



UNIVERSIDAD ESAN
FACULTAD DE INGENIERÍA
INGENIERÍA DE TECNOLOGÍAS DE INFORMACIÓN Y SISTEMAS

Predicción del estado de financiamiento de proyectos de tecnología en web de crowdfunding
Kickstarter mediante modelo(s) de Aprendizaje Automático

Tesis para optar el Título de Ingeniero de Tecnologías de Información y Sistemas que
presenta:

Alonso Augusto Puente Ríos
Asesor: Marks Arturo Calderón Niquin

Lima, 19 de diciembre de 2020

Esta tesis denominada:

PREDICCIÓN DEL ESTADO DE FINANCIAMIENTO DE PROYECTOS DE
TECNOLOGÍA EN WEB DE CROWDFUNDING KICKSTARTER MEDIANTE
MODELO(S) DE APRENDIZAJE AUTOMÁTICO

ha sido aprobada.

.....
(Jurado Presidente)

.....
(Jurado)

.....
(Jurado)

Universidad ESAN
2020

PREDICCIÓN DEL ESTADO DE FINANCIAMIENTO DE PROYECTOS DE
TECNOLOGÍA EN WEB DE CROWDFUNDING KICKSTARTER MEDIANTE
MODELO(S) DE APRENDIZAJE AUTOMÁTICO

Agradecimiento y dedicatoria

Durante la inducción en la empresa en la cual realicé mis segundas prácticas pre-profesionales, se realizaron varias actividades, entre ellas, una que me marcó positivamente. Esta consistía en comparar los tiempos de llegada de un punto a otro de una persona corriendo. Se caracterizó porque quien asumió el reto tuvo presente en su mente las personas y las razones por las cuales todos los días lucha y son su principal fuente de motivación.

Por ello, quiero dedicar este gran esfuerzo personal de trabajo de tesis a quienes siempre han estado a mi lado en los mejores y peores momentos, aquellos críticos en que definen el destino. Mi amada hermana Clarisabel, mis queridos padres Augusto e Isabel, mi familia en especial mis abuelos; y mis pocos, pero verdaderos y leales amigos de la universidad, colegio y trabajo. Todos ellos han sido y son cada uno, piedra fundamental en el desarrollo de mi ser como persona y profesional, así como también seres con los cuales siempre comparto gratos momentos. Su presencia en mi vida no ha sido una suerte más sino parte de mi destino. Asimismo, luchar por mis sueños y mi país, y pensar cada día en solidificar su planificación me motivan emocionalmente hasta en aquellos momentos en que parece haber imposibles.

Quiero concluir esta sección, muy especial para mí, agradeciendo también a mi alma máter, la Universidad Esan, y al Programa Nacional de Becas (Pronabec) por hacer que estos 5 años entre el 2015 y 2019 sean mágicos y muy fructíferos. Tuve la oportunidad no solo de incrementar y potenciar mis conocimientos en distintas áreas académicas sino también de aprender de excelentes profesionales como mis profesores, conocer grandes amigos dentro y fuera de su campus (desde el primer ciclo como cachimbo hasta el último ciclo, en el CADE Universitario 2019, estudiantes de diferentes universidades y otras partes del Perú), ponerme a prueba en el exterior (en el II Congreso Internacional de Investigación en Colombia) y formar parte de la gran familia UE.

Por todos ellos, simplemente gracias.

Índice general

Índice de Figuras	8
Índice de Tablas	12
Índice de Ecuaciones	13
1 PLANTEAMIENTO DEL PROBLEMA	16
1.1 Descripción de la Realidad Problemática	16
1.2 Formulación del Problema	19
1.2.1 Problema General	19
1.2.2 Problemas Específicos	20
1.3 Objetivos de la Investigación	20
1.3.1 Objetivo General	20
1.3.2 Objetivos Específicos	20
1.4 Justificación de la Investigación	21
1.4.1 Teórica	21
1.4.2 Práctica	21
1.4.3 Metodológica	21
1.5 Delimitación del Estudio	21
1.5.1 Espacial	21
1.5.2 Temporal	21
1.5.3 Conceptual	22
1.6 Hipótesis	22
1.6.1 Hipótesis General	22
1.6.2 Hipótesis Específicas	22
1.6.3 Matriz de Consistencia	22
2 MARCO TEÓRICO	23
2.1 Antecedentes de la investigación	23
2.1.1 Primer antecedente: «Supervised Learning Model For Kickstarter Campaigns With R Mining» (Kamath & Kamat, 2018)	23

2.1.2 Segundo antecedente: «Predicting Success in Equity Crowdfunding» (Beckwith, 2016)	24
2.1.3 Tercer antecedente: «Money Talks: A Predictive Model on Crowdfunding Success Using Project Description» (Zhou, Zhang, Wang, Du, Qiao & Fan, 2018)	24
2.1.4 Cuarto antecedente: «The Determinants of Crowdfunding Success: A Semantic Text Analytics Approach» (Yuan, Lau & Xu, 2016)	25
2.1.5 Quinto antecedente: «Will your Project get the Green light? Predicting the success of crowdfunding campaigns» (Chen, Chen, Chen, Yang & Lin, 2015)	25
2.1.6 Sexto antecedente: «Project Success Prediction in Crowdfunding Environments» (Y. Li, Rakesh & Reddy, 2016)	26
2.1.7 Séptimo antecedente: «Effect of Social Media Connectivity on Success of Crowdfunding Campaigns» (Kaur & Gera, 2017)	26
2.1.8 Octavo antecedente: «Prediction of Crowdfunding Project Success with Deep Learning» (Yu, Huang, Yang, Liu, Li & Tsai, 2018)	27
2.1.9 Noveno antecedente: «Estimating the Days to Success of Campaigns in Crowdfunding: A Deep Survival Perspective» (Jin, Zhao, Chen, Liu & Ge, 2019)	28
2.1.10 Décimo antecedente: «Success Prediction on Crowdfunding with Multimodal Deep Learning» (Cheng, Tan, Hou & Wei, 2019)	29
2.2 Bases Teóricas	30
2.2.1 Inteligencia Artificial	30
2.2.2 Aprendizaje Automático	32
2.2.3 Aprendizaje Profundo	36
2.2.4 Modelo Predictivo	37
2.2.5 Minería de Datos	38
2.2.6 Metodologías de Minería de Datos	38
2.2.7 Técnicas de Minería de Datos	42
2.2.8 Procesamiento del Lenguaje Natural	62
2.3 Marco Conceptual	63
2.3.1 Crowdfunding	63
2.3.2 Kickstarter	65
2.3.3 Proyecto	65
2.3.4 Campaña	66
3 METODOLOGÍA DE LA INVESTIGACIÓN	68

3.1	Diseño de la investigación	68
3.1.1	Enfoque de la investigación	68
3.1.2	Alcance de la investigación	68
3.1.3	Tipo de la investigación	69
3.1.4	Descripción del prototipo de investigación	69
3.2	Población y muestra	70
3.2.1	Población	70
3.2.2	Muestra	70
3.2.3	Unidad de análisis	70
3.3	Operacionalización de Variables	70
3.4	Instrumentos de medida	71
3.5	Técnicas de recolección de datos	71
3.6	Técnicas para el procesamiento y análisis de la información	72
3.7	Cronograma de actividades y presupuesto	72
4	DESARROLLO DEL EXPERIMENTO	73
4.1	Construcción de los conjuntos finales de datos	73
4.1.1	Metainformación	73
4.1.2	Descripción	77
4.1.3	Comentarios	80
4.2	Análisis Exploratorios de los datos	82
4.2.1	Metainformación	83
4.2.2	Descripción	88
4.2.3	Comentarios	88
4.3	Pre-procesamiento de los conjuntos de datos	92
4.3.1	Metainformación	92
4.3.2	Descripción	93
4.3.3	Comentarios	94
4.4	Creación de los modelos predictivos	95
4.4.1	Metainformación	96
4.4.2	Descripción	97
4.4.3	Comentarios	98
5	ANÁLISIS Y DISCUSIÓN DE RESULTADOS	100
5.1	Metadata	100
5.2	Descripción	100
5.3	Comentarios	102

6 CONCLUSIONES Y RECOMENDACIONES	105
6.1 Conclusiones	105
6.2 Recomendaciones	105
BIBLIOGRAFÍA	106
Anexo	113
Anexo A Árbol de Problemas	114
Anexo B Árbol de Objetivos	115
Anexo C Matriz de Consistencia	116
Anexo D Resumen de Papers investigados	118

Índice de Figuras

1.1 Resultados y ratios obtenidos en la encuesta por GEM y ESAN. Fuente: Redacción Gestión, 2018	17
1.2 Ratio de éxito de proyectos en Kickstarter desde 2009 hasta 2019 (Febrero). Fuente: The Hustle, 2019	19
2.1 Ejemplo de algoritmo de regresión. Fuente: Zambrano, 2018	33
2.2 Ejemplo de algoritmo de clasificación. Fuente: Zambrano, 2018	33
2.3 Algoritmo de K Vecinos más cercanos con pesos ponderados. Fuente: Sancho Caparrini, 2018	34
2.4 Funcionamiento del algoritmo de K medias. Fuente: Sancho Caparrini, 2018 . .	36
2.5 Componentes del Aprendizaje por Refuerzo. Fuente: Sutton y Barto, 2018 . .	36
2.6 Diferencia entre Aprendizaje Automático y Aprendizaje Profundo. Fuente: House of Bots, 2018	37
2.7 Diagrama de los seis pasos básicos. Fuente: Microsoft, 2019	39
2.8 Fases de la metodología CRISP-DM. Fuente: Braulio Gil y Curto Díaz, 2015 .	40
2.9 Fases de la metodología SEMMA. Fuente: Braulio Gil y Curto Díaz, 2015 .	41
2.10 Fases de la metodología KDD. Fuente: Braulio Gil y Curto Díaz, 2015 . . .	41
2.11 Modelo para representar una neurona propuesto por McCulloch y Pitts (1943). Fuente: Russell y Norvig, 2004	43
2.12 Nodos con funciones de activación umbral en forma de puertas lógicas. Fuente: Russell y Norvig, 2004	44
2.13 Función de activación sigmoide. Fuente: Dorofki y col., 2012	45
2.14 Ilustración del algoritmo gradiente descendiente. Fuente: Sancho Caparrini, 2017	46
2.15 Actualización de pesos W con el algoritmo. Fuente: IArtificial.net, 2019b . .	47
2.16 Capa oculta simple MLP con propagación hacia atrás. Fuente: IArtificial.net, 2019b	48
2.17 Redes neuronales de ejemplo. Fuente: A Not So Random Walk, 2019	49
2.18 Función de activación tangente hiperbólica. Fuente: Dorofki y col., 2012 . . .	50
2.19 Función de activación puramente lineal. Fuente: Dorofki y col., 2012	51

2.20 Función de activación Unidad Lineal Rectificada. Fuente: Machine Learning for Artists, 2019	52
2.21 Ejemplo de perceptrón simple. Fuente: Calvo, 2017	52
2.22 Ejemplo de perceptrón multicapa. Fuente: Calvo, 2017	53
2.23 Ejemplo de red neuronal convolucional. Fuente: Calvo, 2017	53
2.24 Modelo Neocognitron de Fukushima (1980). Fuente: F.-F. Li y col., 2019 . . .	54
2.25 Modelo LeNet-5 de LeCun (1998). Fuente: F.-F. Li y col., 2019	54
2.26 Ejecución de la convolución en una entrada. Fuente: López Briega, 2016	55
2.27 Generación de una nueva imagen a partir de filtros. Fuente: F.-F. Li y col., 2019	55
2.28 Secuencia de varias capas convolucionales. Fuente: F.-F. Li y col., 2019	56
2.29 Extracción de características a partir de convoluciones. Fuente: F.-F. Li y col., 2019	56
2.30 Ejemplo de matriz de imagen de entrada y un filtro. Fuente: Prabhu, 2018 . . .	57
2.31 Dimensiones de una entrada y un filtro. Fuente: F.-F. Li y col., 2019	58
2.32 Paso de 2 píxeles por parte de un filtro. Fuente: Prabhu, 2018	58
2.33 Aplanado de matrices luego de agrupar la capa. Fuente: Prabhu, 2018	59
2.34 Arquitectura completa de una CNN. Fuente: Prabhu, 2018	59
2.35 Ejemplo de red neuronal recurrente. Fuente: Calvo, 2018a	60
2.36 Hiperplano con dos clases separadas por una distancia m. Fuente: Betancourt, 2005	60
2.37 Ejemplo de caso linealmente separable. Fuente: Betancourt, 2005	61
2.38 Ejemplo de caso no linealmente separable. Fuente: Betancourt, 2005	61
2.39 Aplicación de un kernel para transformar el espacio de los datos. Fuente: Be- tancourt, 2005	61
2.40 Ejemplo del algoritmo de árbol de decisión. Fuente: Russell y Norvig, 2004 . .	62
3.1 Marco de trabajo del prototipo final. Fuente: Elaboración propia	69
3.2 Cronograma de actividades de la investigación. Fuente: Elaboración propia . .	72
4.1 Vista (agosto 2019) del website Web Robots. Fuente: Web Robots, 2019 . . .	74
4.2 Tamaño de conjunto de datos al corte de Julio 2019. Fuente: Elaboración propia.	74
4.3 Tamaño de conjunto de datos filtrado del corte de Julio 2019. Fuente: Elabora- ción propia.	75
4.4 Flujo de trabajo de la data final en Alteryx Designer. Fuente: Elaboración propia.	76
4.5 Visualización del archivo de metainformación subido a Kaggle. Fuente: Elabo- ración propia.	77
4.6 Función del algoritmo web scraping de la descripción de proyectos. Fuente: Elaboración propia.	79

4.7 Visualización del archivo de descripción subido a Kaggle. Fuente: Elaboración propia.	79
4.8 Función del algoritmo web scraping de los comentarios. Fuente: Elaboración propia.	80
4.9 Visualización del archivo de comentarios subido a Kaggle. Fuente: Elaboración propia.	81
4.10 Instancias lanzadas en paralelo para la extracción de comentarios. Fuente: Elaboración propia.	81
4.11 Distribución de proyectos tecnológicos según su estado. Fuente: Elaboración propia.	82
4.12 Evolución de cantidad de proyectos tecnológicos por año. Fuente: Elaboración propia.	82
4.13 Evolución de proyectos tecnológicos, por su estado y año. Fuente: Elaboración propia.	83
4.14 Distribución de categorías de tecnología. Fuente: Elaboración propia.	84
4.15 Distribución de países. Fuente: Elaboración propia.	84
4.16 Distribución de divisa del monto patrocinado. Fuente: Elaboración propia	84
4.17 Diagrama de caja y bigote de patrocinadores. Fuente: Elaboración propia.	85
4.18 Diagrama de caja y bigote de meta. Fuente: Elaboración propia.	85
4.19 Diagrama de caja y bigote de monto patrocinado. Fuente: Elaboración propia.	86
4.20 Diagrama de caja y bigote de duración. Fuente: Elaboración propia.	86
4.21 Matriz de correlaciones entre variables independientes. Fuente: Elaboración propia.	87
4.22 Gráfico de dispersión de correlaciones entre variables independientes. Fuente: Elaboración propia.	87
4.23 Gráfico alterno de dispersión de correlaciones entre variables independientes. Fuente: Elaboración propia.	88
4.24 Nube de palabras de descripciones. Fuente: Elaboración propia.	89
4.25 Aperturas de proyectos por estado de financiamiento y presencia de comentarios. Fuente: Elaboración propia.	89
4.26 Aperturas de proyectos por presencia de comentarios y estado de financiamiento. Fuente: Elaboración propia.	90
4.27 Distribución de comentarios en proyectos exitosos y fracasados. Fuente: Elaboración propia.	90
4.28 Nube de palabras de comentarios por unidades más frecuentes. Fuente: Elaboración propia.	91

4.29 Nube de palabras de comentarios por unidades o parejas más frecuentes. Fuente: Elaboración propia.	91
4.30 Matriz de correlaciones entre variables independientes considerando adicionales. Fuente: Elaboración propia.	92
4.31 Flujograma de limpieza de conjunto de datos de descripciones. Fuente: Elaboración propia.	93
4.32 Nube de palabras de descripciones posterior al pre-procesamiento. Fuente: Elaboración propia.	94
4.33 Nube de palabras de comentarios posterior al pre-procesamiento. Fuente: Elaboración propia.	95
4.34 Arquitectura de modelo MLP para la metadata. Fuente: Elaboración propia.	96
4.35 Función de tokenización de palabras de descripciones. Fuente: Elaboración propia.	98
4.36 Función de codificación de palabras y relleno de arreglos. Fuente: Elaboración propia.	98
4.37 Elaboración de matriz de incrustaciones de palabras. Fuente: Elaboración propia.	98
4.38 Arquitectura de modelo CNN para las descripciones. Fuente: Elaboración propia.	99
 5.1 Evolución de la exactitud de los subconjuntos de entrenamiento y validación para el modelo MLP de metadata para un entrenamiento de 100 épocas. Fuente: Elaboración propia.	101
5.2 Evolución de la pérdida de los subconjuntos de entrenamiento y validación para el modelo MLP de metadata para un entrenamiento de 100 épocas. Fuente: Elaboración propia.	101
5.3 Matriz de confusión para el modelo MLP de metadata. Fuente: Elaboración propia.	102
5.4 Evolución de la exactitud de los subconjuntos de entrenamiento y validación para el modelo CNN de descripciones para un entrenamiento de 100 épocas. Fuente: Elaboración propia.	103
5.5 Evolución de la pérdida de los subconjuntos de entrenamiento y validación para el modelo CNN de descripciones para un entrenamiento de 100 épocas. Fuente: Elaboración propia.	103
5.6 Matriz de confusión para el modelo CNN de descripciones. Fuente: Elaboración propia.	104

Índice de Tablas

2.1 Cuadro comparativo entre características de las tres metodologías. Fuente: Shafique y Qaiser, 2014	42
3.1 Diccionario de datos del conjunto final a entrenar. Fuente: Elaboración propia. . .	71
4.1 Diccionario de datos del conjunto final de metainformación. Fuente: Elaboración propia.	78

Índice de Ecuaciones

2.1 Cálculo de los pesos para el algoritmo K-NN mediante ponderación de sus distancias. Fuente: Sancho Caparrini, 2018	34
2.2 Fórmula alternativa del algoritmo K-NN mediante sumatoria de pesos. Fuente: Sancho Caparrini, 2018	34
2.3 Fórmula del algoritmo k-means. Fuente: Sancho Caparrini, 2018	35
2.4 Fórmula del cálculo del valor de un nodo i. Fuente: Russell y Norvig, 2004	44
2.5 Fórmula de una función de activación g para la salida del nodo. Fuente: Russell y Norvig, 2004	44
2.6 Fórmula de la función de activación sigmoide. Fuente: Dorofki y col., 2012	45
2.7 Fórmula de función de coste de una regresión logística. Fuente: Pardo, s.f.	45
2.8 Actualización de pesos W mediante gradiente descendiente. Fuente: IArtificial.net, 2019b	47
2.9 Fórmula del algoritmo de propagación hacia atrás. Fuente: Bertona, 2005	48
2.10 Cálculo del error cometido en una red neuronal. Fuente: Viera Balanta, 2013	48
2.11 Actualización de pesos mediante propagación hacia atrás. Fuente: Viera Balanta, 2013	49
2.12 Cálculo de errores de nodos usando pesos actualizados. Fuente: Viera Balanta, 2013	49
2.13 Fórmula de la función de activación tangente hiperbólica. Fuente: Dorofki y col., 2012	50
2.14 Fórmula de la función de activación puramente lineal. Fuente: Dorofki y col., 2012	50
2.15 Fórmula de la función de activación ReLU. Fuente: Calvo, 2018b	51
2.16 Fórmula matemática de la convolución. Fuente: Figueroa M., s.f.	54
2.17 Cálculo del volumen del mapa de activación. Fuente: Prabhu, 2018	56
2.18 Cálculo del tamaño de la imagen reducida. Fuente: F.-F. Li y col., 2019	57
2.19 Ecuación del hiperplano para clasificar dos clases. Fuente: Betancourt, 2005	60

Resumen

Conocer el destino del financiamiento de proyectos siempre ha sido el principal deseo de todos los emprendedores que los promocionan en Internet, en especial, de la categoría de tecnología por ser los que presentan las ratios más bajas de éxito debido a sus altas metas que buscan alcanzar. El presente trabajo de investigación se basó en construir un modelo predictivo cuyo objetivo es la de estimar el éxito o fracaso de financiamiento de proyectos tecnológicos en la plataforma de crowdfunding Kickstarter durante la duración de su campaña a partir de su metainformación, imagen y/o descripción. Para ello, se crearon modelos de Aprendizaje Automático (SVM, MLP y CNN) para cada una de estas partes. Luego de analizar todos los modelos con las mismas métricas mencionadas en el décimo antecedente, se concluyó que solo los de metainformación tuvieron niveles excelentes de acuerdo a sus puntajes AUC (0.8377 para SVM y 0.7043 para MLP), mientras que los modelos de descripciones tuvieron un rendimiento regular (0.6746 para SVM con TF-IDF y 0.6709 para SVM con BoW) y finalmente el modelo de imágenes no logró clasificar las clases correctamente. Como trabajo a futuro, estos últimos modelos de descripción e imagen serán mejorados para construir, junto con los de metainformación, un modelo ensamblado basado en considerables variables del proyecto.

Palabras claves: metainformación, descripción, imagen del proyecto, Máquina de Vectores de Soporte (SVM), Perceptrón Multicapa (MLP), Red Neuronal Convolucional (CNN).

Knowing funding projects destiny has always been the main desire of all entrepreneurs who promote them on the Internet, especially in the technology category because they have the lowest success rates due to their high goals. The present research work was based on building a predictive model whose objective is to estimate the success or failure of technological projects funding in the Kickstarter crowdfunding platform during the duration of its campaign based on its metadata, image and/or description. For this, Machine Learning models (SVM, MLP and CNN) were created for each of these parts. After analyzing all the models with the same metrics mentioned in the tenth antecedent, it was concluded that only metadata models had excellent levels according to their AUC scores (0.8377 for SVM and 0.7043 for MLP), while the description models had a regular performance (0.6746 for SVM with TF-IDF and 0.6709 for SVM with BoW) and finally the image model failed to classify the classes correctly. As future work, these latest models of description and image will be improved to build, together with metadata ones, an assembled model base don considerable project variables.

Palabras claves: project metadata, project description, project image, Support Vector Machine (SVM), Multilayer Perceptron (MLP), Convolutional Neural Network (CNN).

Introducción

Por muchos años, en especial en las dos últimas décadas, diversos proyectos emprendedores han sido lanzados en distintas plataformas web, buscando un objetivo compartido por todos: ser financiados en un determinado plazo para hacer realidad estas ideas. Entre fracasos y éxitos, han surgido nuevas tendencias, así como nuevos enfoques de estudios de estos casos para encontrar la clave que descifre las variables de éxito.

El presente trabajo de investigación se basó formular un modelo ensamblado robusto que determine el estado final de un proyecto, agregando un nuevo enfoque: basarse solamente en proyectos de tecnología, la segunda categoría con más baja probabilidad de éxito al final de una campaña. En estudios previos, los modelos planteados resultaron bastante aceptables debido a que el resto de categorías de proyectos balancearon la inequidad existente en las dos clases del estado.

El reto principal fue el de construir modelos predictivos que consideren las tres características más importantes de un proyecto: la primera basada en la metainformación, la segunda, en la imagen principal del proyecto y la tercera, en la descripción del mismo, para ser ensamblados más adelante en uno solo.

Para ello, se recolectó un total de 27,251 proyectos tecnológicos en Kickstarter entre los períodos 2009-2019, de los cuales 27,035 registros finalmente fueron usados para cada uno de los tres modelos. Algunos proyectos provenientes de países fuera del territorio de los Estados Unidos y en distintos idiomas fueron considerados dentro de esta cantidad ya que no afectó al rendimiento general como en casos particulares de algunos estudios previos.

La principal motivación de este trabajo fue la de aportar una herramienta de ayuda para los emprendedores que les permita estimar el estado final del financiamiento de su proyecto de tecnología, es decir, éxito o fracaso, con un nivel confiable de probabilidad de éxito del mismo durante el transcurso de su campaña, permitiendo además servir de soporte en la toma de decisiones de cara a lograr su principal objetivo.

Capítulo 1

PLANTEAMIENTO DEL PROBLEMA

1.1. Descripción de la Realidad Problemática

El emprendimiento hoy en día es una realidad en todo el mundo. Desde crear productos nuevos hasta crear nuevas formas de hacer las cosas, todo gracias a ideas nacidas a partir de querer satisfacer nuestras propias necesidades.

