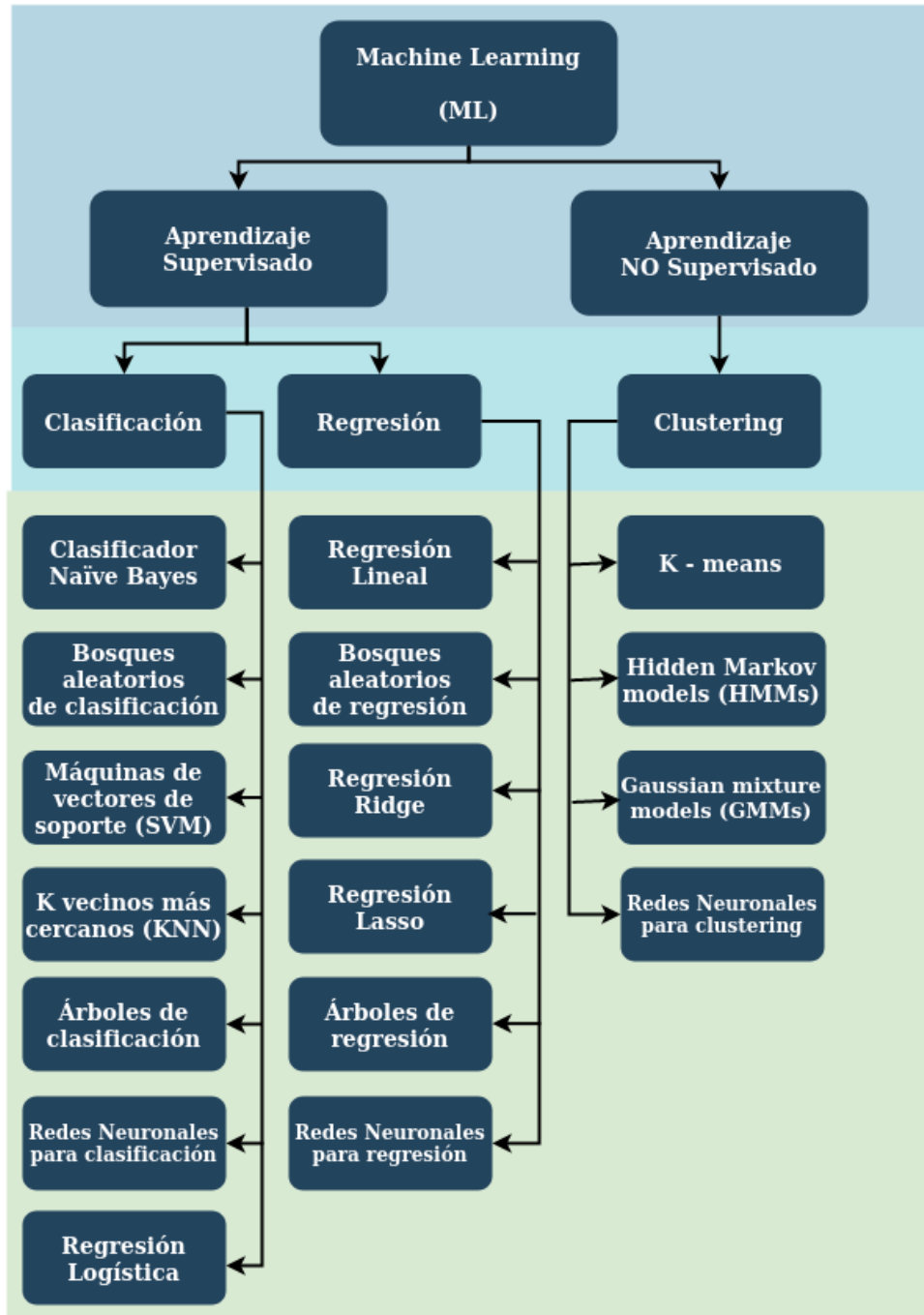


Figura 3.2: Algoritmos de Machine Learning más conocidos

Cabe destacar que algunos algoritmos se pueden adaptar para resolver problemas de otro tipo al cual están inicialmente categorizados. Por ejemplo, KNN que está categorizado como un algoritmo de clasificación, también se puede utilizar para resolver problemas de regresión. Otro caso conocido es Support Vector Regression (SVR), que utiliza las bases de Máquinas de vectores de soporte (SVM) para resolver problemas de regresión.

3.2.2 Aprendizaje transductivo

Existe una cierta relación entre los problemas de clustering y de clasificación, por ejemplo dado un problema de clasificación podríamos aplicar clustering primero y luego clasificar a cada punto de acuerdo al cluster al cual pertenece en base a la clase mayoritaria de dicho cluster. Este procedimiento no es muy frecuente pero es conveniente tenerlo en cuenta porque permite entender el funcionamiento de ciertos algoritmos que combinan las propiedades de un problema de clustering y uno de clasificación.

Una aplicación que combina clustering (aprendizaje no supervisado) y clasificación (aprendizaje supervisado) es la que denominamos **Aprendizaje transductivo**[24][4]. De esta manera, para predecir nuevos datos no etiquetados se utilizan los datos previamente etiquetados, así como los datos sin etiquetar, como forma de ayuda a un clasificador tradicional.

Consideremos el ejemplo de la Figura 3.3. Aquí tenemos solo dos puntos clasificados, uno clasificado como "blanco" el otro como "negro". Sin mayor información el punto marcado con el signo de pregunta, cuya clase desconocemos, quedaría clasificado como "blanco" que está mucho mas cerca del punto blanco que del punto negro.



Figura 3.3: Aprendizaje transductivo

En la Figura 3.4 al agregar puntos cuya clase desconocemos, vemos que en nuestros datos existen dos clusters. De esta manera, si tenemos que asociar cada cluster con un color entonces el de arriba es "negro" y el de abajo es "blanco", ya que el punto negro pertenece al cluster de arriba y el blanco al de abajo. De este modo, el punto que inicialmente consideramos blanco, con esta nueva información de clusters, sería considerado negro.

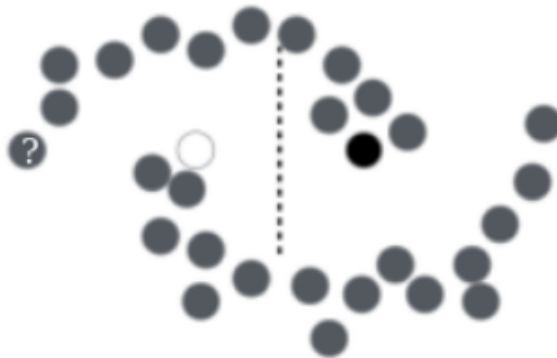


Figura 3.4: Aprendizaje transductivo

El aprendizaje transductivo es un área muy nueva dentro de Data Science y que sin dudas merece ser explorada.

En este proyecto de Tesis se utilizará el aprendizaje transductivo para lograr inicialmente una clusterización de nuestros candidatos en 5 grupos mediante un algoritmo de clustering (K-means), y luego se realizará una clasificación mediante un algoritmo de clasificación (KNN).

3.2.3 Separación de los datos.

Dos conceptos importantes que mencionaremos a lo largo del Informe son: los “datos de entrenamiento” y los “datos de prueba”.