Nuestro país no es ajeno a ello. El 50.6% de la población entre 18 y 64 años tiene la expectativa de iniciar un emprendimiento dentro de los tres próximos años de acuerdo al último reporte de Global Entrepreneurship Monitor (GEM) 2014. El 62.3% de la población entre ese rango de edad, además, tiende a ser más optimista en su percepción de oportunidades. Asimismo, según informa la Cámara de Comercio de Lima, la iniciativa emprendedora responde más a la identificación de una oportunidad de negocio que a una falta de oportunidad de empleo ([Redacción Gestión, 2015](#)). Sin embargo, en un estudio más reciente basado en una encuesta realizada a residentes peruanos entre junio y julio del 2017 desarrollada por el equipo GEM Perú y ESAN a 2080 personas entre el mismo rango de edad, el 24.6% de emprendimientos se encuentra en fase temprana, es decir, representa una dificultad para el emprendedor peruano llegar a etapas más avanzadas como un emprendimiento establecido (negocios con más de 3.5 años, que representan solo el 7.4% para Perú), ubicando así a nuestro país en la posición 25 de 54 economías a nivel mundial ([Redacción Gestión, 2018](#)). En la Figura 1.1 se aprecian algunas ratios del estudio.

Estos resultados desfavorables tienen como base el ecosistema poco beneficioso para los emprendimientos que permitan su establecimiento en el entorno nacional, con condiciones asociadas al acceso de financiamiento, políticas gubernamentales que alienen la implementación de Innovación y Desarrollo en las empresas, acceso a infraestructura física y asesoría

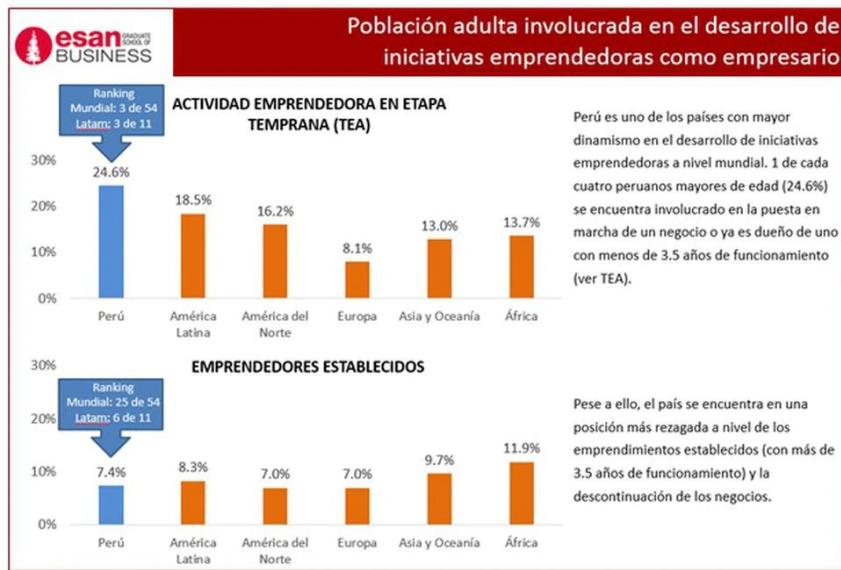


Figura 1.1: Resultados y ratios obtenidos en la encuesta por GEM y ESAN. Fuente: [Redacción Gestión, 2018](#)

a nivel comercial y profesional, como sostiene el investigador del equipo GEM Perú Carlos Guerrero ([Redacción Gestión, 2018](#)). La Asociación de Emprendedores de Perú (ASEP) afirma, asimismo, que en la región solo se invierte el 1.5 % del PIB en actividades de ciencia, tecnología e innovación, y las limitaciones son dadas por barreras burocráticas ejercidas por el Gobierno y el sector privado ([Asociación de Emprendedores de Perú, 2018](#)). En adición a esto, otras razones que representan barreras para emprender son la falta de conocimientos en la iniciación de un negocio, su tramitación, la fuente de financiamiento del proyecto o búsqueda de inversionistas, la cultura, la falta de fomento de emprendimiento y la falta de una red de contactos ([Sandoval, s.f.](#)).

Ante estas limitaciones, en la actualidad muchos emprendedores se ven forzados a mostrar sus proyectos al público en la Internet con el fin de captar personas interesadas en ayudarlos en el financiamiento de estos. Por ello, se han creado plataformas web con el fin de permitir la interacción entre los proyectos publicados en un determinado tiempo, el cual puede variar entre 30 y 120 días, y la comunidad en general que deseé colaborar con una cantidad de dinero para su financiamiento. El sitio web solo servirá para mostrar los proyectos presentados a detalle por los creadores y la promoción de estos al público. La idea es que, al término de este plazo de tiempo, el proyecto sea financiado y se logre convertir en una realidad. A esta práctica se le conoce como crowdfunding ([Universo Crowdfunding, s.f.](#)).

En Latinoamérica, son muy pocos los países los que se incorporan en el crowdfunding, entre los principales se destacan Chile, México, Argentina y Brasil. Sin embargo, el modelo funciona distinto a países de Norteamérica y Europa debido a la cultura diferente y resistencia

a su implementación por la poca confianza en el éxito de los proyectos. En los países mencionados, se encontró que los proyectos audiovisuales, a pesar de encontrarse en desventaja que en sus pares europeos y norteamericanos, se estrenaron 4,135 títulos extranjeros en la región iberoamericana frente a 791 propios, de los cuales 162 fueron obras brasileñas y 131, argentinas, en el año 2015. Estas estadísticas se apoyan con el incremento de aperturas de salas de cines en nuestra región en las dos últimas décadas. Los factores principales: el apoyo de gobiernos latinoamericanos, la masificación de las comunicaciones digitales y por ende, la participación activa de usuarios en redes sociales y el papel protagónico que toma el crowdfunding para apoyar estos proyectos (22,179 proyectos financiados con éxito en Kickstarter con más de 1 millón de dólares de recaudación en el 2017) ([López-Golán y col., 2017](#)). Asimismo, en los últimos años se decidió seguir una manera muy similar a los modelos de Estados Unidos, basados en la creación de campañas de un emprendedor para obtener fondos para sus ideas con la moneda norteamericana pero limitados a las leyes económicas de cada país ([Solidaridad Latina, s.f.](#)).

En el caso de nuestro país, el crowdfunding sí existe y es apoyada por universidades y organizaciones sin fines de lucro, cuyo objetivo es generar empresas comerciales que aumenten la calidad de vida en la comunidad, además de convertir a las personas en socios financieros ([Fernández-Bedoya y col., 2020](#)). Esta actividad es parte de uno de los 4 grupos básicos de economía colaborativa: el financiamiento colaborativo ([Stokes y col., 2014](#)).

Entre los sitios web más conocidos de crowdfunding están Kickstarter e Indiegogo. Kickstarter, desde su inicio en 2009, es una plataforma de financiamiento de proyectos creativos de todo tipo, los cuales incluyen películas, juegos, música, arte, diseño y tecnología. Actualmente, se han registrado más de 162 mil proyectos realizados, 16 millones de contribuyentes y 4,3 miles de millones de dólares fondeados ([Kickstarter, s.f.-a](#)). La plataforma utiliza un modelo de financiamiento llamado “todo o nada”, el cual consiste en que si un proyecto no alcanza su meta de financiamiento en un determinado plazo de tiempo, no se realiza ninguna transacción de fondos ([Kickstarter, s.f.-d](#)). Si bien los patrocinadores apoyan estos proyectos por motivos personales y distintos para hacerlos realidad, ellos no obtienen la propiedad o los ingresos de los proyectos que financian, sino que los creadores conservan la totalidad de su trabajo ([Kickstarter, s.f.-f](#)).

Para los proyectos tecnológicos, en contraste, la ratio de éxito es uno de los más bajos de las categorías existentes (20%) solo por delante de Artesanía y Periodismo, como se aprecia en la Figura 1.2.

Ya existen estudios previos para predecir la probabilidad de éxito de financiamiento para este tipo de proyectos utilizando técnicas de Aprendizaje Automático. Sin embargo, la mayoría de los modelos predictivos propuestos no arrojan resultados con exactitud muy alta

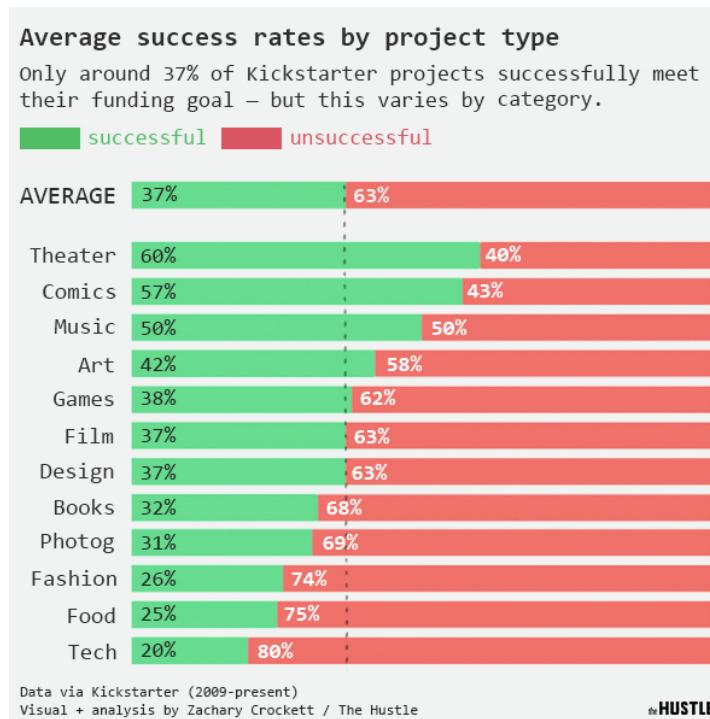


Figura 1.2: Ratio de éxito de proyectos en Kickstarter desde 2009 hasta 2019 (Febrero). Fuente: [The Hustle, 2019](#)

ya que su rango varía entre 60 y 70%. Esto conlleva a generar imprecisión para pronosticar confiablemente el éxito de financiamiento de estos proyectos de tecnología. Para el presente trabajo de tesis, se creó un modelo predictivo alimentado de datos históricos de la plataforma para estimar el estado final de financiamiento de un proyecto aleatorio, así como su probabilidad de éxito.

1.2. Formulación del Problema

Para la formulación de los problemas de la presente investigación, se elaboró un «árbol de problemas» (véase Anexo A).

1.2.1. Problema General

Bajos niveles de precisión de modelos entrenados de Aprendizaje Automático para cualquier categoría para predecir estado de financiamiento de proyectos de tecnología.

1.2.2. Problemas Específicos

- Variables de proyectos no normalizadas y varianzas altas.
- Datos faltantes o incompletos de proyectos.
- Parámetros de modelos no ajustados.
- Sobreajuste de aprendizaje de modelos y clasificación incorrecta de las dos clases del estado final de financiamiento (exitoso o fracasado).
- Predicción incorrecta de estado de financiamiento de un proyecto tecnológico.

1.3. Objetivos de la Investigación

Para la formulación de los objetivos de la presente investigación, se elaboró un «árbol de objetivos» (véase Anexo B)

1.3.1. Objetivo General

Construir modelo(s) de Aprendizaje Automático entrenado(s) para predecir correctamente proyectos de tecnología con nivel de precisión aceptable.

1.3.2. Objetivos Específicos

- Normalizar variables de proyectos y reducir niveles altos de varianza.
- Eliminar datos faltantes o incompletos de proyectos.
- Ajustar parámetros de modelos.
- Evitar sobreajuste de aprendizaje de modelos.
- Predecir correctamente el estado final de financiamiento de cualquier proyecto tecnológico (éxito o fracaso).

1.4. Justificación de la Investigación

1.4.1. Teórica

Esta investigación se basa en crear un modelo de Aprendizaje Automático que sea aplicable a proyectos de tecnología de la plataforma Kickstarter por presentar bajas performances en antecedentes.

1.4.2. Práctica

Al culminar la investigación, se ofrecerá un modelo predictivo confiable que ayude a los emprendedores en la toma de decisiones respecto a sus proyectos a partir del insight obtenido de los resultados que deriven a la manipulación de los datos de entrada.

1.4.3. Metodológica

Se creará un modelo predictivo a partir de las variables finales seleccionadas, previa limpieza de datos. Luego, será entrenado y evaluado por las métricas correspondientes. Finalmente, se lanzará una versión de prueba que reciba datos de entrada para predecir la viabilidad de un proyecto de tecnología.

1.5. Delimitación del Estudio

1.5.1. Espacial

Para la presente investigación, se considerará el territorio de los Estados Unidos ya que tanto la campaña del proyecto a servir para la investigación como los datos fuentes de proyectos relacionados financiados previamente, que servirán para la elaboración del modelo predictivo, se encuentran en dicho país.

1.5.2. Temporal

El periodo de tiempo abarcará desde el año 2009, fecha en el cual se tiene registrado los primeros conjuntos de datos de proyectos en Kickstarter hasta el mes de agosto del año 2019,

últimos registros descargados hasta el inicio del presente trabajo.

1.5.3. Conceptual

La presente investigación consistirá en la implementación de un modelo predictivo del estado de financiamiento de un proyecto tecnológico en Kickstarter basado en técnicas y conceptos de Aprendizaje Automático, previamente evaluando cuál de todas las existentes genera un mejor desempeño para su uso y análisis de resultados.

1.6. Hipótesis

1.6.1. Hipótesis General

El modelo entrenado de Aprendizaje Automático logrará predecir correctamente proyectos de tecnología con nivel de precisión aceptable.

1.6.2. Hipótesis Específicas

- Las variables de los proyectos descargados se normalizarán y se reducirán los niveles altos de varianza.
- Los datos faltantes o incompletos de los proyectos serán eliminados.
- Los parámetros de los modelos usados serán ajustados.
- Se evitará el sobreajuste de aprendizaje de modelos para clasificar correctamente las dos clases del estado final de financiamiento.
- El estado final de financiamiento de cualquier proyecto tecnológico será predicho correctamente.

1.6.3. Matriz de Consistencia

A continuación se presenta la matriz de consistencia elaborada para la presente investigación (véase Anexo C).

Capítulo 2

MARCO TEÓRICO

2.1. Antecedentes de la investigación

En esta sección se presentarán diversos trabajos de investigación basados en la predicción de éxito o fracaso de campañas en Kickstarter o plataformas similares y el análisis de estas utilizando conjunto de datos de la propia plataforma o almacenadas en otros repositorios, con sus variables respectivas. En la mayoría de casos consideraron variables básicas que se obtienen del repositorio de las plataformas de crowdfunding, en otros casos consideraron variables cualitativas basadas en texto y descripción de proyectos, y en otros antecedentes usaron nuevas técnicas poco convencionales como el Aprendizaje Profundo e híbridos de modelos para obtener los mejores resultados posibles. Asimismo, a continuación se presenta un cuadro resumen (véase Anexo D) de lo que se presenta en esta sección.

2.1.1. Primer antecedente: «Supervised Learning Model For Kickstarter Campaigns With R Mining» ([Kamath & Kamat, 2018](#))

Kamath y Kamat realizaron un artículo de investigación el cual fue publicado en la revista «Resources Policy» en el año 2018. Este fue titulado «Supervised Learning Model For Kickstarter Campaigns With R Mining» la cual traducida al español significa «Estimación del precio del cobre utilizando el algoritmo bat».

2.1.1.1. Planteamiento del Problema y objetivo

2.1.1.2. Técnicas empleadas por los autores

2.1.1.3. Metodología empleada por los autores

2.1.1.4. Resultados obtenidos

2.1.2. Segundo antecedente: «Predicting Success in Equity Crowdfunding» ([Beckwith, 2016](#))

Beckwith realizó un artículo de investigación el cual fue publicado en la revista «Resources Policy» en el año 2018. Este fue titulado «Predicting Success in Equity Crowdfunding» la cual traducida al español significa «Estimación del precio del cobre utilizando el algoritmo bat».

2.1.2.1. Planteamiento del Problema y objetivo

2.1.2.2. Técnicas empleadas por los autores

2.1.2.3. Metodología empleada por los autores

2.1.2.4. Resultados obtenidos

2.1.3. Tercer antecedente: «Money Talks: A Predictive Model on Crowdfunding Success Using Project Description» ([Zhou, Zhang, Wang, Du, Qiao & Fan, 2018](#))

Zhou y col. realizaron un artículo de investigación el cual fue publicado en la revista «Resources Policy» en el año 2018. Este fue titulado «Money Talks: A Predictive Model on Crowdfunding Success Using Project Description» la cual traducida al español significa «Estimación del precio del cobre utilizando el algoritmo bat».

2.1.3.1. Planteamiento del Problema y objetivo

2.1.3.2. Técnicas empleadas por los autores

2.1.3.3. Metodología empleada por los autores

2.1.3.4. Resultados obtenidos

2.1.4. Cuarto antecedente: «The Determinants of Crowdfunding Success: A Semantic Text Analytics Approach» ([Yuan, Lau & Xu, 2016](#))

Yuan y col. realizaron un artículo de investigación el cual fue publicado en la revista «Resources Policy» en el año 2018. Este fue titulado «The Determinants of Crowdfunding Success: A Semantic Text Analytics Approach» la cual traducida al español significa «Estimación del precio del cobre utilizando el algoritmo bat».

2.1.4.1. Planteamiento del Problema y objetivo

2.1.4.2. Técnicas empleadas por los autores

2.1.4.3. Metodología empleada por los autores

2.1.4.4. Resultados obtenidos

2.1.5. Quinto antecedente: «Will your Project get the Green light? Predicting the success of crowdfunding campaigns» ([Chen, Chen, Chen, Yang & Lin, 2015](#))

Chen y col. realizaron un artículo de investigación el cual fue publicado en la revista «Resources Policy» en el año 2018. Este fue titulado «Will Your Project Get the Green Light? Predicting the Success of Crowdfunding Campaigns» la cual traducida al español significa «Estimación del precio del cobre utilizando el algoritmo bat».

2.1.5.1. Planteamiento del Problema y objetivo

2.1.5.2. Técnicas empleadas por los autores

2.1.5.3. Metodología empleada por los autores

2.1.5.4. Resultados obtenidos

2.1.6. Sexto antecedente: «Project Success Prediction in Crowdfunding Environments» ([Y. Li, Rakesh & Reddy, 2016](#))

Y. Li y col. realizaron un artículo de investigación el cual fue publicado en la revista «Resources Policy» en el año 2018. Este fue titulado «Project Success Prediction in Crowdfunding Environments» la cual traducida al español significa «Estimación del precio del cobre utilizando el algoritmo bat».

2.1.6.1. Planteamiento del Problema y objetivo

2.1.6.2. Técnicas empleadas por los autores

2.1.6.3. Metodología empleada por los autores

2.1.6.4. Resultados obtenidos

2.1.7. Séptimo antecedente: «Effect of Social Media Connectivity on Success of Crowdfunding Campaigns» ([Kaur & Gera, 2017](#))

Kaur y Gera realizaron un artículo de investigación el cual fue publicado en la revista «Resources Policy» en el año 2018. Este fue titulado «Effect of Social Media Connectivity on Success of Crowdfunding Campaigns» la cual traducida al español significa «Estimación del precio del cobre utilizando el algoritmo bat».

2.1.7.1. Planteamiento del Problema y objetivo

2.1.7.2. Técnicas empleadas por los autores

2.1.7.3. Metodología empleada por los autores

2.1.7.4. Resultados obtenidos

2.1.8. Octavo antecedente: «Prediction of Crowdfunding Project Success with Deep Learning» ([Yu, Huang, Yang, Liu, Li & Tsai, 2018](#))

Yu y col. realizaron un resumen de la conferencia 2018 IEEE 15th International Conference on e-Business Engineering (ICEBE) del 12 al 14 de octubre del 2018 en Xi'an, China. Este fue titulado «Prediction of Crowdfunding Project Success with Deep Learning», la cual traducida al español significa «Predicción del éxito de proyecto de financiamiento colectivo con Aprendizaje Profundo».

2.1.8.1. Planteamiento del Problema y objetivo

Debido al aumento dramático de la actividad de financiamiento colectivo en los últimos años, en el cual tanto creadores como patrocinadores participan, son más las campañas que buscan alcanzar su objetivo. Sin embargo, según el análisis empírico, solo la tercera parte lo logra.

Por ello, los autores plantean como objetivo el desarrollo de un modelo que prediga el éxito del proyecto de crowdfunding con Aprendizaje Profundo. Se recolectaron más de 378 mil campañas de Kickstarter entre Mayo 2009 y Marzo 2018.

2.1.8.2. Técnicas empleadas por los autores

Los autores elaboraron un modelo basado en un Perceptrón Multicapa (MLP), alimentado por 6 variables visibles de la campaña (metadata). Además, sus resultados se compararon con otros modelos considerados en sus antecentes como Bosques aleatorios, AdaBoost, Máquina de vectores de soporte, Árboles de decisión, Regresión logística y Naïve Bayes.

2.1.8.3. Metodología empleada por los autores

Se recolectaron registros históricos de campañas de Kickstarter recopilados retrospectivamente de Kaggle. A continuación, realizaron análisis exploratorio de los datos para entender la data y seleccionar variables, y finalmente construyeron un modelo, el cual se comparó sus resultados con los de otros modelos.

2.1.8.4. Resultados obtenidos

Para comparar los resultados del modelo propuesto con los antecedentes, se tomaron en cuenta como métricas la exactitud y el área bajo la curva (AUC). De acuerdo a ellas, para ambos escenarios se lograron mejor performance, con una exactitud de 0.9320 y un área bajo la curva de 0.9323 frente al mejor modelo de los antecedentes, Bosques Aleatorios, que obtuvo como resultados 0.9293 y 0.9272 para las mismas métricas respectivamente.

2.1.9. Noveno antecedente: «Estimating the Days to Success of Campaigns in Crowdfunding: A Deep Survival Perspective» ([Jin, Zhao, Chen, Liu & Ge, 2019](#))

Jin y col. realizaron un resumen de la conferencia The 33rd AAAI Conference on Artificial Intelligence (AAAI'2019) del 27 de enero al 1 de febrero del 2019 en Honolulu, Estados Unidos. Este fue titulado «Estimating the Days to Success of Campaigns in Crowdfunding: A Deep Survival Perspective» la cual traducida al español significa «Estimación de días de éxito de campañas en financiamiento colectivo: Una perspectiva de supervivencia profunda».

2.1.9.1. Planteamiento del Problema y objetivo

Dado el incremento de actividad en campañas de crowdfunding en sitios como Kickstarter o Indiegogo, los estudios sobre este tema también han ido de la mano. Sin embargo, la mayoría de investigaciones se enfocan en predecir el ratio de éxito de financiamiento en una campaña.

Los autores plantean la predicción de la fecha exacta de finalización de una campaña, así como controlar la distribución de patrocinios conforme a la evolución del tiempo. El reto dado de predecir el tiempo de duración es más complicado porque se ve afectada por la variabilidad de la metadata y los comentarios del proyecto.

2.1.9.2. Técnicas empleadas por los autores

Se elaboró un modelo Seq2seq (encoder + decoder) con antecedentes multifacéticos (SMP) para la distribución de patrocinios y el tiempo de éxito, así como también un modelo evolutivo lineal para mantener el cambio de las distribuciones de patrocinios.

2.1.9.3. Metodología empleada por los autores

Se definió el problema en primer lugar, luego los detalles técnicos de SMP así como el pre-procesamiento de entrada. A continuación, se construyó la arquitectura SMP y se definieron los antecedentes multifacéticos. Finalmente, se entrenó el modelo con más de 14 mil proyectos de Indiegogo.

2.1.9.4. Resultados obtenidos

Respecto a la predicción de distribución de patrocinios, y evaluando mejor con RECM, los modelos propuestos de SMP fueron los más efectivos (0.135 para 70 % de ratio de partición) para el Encoder. Respecto a la predicción del tiempo de éxito, los modelos propuestos de SMP (el mejor con 0.96 para 50 % de ratio de partición) fueron mejores que modelos de referencia para el Decoder.

2.1.10. Décimo antecedente: «Success Prediction on Crowdfunding with Multimodal Deep Learning» ([Cheng, Tan, Hou & Wei, 2019](#))

Cheng y col. realizaron un artículo de investigación el cual fue publicado en la revista «Resources Policy» en el año 2018. Este fue titulado «Success Prediction on Crowdfunding with Multimodal Deep Learning» la cual traducida al español significa «Estimación del precio del cobre utilizando el algoritmo bat».

2.1.10.1. Planteamiento del Problema y objetivo

2.1.10.2. Técnicas empleadas por los autores

2.1.10.3. Metodología empleada por los autores

2.1.10.4. Resultados obtenidos

2.2. Bases Teóricas

2.2.1. Inteligencia Artificial

La Inteligencia Artificial es la inteligencia llevado a cabo por máquinas en las que una máquina “inteligente” ideal es un agente flexible que percibe su entorno y lleva a cabo acciones que maximicen sus posibilidades de éxito en algún objetivo ([Poole y col., 1998](#)). Este término se aplica cuando una máquina imita las funciones “cognitivas” que asocian los humanos con otras mentes ([Russell & Norvig, 2009](#)).