Como mencionamos anteriormente, los algoritmos de Machine Learning aprenden de los datos con los que los entrenamos. A partir de ellos, intentan encontrar o inferir el patrón que les permita predecir el resultado para un nuevo caso. Pero, para poder calibrar si un modelo funciona, necesitaremos probarlo con un conjunto de datos diferente. Por ello, en todo proceso de aprendizaje automático, los datos de trabajo se dividen en dos partes:

- Los **datos de entrenamiento** son los datos que usamos para entrenar un modelo. La calidad de nuestro modelo de aprendizaje automático va a ser directamente proporcional a la calidad de los datos. Por ello son muy importantes las tareas de limpieza y preprocesamiento de los datos..
- Los **datos de prueba o evaluación** son los datos que nos “reservamos” para comprobar si el modelo que hemos generado a partir de los datos de entrenamiento “funciona”. Es decir, si las respuestas predichas por el modelo para un caso totalmente nuevo son acertadas o no.

Es importante que el conjunto de datos de prueba tenga un volumen suficiente como para generar resultados estadísticamente significativos, y a la vez, que sea representativo del conjunto de datos global. Normalmente el conjunto de datos se suele dividir en un **70 %/80 % de datos de entrenamiento** y un **30 %/20 % de datos de prueba**, pero se puede variar la proporción según el caso.

3.2.4 ¿Cómo implementar un modelo de ML?

Resumidamente, los pasos a seguir para implementar un modelo de ML y utilizarlo para realizar predicciones sobre los datos son los siguientes:

- Paso 1: Recolección de Datos.
- Paso 2: Preparación y preprocesamiento de los datos.
- Paso 3: Elección del modelo de ML.
- Paso 4: Entrenamiento del algoritmo.
- Paso 5: Evaluación del modelo.
- Paso 6: Parameter Tuning o configuración de parámetros.
- Paso 7: Utilizando nuestro modelo.

A continuación detallaremos en mayor detalle los mencionados pasos.

- Paso 1: Recolección de Datos.

Dada la problemática que deseamos resolver mediante algoritmos de ML, nuestro primer paso será recolectar los datos que utilizaremos posteriormente para “alimentar” a dicho algoritmo.

Para el primer paso de recolección de datos, hay que tener muy en cuenta la calidad y cantidad de información que consigamos ya que impactará directamente en lo bien o mal que luego funcione nuestro modelo. Estos datos los podemos sacar de bases de datos, planillas de cálculo, utilizando técnicas de web scraping¹⁴ o mediante APIs¹⁵ para recopilar información de manera automática de diversas fuentes de Internet, etc.

- Paso 2: Preparación y preprocesamiento de los datos.

En este paso generalmente se realizan visualizaciones de los datos y se revisa si existen correlaciones entre las distintas “features”¹⁶ de nuestros datos. Al pre-procesar nuestros datos nos referimos a normalizar los mismos: eliminando duplicados y realizando distintas correcciones de errores.

En esta etapa además es importante **separar** nuestros datos en dos grupos:

- un set de entrenamiento.
- un set de prueba.

¹⁴Proceso dentro de Data Science que se utiliza para la extracción de datos de sitios web simulando cómo navegaría un ser humano por los mismos.

¹⁵API o interfaz de programación de aplicaciones, es un conjunto de métodos o funciones que ofrece cierta biblioteca para ser utilizada por otro software como una capa de abstracción

¹⁶Features generalmente son las columnas de nuestro dataframe, archivo o base de datos -dependiendo cómo almacenamos nuestros datos-.

Como mencionamos previamente, el set de test en general es un 20 % o 30 % del set de entrenamiento. Esta partición de los datos en estos dos conjuntos diferenciados permite generar el modelo a partir de los datos de entrenamiento para después comprobar su eficiencia con los datos reservados para test.

- Paso 3: Elección del modelo de ML.

Una vez obtenidos y preprocesados estos datos, lo que se hace es **elegir el modelo de ML**¹⁷ de acuerdo al objetivo que tengamos o problema que deseemos resolver.

De esta manera, utilizaremos algoritmos de clasificación, regresión o clustering para construir nuestro modelo de ML a partir de los datos, de forma tal de luego poder usar dicho modelo para predecir nuevos datos.

- Paso 4: Entrenamiento del algoritmo.

El siguiente paso es **entrenar** a nuestro algoritmo de ML. En este proceso mediante una serie de iteraciones nuestro algoritmo detecta patrones en nuestros datos que luego nos servirán para poder realizar predicciones con los nuevos datos que se incorporen al sistema.

La idea es entrenar a nuestro algoritmo con el set de entrenamiento (mediante una función *fit()*) para luego, en las etapas posteriores, aplicarlo al set de prueba (mediante una función *predict()*). De esta forma, los datos para los cuales queremos probar el algoritmo (set de test) nunca fueron vistos por el mismo en la etapa de entrenamiento, lo cual permite saber si el modelo fue capaz de generalizar correctamente.

- Paso 5: Evaluación del modelo[4].

Hacer predicciones correctas sobre datos futuros suele ser el principal problema que queremos resolver al utilizar algoritmos de ML. Luego de entrenar el modelo se tiene que **evaluar** el mismo. Evaluar un modelo, resumidamente, es estimar su rendimiento para saber qué tan bien se desempeñará / predecirá para datos nuevos no vistos por el mismo.

Para poder evaluar un modelo correctamente, tenemos que contar con la separación de nuestros datos en set de entrenamiento y set de prueba que realizamos en los pasos previos. Esto lo hacemos, ya que evaluar la precisión predictiva de un modelo de ML con los mismos datos que se han utilizado para el entrenamiento no es útil, ya que compensa a los modelos que pueden recordar los datos de entrenamiento en lugar de generalizar.

De esta manera, luego de haber entrenado nuestro modelo de ML, resumidamente lo que hacemos en esta etapa es comparar las predicciones realizadas sobre set de pruebas devueltas por el modelo de ML contra el valor de destino conocido para el mismo set de pruebas; y por último generar alguna **métrica de evaluación** que

¹⁷Un modelo de machine learning es la salida de información que se genera cuando se entrena un algoritmo de ML con datos. Después del entrenamiento, al proporcionar un modelo con una entrada, se le dará una salida. Por ejemplo, un algoritmo predictivo creará un modelo predictivo.

nos permite verificar la performance de nuestro modelo indicando la efectividad de las predicciones.

Dependiendo del tipo de modelo que tengamos, podemos utilizar distintas métricas de evaluación para verificar su performance. Como observación, se detallan algunas de las métricas más conocidas en el Cuadro 1.

Modelos de regresión	Modelos de clasificación	Modelos de clustering
Error cuadrático medio (MSE)	Matriz de confusión o error	Inertia
Error cuadrático medio (RMSE)	Accuracy (Exactitud)	Silhouette Score
Error absoluto medio (MAE)	Precision (Precisión)	Rand Index
R al cuadrado (R^2)	Recall o sensibilidad o TPR (Tasa positiva real)	Adjusted Rand Index
Error de porcentaje cuadrático medio (MSPE)	Especificidad o TNR (Tasa negativa real)	Mutual Information
Error porcentual absoluto medio (MAPE)	F1-Score	Calinski-Harabasz Index
Error logarítmico cuadrático medio (RMSLE)		Davies-Bouldin Index

Cuadro 1: Métricas para evaluar distintos tipos de modelos.

Por ejemplo, si queremos verificar la performance de un modelo de clasificación podemos utilizar el **accuracy**. Esta métrica mide el % de aciertos: es el ratio de las predicciones correctas sobre el número total de instancias evaluadas. Por ejemplo, si el Accuracy es menor o igual al 50 % este modelo no será útil ya que sería como lanzar una moneda al aire para tomar decisiones. Si alcanzamos un 90 % o más podremos tener una buena confianza en los resultados que nos otorga el modelo.

■ Paso 6: Parameter Tuning o configuración de parámetros[4].

Si durante la evaluación no obtuvimos buenas predicciones y nuestra métrica de evaluación no logró ser la mínima deseada, es posible que tengamos problemas de overfitting (ó underfitting) y deberemos retornar al paso de entrenamiento (Paso 4) haciendo antes una nueva configuración de hiper-parámetros de nuestro modelo.

Cada modelo tiene un conjunto de parámetros e hiper-parámetros que necesita para funcionar:

- Los **parámetros** son los valores obtenidos por el propio algoritmo a partir de los datos, no son indicados manualmente.
- Los **hiper-parámetros**, en cambio, son datos que debemos pasarle al algoritmo para funcionar.

Como ejemplos de parámetros tenemos los coeficientes en una regresión lineal o logística, o los pesos en una red neuronal artificial. Y, como ejemplos de hiper-parámetros, tenemos al 'k' en KNN, o los 'EPOCHs' que nos permiten incrementar la cantidad de veces que iteramos sobre nuestros datos de entrenamiento en una red neuronal artificial.

Para encontrar los hiper-parámetros óptimos, es decir, aquellos que mejor funcionan para nuestro set de datos, lo que se hace es, nuevamente, realizar un entrenamiento y una evaluación de nuestro modelo de ML seleccionando e iterando sobre los distintos hiper-parámetros que tengamos: a esto se le conoce como “*Parameter Tuning*”. Esta iteración puede ser de manera aleatoria (método conocido como *random search*) o completa (*grid search*). Luego de realizar esta iteración se obtiene la configuración de hiper-parámetros más óptima.

Algo a tener en cuenta es que, para realizar la evaluación, no debemos usar los datos del set de prueba, ya que podríamos caer en un caso de overfitting seleccionando los parámetros que funcionan mejor para los datos del set de pruebas, pero tal vez no los parámetros que generalicen mejor. Para evaluar el modelo y realizar dicho “*Parameter Tuning*” lo que necesitamos es dividir el set de entrenamiento original en dos: un set de entrenamiento y un **set de validación**.

La idea es entrenar el modelo con el set de entrenamiento y luego probarlo con dicho set de validación (NO con el set de pruebas) a efectos de encontrar los mejores hiper-parámetros. El set de validación en general es un 20-30 % del set de entrenamiento original.

Por último, una vez hallados los hiper-parámetros más óptimos, lo que haremos es realizar una **evaluación final** sobre el set de pruebas con los hiper-parámetros que encontramos, y de esta manera nos dará el accuracy (o la métrica de evaluación que elegimos) para los datos que el modelo no vió.

No obstante, existen 2 problemas con el esquema anterior:

- Siendo el set de validación siempre el mismo podemos caer en el problema de que los hiper-parámetros que encontremos solo sean óptimos para nuestro set de validación (el cual es un pequeño conjunto de nuestros datos).
- La división del set de entrenamiento en un nuevo set de entrenamiento y un set de validación hace que haya menos datos disponibles para el entrenamiento. Esto resulta un problema, especialmente para conjuntos de datos pequeños, ya que siempre es mejor utilizar el mayor número de datos posible para el entrenamiento.

Para evitar estos 2 problemas utilizamos el método de **Cross Validation** (o validación cruzada), el cual es prácticamente universal para optimizar algoritmos de ML. Este método nos permite evaluar modelos de ML solucionando los 2 problemas mencionados anteriormente previniendo el overfitting. Para mayor detalle de su funcionamiento ver *Cross Validation*.

■ Paso 7: Utilizando nuestro modelo.

Una vez que obtuvimos buenas predicciones en la etapa de evaluación y nuestra métrica de evaluación llegó a ser la mínima deseada, ya podemos afirmar que estamos en condiciones de utilizar nuestro modelo de Machine Learning entrenado para realizar predicciones con nuevos datos que están fuera del set de entrenamiento y del set de test.

3.2.4.1 Cross Validation.

gadgha gagaga

3.2.4.2 Los Problemas de ML: Overfitting y Underfitting.

4 Natural Language Processing.

FALTA

4.1 Introducción.

FALTA

4.2 Preprocesamiento de textos.

FALTA

4.3 Similitud entre textos.

FALTA

4.4 Técnicas para medir Similitud entre textos.

FALTA

4.4.1 Cosine Similarity.

FALTA

Cosine Similarity se basa en la medición del coseno del ángulo entre dos vectores proyectados en un espacio multidimensional para lograr medir la similitud de los documentos (un menor ángulo indica una mayor similitud). En este contexto, estos dos vectores representan matrices que contienen el recuento de palabras de dos documentos. Una de sus limitaciones es que no tiene la habilidad de reconocer si las palabras que compara son semánticamente similares.

4.4.2 Word Mover's Distance (WMD).

FALTA

Anteriormente, mencionamos que una de las limitaciones de Cosine Similarity es que no tiene habilidad de reconocer si las palabras que compara son semánticamente similares. En cambio, Word Mover's Distance (WMD) es un algoritmo más complejo que sí permite reconocer las relaciones semánticas; su limitación son las relaciones sintácticas (VER.....). WMD se basa en word embeddings y permite medir la distancia entre documentos (una menor distancia indica una mayor similitud).

4.5 Algoritmos de vectorización.

FALTA

Previamente a utilizar Cosine Similarity y WMD para (—completar—) se debe emplear algún algoritmo de vectorización que permita representar las palabras de nuestros textos a un espacio vectorial. De esta forma Cosine Similarity y WMD podrán interpretarlos de la mejor manera. Como algoritmos de vectorización se utilizarán TF-IDF y Word Embeddings.

4.5.1 TF-IDF.

FALTA

El algoritmo TF-IDF asigna valores numéricos a las palabras en función de la frecuencia con que aparecen en los textos para medir la frecuencia de ocurrencia de un término en la colección de documentos, expresando cuán relevante es una palabra para un documento en una colección.

4.5.2 Word Embeddings.

FALTA

Los Word Embeddings son necesarios para utilizar WMD. Los Word Embeddings son una de las variantes más populares para representar textos. Son vectores previamente entrenados y generados mediante un modelo de red neuronal secuencial; de esta manera son capaces de capturar los contextos de una palabra en el documento llegando a poder contener información semántica y sintáctica.

4.5.2.1 ¿Cómo entrenar Word Embeddings?

FALTA

5 Implementación.

La implementación de este Sistema se trabajó en dos grandes partes:

1. Obtención del modelo de clasificación.

En esta primera parte se obtuvieron y preprocesaron datasets de Curriculum Vitae de distintos candidatos y descripciones de puestos de trabajo de IT publicados por distintas empresas, para luego ser comparados y obtener similitudes entre los textos utilizando las técnicas para medir distancias y obtener dichas similitudes (WMD y Cosine Similarity) y los algoritmos de vectorización (TF-IDF y Word Embeddings).