Durante la historia de la humanidad, se han seguido 4 enfoques: dos centrados en el comportamiento humano y dos enfocados en torno a la racionalidad. El enfoque centrado en el comportamiento humano se basa en una ciencia empírica, es decir, mediante experimentos que incluyen hipótesis y confirmaciones. Este enfoque nace a partir de la prueba de Alan Turing, en 1950, en la cual, el célebre matemático inglés diseñó una prueba basada en la incapacidad de diferenciar entre entidades inteligentes indiscutibles y seres humanos por parte de un computador. Si este era capaz de diferenciar y superar la prueba mientras que el humano no, se afirma que se trataba de una “máquina inteligente”. Por ello, el computador debía contar con las siguientes capacidades: procesamiento de lenguaje natural para poder comunicarse, representación del conocimiento describiendo lo que percibe de su entorno, razonamiento automático utilizando la información procesada en su interior, y aprendizaje automático para adaptarse a nuevos eventos. Si el evaluador decide incluir una señal de video para evaluar la percepción de la computadora, se dice que se está realizando la Prueba Global de Turing. Para superarla, además de las 4 anteriormente mencionadas, la computadora debe contar además con las capacidades de visión computacional para percibir objetos y robótica con el fin de manipularlos. Todas estas seis capacidades o disciplinas abarcan la mayor parte de la Inteligencia Artificial ([Russell & Norvig, 2004](#)).

Por el otro lado, el enfoque racional implica una combinación de ingeniería y matemáticas basándose en las “leyes del pensamiento”. Estas parten de la Grecia antigua, planteadas por

grandes filósofos como Aristóteles en su intento de codificar la “manera correcta de pensar”, lo que más adelante derivó al estudio de la lógica. Más adelante, en el siglo XIX, se construyeron programas capaces de resolver problemas en notación lógica. De ahí que la tradición logista dentro del campo de la Inteligencia Artificial trata de construir sistemas inteligentes con estas capacidades. De todo lo anterior dicho respecto al enfoque racional se creó el término de un agente racional, el cual actúa intentando lograr el mejor resultado, o de existir incertidumbre, el mejor resultado esperado. Finalmente, la amplia aplicación de la Inteligencia Artificial y sus fundamentos derivan en muchas ciencias de las cuales se pueden mencionar, además de la filosofía y las matemáticas, a la economía, neurociencia, psicología, la ingeniería computacional, la teoría de control y cibernetica, y hasta la lingüística ([Russell & Norvig, 2004](#)).

Pero, ¿cómo es surge este amplio estudio de la Inteligencia Artificial? En 1943, basándose en la fisiología básica y funcionamiento de las neuronas en el cerebro, el análisis formal de la lógica proposicional de Russell y Whitehead, y la teoría computacional de Turing, dos estudiantes en neurociencia realizaron juntos el que sería considerado primer trabajo de Inteligencia Artificial. Warren McCulloch y Walter Pitts propusieron un modelo constituido por neuronas artificiales, en el que cada una de ellas se caracterizaba por estar “activada” o “desactivada”; la del primer tipo daba como resultado a la estimulación producida por una cantidad suficiente de neuronas vecinas. Como ejemplo, mostraron que cualquier función de cómputo podría calcularse mediante alguna red de neuronas interconectadas y que todos los conectores lógicos eran capaces de ser implementados usando estructuras sencillas de red. Seis años más adelante, Donald Hebb propuso una regla de actualización de intensidades de conexiones entre las neuronas, la que actualmente se le conoce como la “regla de aprendizaje Hebbiano” vigente hasta nuestros días. En 1956, Allen Newell y Herbert Simon inventaron un programa de computación en el taller de Dartmouth de John McCarthy, que era capaz de pensar de forma no numérica, basado en el Teórico Lógico, artículo que, además, fue rechazado de ser publicado en la revista *Journal of Symbolic Logic*. A pesar de ello, los trabajos de los colaboradores presentes en dicho taller se mantuvieron por 20 años más, siendo McCarthy quien acuñó el término de “Inteligencia Artificial” a este campo ([Russell & Norvig, 2004](#)).

En la década de los años 80, la Inteligencia Artificial dio el gran salto de formar parte de la industria, en especial, de las compañías más grandes de los países desarrollados a través de grupos especializados para la realización de investigaciones de sistemas expertos, así como en la construcción de computadoras cada vez más potentes y capaces de resolver tareas más complejas.

Actualmente, la IA cuenta con muchas aplicaciones como la Minería de Datos, el procesamiento de lenguaje natural, la robótica, los videojuegos, entre otros. Dentro de ella se pueden encontrar otras ramas como por ejemplo el Aprendizaje Automático, Visión computacional,

etcétera.

2.2.2. Aprendizaje Automático

El Aprendizaje Automático (*Machine Learning* por su nombre en inglés) es una rama de la Inteligencia Artificial cuyo fin es desarrollar técnicas que las computadoras pueden aprender a través de encontrar algoritmos y heurísticas que conviertan muestras de datos en programas sin necesidad de hacerlos (Russell & Norvig, 2009). Sus algoritmos están compuestos por muchas tecnologías, como por ejemplo Aprendizaje Profundo, Redes Neuronales y Procesamiento de lenguaje natural, utilizadas en el aprendizaje supervisado y no supervisado, las cuales operan guiadas por lecciones de información existente (Gartner, 2019a). La premisa básica del aprendizaje automático es construir algoritmos que puedan recibir datos de entrada y usar análisis estadísticos para predecir una salida mientras se actualizan las salidas a medida que se dispone de nuevos datos (Alpaydin, 2014).

Los tres tipos de aprendizaje principales son:

- **Aprendizaje supervisado:** Se trabajan con datos etiquetados buscando obtener una función que asigne una respuesta de salida adecuada, denominadas etiquetas, a partir de unos datos de entrada denominadas características (Zambrano, 2018). Por lo general, los datos de entrada son conocidos como variables dependientes o X, mientras que los datos de salida son llamadas variables independientes o Y. Se le dice supervisado ya que el resultado depende de los datos que recibe de entrada, afectando su performance si estos son alterados.

Existen dos tipos de aprendizaje supervisado. El primero es la regresión, que consiste en obtener como resultado un número específico a partir de un conjunto de variables de las características, representado en la Figura 2.1; mientras que por otra parte está la clasificación, el cual se basa en encontrar distintos patrones ocultos para clasificar los elementos del conjunto de datos en diferentes grupos, como se aprecia en la Figura 2.2 (Zambrano, 2018).

Para el segundo tipo de aprendizaje supervisado, el algoritmo más usado es el de los K Vecinos más cercanos o *k-NN Nearest Neighbour* en inglés. Este se basa en la idea de que los nuevos ejemplos serán clasificados a la clase a la cual pertenezca la mayor cantidad de vecinos más cercanos del conjunto de entrenamiento más cercano a él. Sin embargo, el número k de vecinos más cercanos lo decide el usuario, de preferencia impar, para evitar ambigüedad al momento de clasificar un registro por parte del algoritmo (esto puede ocurrir por las mismas distancias existentes entre dos o más registros). Otra variante

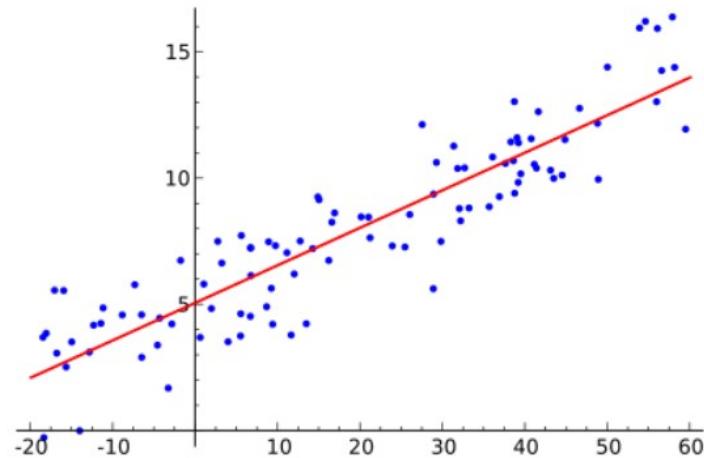


Figura 2.1: Ejemplo de algoritmo de regresión. Fuente: [Zambrano, 2018](#)

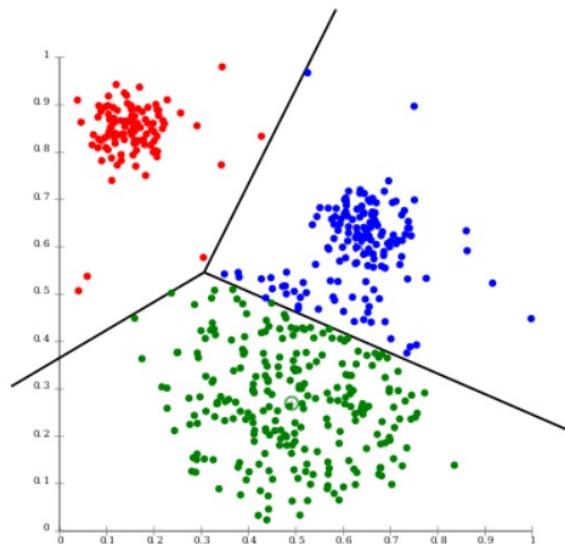


Figura 2.2: Ejemplo de algoritmo de clasificación. Fuente: [Zambrano, 2018](#)

aplicada consiste en la ponderación de cada vecino de acuerdo a la distancia entre él y el ejemplar a ser clasificado, asignando mayor peso a los más próximos ([Sancho Caparrini, 2018](#)). Para ello, se tiene la Ecuación 2.1 y su variante en la Ecuación 2.2:

$$W_i = \frac{1}{d(x, x_i)^2}$$

Ecuación 2.1: Cálculo de los pesos para el algoritmo K-NN mediante ponderación de sus distancias.

Fuente: [Sancho Caparrini, 2018](#)

$$\operatorname{argmax}_{v \in V} \sum_{i=1 \dots k, x_i \in v} W_i$$

Ecuación 2.2: Fórmula alternativa del algoritmo K-NN mediante sumatoria de pesos. Fuente: [Sancho Caparrini, 2018](#)

Donde:

x = ejemplo que se desea clasificar

V = posibles clases de clasificación

x_i = conjunto de los k ejemplos de entrenamiento más cercanos

y finalmente, la clase asignada a x es aquella que verifique que la suma de los pesos de sus representantes sea la máxima, representándose en la Figura 2.3:

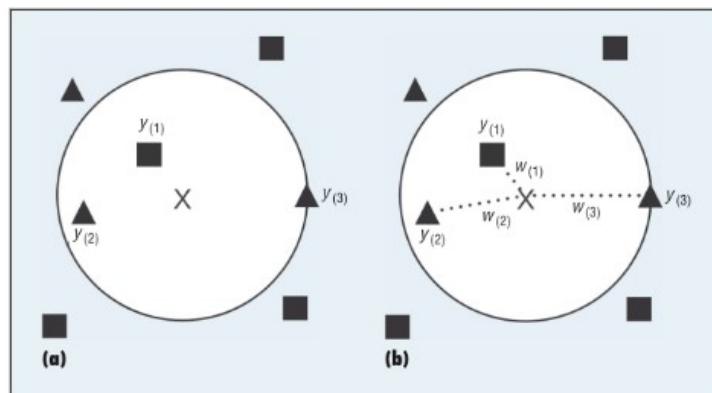


Figura 2.3: Algoritmo de K Vecinos más cercanos con pesos ponderados. Fuente: [Sancho Caparrini, 2018](#)

- **Aprendizaje no supervisado:** A diferencia de la anterior, aquí se trabaja con datos no etiquetados para entrenar el modelo, ya que el fin es de carácter exploratorio y descriptivo de la estructura de los datos. No existen variables independientes o Y.

La función es agrupar ejemplares, por lo que el algoritmo los cataloga por similitud en sus características y a partir de ahí, crea grupos o clústeres sin tener la capacidad de definir cómo es cada individualidad de cada uno de los integrantes de los mismos ([Zambrano, 2018](#)).

El algoritmo usado para este tipo de aprendizaje es el de las K medias o *k-means* en inglés. Este intenta encontrar una partición de las muestras en K agrupaciones, de manera que cada ejemplar pertenezca a una de ellas de acuerdo al centroide más cercano. Si bien el valor de K es definido por el usuario, a partir de pruebas de varias iteraciones se le puede consultar al algoritmo cuál es su valor óptimo. La función para lograr esto se observa en la Ecuación [2.3](#):

$$\sum_i \sum_j d(x_j^i, c_i)^2$$

Ecuación 2.3: Fórmula del algoritmo k-means. Fuente: [Sancho Caparrini, 2018](#)

Donde:

c_i = centroide de la agrupación i-ésima

x_j^i = conjunto de ejemplos clasificados de la agrupación i-ésima

Representándose en la Figura [2.4](#), los pasos seguidos para este algoritmo comienzan con la selección de los K puntos como centros de los grupos. Luego, se asignarán los ejemplos al centro más cercano y se calculará el centroide de los ejemplos asociados a cada grupo. Finalmente, estos dos últimos pasos se repetirán hasta que ninguno de los centros pueda ser reasignados en las iteraciones.

- **Aprendizaje por refuerzo:** Se basa en que un agente racional puede tomar una decisión a partir de una retroalimentación llamada recompensa o refuerzo. A diferencia del Aprendizaje Supervisado, en donde el agente puede aprender solamente a partir de ejemplos dados, en este caso no basta solamente con proporcionárselos sino también de “informarle” si lo está haciendo de la manera correcta o no. Por ejemplo, un agente que intenta aprender a jugar ajedrez necesita saber que algo bueno ha ocurrido cuando gana y algo malo ha ocurrido cuando pierde. La mejor recompensa que busca al finalizar el juego es vencer al oponente, y para ello debe estudiar todos los movimientos que este haga, la posición de las fichas en el tablero, entre otros. A este conjunto se le conoce como entorno o medio ambiente ([Russell & Norvig, 2004](#)). Entonces, en resumen, representando en la Figura [2.5](#), y mencionando otro ejemplo, el aprendizaje por refuerzo está compuesto por un agente (Pacman) en un estado determinado (su ubicación o posición actual) dentro

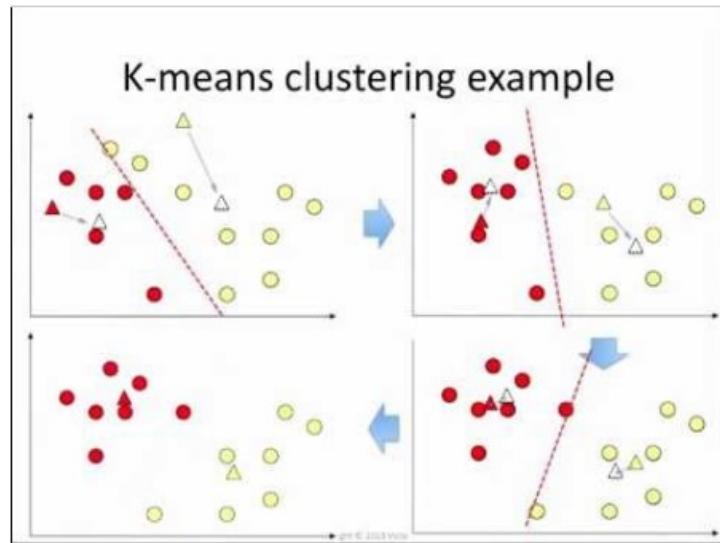


Figura 2.4: Funcionamiento del algoritmo de K medias. Fuente: [Sancho Caparrini, 2018](#)

de un medio ambiente (el laberinto). La recompensa positiva que busca Pacman son los puntos por comer, mientras que la negativa será la de morir si se cruza con un fantasma, en base a la acción (desplazamiento a un nuevo estado) que realice ([Merino, 2019](#)).

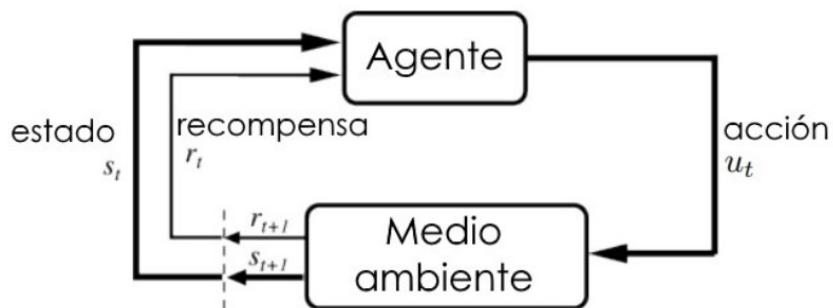


Figura 2.5: Componentes del Aprendizaje por Refuerzo. Fuente: [Sutton y Barto, 2018](#)

2.2.3. Aprendizaje Profundo

El Aprendizaje Profundo (*Deep Learning* por su nombre en inglés) es un tipo de Aprendizaje Automático que entrena a una computadora para que realice tareas como las realizadas por los seres humanos, desde la identificación de imágenes hasta realizar predicciones y reconocer el lenguaje humano. El Aprendizaje Profundo configura parámetros básicos acerca de los datos y entrena a la computadora para que aprenda por su cuenta reconociendo patrones mediante el uso de múltiples capas de procesamiento ([SAS Institute, s.f.](#)). Se basa en teorías acerca de cómo funciona el cerebro humano ([BBVA OpenMind, 2019](#)).

La principal diferencia con el Aprendizaje Automático es que el Aprendizaje Profundo se basa en la extracción de características y clasificación al mismo tiempo luego de recibir una entrada, algo que en la primera técnica ocurre por separado, como se aprecia en la Figura 2.6.

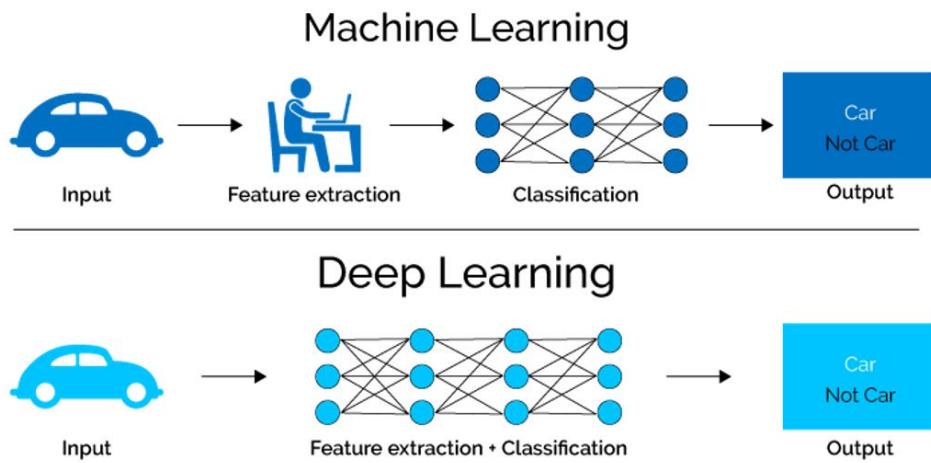


Figura 2.6: Diferencia entre Aprendizaje Automático y Aprendizaje Profundo. Fuente: [House of Bots, 2018](#)

Por un lado, mientras en el aprendizaje automático o de máquina, el ordenador extrae conocimiento a través de experiencia supervisada, en el aprendizaje profundo está menos sometido a supervisión. Mientras que el primer tipo de aprendizaje consume muchísimo tiempo y se basa en proponer abstracciones que permiten aprender al ordenador, en el segundo no consume demasiado tiempo y por el contrario de su par, crea redes neuronales a gran escala que permiten que el ordenador aprenda y piense por sí mismo sin necesidad directa de intervención humana. Actualmente, el aprendizaje profundo se usa para crear softwares capaces de determinar emociones o eventos descritos en textos, reconocimiento de objetos en fotografías y realizar predicciones acerca del posible comportamiento futuro de las personas. Empresas como Google (proyecto Google Brain) o Facebook (Unidad de investigación en IA) han puesto en marcha proyectos basados en esta rama para potenciar y mejorar sus algoritmos con el fin de ofrecer una mejor experiencia de sus servicios a sus clientes ([BBVA OpenMind, 2019](#)).

2.2.4. Modelo Predictivo

Son modelos de datos estadísticos utilizados para predecir el comportamiento futuro. En estos, se recopilan datos históricos y actuales, se formula un modelo estadístico, se realizan predicciones y el modelo se valida a medida que se dispone de datos adicionales. Los modelos predictivos analizan el rendimiento pasado para evaluar la probabilidad de que un cliente mues-

tre un comportamiento específico en el futuro. En esta categoría también abarca la búsqueda de patrones ocultos ([Gartner, 2019b](#)).

2.2.5. Minería de Datos

La Minería de Datos es un campo de la estadística y las ciencias de la computación referido al proceso que intenta descubrir patrones en grandes volúmenes de conjuntos de datos ([Maimon & Rokach, 2010](#)). Normalmente, estos patrones no pueden detectarse mediante la exploración tradicional de datos porque sus relaciones son demasiado complejas o por su gran volumen. Para ello, utiliza métodos de Inteligencia Artificial, Aprendizaje Automático, estadística y sistemas de bases de datos. Estos patrones son recopilados y definidos como un modelo de minería de datos, los cuales pueden aplicarse en los siguientes escenarios ([Microsoft, 2019](#)):

- Previsión.
- Riesgo y probabilidad.
- Recomendaciones.
- Buscar secuencias.
- Agrupación.

La generación de un modelo de minería de datos forma de un macro-proceso descrita en los siguientes seis pasos representados en la Figura 2.7 ([Microsoft, 2019](#)):

2.2.6. Metodologías de Minería de Datos

Dentro de los sistemas de analítica de negocio, Big Data y Minería de Datos, las tres metodologías más usadas se encuentran CRISP-DM, SEMMA y KDD ([Braulio Gil & Curto Díaz, 2015](#)).

- **CRISP-DM** (Cross Industry Standard Process for Data Mining):

Esta metodología presenta seis fases representadas en la Figura 2.8 a continuación.

- En la comprensión del negocio se determinan los objetivos y requerimientos desde el lado del negocio, así como generar plan del proyecto.

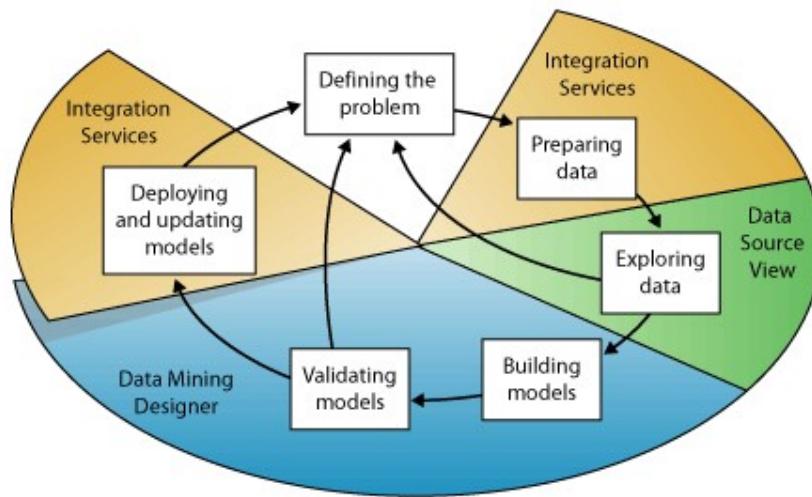


Figura 2.7: Diagrama de los seis pasos básicos. Fuente: [Microsoft, 2019](#)

- En la comprensión de los datos se logra entender el significado de las variables existentes, así como el entendimiento de los datos desde su recopilación hasta su verificación de calidad.
 - En la preparación de los datos se prepara el conjunto de datos adecuado que servirán para la construcción del modelo. Por ello, la calidad de los datos es un factor relevante y ello requiere la exclusión de redundancia y valores que no ayuden a establecer buena comprensión y resultados más adelante. A esto se le conoce como limpieza de datos.
 - En el modelado se aplican técnicas de minería de datos en el conjunto de datos creado en el paso anterior. Para ello, se evalúan entre varias la que mejor performance desempeñe y luego se construye el o los modelos que busquen determinar un objetivo.
 - En la evaluación se evalúan los posibles modelos del paso anterior a partir del nivel de importancia de acuerdo a las necesidades del negocio y performance que estos cuentan.
 - El despliegue, finalmente, utiliza el modelo final creado para determinar los objetivos que se buscan cumplir en los requerimientos y ayudar en la toma de decisiones.
- **SEMMA (Sample – Explore – Modify – Model – Assess):**
- Esta metodología cuenta con cinco fases como se aprecia en la Figura 2.9. A diferencia de la anterior, esta metodología se enfoca más en el modelado.
- En la Muestra (Sample) se crea una muestra significativa.