Una vez obtenidas estas mediciones de similitud entre los Curriculum Vitae de los candidatos y las descripciones de los puestos laborales de IT, estos valores se utilizaron para alimentar un algoritmo de clustering K-means que a su vez, con sus datos de salida (4 clusters), alimentan a un modelo de clasificación KNN. Finalmente, con este modelo KNN logramos, en base a los valores de similitud de nuevos candidatos, clasificar qué tan similares son dichos candidatos con respecto a la descripción de un puesto de IT: similitud escasa, similitud media, similitud alta, similitud muy alta.

Estos análisis se realizaron en documentos de Jupyter Notebook utilizando Python; y sirvieron para evaluar el comportamiento del modelo de clasificación y los distintos algoritmos de medición de similitudes para luego ser utilizados en la siguiente etapa.

2. Integración al Sistema Web.

Etapas posteriores a la primera parte. Una vez observado que los resultados fueron los esperables, lo que se hizo fue reutilizar las funciones que contenían la lógica de los distintos algoritmos utilizados junto con el modelo de clasificación KNN obtenidos previamente en la parte 1, para integrar todo esto en el sistema Web. Este sistema web está realizado en Django ¹⁸, y cuenta con una base de datos relacional que contiene la información de los candidatos y reclutadores junto con los Curriculum Vitae y puestos que hayan cargado.

De esta manera, nuestro sistema cuenta con una interfaz gráfica permitiendo interactuar entre candidatos y reclutadores y, principalmente, permitiendo que el reclutador sea capaz de obtener un listado con los N candidatos más similares a un puesto determinado, y ordenados de mayor a menor de acuerdo a esta *similitud*. Dicha *similitud* representa el resultado obtenido de la clasificación por nuestro modelo KNN.

¹⁸Framework de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como modelo-vista-controlador.

5.1 Obtención del modelo de clasificación.

5.1.1 Introducción.

Como inicio definamos qué es un modelo. En nuestro caso, un modelo representa Este modelo se construyó en base a ... La implementación de este Sistema se realizó en Python...

5.1.2 Esquema.

Como podemos ver la figura 5.1 representa el procedimiento utilizado para la obtención de nuestro modelo de clasificación.

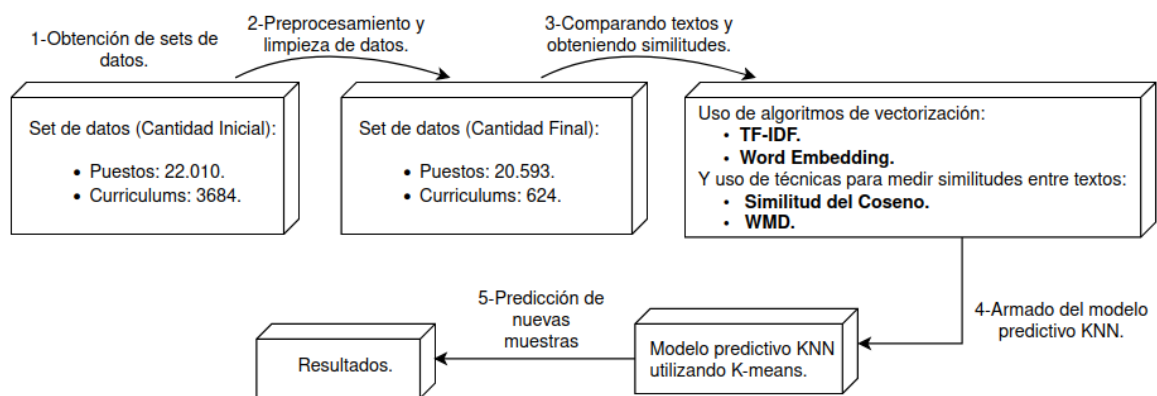


Figura 5.1: Pipeline Flow para la obtención del modelo de clasificación KNN

5.1.3 Obtención de sets de datos.

En primer lugar debemos definir qué es un set o conjunto de datos. Un set o conjunto de datos es una tabla de una base de datos o, matemáticamente, una matriz estadística de datos. Cada columna de la tabla representa una variable del set de datos; y cada fila representa a un miembro determinado del mismo.

Para este Proyecto utilizamos dos grandes sets de datos que se obtuvieron mediante la recolección de distintos archivos alojados en la Web, los cuales estan descriptos a continuación.

5.1.3.1 Curriculum Vitae.

Los set de datos de Curriculum Vitae de los candidatos se obtuvieron de las siguientes fuentes:

1. 228 Curriculums en formato docx y posteriormente convertidos a pdf, obtenidos del sitio Kaggle¹⁹. Estos pdfs son candidatos de la India con experiencia en el rubro de IT.
2. 2484 Curriculums en formato CSV, obtenidos del sitio Kagle²⁰. Este CSV cuenta con curriculums vitae obtenidos del sitio web de postulación de trabajos 'livecareer.com'.
3. 962 Curriculumns en formato CSV, obtenidos del sitio Kaggle²¹. Este CSV cuenta con curriculums vitae repartidos en distintas categorías de IT.
4. 10 Curriculums en formato PDF, los cuales los cuales fueron obtenidos como ejemplos mediante una recolección propia de distintos sitios web.

5.1.3.2 Descripciones Puestos Laborales.

Los set de datos de descripciones de puestos laborales se obtuvieron de las siguientes fuentes:

1. 22.000 descripciones en formato CSV; obtenido del sitio Kaggle²². El CSV cuenta con descripciones de puestos obtenidos del sitio web de USA de postulación de trabajos del rubro de IT 'Dice.com'.
2. 10 descripciones en formato CSV; obtenidas como ejemplos mediante una recolección propia del sitio Indeed²³ para puestos de trabajo de IT.

¹⁹<https://www.kaggle.com/palaksood97/resume-dataset>

²⁰<https://www.kaggle.com/snehaanbawal/resume-dataset>

²¹<https://www.kaggle.com/gauravduttakiit/resume-dataset>

²²<https://www.kaggle.com/PromptCloudHQ/us-technology-jobs-on-dicecom>

²³<https://www.indeed.com/q-USA-jobs.html>

5.1.4 Preprocesamiento y limpieza de datos.

Previamente a utilizar las técnicas para medir distancias y obtener similitudes entre textos (WMD y Cosine Similarity) y los algoritmos de aprendizaje (KNN y K-Means) necesitamos que los datos que comparemos e introduzcamos en los algoritmos estén lo más limpios posible; ya que de lo contrario las mismos podrían clasificar o predecir de forma errónea. Este análisis previo sobre los datos debe ser minucioso ya que puede haber valores incoherentes o absurdos.

El procedimiento para la Limpieza de los Curriculum Vitae y las descripciones de los puestos laborales fue el siguiente:

1. Convertimos todo a minúscula.
2. Eliminamos datos no relevantes para nuestros análisis (mails y páginas web).
3. Eliminamos signos de puntuación y caracteres especiales (incluyendo números).
4. Eliminamos stop words.
5. Eliminamos common words no relevantes para nuestros análisis.
6. Aplicamos Lematización y Tokenización.
7. Eliminamos repetidos.
8. Obtenemos y usamos bi-gramas.

Luego de aplicar preprocesamiento y limpieza de datos nos quedarán los siguientes tamaños de nuestros datasets:

- 624 curriculums vitae de candidatos (en formato pdf y csv).
- 20593 descripciones de puestos de IT (en formato csv).

5.1.5 Cantidad final del set de datos y su uso en las distintas etapas.

El total de 624 curriculums vitae de candidatos y 20593 descripciones de puestos de IT que mencionamos previamente, serán utilizados para el entrenamiento y obtención de vectores mediante TF-IDF (para el posterior cálculo de Cosine Similarity) y para el entrenamiento de Word2Vec y obtención de los Word Embeddings (para el posterior cálculo de WMD).

Por otro lado, para calcular Cosine Similarity y WMD, para utilizarlos en K-means y para entrenar a nuestro algoritmo KNN, utilizaremos únicamente una porción de nuestros datasets:

1-Para el cálculo de Cosine Similarity y WMD:

- 301 curriculums vitae de candidatos.
- 201 descripciones de puestos de IT.

Nota: No obstante, al realizar los cálculos de distancias compararemos cada curriculum vitae con cada Job Description, obteniendo un dataframe total de 3131 filas con sus respectivos valores de WMD y Cosine Sim.

2-Para el uso de K-means y entrenamiento con KNN (eliminamos un curriculum vitae y una descripción de puesto IT que los utilizamos en '3-'):

- 300 curriculums vitae de candidatos.
- 200 descripciones de puestos de IT.

Nota: como se comentó previamente, nos quedarán 3000 filas / puntos para usar en K-means y entrenar KNN; llegando a representar estos 3000 puntos en un plano de 2 dimensiones.

3-Para la clasificación de nuevas muestras mediante KNN:

- 1 curriculum vitae de candidatos.
- 1 descripción de puesto de IT.

Nota: como se comentó previamente, nos quedarán 131 filas para clasificar.

¿Por qué utilizamos solo una porción de nuestros datasets?: Esto es debido a los drawbacks de WMD y KNN.

- WMD: posee una alta complejidad en el cálculo de la distancia, teniendo un tiempo de ejecución muy elevado. Como ejemplo, al correrlo localmente, el cálculo de WMD para 3131 filas tardó 7 horas; frente a los 3 segundos que tardó el cálculo de Cosine Similarity para la misma cantidad de filas.
- KNN: KNN es una gran opción para datasets pequeños con pocas variables de entrada; pero tiene problemas cuando la cantidad de entradas es muy grande. En

grandes dimensiones, los puntos que pueden ser similares pueden tener distancias muy grandes. Además, cada vez que se va a hacer una predicción con KNN, busca al vecino más cercano en el conjunto de entrenamiento completo. Por esto, se debe utilizar un dataset pequeño para que el clasificador KNN complete su ejecución rápidamente.

En conclusión, al utilizar solo una porción de nuestros datasets para obtener los distintos cálculos de distancias y entrenar KNN, el cálculo de WMD se podrá realizar en un tiempo finito, y nuestro clasificador KNN funcionará rápida y eficientemente al realizar predicciones.

5.2 Comparando textos y obteniendo similitudes.

FALTA

Previamente a utilizar Cosine Similarity y WMD para obtener las medidas de similitud entre los textos, se debe emplear algún algoritmo de vectorización que permita representar las palabras de nuestros textos a un espacio vectorial. De esta forma Cosine Similarity y WMD podrán interpretarlos de la mejor manera. Como mencionamos previamente, como algoritmos de vectorización se utilizarán TF-IDF y Word Embeddings.

5.3 Armado del modelo de clasificación KNN.

FALTA

Una vez obtenidas estas mediciones de similitud entre los Curriculum Vitae de los candidatos y las descripciones de los puestos laborales de IT, estos valores se utilizarán para alimentar un algoritmo de clustering K-means que a su vez, con sus datos de salida (4 clusters), alimentarán a un modelo de clasificación KNN. Finalmente, con este modelo KNN lograremos, en base a los valores de similitud de nuevos candidatos, clasificar qué tan similares son dichos candidatos con respecto a la descripción de un puesto de IT: similitud escasa, similitud media, similitud alta, similitud muy alta.

5.4 Clasificación de nuevas muestras y resultados obtenidos.

FALTA

5.5 Integración al Sistema Web.

FALTA

Anteriormente lo que se hizo fue un análisis mediante documentos en Jupyter Notebooks para evaluar el comportamiento del modelo de clasificación y los distintos algoritmos de medición de similitudes.

Al observar que los resultados fueron los esperables, lo que se hizo en esta última etapa fue reutilizar las funciones que contenían la lógica de los distintos algoritmos utilizados junto con el modelo de clasificación KNN obtenidos en la fase previa, para integrar todo esto en el sistema Web.

Como mencionamos previamente, este sistema web está realizado en Django, y cuenta con una base de datos relacional que contiene la información de los candidatos y reclutadores junto con los Curriculum Vitae y puestos que hayan cargado.

De esta manera, nuestro sistema cuenta con una interfaz gráfica permitiendo interactuar entre candidatos y reclutadores y, principalmente, permitiendo que el reclutador sea capaz de obtener un listado de los N candidatos más similares a un puesto determinado, y ordenados de mayor a menor de acuerdo a esta *similitud*. Dicha *similitud* representa el resultado obtenido de la clasificación por nuestro modelo KNN.

El sistema web contará con 2 tipos de usuario:

- Candidato: quienes cargarán en el sistema sus Curriculum Vitae y aplicarán a los distintos puestos disponibles.
- Reclutador: quienes cargarán en el sistema los puestos de trabajo que tengan disponibles y podrán consultar, entre otras cosas, un listado con los N candidatos más similares a un puesto determinado, y ordenados de mayor a menor de acuerdo a esta *similitud*.

5.5.1 Base de datos.

Nuestros datos los almacenaremos en una base de datos **FALTA definir cual**.

Para modelar y gestionar nuestros datos utilizamos el modelo relacional ²⁴.

Sacar la mayoría del documento modelo-entidad-relacion-case-method-richar-barker.pdf

Para comprender los datos que se almacenan en dicha base de datos, los representaremos utilizando un diagrama entidad relación ²⁵. Previamente a esto explicaremos los elementos del diagrama de entidad relación:

FALTA PONER IMAGEN CON LOS ELEMENTOS: Entidad rectángulo, Unión entre entidades s
las líneas que puede ser obligatoria u opcional, cardinalidad son los 1:M / 1:1 / M:M

- Entidad: objeto concreto o abstracto que figura en nuestra base de datos. Por ejemplo una entidad puede ser un alumno, un cliente, una empresa, etc. Dentro de las entidades estan los atributos, atributos principales o clave primaria (PK) y atributos foraneos o clave secundaria (FK). Las entidades que necesitamos para crear nuestra BD son: Candidato, Puesto, Reclutador y Candidato_Puesto -entidad intermedia entre Candidato y Puesto-.
- Unión entre entidades: pueden ser obligatorias u opcionales. En nuestro diagrama nuestras uniones son todas opcionales, ya que el reclutador puede o no CARGAR un puesto, el candidato puede o no APLICAR a un puesto y a su vez el puesto puede o no ser aplicado por un candidato.
- Cardinalidad: Relación entre entidades o mapeo. La cardinalidad es el tipo de relación entre entidades. Observando la figura 5.2 y considerando que los rectángulos azules son una entidad y los naranjas son otra entidad observamos que pueden haber 4 tipos de cardinalidades posibles:
 1. Uno a uno: a cada entidad azul le corresponde solo una entidad naranja.
 2. Uno a muchos: a cada entidad azul le corresponde una o varias entidades naranjas.
 3. Muchos a uno: a cada entidad naranja le corresponde una o varias entidades azules.
 4. Muchos a muchos: las entidades azules pueden tener varias entidades naranjas y las entidades naranjas también pueden tener varias entidades azules.