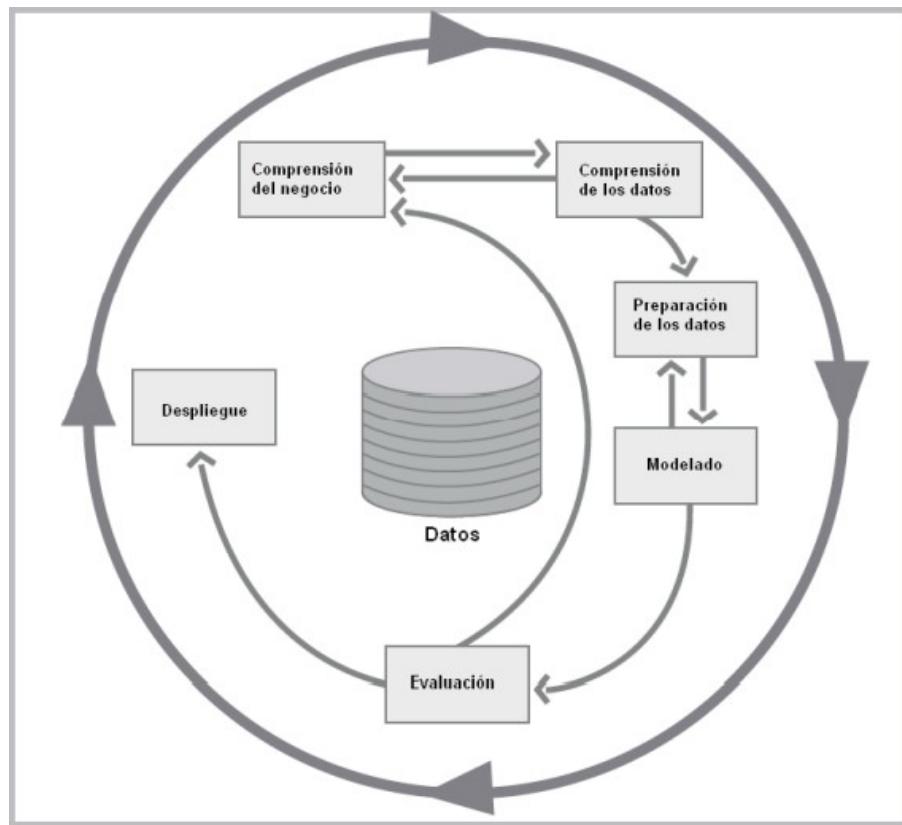


Figura 2.8: Fases de la metodología CRISP-DM. Fuente: [Braulio Gil y Curto Díaz, 2015](#)

- En la Exploración (Explore) se comprenden los datos con el fin de encontrar relaciones entre variables y anomalías.
 - En la Modificación (Modify) se transforman las variables para las necesidades del modelo.
 - En la Modelización (Model) se aplican uno o varios modelos sobre el conjunto de datos para buscar resultados.
 - En el Asesoramiento (Assessment) se evalúan los resultados obtenidos del modelo.
- **KDD (Knowledge Discovery and Data Mining):**

Esta metodología se refiere al proceso de encontrar conocimiento alguno en el dato y, a diferencia de sus predecesores, se enfoca en crear aplicaciones de minería de datos. Consiste de cinco fases más 1 previa y 1 posterior basadas en la generación de conocimiento como se muestra en la Figura 2.10.

- En la fase Pre KDD se comprende el dominio del negocio, así como también se identifican las necesidades del cliente.
- En la selección, primero se identifica el conjunto de datos a usar y luego se seleccionan la muestra y las variables para la exploración.

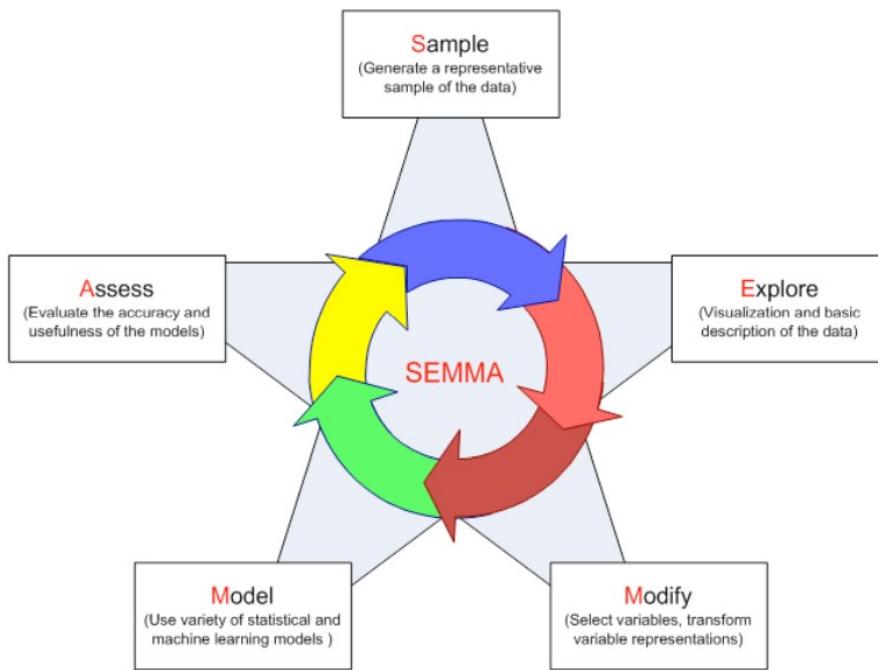


Figura 2.9: Fases de la metodología SEMMA. Fuente: [Braulio Gil y Curto Díaz, 2015](#)

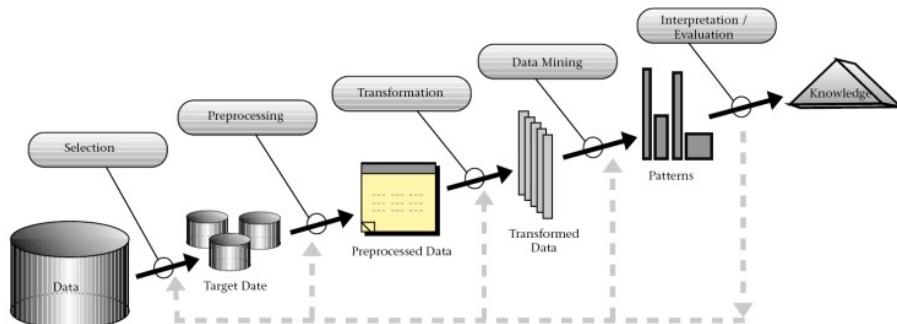


Figura 2.10: Fases de la metodología KDD. Fuente: [Braulio Gil y Curto Díaz, 2015](#)

- En el pre-procesamiento, se realiza la limpieza de datos y se elimina el ruido, así como los valores atípicos.
- En la transformación se implementan métodos de reducción de dimensiones para reducir el número de variables efectivas.
- En la Minería de datos, se elige el tipo de tarea de minería de datos (clasificación, regresión, agrupamiento, entre otros) así como el algoritmo, los métodos, los modelos y parámetros apropiados.
- En la interpretación y evaluación se analizan los resultados dados.
- En la fase Post KDD finalmente se consolida el conocimiento adquirido.

Luego de presentar las tres metodologías más usadas, la pregunta dada es ¿cuál de los

tres representa la mejor opción para usar? Las tres metodologías tienen distinto número de pasos, así como distintos enfoques, tal cual se observa en el siguiente resumen de la Tabla 2.1.

Modelo de Procesos de Minería de Datos	KDD	CRISP-DM	SEMMA
Número de pasos	9	6	5
Nombre de los pasos	Desarrollo y entendimiento de la aplicación	Entendimiento del negocio	-
	Creación de un conjunto de datos de destino	Entendimiento de los datos	Muestreo
	Limpieza de datos y pre-procesamiento		Exploración
	Transformación de datos	Preparación de los datos	Modificación
	Elección de la tarea adecuada de Minería de datos	Modelamiento	Modelo
	Elección del algoritmo adecuado de Minería de datos		
	Implementación del algoritmo de Minería de datos		
	Interpretación de patrones minados	Evaluación	Evaluación
	Uso de conocimiento descubierto	Despliegue	-

Tabla 2.1: Cuadro comparativo entre características de las tres metodologías. Fuente: [Shafique y Qaiser, 2014](#)

Sin embargo, la elección depende de los involucrados que finalmente usarán el modelo en el negocio. La mayoría de investigadores siguen la metodología KDD debido a que es más completo y su exactitud. Para aquellos objetivos enfocados más en la compañía como la integración usada por SAS Enterprise Miner con su software se utilizan SEMMA y CRISP-DM. Esta última resulta ser más completa de acuerdo a los estudios.

2.2.7. Técnicas de Minería de Datos

Existe una gran variedad de técnicas para la Minería de Datos. Las más importantes y utilizadas en los antecedentes de la investigación se mencionan a continuación ([Microsoft, 2018](#)).

- **Redes Neuronales Artificiales (RNA):** Es un sistema de computación que consiste en un número de elementos o nodos simples, pero altamente interconectados, llamados “neuronas”, que se organizan en capas que procesan información utilizando respuestas de estado dinámico a entradas externas ([Inzaugarat, 2018](#)).

Este sistema de programas y estructura de datos se aproxima al funcionamiento del cerebro humano. Una red neuronal implica tener un gran número de procesadores funcionando en paralelo, teniendo cada uno de ellos su propia esfera de conocimiento y acceso a datos en su memoria local. Normalmente, una se alimenta con grandes cantidades de datos y un conjunto dado de reglas acerca de las relaciones. Luego, un programa puede indicar a la red cómo debe comportarse en respuesta a un estímulo externo o si puede iniciar la actividad por sí misma ([BBVA OpenMind, 2019](#)).

Para entender mejor cómo funciona una red neuronal, hay que describir qué es una neurona. Una neurona es una célula del cerebro cuya función principal es la recogida, procesamiento y emisión de señales eléctricas. Debido a que se piensa que la capacidad de procesamiento de información del cerebro proviene de redes de este tipo de neuronas, los primeros trabajos en Inteligencia Artificial se basaron en crear redes neuronales artificiales para emular este comportamiento, en 1943 con un modelo matemático, mostrado en la Figura 2.11, por los ya mencionados anteriormente McCulloch y Pitts. Estos y posteriores trabajos potenciaron lo que hoy en día se conoce como el campo de la neurociencia computacional ([Russell & Norvig, 2004](#)). Años más tarde, en 1958, se desarrolló el concepto del perceptrón por Rosenblatt, el cual tenía la capacidad de aprender y reconocer patrones sencillos, formado por entradas, neurona, función de adaptación (sigmoidal, tangencial, en escalón, etc.) y salida.

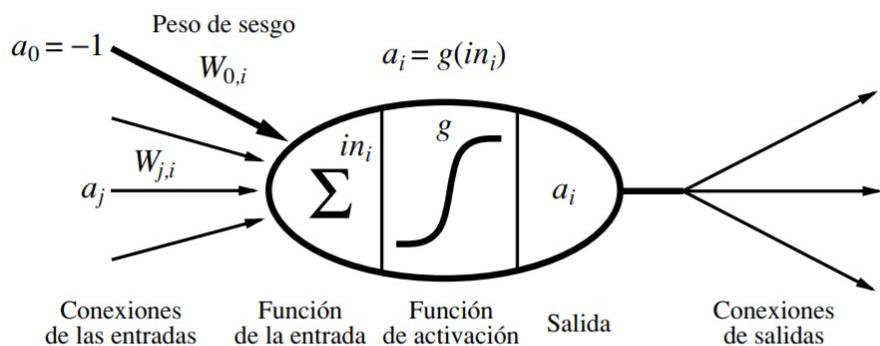


Figura 2.11: Modelo para representar una neurona propuesto por McCulloch y Pitts (1943). Fuente: [Russell y Norvig, 2004](#)

La última figura descrita muestra, además de los pesos, funciones de activación tanto para la entrada (a_j) como para la salida (a_i). Pero, ¿qué son estas funciones y para qué sirven?

Para comenzar, las redes neuronales están compuestas de nodos (la elipse) conectados a través de conexiones dirigidas (las flechas). Una conexión del nodo j a la unidad i sirve para propagar la activación a_j de j a i . Asimismo, cada conexión tiene un peso numérico

$W(j,i)$ que determina la fuerza y el signo de la conexión. Para calcular cada nodo i , se realiza una suma ponderada de sus entradas (producto entre pesos y nodos de entrada j), y se le añade el sesgo (*bias*) θ_i (aumenta/disminuye el valor de la combinación lineal de las entradas) como se observa en la Ecuación 2.4:

$$in_i = \sum_{j=0}^n W_{j,i} * a_j + \theta_i$$

Ecuación 2.4: Fórmula del cálculo del valor de un nodo i . Fuente: [Russell y Norvig, 2004](#)

Posteriormente, se efectúa una función de activación g a esta suma para producir la salida de la Ecuación 2.5:

$$a_i = g(in_i) = g\left(\sum_{j=0}^n W_{j,i} * a_j + \theta_i\right)$$

Ecuación 2.5: Fórmula de una función de activación g para la salida del nodo. Fuente: [Russell y Norvig, 2004](#)

Entonces, aquí se explica los dos objetivos de una función de activación. En primer lugar, se desea que el nodo esté “activo” (cercano a +1) cuando las entradas correctas sean dadas, e “inactiva” (cercano a 0) cuando las entradas erróneas sean proporcionadas. En segundo lugar, la activación tiene que ser no lineal porque, de lo contrario, la red neuronal colapsaría en su totalidad con una función lineal sencilla, como se aprecia en el ejemplo de la Figura 2.12.

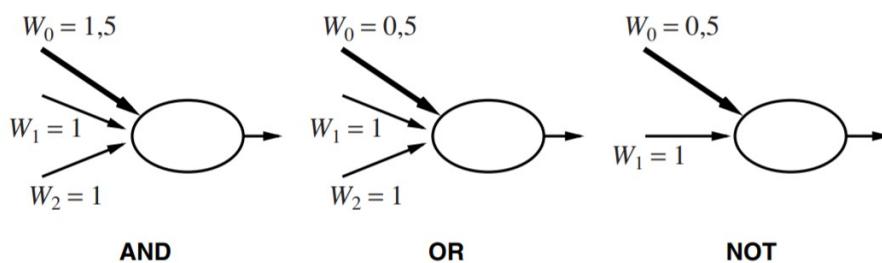


Figura 2.12: Nodos con funciones de activación umbral en forma de puertas lógicas. Fuente: [Russell y Norvig, 2004](#)

Entre las funciones de activación que más destacan son las siguientes:

- **Función sigmoide o logística:** Toma los valores de entrada que oscilan entre infinito negativo y positivo, y restringe los valores de salida al rango entre 0 y 1.

Frecuentemente es usada en Redes Multicapa (MLP) entrenadas con el algoritmo de propagación inversa. Se representa como en la Figura 2.13 y su fórmula para calcular su nuevo valor es la Ecuación 2.6:

$$a = \text{Logsig}(n) = \frac{1}{1 + e^{-n}}$$

Ecuación 2.6: Fórmula de la función de activación sigmoide. Fuente: [Dorofki y col., 2012](#)

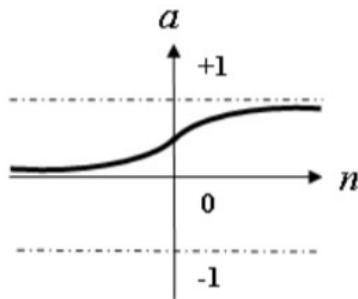


Figura 2.13: Función de activación sigmoide. Fuente: [Dorofki y col., 2012](#)

Un dato curioso de esta función relacionado con la regresión logística es que el nombre de esta última no deriva de una regresión. Por el contrario, se debe a que, al principio de la neurona, se realiza una combinación lineal muy parecida a una regresión lineal y después se aplica la función logística o sigmoide. De ahí el origen del nombre ([IArtificial.net, 2019a](#)).

- **Regresión Logística:** Como se mencionó antes, es similar a un modelo de regresión lineal, pero está adaptado para modelos en los que la variable dependiente es dicotómica, es decir, presenta solo dos posibles valores. Resulta muy útil para los casos en los que se desea predecir la presencia o ausencia de una característica o resultado según los valores de un conjunto de predictores ([IBM, 2019](#)). Su función de coste que se optimiza con gradiente descendiente se representa mediante la Ecuación 2.7:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^m [-y_i * \log(h_\theta(x_i)) - (1 - y_i) * \log(1 - h_\theta(x_i))]$$

Ecuación 2.7: Fórmula de función de coste de una regresión logística. Fuente: [Pardo, s.f.](#)

Donde:

$h_{\theta}(x_i)$ = función sigmoide de $\theta^T x$

θ = vector de longitud theta para $j=0,1,2,3...n$

x = matriz de entradas

y = vector de salidas

La primera parte de la ecuación está conformada por el logaritmo de la probabilidad de éxito y la segunda, por la de fracaso.

- **Gradiente descendiente:** Es un método de optimización numérica para estimar los mejores coeficientes, fundamental en Deep Learning para entrenar redes neuronales y en muchos casos, para la regresión logística, siendo mejor que el método de mínimos cuadrados (poner ref). A través de una función E(W), proporciona el error que comete la red en función del conjunto de pesos sinápticos W. El objetivo del aprendizaje será encontrar la configuración de pesos que corresponda al mínimo global de la función de error o coste ([Bertona, 2005](#)).

En general, la función de error es una función no lineal, por lo que el algoritmo realiza una búsqueda a través del espacio de parámetros que, se aproxime de forma iterada a un error mínimo de la red para los parámetros adecuados, como se aprecia en la Figura 2.14 ([Sancho Caparrini, 2017](#)).

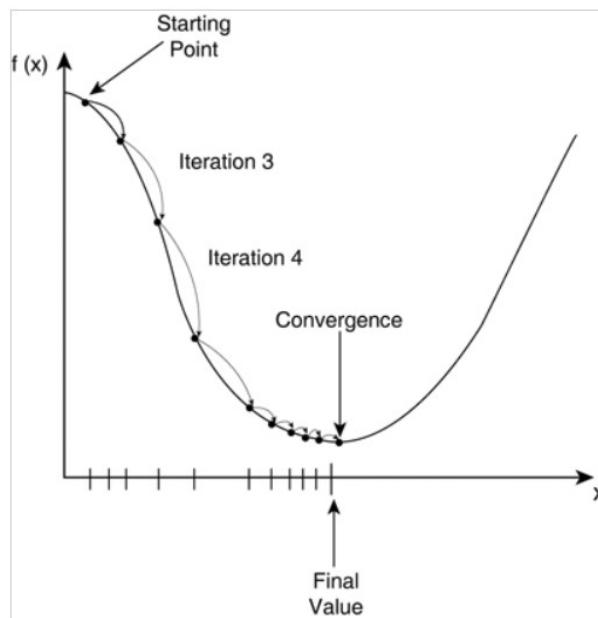


Figura 2.14: Ilustración del algoritmo gradiente descendiente. Fuente: [Sancho Caparrini, 2017](#)

El Descenso del Gradiente, como también se le conoce, es el algoritmo de entrenamiento más simple y también el más extendido y conocido. Solo hace uso del vector gradiente, y por ello se dice que es un método de primer orden

([Sancho Caparrini, 2017](#)). Un gradiente es la generalización de la derivada. Matemática, la derivada de una función mide la rapidez con la que cambia el valor de esta, según varíe el valor de su variable independiente. La gradiente se calcula con derivadas parciales, por lo que al actualizar los coeficientes W para un tiempo t, se usa la Ecuación 2.8 ([IArtificial.net, 2019b](#)):

$$W_{(t+1)} = W_{(t)} - \alpha \left(\frac{\partial MSE}{\partial W} \right)$$

Ecuación 2.8: Actualización de pesos W mediante gradiente descendente. Fuente: [IArtificial.net, 2019b](#)

Donde:

α = ratio de aprendizaje

Este ratio controla el tamaño de la actualización. Si este es demasiado grande, será más difícil encontrar los coeficientes que minimicen la función de coste o error; la actualización de W es proporcional al gradiente; y se usa la resta para ir en dirección opuesta al gradiente como en la Figura 2.15.

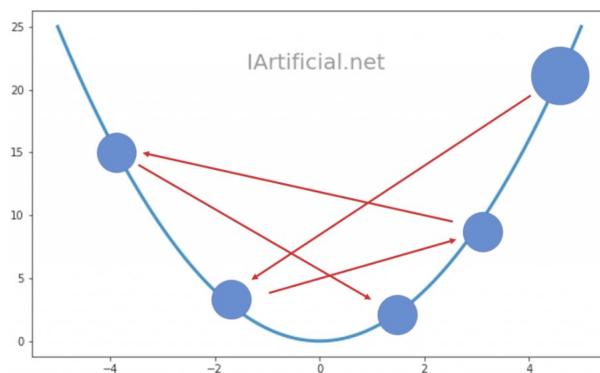


Figura 2.15: Actualización de pesos W con el algoritmo. Fuente: [IArtificial.net, 2019b](#)

- **Propagación hacia atrás:** También conocido en inglés como *Backpropagation*, es un método que consta de dos fases: en la primera se aplica un patrón, el cual se propaga por las distintas capas que componen la red hasta producir la salida de la misma. Luego, esta se compara con la salida deseada y se calcula el error cometido por cada neurona de salida. Estos errores se transmiten hacia atrás, partiendo de la capa de salida, hacia todas las neuronas de las capas intermedias [Fritsch, 1996] ([Bertona, 2005](#)). La actualización iterativa de los pesos que el algoritmo propone es mediante la Ecuación 2.9:

$$W_{ji}(t+1) = W_{ji}(t) + [\alpha \delta_{pj} y_{pj} + \beta \Delta W_{ji}(t)]$$

Ecuación 2.9: Fórmula del algoritmo de propagación hacia atrás. Fuente: [Bertona, 2005](#)

siendo $\delta_{pj} = \begin{cases} (d_{pj} - y_{pj})f'_j(h_j) & \text{si } j \text{ es una neurona de salida} \\ \left(\sum_k \delta_{pk} W_{kj} \right) f'_j(h_j) & \text{si } j \text{ es una neurona oculta} \end{cases}$

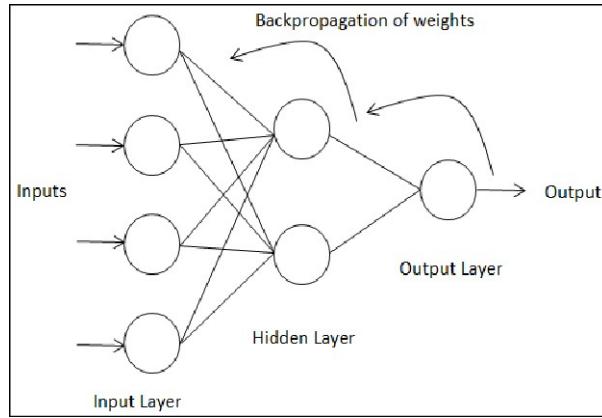


Figura 2.16: Capa oculta simple MLP con propagación hacia atrás. Fuente: [IArtificial.net, 2019b](#)

Para entender mejor la teoría y la fórmula de actualización de pesos, se seguirá el siguiente ejemplo del conjunto de redes de la Figura 2.17.

Se tiene una red neuronal con tres nodos de entrada ($x_1=1$, $x_2=4$ y $x_3=5$) con dos pesos respectivos cada una ($W_1=0.1$ y $W_2=0.2$ para x_1 ; $W_3=0.3$ y $W_4=0.4$ para x_2 ; $W_5=0.5$ y $W_6=0.6$ para x_3), dos capas ocultas (h_1 y h_2) con dos peso cada una ($W_7=0.7$ y $W_8=0.8$ para h_1 ; $W_9=0.9$ y $W_{10}=0.1$ para h_2) y dos nodos de salida (O_1 y O_2). El proceso normal para calcular el valor del nodo final se da, tanto con los nodos de entrada y los de capa oculta, mediante la sumatoria de producto de cada peso con su valor, es decir, mediante la fórmula de las RNA $in_i = \sum_{j=0}^n W_{j,i} * a_j + b_{j,i}$, al mismo tiempo que devuelve un valor del error cometido. Este último se calcula mediante la Ecuación 2.10:

$$E_k = (T_k - O_k) * O_k * (1 - O_k)$$

Ecuación 2.10: Cálculo del error cometido en una red neuronal. Fuente: [Viera Balanta, 2013](#)

Donde:

T_k = salida correcta de cada nodo de salida

O_k = salida actual que cada nodo genera

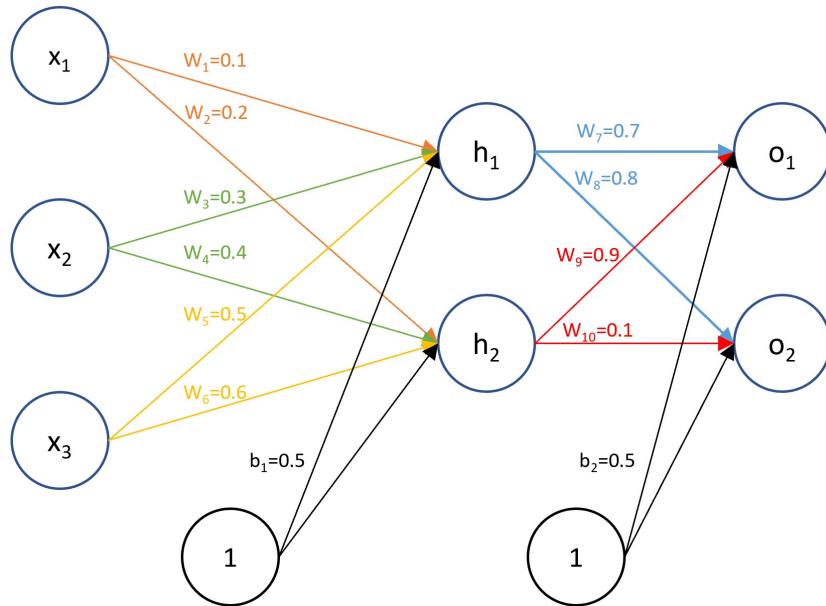


Figura 2.17: Redes neuronales de ejemplo. Fuente: [A Not So Random Walk, 2019](#)

Con estos errores calculados, se retrocede hacia la capa oculta y se procede a calcular los nuevos pesos para sus nodos. La fórmula del cálculo de los mismos es la Ecuación 2.11:

$$W_{jk} = W_{jk} + L * E_k * O_j$$

Ecuación 2.11: Actualización de pesos mediante propagación hacia atrás. Fuente: [Viera Balanta, 2013](#)

Donde:

W_{jk} = peso a actualizar para cada nodo de la capa oculta (W_7, W_8, W_9 y W_{10})

L = porcentaje de aprendizaje

O_j = valor de los nodos que entrarán a las salidas

Estos nuevos pesos permitirán redefinir los errores de ambos nodos, con una pequeña diferencia en su cálculo (Ecuación 2.12):

$$E_j = O_j * (1 - O_j) * \sum E_k * W_{jk}$$

Ecuación 2.12: Cálculo de errores de nodos usando pesos actualizados. Fuente: [Viera Balanta, 2013](#)

El error de cada nodo de la capa oculta se obtiene multiplicando su valor por su complemento por la sumatoria del producto de sus pesos y los errores de los nodos de salida. Por ejemplo, para h_1 sería $E_{h1} = h_1 * (1 - h_1) * (0.7 * O_1 + 0.8 * O_2)$.