²⁴Una base de datos relacional es un conjunto de una o más tablas estructuradas en registros (líneas) y campos (columnas), que se vinculan entre sí por un campo en común.

²⁵Un modelo entidad-relación es una herramienta para el modelo de datos, la cual facilita la representación de entidades de una base de datos.



Figura 5.2: Tipos de cardinalidad.

La cardinalidad entre nuestras entidades son:

- Entre Candidato y Puesto existe una relación muchos a muchos (M:M), ya que un candidato puede aplicar a M puestos y un puesto puede ser aplicado por M candidatos. Es por esto que se creó la tabla intermedia Candidato_Puesto conllevando dos relaciones uno a muchos (1:M) con Puesto y Candidato.
- Entre Reclutador y Puesto existe una relación de uno a muchos (1:M), ya que un reclutador puede cargar M puestos, y un puesto pertenece a un solo reclutador.

En cuanto a las claves pueden haber dos tipos. Por un lado está la clave primaria o atributo principal (PK) es única y toda entidad debe tener la suya. Pueden haber múltiples PKs; estas se llaman PKs compuestas. Y por el otro está la clave secundaria o atributo foráneo (FK). Estas claves identifican a una entidad externa en otra, utilizándose para generar relaciones entre nuestras entidades. Si tenemos una clave FK en una entidad, significa que dicha clave FK es clave PK en otra entidad.

El diagrama de relación que utilizamos para nuestro trabajo lo observamos en la figura 5.3.

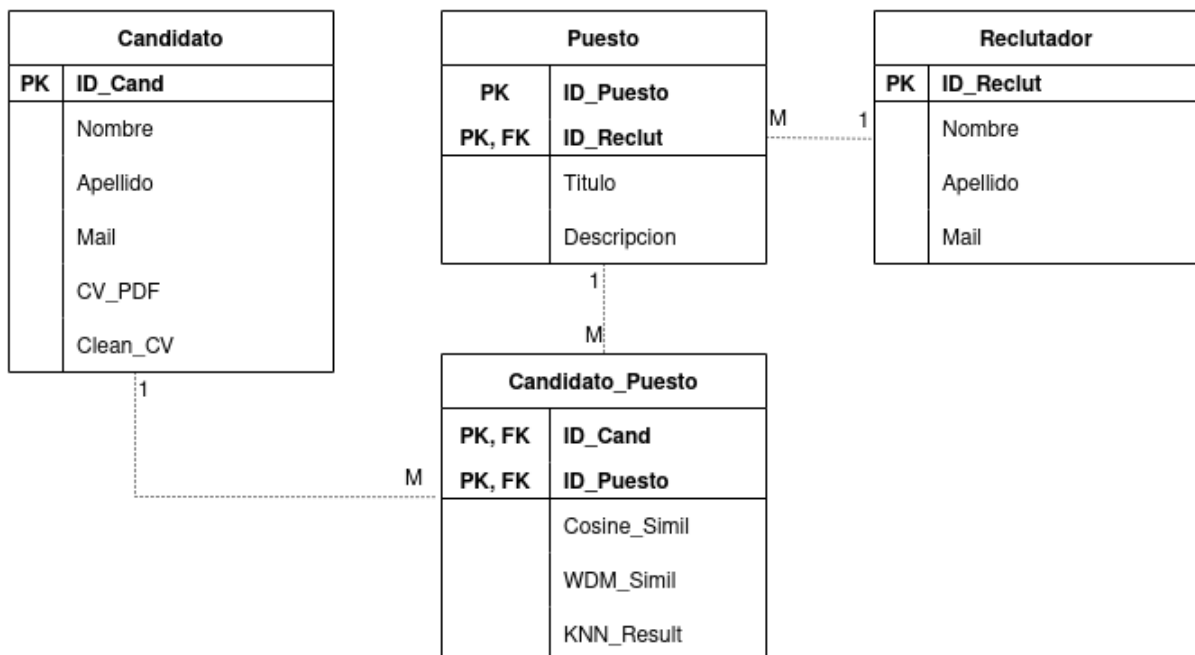
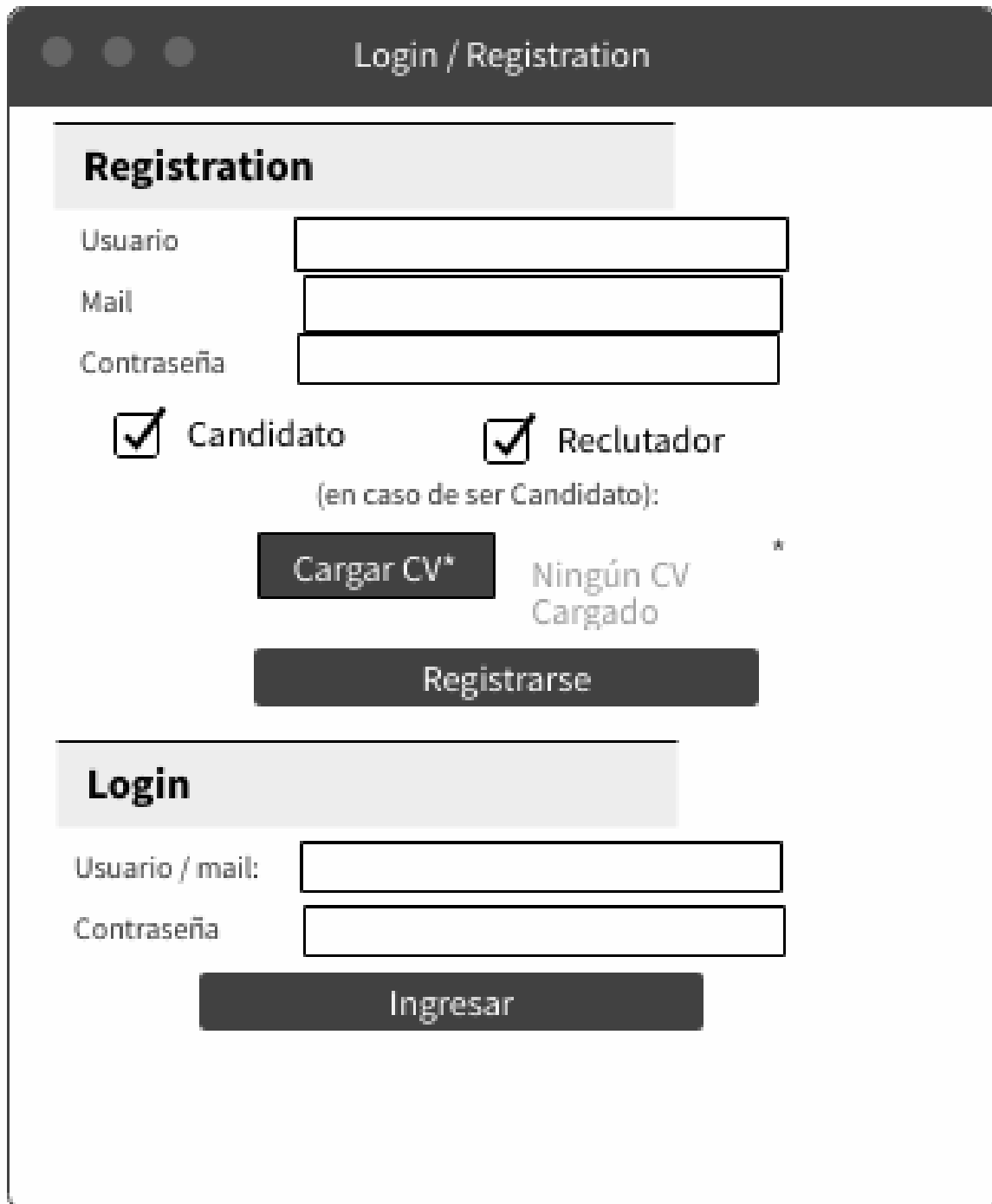


Figura 5.3: Diagrama de relación utilizado.

5.5.2 Secciones del sistema

Para registrarse o loguearse al sistema, se implementará la interfaz provista en la figura 5.4.



The image is a sketch of a web interface titled "Login / Registration". It is divided into two main sections: "Registration" and "Login".

Registration Section:

- Fields for "Usuario", "Mail", and "Contraseña" (Password).
- Two checkboxes: ☒ "Candidato" and ☒ "Reclutador".
- Text "(en caso de ser Candidato):" below the checkboxes.
- A button "Cargar CV*" and a status message "Ningún CV Cargado" with an asterisk.
- A large "Registrarse" button.

Login Section:

- Fields for "Usuario / mail:" and "Contraseña".
- A large "Ingresar" button.

Figura 5.4: Logueo y Registración.

ES UN BOCETO, FALTA PONER LA IMAGEN REAL

El Candidato tendrá acceso al menú indicado en la figura 5.5.

Candidato

Mi Perfil

Nombre y apellido

DNI

Fecha nacimiento

Teléfono

Email

Domicilio

Actualizar datos

Cargar y analizar nuevo CV

Puestos disponibles

ID Puesto	Puesto	Descripcion Pu...	Ubicación
1	Programador F...	Descripcion larga	Buenos Aires
2	DB Engineer	Descripcion larga	Buenos Aires
3	Data Scientist	Descripcion larga	Buenos Aires

(se podrá filtrar/ordenar en cada columna)

Postularse

Postulaciones

ID Puesto	Puesto	Descripci...	Ubicación	Fecha Pos...
1	Programa...	Descripci...	Buenos Ai...	15/06/2021

(se podrá filtrar/ordenar en cada columna)

Figura 5.5: Vista del Candidato.

ES UN BOCETO, FALTA PONER LA IMAGEN REAL

Por su parte, el Reclutador tendrá acceso al menú indicado en la figura 5.6.

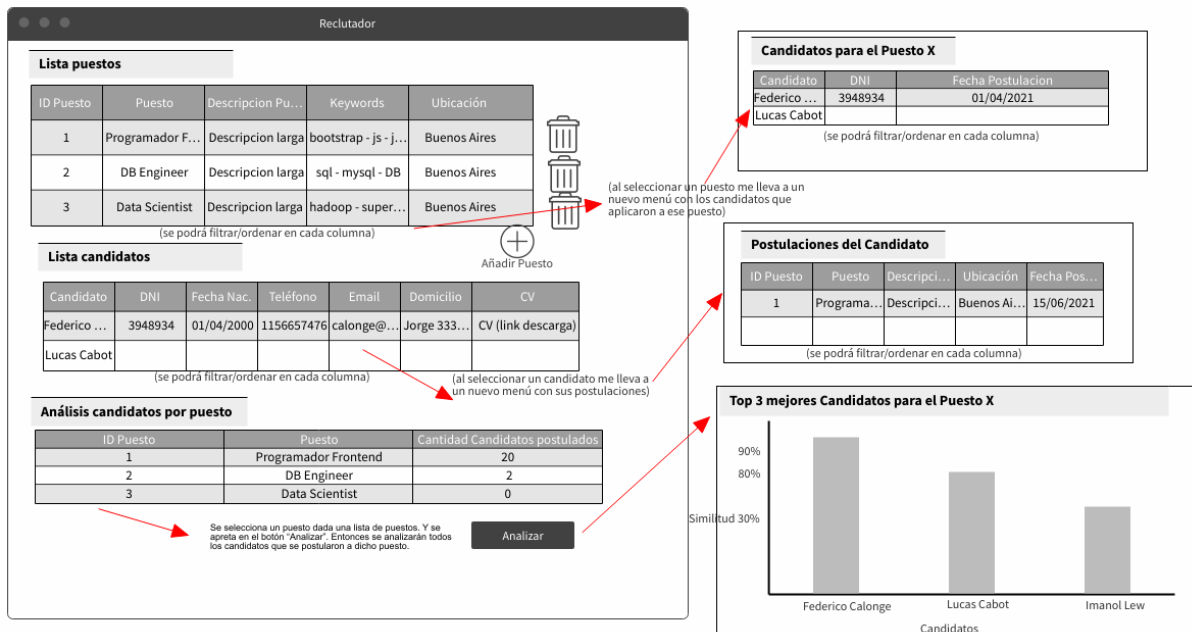


Figura 5.6: Vista del Reclutador.

ES UN BOCETO, FALTA PONER LA IMAGEN REAL

5.5.3 Manejo de los datos.

FALTA

5.5.3.1 Modelado.

FALTA

5.5.3.2 Filtrado.

FALTA

5.5.3.3 Visualización.

FALTA

5.6 Pipeline Flow final del Sistema.

FALTA

Una vez que el reclutador dentro de la sección observada en la figura 5.6 haga click en ".Analizar", el sistema reflejará el pipeline indicado en la figura 5.7 para obtener como resultado un listado con los N candidatos más similares a un puesto determinado, y ordenados de mayor a menor de acuerdo a esta *similitud*. Dicha *similitud* representa el resultado obtenido de la clasificación por nuestro modelo KNN.

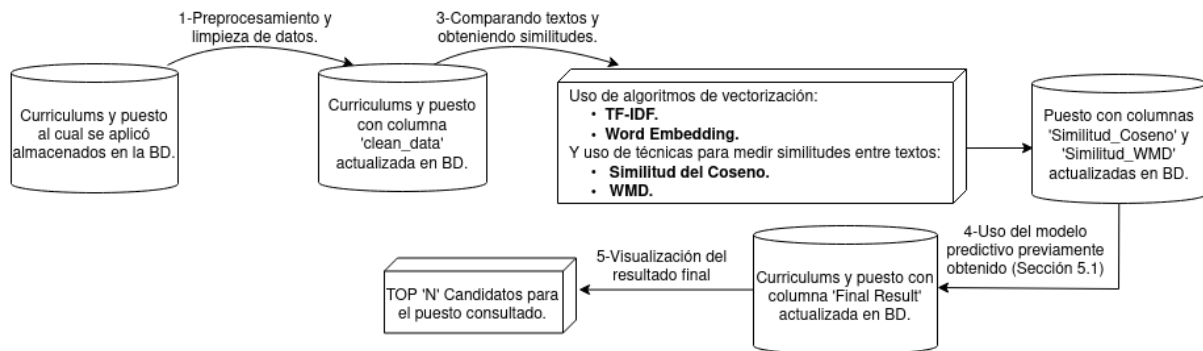


Figura 5.7: Pipeline Flow final del Sistema.