Finalmente, se retrocede hacia los nodos de entrada y se repite el mismo proceso para la actualización de sus pesos y errores.

- **Función tangente hiperbólica:** Esta función está relacionada con una sigmoid bipolar. Sin embargo, sus salidas estarán en el rango de -1 y +1. Para redes neuronales, donde la velocidad es más importante que la forma de la función misma, es recomendable usar esta. Se representa como en la Figura 2.18 y su fórmula para calcular su nuevo valor es la Ecuación 2.13:

$$a = \text{Tansig}(n) = \frac{2}{1+e^{-2n}} - 1$$

Ecuación 2.13: Fórmula de la función de activación tangente hiperbólica. Fuente: [Dorofki y col., 2012](#)

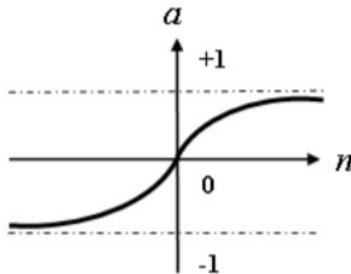


Figura 2.18: Función de activación tangente hiperbólica. Fuente: [Dorofki y col., 2012](#)

- **Función puramente lineal (purelin):** Esta función se caracteriza porque su salida es igual a su entrada debido a su linealidad. Normalmente se usa para obtener los mismos valores de la entrada. Se representa como en la Figura 2.19 y su fórmula para calcular su nuevo valor es la Ecuación 2.14:

$$a = n$$

Ecuación 2.14: Fórmula de la función de activación puramente lineal. Fuente: [Dorofki y col., 2012](#)

- **Función Unidad Lineal Rectificada (ReLU):** Esta función se caracteriza por, además de conservar los valores positivos, convertir los valores negativos de entrada en 0, esto con la finalidad de no considerarlos en la siguiente capa de convolución como en el caso de procesamiento de imágenes ([SitioBigData.com, 2019](#)). Si bien tiene un buen desempeño en redes convolucionales y es muy usada para el procesamiento de imágenes, al no estar acotada pueden morirse demasiadas neuronas ([Calvo, 2018b](#)). Se representa como en la Figura 2.20 y su fórmula para calcular su nuevo valor es la Ecuación 2.15:

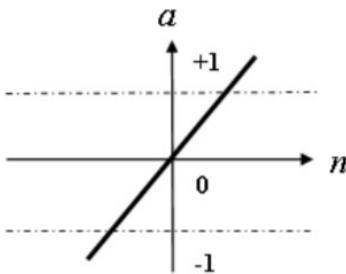


Figura 2.19: Función de activación puramente lineal. Fuente: [Dorofki y col., 2012](#)

$$f(x) = \max(0, x) = \begin{cases} 0 & \text{para } x < 0 \\ x & \text{para } x \geq 0 \end{cases}$$

Ecuación 2.15: Fórmula de la función de activación ReLU. Fuente: [Calvo, 2018b](#)

Además de existir distintas funciones de activación, las redes neuronales artificiales se clasifican según la topología de red, siendo algunas de las más importantes ([Calvo, 2017](#)).

- **Red Neuronal Monocapa – Perceptrón simple:** Es la red neuronal más simple ya que está compuesta solamente de una capa de neuronas que componen varios nodos de entrada para proyectar una capa de neuronas de salida, como se aprecia en la Figura 2.21. Esta última capa se calcula usando la misma Ecuación 4 que implica la suma de productos de cada uno de los pesos de los nodos de entrada con sus instancias, añadiéndole finalmente el sesgo, aquel que controla la predisposición de la neurona a disparar un 1 o 0 independientemente de los pesos, para que el valor resultante se le aplique la función de activación que ayudarán a modelar funciones curvas o no triviales ([Machine Learning for Artists, 2019](#)).
- **Red Neuronal Multicapa – Perceptrón multicapa:** Con arquitectura similar al perceptrón simple, con el añadido de contener capas intermedias entre la capa de neuronas de entrada y la de salida, conocidas como capas ocultas, como en el ejemplo de la Figura 2.22.
- **Redes Neuronales Convolucionales (CNN):** También conocidas por su nombre en inglés *Convolutional Neural Networks*, se diferencia del perceptrón multicapa en que cada neurona no necesita estar unida con todas las que le siguen, sino más bien solo con un subgrupo de estas con el fin de reducir la cantidad de neuronas necesarias para su funcionamiento, como se observa en la Figura 2.23 ([Calvo, 2017](#)).

Hoy en día, las redes neuronales convolucionales tienen múltiples usos desde que la idea fue concebida. Algunos de los problemas en las que pueden ser usados son

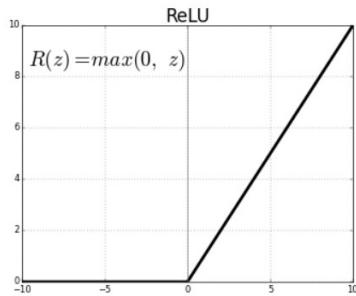


Figura 2.20: Función de activación Unidad Lineal Rectificada. Fuente: [Machine Learning for Artists, 2019](#)

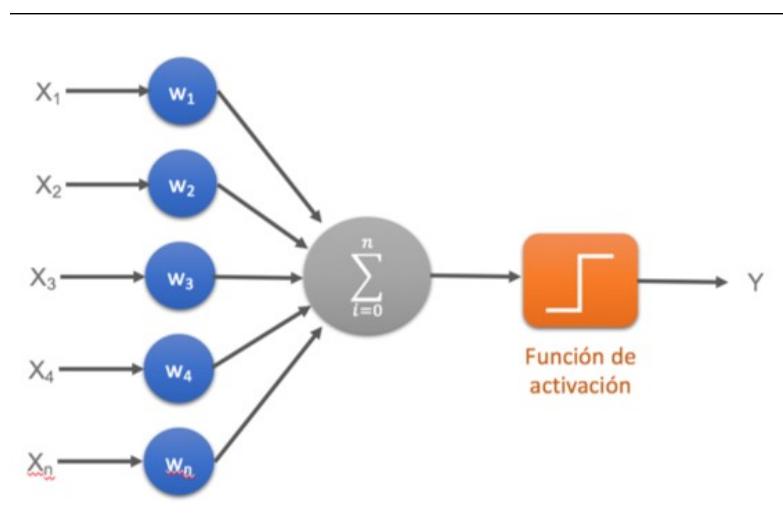


Figura 2.21: Ejemplo de perceptrón simple. Fuente: [Calvo, 2017](#)

de clasificación de objetos, recuperación de imágenes, detección y segmentación de objetos, distorsión y filtros de imágenes, por citar los ejemplos más comunes. El modelo de CNN más conocido es “AlexNet” (2012) por ser uno de los pioneros en clasificar imágenes ([F.-F. Li y col., 2019](#)).

Estas redes tienen su origen en el Neocognitron introducido por Fukushima en 1980 como modelo de red neuronal para el mecanismo de reconocimiento de patrón visual sin la enseñanza de un “profesor” (ver Figura 2.24), mismo que en el año 1998 sería mejorado por LeCun, Bottou, Bengio y Haffner al agregar un método de aprendizaje de gradiente aplicado al reconocimiento de documento basado en la propagación hacia atrás (ver Figura 2.25) ([F.-F. Li y col., 2019](#)).

Estos modelos se inspiraron en el estudio de la información visual en la corteza donde se ubican hasta 5 áreas. La primera, V1, contiene la información visual donde sus neuronas se ocupan de características visuales de bajo nivel, alimentando así a otras áreas adyacentes. Cada una de ellas se encarga de aspectos más específicos

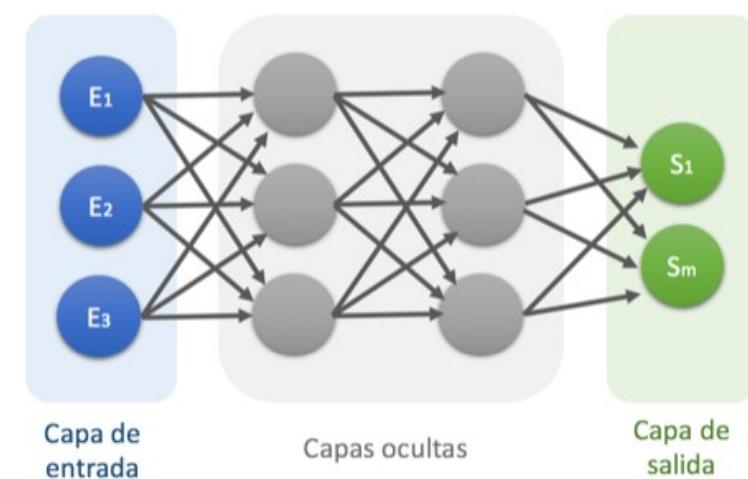


Figura 2.22: Ejemplo de perceptrón multicapa. Fuente: [Calvo, 2017](#)

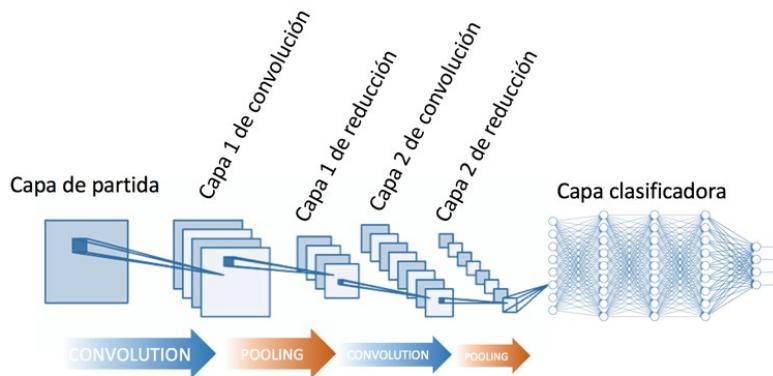


Figura 2.23: Ejemplo de red neuronal convolucional. Fuente: [Calvo, 2017](#)

y detallados de la información obtenida. La idea de su implementación es la de solucionar el problema que surgen al escalar imágenes de mucha definición por las redes neuronales ordinarias. Por ello, este tipo de redes trabajan modelando de forma consecutiva piezas pequeñas de información para luego combinarlas en sus capas más profundas ([López Briega, 2016](#)).

Su nombre deriva del concepto convolución. La convolución es un término en las matemáticas usado como operador matemático que convierte dos funciones f y g en una tercera función en donde la primera se superpone a una versión invertida y trasladada de la segunda, así como para denotar la distribución de la función de probabilidad de la suma de dos variables independientes aleatorias. Esta se da por la Ecuación 2.16 ([Figueroa M., s.f.](#)):

Donde el rango puede variar entre un conjunto finito (como en la fórmula desde 0 hasta un valor t) o uno infinito.

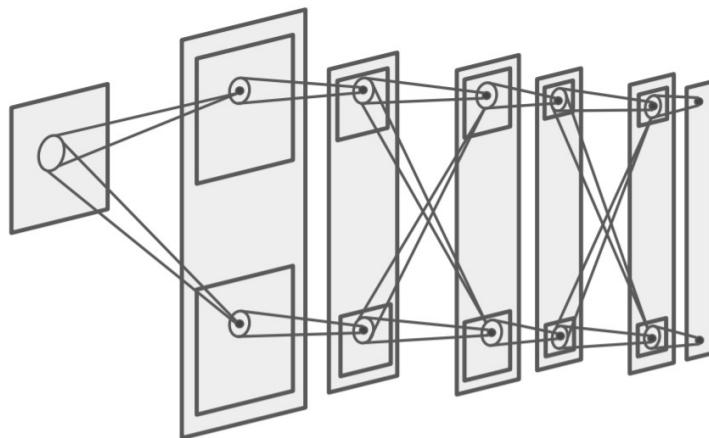


Figura 2.24: Modelo Neocognitron de Fukushima (1980). Fuente: [F.-F. Li y col., 2019](#)

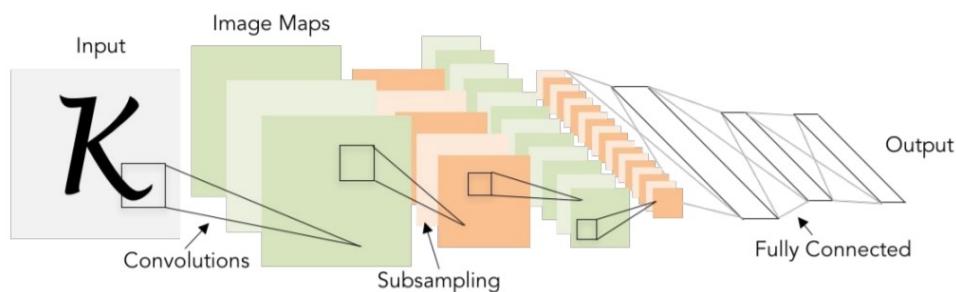


Figura 2.25: Modelo LeNet-5 de LeCun (1998). Fuente: [F.-F. Li y col., 2019](#)

$$(f * g)(t) = \int_0^t f(t - \tau)g(\tau)d\tau$$

Ecuación 2.16: Fórmula matemática de la convolución. Fuente: [Figueroa M., s.f.](#)

La estructura de las Redes Neuronales Convolucionales se constituye en tres tipos de capas ([López Briega, 2016](#)).

- **Capa convolucional (*Convolutional Layer*):** Es la capa que hace distinta a esta red de otros tipos de redes neuronales artificiales. Se aplica la operación de la convolución, que recibe como entrada (*input* en inglés) a la imagen para luego aplicarle un filtro (*kernel* en inglés), devolviendo un mapa de las características de la imagen original, logrando así reducir el tamaño de los parámetros, como se observa en la Figura 2.26.

Por ejemplo, en la anterior figura se tiene una imagen de entrada con dimensiones de 32 de alto, 32 de ancho y 3 de profundidad (32x32x3). A ella se le aplica un filtro de dimensiones (5x5x3) que recorrerá toda la imagen para extraer características de cada pixel. Tanto la profundidad de la entrada como del filtro siempre son iguales. El resultado de tomar un producto escalar entre el

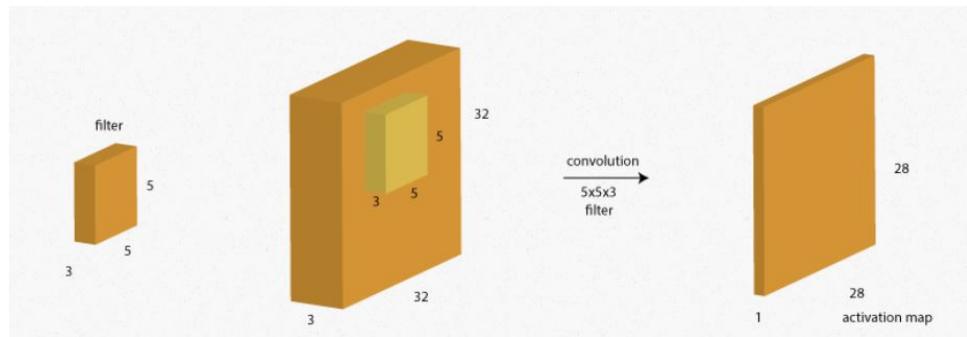


Figura 2.26: Ejecución de la convolución en una entrada. Fuente: [López Briega, 2016](#)

filtro y un pequeño fragmento de $5 \times 5 \times 3$ de la imagen es un número, generando así un mapa de activación de nuevas dimensiones ($28 \times 28 \times 1$). Por cada n filtros aplicados a la entrada se generan n de estos mapas. Al final, la cantidad de mapas de activación determinará una nueva imagen de n de profundidad, como en la Figura 2.27.

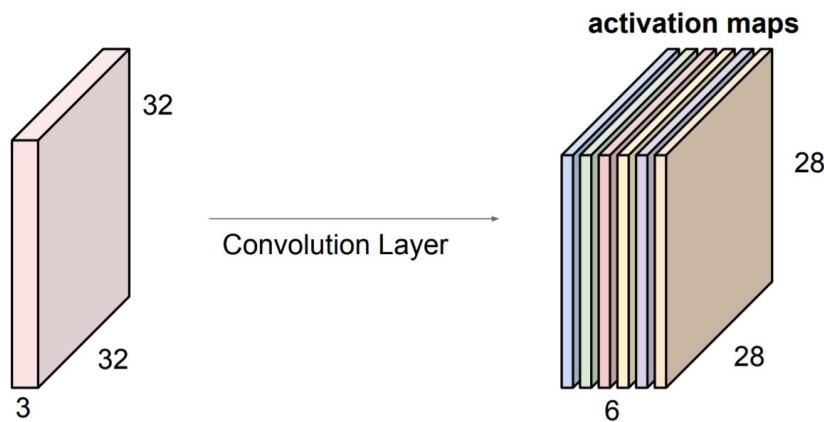


Figura 2.27: Generación de una nueva imagen a partir de filtros. Fuente: [F.-F. Li y col., 2019](#)

Asimismo, cada vez que se aplica una convolución a una imagen, se aplicará una función de activación como en la secuencia de la Figura 2.28.

A nivel visual, en la Figura 2.29 se aprecia un ejemplo de los resultados de aplicar varias convoluciones a una imagen.

Finalmente, se calcula el volumen de la dimensión de la salida de la Figura 2.30 mediante la Ecuación 2.17:

- Se tiene una entrada de dimensiones $(h * w * d)$.
- Se tiene un filtro de dimensiones $(f_h * f_w * d)$.
- **Capa de reducción (Pooling Layer):** Esta capa le sucede a la capa convolucional (luego de aplicar la función de activación). Sirve principalmente para reducir las dimensiones espaciales del volumen de la entrada (alto x ancho)

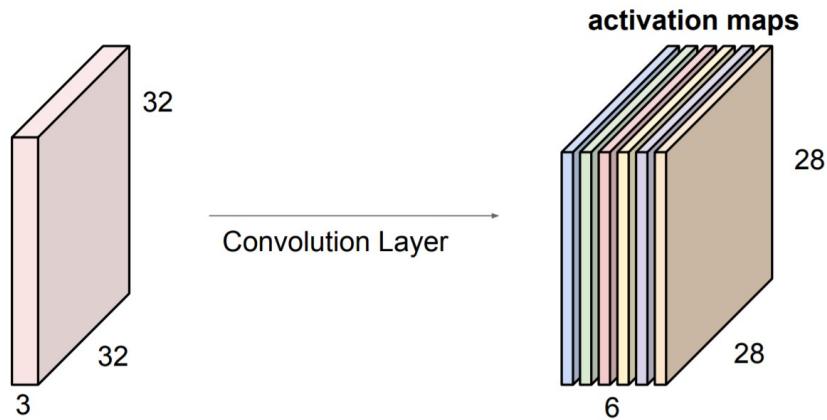


Figura 2.28: Secuencia de varias capas convolucionales. Fuente: [F.-F. Li y col., 2019](#)

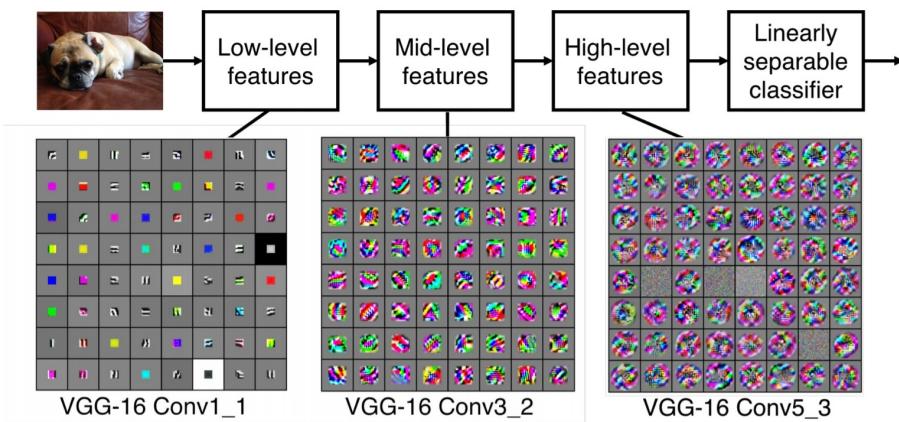


Figura 2.29: Extracción de características a partir de convoluciones. Fuente: [F.-F. Li y col., 2019](#)

$$Volumen = (h - f_h + 1) * (w - f_w + 1) * 1$$

Ecuación 2.17: Cálculo del volumen del mapa de activación. Fuente: [Prabhu, 2018](#)

para la siguiente capa convolucional. Sin embargo, no afecta la profundidad de la misma. Esta operación que realiza se le conoce también como “reducción de muestreo” debido a que, si bien logra reducir las dimensiones para procesar mejor en la siguiente capa, también conlleva perder información. Por el contrario de lo que se piensa, además de reducir la sobrecarga del cálculo para las siguientes capas, el modelo se beneficia también disminuyendo el sobreajuste. Para determinar las dimensiones de la nueva imagen generada (siempre que sea cuadrada, es decir, lados iguales como en la Figura 2.31) con esta capa, se aplica la Ecuación 2.18:

Donde:

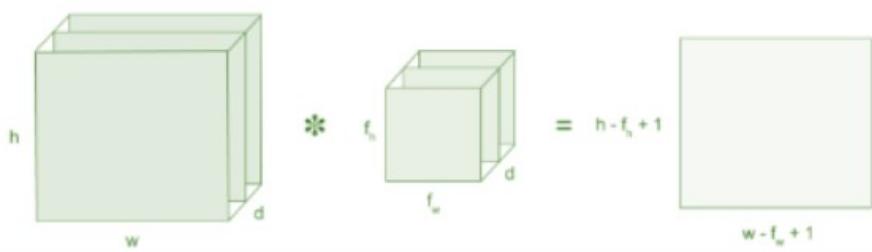


Figura 2.30: Ejemplo de matriz de imagen de entrada y un filtro. Fuente: [Prabhu, 2018](#)

$$\text{Salida} = \frac{N - F}{S} + 1$$

Ecuación 2.18: Cálculo del tamaño de la imagen reducida. Fuente: [F.-F. Li y col., 2019](#)

N = tamaño del lado de la imagen de entrada

F = tamaño del lado del filtro

S = número de desplazamiento de pixeles sobre la matriz de entrada

Por ejemplo, cuando el paso es 1, los filtros se mueven a 1 pixel por vez, cuando el paso es 2, se mueven a 2 píxeles (como en la Figura 2.32) y así sucesivamente (poner ref).

Si, por el contrario, se desea aplicar convolución a una imagen sin afectar sus dimensiones luego de pasar por la capa de reducción, se construye bordes de ceros de n píxeles. A este tamaño de borde se le llama Relleno (*pad* en inglés), por lo que el tamaño de la nueva salida se obtiene mediante la siguiente fórmula:

Existen diferentes tipos de reducción ([Prabhu, 2018](#)):

- ◊ Max Pooling: Toma el elemento más grande dentro del mapa de características.
- ◊ Average Pooling: Toma el promedio de los elementos dentro del mapa de características.
- ◊ Sum Pooling: Toma la suma total de los elementos dentro del mapa de características.
- ◊ **Capa totalmente conectada (Fully Connected Layer):** Al final de las capas de convolución y reducción, se usan redes completamente conectadas a cada pixel considerando que cada uno como una neurona separada al igual que en una red neuronal regular ([López Briega, 2016](#)). En esta capa, se aplana la matriz de todas las características obtenidas anteriormente a un vector y se alinea en una capa completamente conectada a una red neuronal (Figura 2.33).