5.7 Conclusiones.

FALTA

5.8 Caso de Uso.

FALTA

5.9 Limitaciones del sistema.

Las limitaciones del sistema son las siguientes:

- Solo acepta curriculums en formato PDF.
- Curriculums y Puestos de trabajo en idioma inglés.
- Curriculums y Puestos de trabajo de IT.

6 Próximos pasos.

FALTA - acá poner mejoras

7 Anexos.

FALTA

Referencias

- [1] World Economic Forum. (2020, Octubre). *The Future of Jobs Report*. (pp. 29-31).
- [2] Derek S. Chapman, & Jane Webster. (2003, Junio-Septiembre). *The Use of Technologies in the Recruiting, Screening, and Selection Processes for Job Candidates*. International journal of selection and assessment. 11(2/3). (pp. 113-114, 117-119).
- [3] Pshdar Abdalla Hamza, Baban Jabbar Othman, Bayar Gardi, Sarhang Sorguli, Hassan Mahmood Aziz, Shahla Ali Ahmed, Bawan Yassin Sabir, Nechirwan Burhan Ismael, Bayad Jamal Ali, & Govand Anwar. (2021, Mayo-Junio). *Recruitment and Selection: The Relationship between Recruitment and Selection with Organizational Performance*. International journal of Engineering, Business and Management (IJEEM). 5(3). (pp. 1-6).
- [4] Luis Argerich, Natalia Golmar, Damián Martinelli, Martín Ramos Mejía, & Juan Andrés Laura. (2019, Enero). *75.06, 95.58 Organización de Datos*. Apunte del Curso Organización de Datos, Universidad de Buenos Aires, Facultad de Ingenieria. (pp. 4-8,351,352,377-379, 387-389,470-477).
- [5] Ladders Company. (2018). *Eye-Tracking Study*. (pp. 2,6).
- [6] Riza Tanaz Fareed, Sharadadevi Kaganurmamath, & Rajath V. (2021, Agosto). *Resume Classification and Ranking using KNN and Cosine Similarity*. International Journal of Engineering Research & Technology (IJERT). 10(8). (pp. 192-194).
- [7] Senthil Kumaran V, & Annamalai Sankar. (2013, Mayo). *Towards an automated system for intelligent screening of candidates for recruitment using ontology mapping (EXPERT)*. International Journal of Metadata, Semantics and Ontologies. 8(1). (pp. 56-64).
- [8] Nuno Silva, & Joao Rocha. (2003). *Ontology Mapping for Interoperability in Semantic Web*, IADIS International Conference WWW/Internet (ICWI), Portugal. (pp. 1).
- [9] Wahiba Ben Abdessalem Karaa, & Nouha Mhimdi. (2011) *Using ontology for resume annotation*. International Journal of Metadata, Semantics and Ontologies. 6(3). (pp. 166-174).
- [10] Duygu Çelik, Askýn Karakas, Gülsen Bal, Cem Gültunca, Atilla Elçi, Basak Buluz, & Murat Can Alevli. (2013, Septiembre). *Towards an Information Extraction System Based on Ontology to Match Resumes and Jobs*. Computer Software and Applications Conference Workshops (COMPSACW), IEEE 37th. (pp. 333-338).
- [11] Frank Färber, Tim Weitzel, & Tobias Keim. (2003, Agosto). *An automated recommendation approach to selection in personnel recruitment*. 9th Americas Conference on Information Systems (AMCIS), USA. (pp. 1-11).
- [12] Chirag Daryania, Gurmeet Singh Chhabrab, Harsh Patel, Indrajeet Kaur Chhabrad, & Ruchi Patel. (2020). *An Automated Resume Screening System using Natural Language Processing and Similarity*. Topics In Intelligent Computing And Industry Design. 2(2). (pp. 99-103).

- [13] Juneja Afzal Ayub Zubeda, Momin Adnan Ayyas Shaheen, Gunduka Rakesh Narsayya Godavari, & Sayed ZainulAbideen Mohd Sadiq Naseem. (2016, Mayo). *Resume Ranking using NLP and Machine Learning*. Proyecto de Tesis para carrera de grado *Bachiller en Ingeniería*. School of Engineering and Technology Anjuman-I-Islam's Kalsekar Technical Campus. (pp. 1-3).
- [14] Jai Janyani, Kartik Agarwal, & Abhishek Sharma. (2018). *Automated Resume Screening System*. Proyecto de Tesis para carrera de grado *Bachiller en Tecnología*. Rajasthan Technical University. (pp. 5, 9-15).
- [15] V. V. Dixit, Trisha Patel, Nidhi Deshpande, & Kamini Sonawane. (2019, Abril). *Resume Sorting using Artificial Intelligence*. International Journal of Research in Engineering, Science and Management. 2(4). (pp. 423-425).
- [16] Dr. K.Satheesh, A.Jahnavi, L Aishwarya, K.Ayesha, G Bhanu Shekhar, & K.Hanisha. (2020). *Resume Ranking based on Job Description using SpaCy NER model*. International Research Journal of Engineering and Technology (IRJET). 7(5). (pp. 74-77).
- [17] Paolo Montuschi, Valentina Gatteschi, Fabrizio Lamberti, Andrea Sanna, & Claudio Demartini. (2014, Septiembre-Octubre). *Job recruitment and job seeking processes: how technology can help*. IT Professional, 16(5). (pp. 41-49).
- [18] Leila Yahiaoui, Zizette Boufaïda, & Yannick Prié. (2006). *Semantic Annotation of Documents Applied to E-Recruitment*. SWAP 2006, the 3rd Italian Semantic Web Workshop. (pp. 1-6).
- [19] Rémy Kessler, Nicolas Béchet, Mathieu Roche, Juan Manuel Torres-Moreno, & Marc El-Bèze. (2012). *A hybrid approach to managing job offers and candidates*. Information Processing & Management. 48(6). (pp. 1124-1135).
- [20] Pradeep Kumar Roy, Sarabjeet Singh Chowdhary, & Rocky Bhatia. (2020). *A Machine Learning approach for automation of Resume Recommendation system*. International Conference on Computational Intelligence and Data Science (ICCIDS). Procedia Computer Science 167. (pp. 2318-2327).
- [21] Ioannis Paparrizos, B. Barla Cambazoglu, & Aristides Gionis. (2011). *Machine learned job recommendation*. 5th ACM Conference on Recommender Systems, ACM. (pp. 325-328).
- [22] Paul Resnick, & Hal R. Varian. (1997). *Recommender Systems*. Communications of the ACM40. (pp. 56-59).
- [23] M. Emre Celebi, Michael W. Berry, Azlinah Mohamed, & Bee Wah Yap. (2020). *Supervised and Unsupervised Learning for Data Science*. Springer book. (pp. 3,4,14,15).
- [24] Karthik Tangirala. (2011). *Semi-supervised and transductive learning algorithms for predicting alternative splicing events in genes*. Proyecto de Tesis para carrera de grado *Master of Science*. Kansas State University. (pp. 1-4).