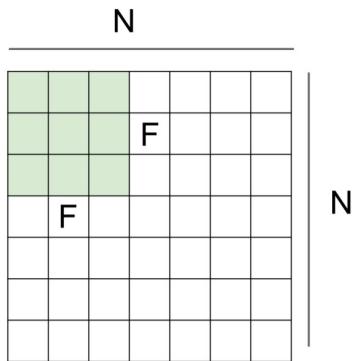


Figura 2.31: Dimensiones de una entrada y un filtro. Fuente: [F.-F. Li y col., 2019](#)



Figura 2.32: Paso de 2 píxeles por parte de un filtro. Fuente: [Prabhu, 2018](#)

Para concluir, se muestra a continuación (Figura 2.34) la representación de la arquitectura completa de una Red Neuronal Convolutacional resumiendo los conceptos anteriores.

- **Redes Neuronales Recurrentes (RNN):** También conocidas por su nombre en inglés *Recurrent Neural Networks*, se caracterizan por no tener una estructura de capas como se aprecia en la Figura 2.35, sino más bien por permitir conexiones entre sus neuronas de manera arbitraria para crear temporalidad y que toda la red obtenga memoria. Todo esto permite generar una red muy potente para el análisis de secuencias, entre algunos ejemplos se mencionan el análisis de textos, sonidos o video ([Calvo, 2018a](#)).
- **Máquina de Vectores de Soporte (SVM):** Es un algoritmo usado para tareas de regresión y clasificación, buscando un hiperplano en un espacio N -dimensional que clasifique claramente los puntos de datos a partir de la distancia máxima entre los puntos de datos de ambas clases. Para ello, maximiza la distancia del margen proporcionando cierto refuerzo para que los puntos de datos futuros puedan clasificarse con más confianza, es decir, que permita distinguir claramente dos clases, como se muestra en la Figura 2.36 ([Gandhi, 2018](#)).

Este algoritmo tiene sus orígenes en la década de los años 60 en Rusia, desarrollados por

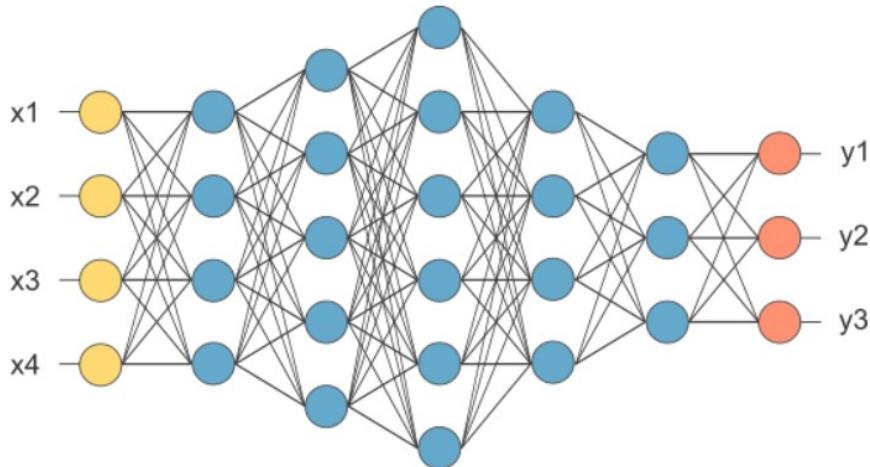


Figura 2.33: Aplanado de matrices luego de agrupar la capa. Fuente: [Prabhu, 2018](#)

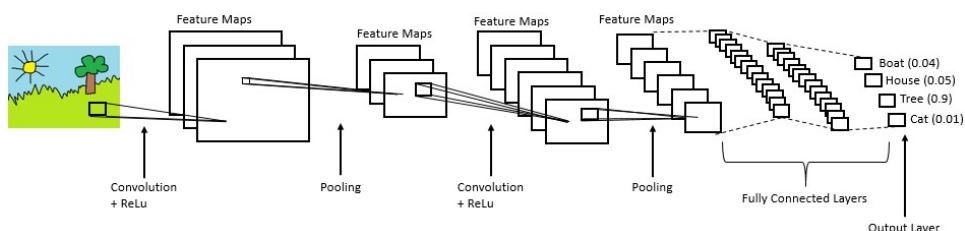


Figura 2.34: Arquitectura completa de una CNN. Fuente: [Prabhu, 2018](#)

Vapnik y Chervonenkis. Inicialmente se enfocó en el reconocimiento óptico de caracteres (OCR). Más tarde, los clasificadores de Vectores de Soporte se volvieron competitivos con los mejores sistemas disponibles en ese momento para resolver no solamente el anterior tipo de problema, sino también abarcar tareas de reconocimiento de objetos. En 1998, se publicó el primer manual de estos algoritmos por Burges. Y debido a sus grandes resultados obtenidos en la industria, actualmente se usa con frecuencia en el campo del aprendizaje automático ([Smola & Schölkopf, 2004](#)).

Los vectores de soporte hacen referencia a un pequeño subconjunto de las observaciones de entrenamiento que se utilizan como soporte para la ubicación óptima de la superficie de decisión ([MathWorks, s.f.](#)).

Una Máquina de Vectores de Soporte aprende la superficie decisión de dos clases distintas de los puntos de entrada. Como un clasificador de una sola clase, la descripción dada por los datos de los vectores de soporte es capaz de formar una frontera de decisión alrededor del dominio de los datos de aprendizaje con muy poco o ningún conocimiento de los datos fuera de esta frontera. Los datos son mapeados por medio de un *kernel* Gaussiano u otro tipo de *kernel* a un espacio de características en un espacio dimensional más alto, donde se busca la separación máxima entre clases. Cuando es traída de regreso al espacio

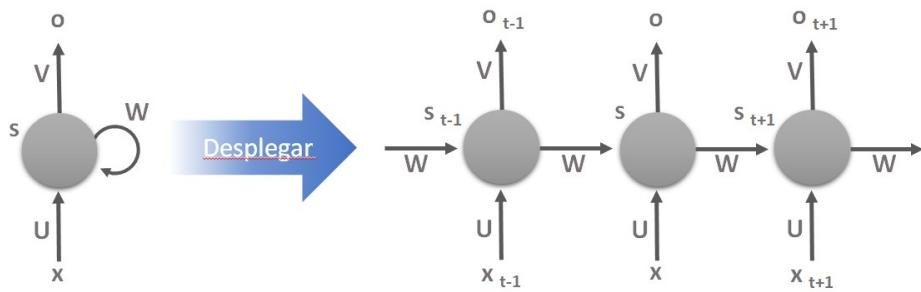


Figura 2.35: Ejemplo de red neuronal recurrente. Fuente: [Calvo, 2018a](#)

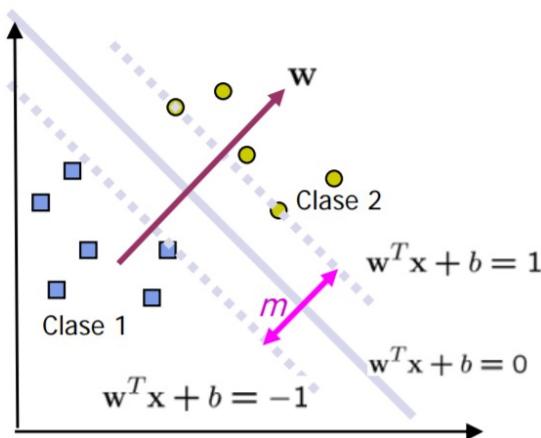


Figura 2.36: Hiperplano con dos clases separadas por una distancia m. Fuente: [Betancourt, 2005](#)

de entrada, la función de frontera puede separar los datos en todas las clases distintas, cada una formando un agrupamiento. Esta teoría se basa en la idea de minimización de riesgo estructural (SRM), demostrando en muchas aplicaciones tener mejor desempeño que otros algoritmos de aprendizaje tradicional como las redes neuronales para resolver problemas de clasificación ([Betancourt, 2005](#)).

Cabe mencionar que hay casos en que el conjunto de datos de dos clases puede ser separables no necesariamente de forma lineal. En las Figura 2.37 y Figura 2.38 se observan casos linealmente y no linealmente separables, respectivamente.

Lo que se debe hacer para el primer caso es crear el hiperplano a través de una función lineal $w * z + b = 0$ y, definido el par (w, b) , separar el punto x_i según la Ecuación 2.19:

$$f(x_i) = \text{sign}(w * z + b) = \begin{cases} 1 & y_i = 1 \\ -1 & y_i = -1 \end{cases}$$

Ecuación 2.19: Ecuación del hiperplano para clasificar dos clases. Fuente: [Betancourt, 2005](#)

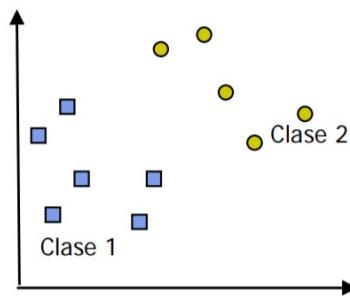


Figura 2.37: Ejemplo de caso linealmente separable. Fuente: [Betancourt, 2005](#)

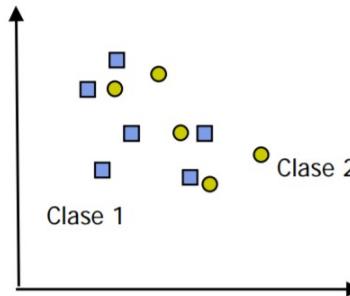


Figura 2.38: Ejemplo de caso no linealmente separable. Fuente: [Betancourt, 2005](#)

Para el segundo caso, debido a su mayor complejidad, se puede introducir algunas variables no-negativas a la función del hiperplano para hallar su valor óptimo; o también es viable utilizar una función *kernel* que calcule el producto punto de los puntos de entrada en el espacio de características Z, como se aprecia en la Figura 2.39.

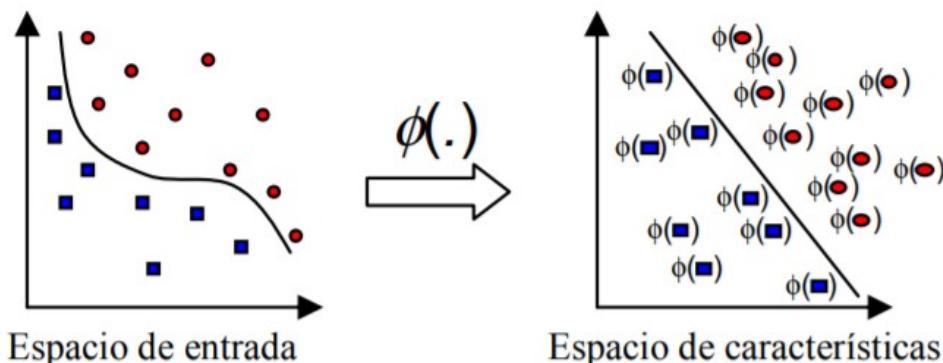


Figura 2.39: Aplicación de un kernel para transformar el espacio de los datos. Fuente: [Betancourt, 2005](#)

- **Árboles de Decisión:** Representación visual de decisiones y toma de decisiones utilizada en la minería de datos para derivar una estrategia y alcanzar un objetivo particular. Se dibuja boca abajo con su raíz en la parte superior. Consta de nodos internos, los cuales se subdividen en ramas o bordes y su contenido, las hojas o decisiones ([Gupta, 2017](#)).

Un árbol de decisión toma como entrada un objeto descrito a través de un conjunto de atributos y devuelve una “decisión”. Estos pueden ser discretos o continuos. La salida puede tomar cualquiera de estos dos tipos de valores; en el caso que aprenda una función tomando valores discretos se le denominará clasificación, y en el caso que la función sea continua será llamada regresión. En las clasificaciones booleanas, es decir de dos valores o binaria, clasificará como verdadero (positivo) o falso (negativo). Para alcanzar una decisión, el árbol desarrolla una serie de pruebas a través de sus nodos y las ramas que salen del nodo son etiquetadas con los valores posibles de dicha propiedad. Además, cada nodo hoja del árbol representa el valor que ha de ser devuelto si es alcanzado ([Russell & Norvig, 2004](#)).

Por ejemplo, representando un ejemplo de este algoritmo, se ilustra en la Figura 2.40 para decidir si se debe esperar por una mesa en un restaurante.

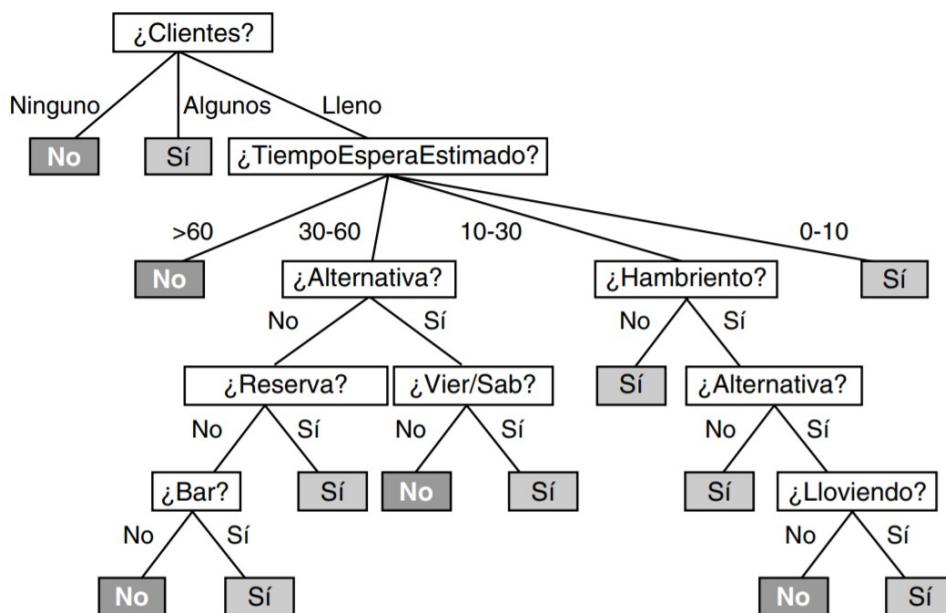


Figura 2.40: Ejemplo del algoritmo de árbol de decisión. Fuente: [Russell y Norvig, 2004](#)

2.2.8. Procesamiento del Lenguaje Natural

El Procesamiento del Lenguaje Natural (*Natural Language Processing (NLP)* por su nombre en inglés) es un campo interdisciplinario que combina lingüística computacional, ciencias de la computación, ciencia cognitiva e inteligencia artificial. Investiga el uso de las computadoras para entender el lenguaje humano con el fin de realizar tareas útiles, así como lograr una interacción entre ambas partes. Algunas de estas tareas son, por ejemplo, reconocimiento de voz, comprensión del lenguaje hablado, sistemas de diálogo, análisis léxico, análisis de

sentimientos, entre otros ([Deng & Liu, 2018](#)).

Los autores Deng y Liu explican en su libro *Deep Learning in Natural Language Processing* el desarrollo del estudio de este campo, separando desde una perspectiva histórica en 3 olas: el Racionalismo, el Empiricismo y el Aprendizaje Profundo.

El Racionalismo, nomenclatura establecida entre las décadas de los años 60s y 80s de acuerdo a los argumentos por Noam Chomsky sobre la concepción del lenguaje humano, se basa en el conocimiento de este último fijado por herencia genética y concebido desde el nacimiento. Las primeras apariciones de la aplicación de este enfoque se remontan a la década de los años 50s, donde Alan Turing, en los experimentos que se conocen como "las pruebas de Turing", intentó simular conversaciones de lenguaje natural entre un humano y un computador para generar respuestas similares a la de una persona con el fin de poder evaluar las habilidades que ellas pueden alcanzar.

2.3. Marco Conceptual

2.3.1. Crowdfunding

El financiamiento colectivo o crowdfunding es un sistema que, utilizando internet como base de operaciones, busca generar una respuesta económica activa en el usuario ([López-Golán y col., 2017](#)).

Combinando ideas de microfinanzas y crowdsourcing, el crowdfunding es la práctica de financiar una empresa o un proyecto al recaudar muchas pequeñas cantidades de dinero de un gran número de personas. Este mecanismo de financiamiento ha sido recibido por los recaudadores de fondos, el público en general y los formuladores de políticas. La industria de crowdfunding ha crecido rápidamente en los últimos años, así como el crecimiento exponencial del número de plataformas de crowdfunding, el número de proyectos expuestos allí y el número de capital total recaudado en esos sitios web ([Xuefeng & Zhao, 2018](#)).

La cantidad de grupos de crowdfunding varía según distintos autores. Según Hollas, se clasifican en 4 modelos: basado en donaciones, recompensas, capital y deuda. Colgren reafirma lo anterior con la variante de crédito en vez de capital y por su lado, Collins, préstamo por deuda. Mientras que I. Lee y Shin solo consideran principalmente al crowdfunding basado en recompensas, donaciones y capital. El crowdfunding basado en capital social se encuentra actualmente limitado en los Estados Unidos debido a que la Regulación D de la Ley de Valores de 1933 prohíbe la participación de muchos potenciales inversionistas y los obliga a tener un

ingreso anual mayor a \$ 200,000 o más de \$ 1 millón en patrimonio neto ([Lichtig, 2015](#)).

Además de los tipos mencionados en el párrafo anterior, Collins también afirma que el crowdfunding se puede dividir en 2 tipos de modelo de negocio ([H.-D. Lee, 2019](#)):

- **El modelo “Todo o Nada”** (*All-Or-Nothing* (AON) en inglés), el cual un proyecto logra ser financiado si es que alcanza o sobrepasa la meta durante el periodo de campaña. A partir de esto, el creador tiene como principal objetivo motivar a los patrocinadores. En caso de no lograrse el cometido, los montos aportados al proyecto serán devueltos, significando un riesgo menor para ellos. Esto, asimismo, permite evaluar a los dueños las variables de la campaña para apostar por un algún cambio o decisión que le permita encaminar al éxito del financiamiento. Hasta febrero del 2019, el ratio promedio de proyectos financiados exitosamente en Kickstarter (considerando todas las categorías de la plataforma) fue de 37 %. Sin embargo, aunque parezca un indicador con baja performance, según la misma compañía, el 82 % de los proyectos que alcanzan el 20 % de su meta son financiados exitosamente.
- **El modelo “Mantener todo”** (*Keep-In-All* (KIA) en inglés), muy similar al anterior, con la única diferencia que el creador del proyecto mantiene todo el dinero financiado después de acabar el plazo de días sin importar si alcanza o no la meta establecida. En contraste con el modelo AON, este presenta un ratio menor de éxito. Asimismo, la plataforma de crowdfunding de este tipo de modelo más popular es Indiegogo, que viene funcionando desde el 2008.

Algunos factores clave, relacionados a la eficiencia del proceso y la retención de clientes durante la campaña, que afectan a un crowdfunding exitoso son ([H.-D. Lee, 2019](#)):

- Seleccionar una plataforma correcta. Esta normalmente depende de los objetivos reales que busca el creador del proyecto durante la campaña. En el anterior punto se mencionaron las dos principales dirigidas a un tipo de modelo de negocio en particular cada una.
- Crear un proyecto atractivo. Las personas normalmente se motivan a contribuir económicamente por algún proyecto que sea de su interés. Para ello, deben ser fáciles de entender en general.
- Ofrecer diferentes opciones de recompensa. Estas pueden ir más allá de lo económico. Influirá mucho el nivel de creatividad que tenga el creador para llamar la atención del público.

- Promocionar un video. Además del contenido que pueda tener un proyecto, ayuda mucho el apoyo de material audiovisual que complemente o explique brevemente la descripción de este. Brinda una mejor impresión y además, para las generaciones más jóvenes, es más útil que leer un enunciado extenso.

2.3.2. Kickstarter

Kickstarter, desde su inicio en 2009, es una plataforma de financiamiento de proyectos creativos de todo tipo, los cuales incluyen películas, juegos, música, arte, diseño y tecnología. Actualmente, se han registrado más de 162 mil proyectos realizados, 16 millones de contribuyentes y 4,3 miles de millones de dólares fondeados ([Kickstarter, s.f.-a](#)). La plataforma utiliza un modelo de financiamiento llamado “Todo o Nada” (AON), el cual consiste en que, si un proyecto no alcanza su meta de financiamiento en un determinado plazo de tiempo, no se realiza ninguna transacción de fondos, como se explicó en el anterior punto ([Kickstarter, s.f.-d](#)). Si bien los patrocinadores apoyan estos proyectos por motivos personales y distintos para hacerlos realidad, ellos no obtienen la propiedad o los ingresos de los proyectos que financian, sino que los creadores conservan la totalidad de su trabajo ([Kickstarter, s.f.-f](#)).

2.3.3. Proyecto

Un proyecto es un esfuerzo temporal que se lleva a cabo para crear un producto, servicio o resultado único. La naturaleza temporal de los proyectos indica un principio y un final definidos, y que el propósito se alcanza cuando se logran los objetivos del proyecto o cuando este es terminado por no cumplir sus objetivos, o cuando ya no existe la necesidad inicial que dio origen al mismo ([Project Management Institute, 2017](#)).

A partir de este concepto enunciado por el PMI, se entiende que un proyecto parte de una idea que un individuo o conjunto de individuos tienen en mente para convertirla en realidad con el fin de responder a una necesidad.

En Kickstarter, los proyectos que pueden ser financiados deben pertenecer a las siguientes categorías: Arte, Cómics, Artesanías, Baile, Diseño, Moda, Cine y vídeo, Comida, Juegos, Periodismo, Música, Fotografía, Publicaciones, Tecnología, y Teatro. Cualquier tipo de persona puede contribuir al mismo desde ser patrocinadores hasta formar parte de las principales referencias ([Kickstarter, s.f.-c](#)).

Dentro de las normas internas de la plataforma para la creación de proyectos se especifica que cada proyecto debe resultar ser totalmente nuevo, original e innovador que pueda

ser compartido con el público, así como haber sido presentados de forma honesta y clara. En adición a esto, también se menciona que las recaudaciones que se darán no podrán ser otorgadas a obras benéficas sino cumplir con el objetivo de llevar a cabo el proyecto planteado, al igual que está prohibido asimismo del ofrecimiento de incentivos financieros como resultado ([Kickstarter, s.f.-e](#)).

2.3.4. Campaña

Una campaña puede abarcar diversos términos si es que se busca su concepto en fuentes confiables como la Real Academia Española debido al alcance y empleabilidad del mismo. El significado más próximo que la RAE enuncia sobre campaña es la de un “conjunto de actos o esfuerzos de índole diversa que se aplican a conseguir un fin determinado” ([Real Academia Española, 2020](#)). Para el actual contexto, el término se refiere específicamente a la campaña publicitaria, la cual se define como una estrategia diseñada y ejecutada en diferentes medios para obtener objetivos de notoriedad, ventas y comunicación de una determinada marca o producto a través del uso de la publicidad ([Cyberclic, s.f.](#)).

En la plataforma de Kickstarter, existen personas que tienen ideas y proyectos en mente, buscando financiarlos en el sitio web. Para lograr su objetivo y siguiendo la regla del “todo o nada”, estos individuos realizan eventos de promoción para atraer entre personas conocidas suyas y cibernautas en general. A esta serie de eventos de promoción se le conoce como campañas. Las campañas exitosas se basan en que un determinado proyecto alcanza a ser financiado en el tiempo estimado gracias a algunas de las siguientes características ([Kickstarter, s.f.-b](#)):

- Logran ser claros y concisos sobre lo que su proyecto busca alcanzar al ser financiado. Se definen bien las características del proyecto mediante detalles en la descripción, videos e imágenes precisas.
- Indican los beneficios y recompensas que los patrocinadores obtendrán si es que la campaña es exitosa y el proyecto se financia.
- Atrai e interactúa con una gran cantidad de público, manteniendo constante comunicación acerca de actualizaciones y novedades de la campaña y resolviendo dudas que puedan darse.
- Se estiman costos de manera eficiente que pueden ser solventados para cubrir más allá del proyecto una vez financiado, incluido los que se originan para la entrega del producto a los patrocinadores. Se logra convencerlos.

Adicional a estos puntos, se descubrió en una investigación con más de 27 mil proyectos en Kickstarter que, a pesar que el número de proyectos iniciados por varones es dos veces más que de mujeres, el ratio de éxito de financiamiento es mayor en ellas. Esto debido a que las creadoras, en general, establecen objetivos más realísticos y con un tiempo límite más bajo para cumplir sus objetivos, transmitiendo así calidad y confianza para que los inversores actúen más rápido ([Ullah & Zhou, 2020](#)).

Otro factor clave es el rol importante de los patrocinadores en el éxito de un proyecto financiado. Sus principales motivaciones vienen dadas a factores como las recompensas por el aporte económico al proyecto, el nivel de innovación que este ofrece, la participación social que se logre captar, entre otros. Podrían clasificarse estas motivaciones en 6 grupos: interés, diversión, filantropía, recompensa, relación y reconocimiento. El resultado, 4 tipos de patrocinadores: los angelicales (donantes tradicionales), cazadores de recompensas (inversores del mercado), ávidos fanáticos (apasionados como los miembros de una comunidad de marca) y ermitaños de buen gusto (fanáticos discretos pero entusiastas) ([Tung & Liu, 2018](#)).

Capítulo 3

METODOLOGÍA DE LA INVESTIGACIÓN

3.1. Diseño de la investigación

En esta sección del documento se explicará cual es el diseño, el tipo y el enfoque del trabajo de investigación, así como también la población y la muestra.

3.1.1. Enfoque de la investigación

El presente trabajo tendrá un enfoque cuantitativo ya que se busca diseñar y desarrollar instrumentos, en este caso modelos predictivos, para responder al problema estudiado a partir de medición de datos históricos en la plataforma Kickstarter con herramientas basadas en la estadística y matemáticas que puedan ser interpretadas por cualquier investigador.

3.1.2. Alcance de la investigación

El alcance del presente trabajo será descriptivo ya que se recolectarán datos en un determinado rango de tiempo (desde 2009 hasta el presente año 2019) para describir el comportamiento de las campañas de proyectos tecnológicos en Kickstarter a partir de las características de sus variables y con ello, pronosticar su posible éxito o fracaso antes de finalizar la campaña con un nivel óptimo de precisión.

3.1.3. Tipo de la investigación

Para determinar el tipo de la investigación, primero es necesario definir el actual trabajo como Diseño Experimental ya que las variables que se tienen serán controladas, es decir, serán agregadas o quitadas en el o los modelos construidos en el experimento para analizar el impacto que este o estos tendrán en los resultados obtenidos. Dentro de esta categoría se clasifica como Diseño Experimental Puro ya que se busca medir la variable dependiente, en este caso Status (el estado actual del proyecto en Kickstarter) a partir de la manipulación de las demás variables independientes agregando o desagregándolas para comparar los rendimientos obtenidos de los instrumentos de medición y determinar cuáles de ellas finalmente serán tomadas en cuenta.

3.1.4. Descripción del prototipo de investigación

Teniendo como referencia y base principal el décimo antecedente explicado en el Capítulo II, la idea del prototipo final consistió en ensamblar las tres partes básicas de un proyecto: la primera consiste en el tratamiento de la metainformación (en la cual se realizarán, asimismo, tres experimentos independientes), el segundo, en el contenido visual y el último, el contenido textual respectivamente pero con el valor diferenciado de adaptar el modelo general de acuerdo a las variables y conjuntos de datos disponibles para el presente trabajo. Para ello, se representa cada una de las tres partes agrupadas en el marco de trabajo de la Figura 3.1.

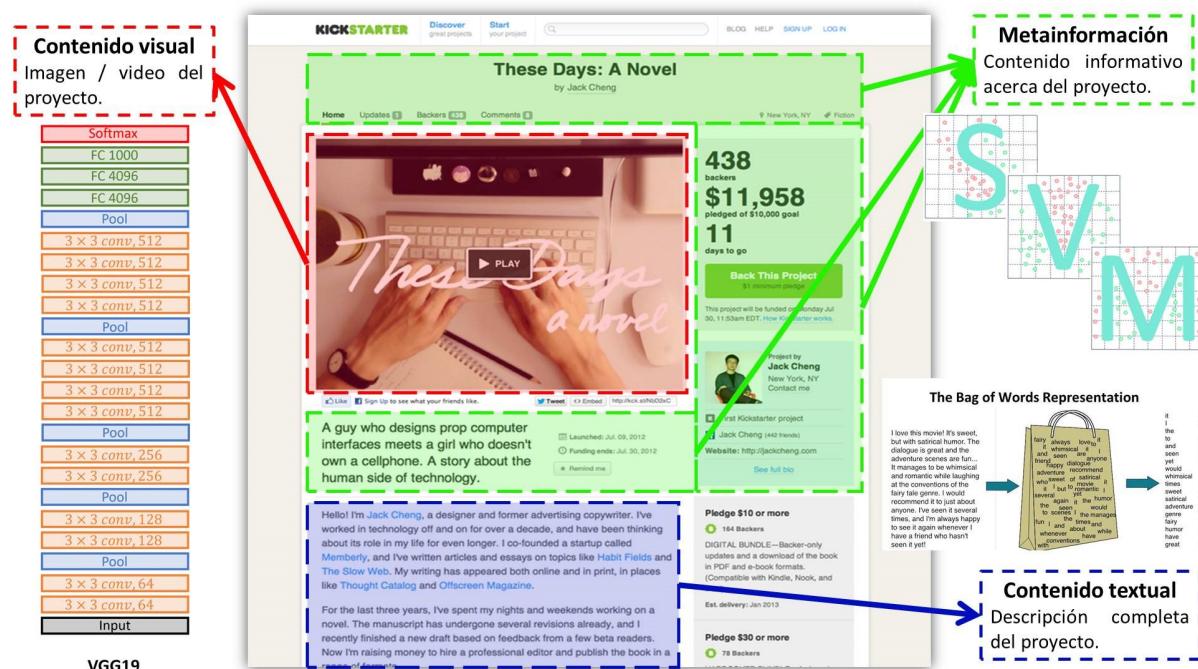


Figura 3.1: Marco de trabajo del prototipo final. Fuente: Elaboración propia

3.2. Población y muestra

3.2.1. Población

La población que será considerada para el presente trabajo será de 27,251 proyectos en Kickstarter de la categoría tecnología de todas las subcategorías entre los períodos 2009-2019, en su mayoría del territorio de los Estados Unidos de América.

3.2.2. Muestra

Debido a que se 214 imágenes del contenido visual no pudieron re-dimensionarse, así como 2 proyectos no contaban con descripciones en el contenido textual, se procedió a remover los 216 proyectos incompletos tanto en la metainformación como en las otras bases de datos, resultando finalmente en 27,035 registros en cada una de los tres conjuntos de datos. Sin embargo, la división en subconjuntos fue distinta en los tres casos y se dio de la siguiente manera:

- Para la metainformación y el contenido textual, el conjunto de datos total de cada uno fue dividido en un subconjunto de entrenamiento (80 %) y uno de prueba (20 %) siguiendo las proporciones dadas en el octavo antecedente.
- Para el contenido visual, el conjunto de datos total fue dividido en tres subconjuntos: entrenamiento (80 %), validación (10 %) y prueba (10 %) siguiendo las proporciones dadas en el décimo antecedente.

3.2.3. Unidad de análisis

La unidad de análisis para el presente trabajo será un proyecto en Kickstarter de la categoría tecnología de cualquier subcategoría entre los períodos 2009-2019 dentro del territorio de los Estados Unidos de América.

3.3. Operacionalización de Variables

En la Tabla 5 se presentan las variables a usar para el conjunto de datos final basado en contenido textual y metainformación. Estas fueron seleccionadas de acuerdo al Benchmarking aplicado a los 10 antecedentes en el Capítulo II.

Los autores citados son los siguientes:

Variable	Detalle	Tipo de dato
Variables independientes		
backers_count	Número de patrocinadores de la campaña del proyecto.	int64
goal	Monto de la meta de financiamiento del proyecto.	float64
duration	Duración de la campaña (en días).	int64
description	Descripción del proyecto.	object
comments	Comentarios de patrocinadores sobre el proyecto.	object
Variable dependiente		
state	Estado de financiamiento del proyecto.	object

Tabla 3.1: Diccionario de datos del conjunto final a entrenar. Fuente: Elaboración propia.

3.4. Instrumentos de medida

3.5. Técnicas de recolección de datos

Los conjuntos de datos recolectados para la investigación son un mix de observaciones cuantitativas (variables numéricas medibles como la meta de financiamiento, montos prometidos y duración de la campaña) y cualitativas (propaganda, descripción y comentarios del proyecto). La base de datos de la metainformación, que comprende las variables cuantitativas y propaganda del proyecto, fue consolidada luego de descargar un histórico público de 10

años de la página web “Web Robots”, fundada por los ex corporativos de TI Tomás Vitulskis y Paulius Jonaitis, y posteriormente pre-procesarla. Mientras que por el lado de la descripción y comentarios de cada proyecto, se usaron técnicas y herramientas de *web scraping* a partir de los URLs. Para encontrar algunos de los papers con la información requerida más cercana, se utilizaron keywords o palabras clave como *crowdfunding*, *Machine Learning*, *Deep Learning*, *prediction*, *Kickstarter*, *accuracy* y *projects*.

3.6. Técnicas para el procesamiento y análisis de la información

3.7. Cronograma de actividades y presupuesto

Se elaboró un cronograma de actividades de toda la investigación, mostrada en la Figura 3.2, contemplando desde el inicio de la misma, desarrollo, evaluación de resultados y sustentación en el mes de diciembre.



Figura 3.2: Cronograma de actividades de la investigación. Fuente: Elaboración propia

Capítulo 4

DESARROLLO DEL EXPERIMENTO

4.1. Construcción de los conjuntos finales de datos

El conjunto total de datos utilizado para la investigación consistió en la recolección de 27,251 proyectos tecnológicos de Kickstarter comprendidos entre los años 2009 y 2019, en 3 partes: metainformación, descripción y comentarios (excluyendo los del creador) del proyectos.

4.1.1. Metainformación

Como se mencionó en las Técnicas de recolección de datos del Capítulo III, el punto de partida para la construcción de los conjuntos de datos fue consolidar las bases públicas, archivos de valores separados por comas (.csv) comprimidos, de la página Web Robots (<https://webrbots.io/kickstarter-datasets/>), fraccionadas por fechas mensuales de captura de información comenzadas desde noviembre del 2015 hasta agosto del 2019 (Figura 4.1), y posteriormente pre-procesada con la ayuda del software Alteryx Designer. De acuerdo con la información de los creadores del sitio Web Robots, se ejecutan robots en dos servidores en la nube encargados de recolectar en un determinado punto del día y una vez al mes información de las campañas que aparecen en Kickstarter ([Web Robots, 2019](#)).

Cada archivo descargado por fecha de captura mensual es descomprimido, y las particiones en formato .csv que contienen son juntadas mediante el siguiente proceso:

- Descargar librerías correspondientes de Python (pandas, numpy, glob, math).
- Dirigirse a la ruta de destino.
- Crear carpetas por mes para almacenar archivos particionados.

Kickstarter Datasets

We have a scraper robot which crawls all [Kickstarter](#) projects and collects data in CSV and JSON formats. From March 2016 we run this data crawl once a month. Datasets are available from the following scrape dates:

2019

- 2019-08-15 [[JSON](#)] – [[CSV](#)]
- 2019-07-18 [[JSON](#)] – [[CSV](#)]
- 2019-06-13 [[JSON](#)] – [[CSV](#)]
- 2019-05-16 [[JSON](#)] – [[CSV](#)]
- 2019-04-18 [[JSON](#)] – [[CSV](#)]
- 2019-03-14 [[JSON](#)] – [[CSV](#)]
- 2019-02-14 [[JSON](#)] – [[CSV](#)]
- 2019-01-17 [[JSON](#)] – [[CSV](#)]

2018

- 2018-12-13 [[JSON](#)] – [[CSV](#)]
- 2018-11-15 [[JSON](#)] – [[CSV](#)]
- 2018-10-18 [[JSON](#)] – [[CSV](#)]
- 2018-09-13 [[JSON](#)] – [[CSV](#)]
- 2018-08-16 [[JSON](#)] – [[CSV](#)]
- 2018-07-12 [[JSON](#)] – [[CSV](#)]
- 2018-06-14 [[JSON](#)] – [[CSV](#)]
- 2018-05-17 [[JSON](#)] – [[CSV](#)]
- 2018-04-12 [[JSON](#)] – [[CSV](#)]
- 2018-03-15 [[JSON](#)] – [[CSV](#)]
- 2018-02-15 [[JSON](#)] – [[CSV](#)]
- 2018-01-12 [[JSON](#)] – [[CSV](#)]

Figura 4.1: Vista (agosto 2019) del website Web Robots. Fuente: [Web Robots, 2019](#)

- Ejecutar algoritmo para unir archivos particionados csv dentro de una misma carpeta.

Cuando el algoritmo finaliza su ejecución, se crea un nuevo archivo separado por comas en cada carpeta por mes, cuyo tamaño oscila entre 1 y 5 gigabytes (GB) en total por cada uno. Con el fin de ahorrar espacio en memoria dentro de la computadora, los archivos particionados son eliminados y solamente se almacenan los nuevos generados.

Por ejemplo, en la Figura 4.2 se detalla el tamaño del conjunto de datos total al corte del periodo de captura de información Julio 2019, aproximadamente más de 212 mil proyectos de todas las categorías y 37 columnas de variables.

```
In [7]: data_combinada_201907.shape ##Originalmente había 212,378 registros de todos los proyectos en Kickstarter
Out[7]: (212378, 37)

In [8]: data_combinada_201907.columns
Out[8]: Index(['backers_count', 'blurb', 'category', 'converted_pledged_amount',
       'country', 'created_at', 'creator', 'currency', 'currency_symbol',
       'currency_trailing_code', 'current_currency', 'deadline',
       'disable_communication', 'friends', 'fx_rate', 'goal', 'id',
       'is_backing', 'is_starrable', 'is_starred', 'launched_at', 'location',
       'name', 'permissions', 'photo', 'pledged', 'profile', 'slug',
       'source_url', 'spotlight', 'staff_pick', 'state', 'state_changed_at',
       'static_usd_rate', 'urls', 'usd_pledged', 'usd_type'],
      dtype='object')
```

Figura 4.2: Tamaño de conjunto de datos al corte de Julio 2019. Fuente: Elaboración propia.

A cada conjunto de datos generado se filtran los que pertenecen a la categoría **Technology**. Debido a que no se cuenta con la información de la columna *main_category*, este proceso de filtración se logra utilizando la variable *source_url* al seleccionar aquellos registros que con-

tengan la cadena de caracteres “<https://www.kickstarter.com/discover/categories/technology>”. En la Figura 4.3 se aprecia cómo queda el conjunto filtrado por categoría Techonlogy al corte de Julio 2019.

```
In [40]: df_201907_filtrada.shape  
Out[40]: (21398, 37)
```

Figura 4.3: Tamaño de conjunto de datos filtrado del corte de Julio 2019. Fuente: Elaboración propia.

Cuando se repitió este procedimiento con cada conjunto de datos, se notó que la proporción de proyectos tecnológicos en Kickstarter representa el 10 % del total de proyectos. Esto se calculó al comparar los tamaños (cantidad de registros por fila) del conjunto de datos inicial y del filtrado.

La siguiente fase del pre-procesamiento fue la de seleccionar las variables de los nuevos conjuntos de datos filtrados que se usarán para generar el dataset final. Para ello, se usó el software Alteryx Designer, el cual permite desarrollar flujos de trabajo para preparar, unir y analizar volúmenes de datos complejos de distintas fuentes.

Con los conjuntos de datos filtrados, se creó el flujo de trabajo representado en la Figura 4.4.

El flujo comienza con la carga de los datos de entrada, los cuales son los 45 archivos separados por coma por cada captura mensual desde noviembre del 2015 hasta agosto del 2019. Luego, se realizaron dos uniones en vez de una sola, esto debido a que, a partir de marzo del 2018, algunas de las variables y valores de estas son diferentes a la de sus predecesoras. Sin embargo, esto no afectará al proceso más adelante ya que ambas uniones fueron juntadas por las mismas variables.

En cada una de las dos uniones, se seleccionaron las variables de la Tabla N° 3, se realizó limpieza de datos para las variables *category*, *location*, *photo* y *urls*, y se transformaron las variables numéricas en milisegundos *created_at*, *launched_at* y *deadline* a variables de fecha. Esto último permitió crear la variable *duration* para determinar la duración de la campaña (en días) de un proyecto calculando la diferencia entre la fecha de culminación (*deadline*) y la fecha de lanzamiento (*launched_at*). Luego se excluyeron los proyectos en proceso de la variable *state* para conservar los culminados, es decir, aquellos cuyo valor sea “successful” o “failed” ya que se analizarán solamente los proyectos que han sido exitosos o fracasados. Por último, la variable *urls* contiene los enlaces directos de los proyectos.

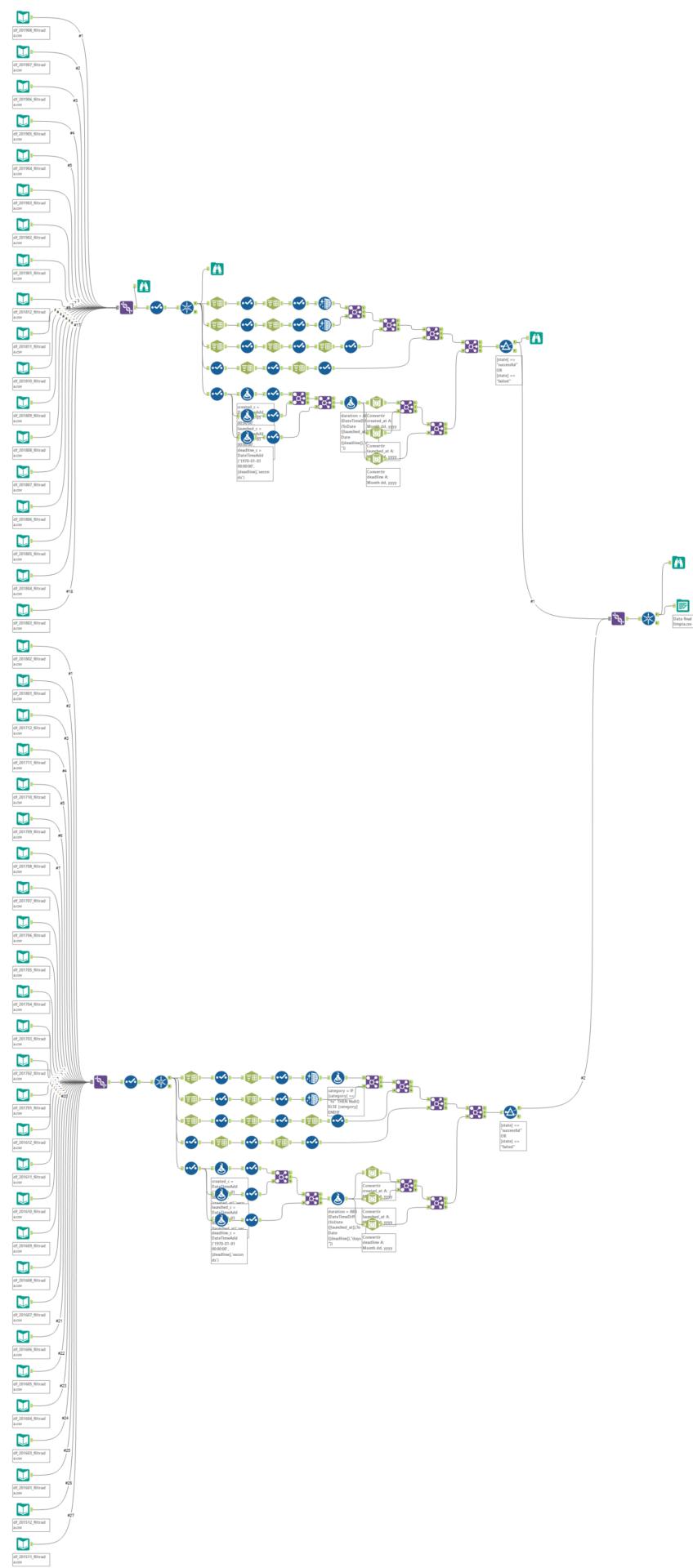


Figura 4.4: Flujo de trabajo de la data final en Alteryx Designer. Fuente: Elaboración propia.

El fluograma culmina con la generación de un archivo de valores separados por coma (.csv) guardado en memoria local. Posteriormente, el archivo generado fue subido a la plataforma Kaggle de manera pública (Figura 4.5) para que pueda ser descargada.

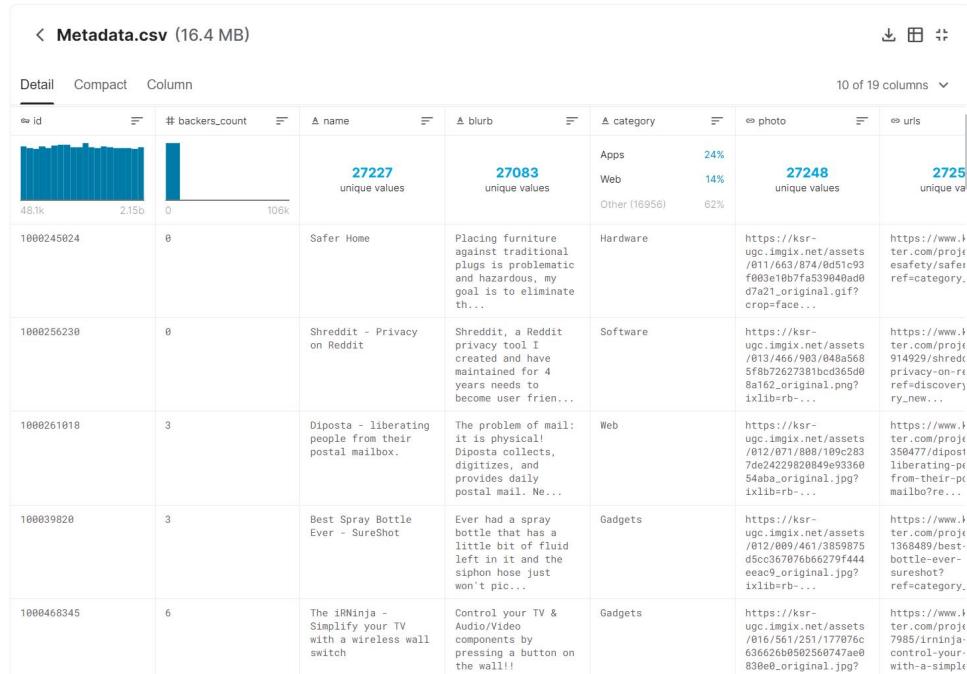


Figura 4.5: Visualización del archivo de metainformación subido a Kaggle. Fuente: Elaboración propia.

La Tabla 4.1 detalla cada variable del conjunto final, adicionando la variable *pledge_amounts* obtenida a partir de web scraping.

4.1.2. Descripción

La descripción (variable *description*) contiene todas las características del producto o servicio del proyecto ofrecido al público, así como otros datos importantes de la campaña. Al proveer gran cantidad de información, en los antecedentes de la investigación se confirma la relación que presenta esta observación con la performance de la campaña y, por ende, el resultado final del financiamiento del proyecto.

La variable se obtuvo utilizando web scraping en cada proyecto gracias a la variable *urls*. Para ello, y como se describe en la Figura 4.6, se elaboró un algoritmo usando la librería BeautifulSoup que, mediante la navegación al contenido de estas páginas a través de un agente falso, se dirigió a las descripciones de los proyectos identificando las etiquetas con clase llamada “**rte_content js-full-description responsive-media**” y las almacenó en un vector vacío,

Variable	Detalle	Tipo de dato
id	Identificador del proyecto.	number
backers_count	Número de patrocinadores de la campaña del proyecto.	number
name	Nombre del proyecto.	string
blurb	Propaganda del proyecto.	string
category	Categoría (dentro de categoría principal) del proyecto.	string
photo	Dirección de enlace de la foto del proyecto.	string
urls	Dirección de la página de la campaña del proyecto.	string
city	Ciudad del creador del proyecto.	string
country	Código de país del creador del proyecto.	string
goal	Monto de la meta de financiamiento del proyecto.	float
pledge_amounts	Montos disponibles para patrocinar la campaña.	string
pledged	Monto final patrocinado de la campaña.	float
currency	Divisa del monto final patrocinado.	string
usd_pledged	Monto final patrocinado de la campaña (en USD).	float
created_at	Fecha de creación de la campaña.	date
launched_at	Fecha de lanzamiento de la campaña.	date
deadline	Fecha de culminación de la campaña.	date
duration	Duración de la campaña (en días).	number
state	Estado de financiamiento del proyecto.	string

Tabla 4.1: Diccionario de datos del conjunto final de metainformación. Fuente: Elaboración propia.

uniendo previamente los párrafos y eliminando caracteres especiales, para posteriormente asignarle la id correspondiente y guardarlo en un archivo de extensión .csv. En caso el algoritmo no encuentre esta clase dentro de las páginas (IndexError), el vector almacenaba un registro nulo.

Debido a la gran cantidad de memoria y tiempo que iba a presentar este proceso, se determinó fraccionar los 27,251 proyectos en tres partes y repetir el mismo en cada uno de ellos. El tiempo aproximado de descarga de cada fracción fue de 6 horas.

Finalmente, las tres partes fueron unidas, se reemplazaron los valores nulos por espacios en blanco y se guardó como un nuevo archivo de valores separados por coma (.csv) en código Unicode UTF-8 para la lectura de caracteres no alfabéticos.

El archivo generado fue subido a la plataforma Kaggle de manera pública (Figura 4.7) para que pueda ser descargada a través del API de la web.

```

def getPageText(url):
    # Crear agente falso para scrapear
    ua = UserAgent()
    user_agent = ua.chrome
    # Solicitar uso de librería urllib con agente falso
    request = urllib.request.Request(url, headers = {"User-Agent":user_agent})
    # Obtener contenido de urls mediante librería urllib
    data = urllib.request.urlopen(request)
    # parse as html structured document
    bs = BeautifulSoup(data, "html.parser")
    # Buscar todos los tags div con una determinada clase
    # A partir de acá, se usa un "try" y un "except" porque hay páginas que no muestran su contenido
    # Para evitar que salga el mensaje de error, se llenarán como null los contenidos que no se puedan descargar
    # El try contiene el algoritmo para descargar la descripción de un proyecto
    try:
        description = bs.find_all("div", {"class":"rte__content js-full-description responsive-media"})
        # Encontrar todos los párrafos
        description = description[0].find_all("p")
        # Crear array vacío donde se almacenará el contenido descargado
        project_description = []
        # Iteración para agregando en lista cada descripción descargada
        for link in description:
            project_description.append(link.text)
        # Junta párrafos separados en un solo vector, separándolos con un espacio
        project_description = ' '.join(project_description)
        # Eliminar caracteres especiales
        project_description = project_description.replace(u'\xa0', u' ')
        project_description = project_description.replace(u'\n', u' ')
        # Y el except sirve para llenar valores nulos en el array en caso de error
    except (IndexError, ValueError):
        project_description = 'null'

    # Eliminar saltos de línea
    return Newlines.sub("\n", project_description)

```

Figura 4.6: Función del algoritmo web scraping de la descripción de proyectos. Fuente: Elaboración propia.

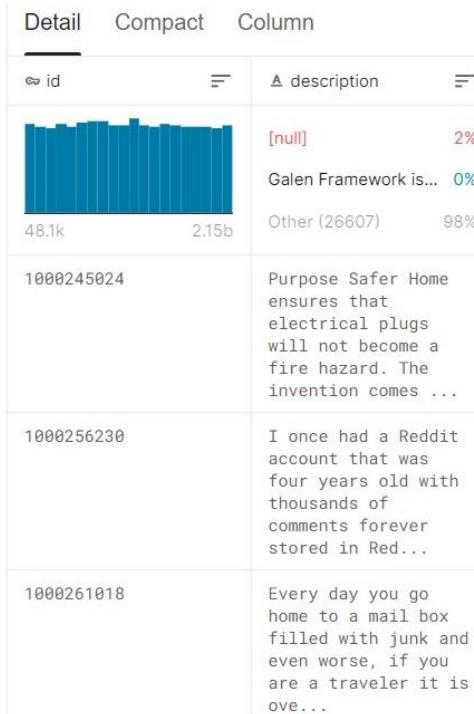


Figura 4.7: Visualización del archivo de descripción subido a Kaggle. Fuente: Elaboración propia.

4.1.3. Comentarios

Al igual que la descripción, los comentarios se obtuvieron utilizando la variable *urls* para web scraping, pero reemplazando caracteres que contengan “?ref=” en adelante por “/comments” para redireccionarse a la sección de comentarios de cada proyecto.

Los comentarios, al ser dinámicos, no podían extraerse mediante la librería BeautifulSoup-Soup como en el caso de las descripciones, por lo que se utilizó la librería Selenium para extraerlos al iniciar una sesión desde Google Chrome, como se observa en el algoritmo de la Figura 4.8.

```
def getPageText(url):
    driver = webdriver.Chrome(options=options)
    driver.set_page_load_timeout(30)
    driver.get(url)
    time.sleep(10)
    count = 0
    while (count<13):
        try:
            loadMoreButton = driver.find_element_by_xpath('//*[@id="react-project-comments"]/div/button')
            time.sleep(2)
            loadMoreButton.click()
            time.sleep(5)
            count += 1
        except (NoSuchElementException, StaleElementReferenceException, TimeoutException):
            break
        time.sleep(5)
    comments = driver.find_elements_by_css_selector('span.bg-ksr-green-700.white.px1.type-14.mr1, div.w100p')
    project_paragraphs = []
    for paragraph in comments:
        project_paragraphs.append(paragraph.text)
    idx_comment_creador = [i for i in range(len(project_paragraphs)) if project_paragraphs[i] == "Creator"]
    idx_comment_creador = [i+1 for i in idx_comment_creador]
    for index in sorted(idx_comment_creador, reverse=True):
        del project_paragraphs[index]
    idx_comment_creador = [i for i in range(len(project_paragraphs)) if project_paragraphs[i] == "Creator"]
    for index in sorted(idx_comment_creador, reverse=True):
        del project_paragraphs[index]
    project_paragraphs = [x for x in project_paragraphs if ("This comment" not in x)]
    project_paragraphs = [x for x in project_paragraphs if ("0:00" not in x)]
    project_paragraphs = [x for x in project_paragraphs if ("Showing " not in x)]
    project_paragraphs = [x for x in project_paragraphs if x]
    project_comments = []
    for project_text in project_paragraphs:
        project_text = project_text.replace(u'\xa0', u' ')
        project_text = project_text.replace(u'\n', u' ')
        project_text = re.sub(useless_characters, '', project_text)
        project_comments.append(project_text)
    del(project_paragraphs)
    driver.quit()
    time.sleep(randint(1,10))
    return project_comments
print("Scraping...")
```

Figura 4.8: Función del algoritmo web scraping de los comentarios. Fuente: Elaboración propia.

La función consiste en redireccionarse a la sección de comentarios del proyecto, esperar un máximo de 30 segundos de carga de la página, hacer clic en el botón “Load More” (Cargar más) hasta un límite de no más de 13 veces y almacenar en un vector cada comentario que no pertenezca al creador (se puede diferenciar gracias a una etiqueta en el extremo superior derecho del recuadro del comentario) de manera independiente. El siguiente paso implica la eliminación de ciertos caracteres especiales, emojis y comentarios completos de un patrocinador que no se encuentren en inglés. Finalmente, estos registros de vectores de comentarios con el id del proyecto correspondiente se almacenaron en un archivo .csv por cada fila terminada, el cual se encuentra disponible públicamente en Kaggle (Figura 4.9) así como los anteriores conjuntos comentados para poder ser descargados.

Detail	Compact	Column
id	comments	
 48.1k	71% ['@Creator: What wi... 0% Other (7857) 29%	
1000245024	[]	
1000256230	[]	
1000261018	[]	
100039820	[]	
1000468345	['Electrical Solutions Group installed the iRNinja at my parents house and they love it! They use to...']	

Figura 4.9: Visualización del archivo de comentarios subido a Kaggle. Fuente: Elaboración propia.

Para optimizar la descarga, se crearon 8 instancias en Google Cloud (Figura 4.10), en donde cada una contenía dos copias del algoritmo, con la cantidad de proyectos fraccionada en 16 partes para que sean ejecutados en paralelo. Si bien el tiempo total de la consolidación de esta base de información duró aproximadamente un mes debido a percances de la conexión interna de las instancias y algunos problemas de ineficiencia de la primera versión del algoritmo, durante el transcurso dentro de este lapso de tiempo fueron solucionados hasta lograr optimizar el algoritmo de web scraping y tener el conjunto final de datos tomó menos de 48 horas.

Compute Engine	Instanc... de VM
Instancias de VM	Nombre Zona Recomendación Usada por
Grupos de instancias	<input type="checkbox"/> Instance-1 us-central1-a
Plantillas de instancias	<input checked="" type="checkbox"/> scraping-01 us-central1-a
Nodos de único cliente	<input checked="" type="checkbox"/> scraping-02 us-central1-a
Imágenes de la máquina	<input checked="" type="checkbox"/> scraping-03 us-central1-a
Discos	<input checked="" type="checkbox"/> scraping-04 us-central1-a
Capturas	<input checked="" type="checkbox"/> scraping-05 us-central1-a
	<input checked="" type="checkbox"/> scraping-06 us-central1-a
	<input checked="" type="checkbox"/> scraping-07 us-central1-a
	<input checked="" type="checkbox"/> scraping-08 us-central1-a

Figura 4.10: Instancias lanzadas en paralelo para la extracción de comentarios. Fuente: Elaboración propia.

4.2. Análisis Exploratorios de los datos

Según su estado de financiamiento, los 27,251 proyectos se distribuyen mediante el gráfico de pie de la Figura 4.11. De esta, se observa que más del 70 % de estos fracasaron.

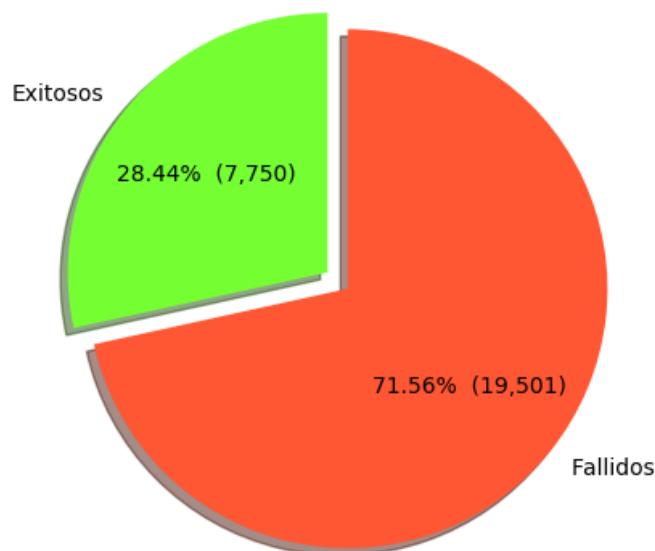


Figura 4.11: Distribución de proyectos tecnológicos según su estado. Fuente: Elaboración propia.

Al visualizarlos por año (Figura 4.12), se puede ver que el histórico comprende entre los períodos 2009 y 2019, siendo la gran mayoría perteneciente al año 2015.

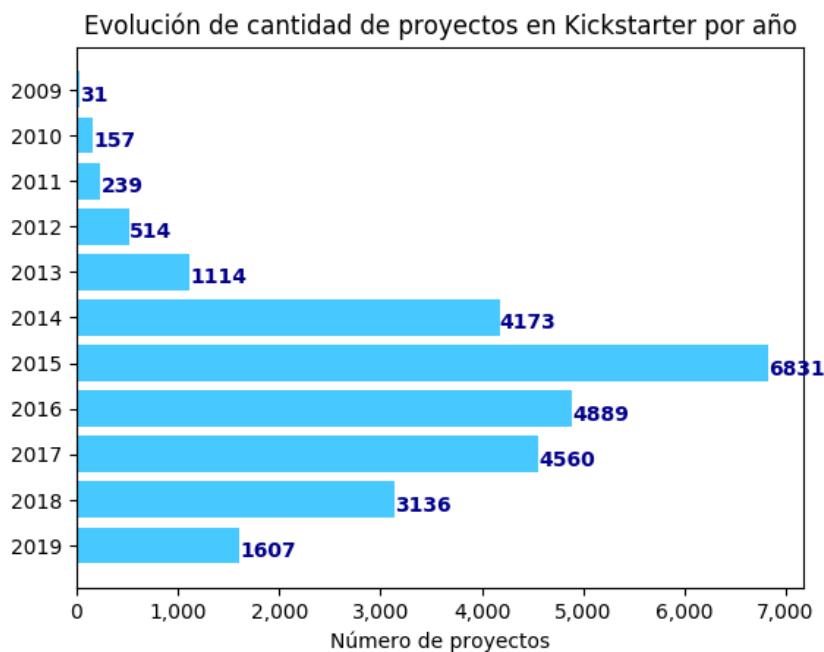


Figura 4.12: Evolución de cantidad de proyectos tecnológicos por año. Fuente: Elaboración propia.

La evolución histórica detallada por su estado se observa en la Figura 4.13.

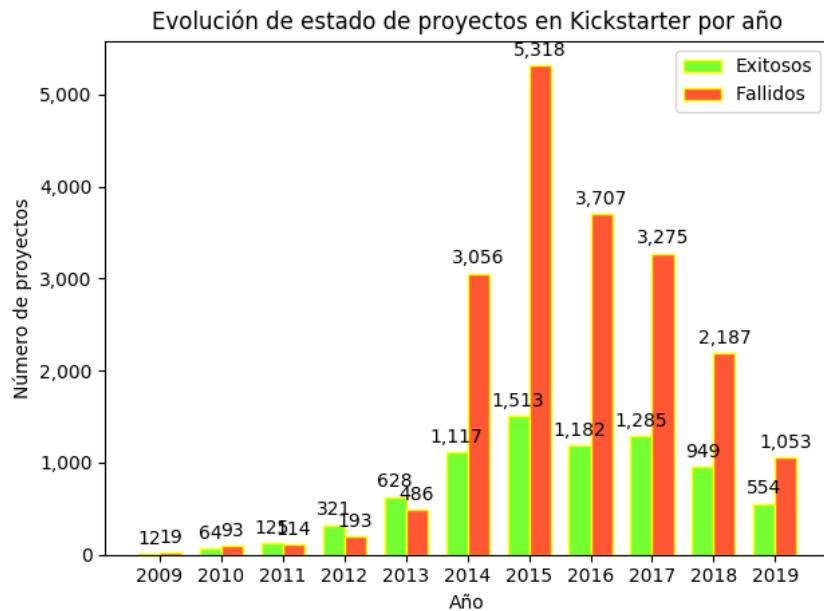


Figura 4.13: Evolución de proyectos tecnológicos, por su estado y año. Fuente: Elaboración propia.

4.2.1. Metainformación

De las 19 variables de la Tabla 4.1, y en base a los antecedentes (citar), se consideraron como potenciales variables independientes a 5 numéricas (*backers_count*, *goal*, *pledged*, *usd_pledged* y *duration*), 3 categóricas (*category*, *country* y *currency*) y 1 del tipo lista compuesta por números (*pledge_amounts*).

Para las variables categóricas, se realizaron gráficos de pastel para observar la distribución de sus valores. Así tenemos la Figura 4.14 para las categorías de tecnología, Figura 4.15 para los países (por código) y la Figura 4.16 para las divisas del monto final patrocinado (por código). De estos tres gráficos, se observa que más de la mitad de los patrocinadores provienen de los Estados Unidos (US) e invierten en dólares (USD). El segundo y tercer lugar los constituyen personas de Gran Bretaña (GB) y Canadá (CA), invirtiendo en sus monedas locales respectivas (GBP y CAD). Por el lado de las categorías, la más recurrente son de Apps, representando casi la cuarta parte del universo observado.

Para las variables numéricas, se calcularon sus datos estadísticos con ayuda de diagramas de caja y bigote:

- Número de patrocinadores de la campaña (*backers_count*):

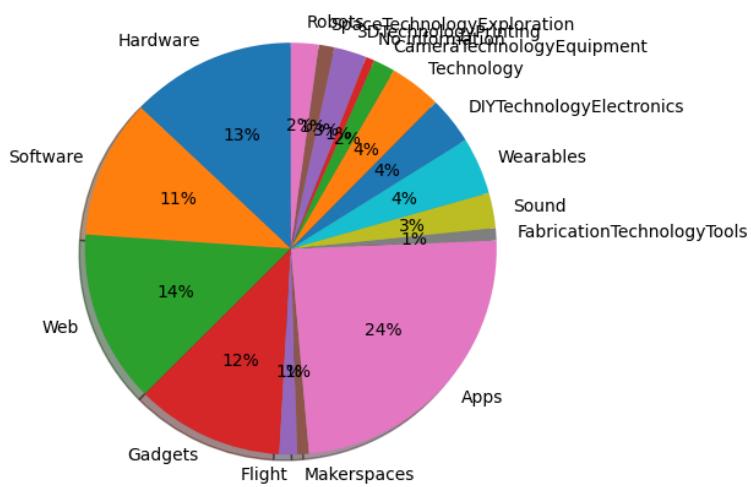


Figura 4.14: Distribución de categorías de tecnología. Fuente: Elaboración propia.

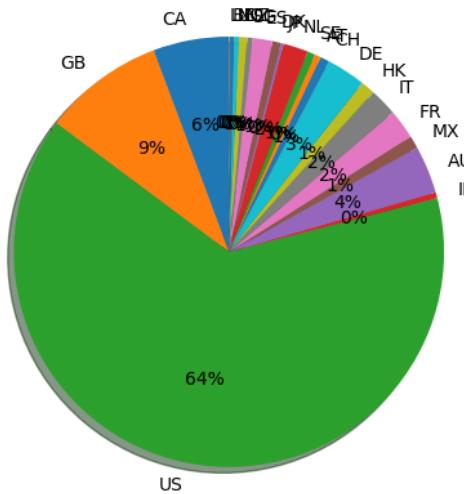


Figura 4.15: Distribución de países. Fuente: Elaboración propia.

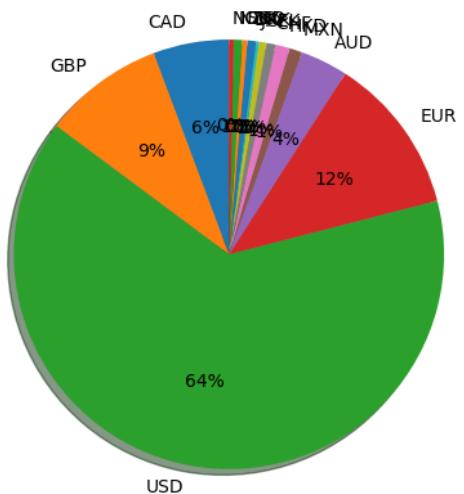


Figura 4.16: Distribución de divisa del monto patrocinado. Fuente: Elaboración propia

- Rango de valores: [0; 105,857]
- Media: 208.710469340575
- Mediana: 9.487
- Desviación estándar: 1,179.68237749203
- Varianza: 1,391,650.51176525

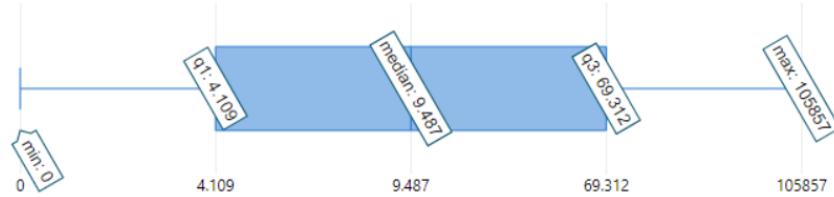


Figura 4.17: Diagrama de caja y bigote de patrocinadores. Fuente: Elaboración propia.

- Monto meta de la campaña (*goal*):
- Rango de valores: [1; 100,000,000]
 - Media: 91,263.9666162825
 - Mediana: 15,762.614
 - Desviación estándar: 1,259,282.1587922
 - Varianza: 1,585,791,555,452.35

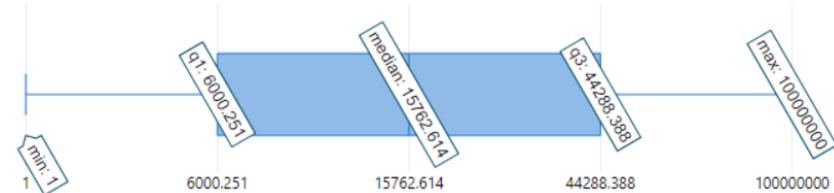


Figura 4.18: Diagrama de caja y bigote de meta. Fuente: Elaboración propia.

- Monto patrocinado al final de la campaña (*pledged*):
- Rango de valores: [0; 17,406,300]
 - Media: 34,668.5134710787
 - Mediana: 1,382.933
 - Desviación estándar: 226,763.900313481
 - Varianza: 51,421,866,485.3822



Figura 4.19: Diagrama de caja y bigote de monto patrocinado. Fuente: Elaboración propia.

■ Duración de la campaña (*duration*).

- Rango de valores: [1; 92]
- Media: 35.4654141132436
- Mediana: 30
- Desviación estándar: 11.84570862999998
- Varianza: 140.320812946853

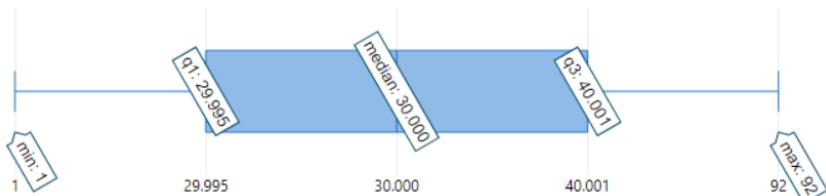


Figura 4.20: Diagrama de caja y bigote de duración. Fuente: Elaboración propia.

Posterior a este entendimiento de datos, se elaboró una matriz de correlaciones (Figura 4.21) para encontrar correlaciones entre ellas y determinar la existencia de alguna variable redundante y descartarla para no afectar el rendimiento del modelo.

Como se puede apreciar en la figura anterior, la variable *usd_pledged* está altamente correlacionada con las variables *backers_count* y *pledged* (ambas con un aproximado de 70 %). Esto quiere decir que dicha variable no es significativa porque explicaría de manera muy similar a las otras dos.

Asimismo, si se observan los registros desde una matriz que contiene, además de gráficos de dispersión de las correlaciones, histogramas de las variables independientes como en la Figura 4.22, se confirma y concluye no utilizar las observaciones comentadas.

El mismo gráfico, segmentado por las dos clases del estado de financiamiento, se aprecia en la Figura 4.23.

Se concluye entonces que la variable *duration* es la única que sigue una distribución normal.

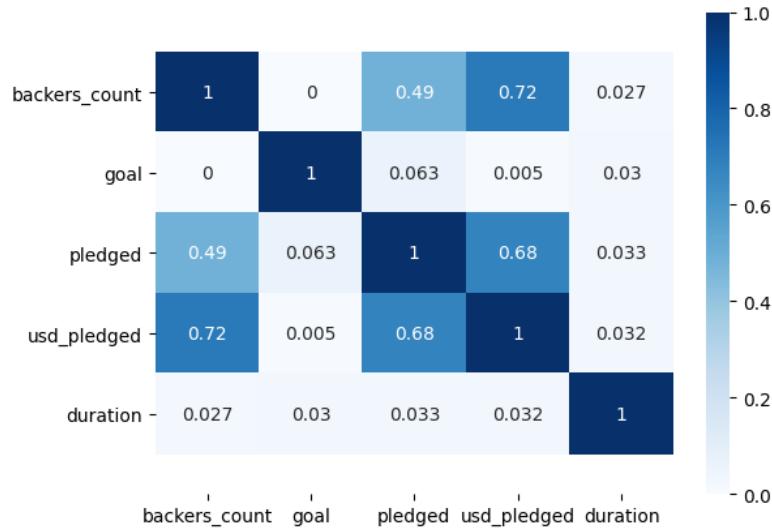


Figura 4.21: Matriz de correlaciones entre variables independientes. Fuente: Elaboración propia.

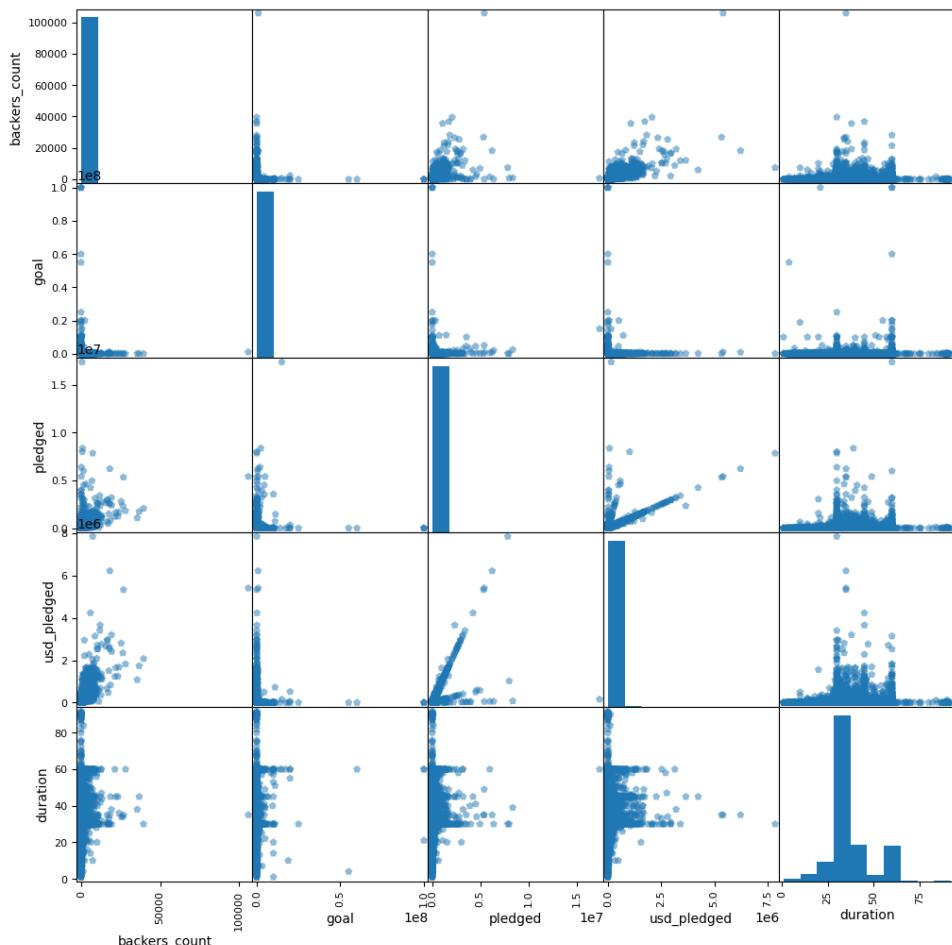


Figura 4.22: Gráfico de dispersión de correlaciones entre variables independientes. Fuente: Elaboración propia.

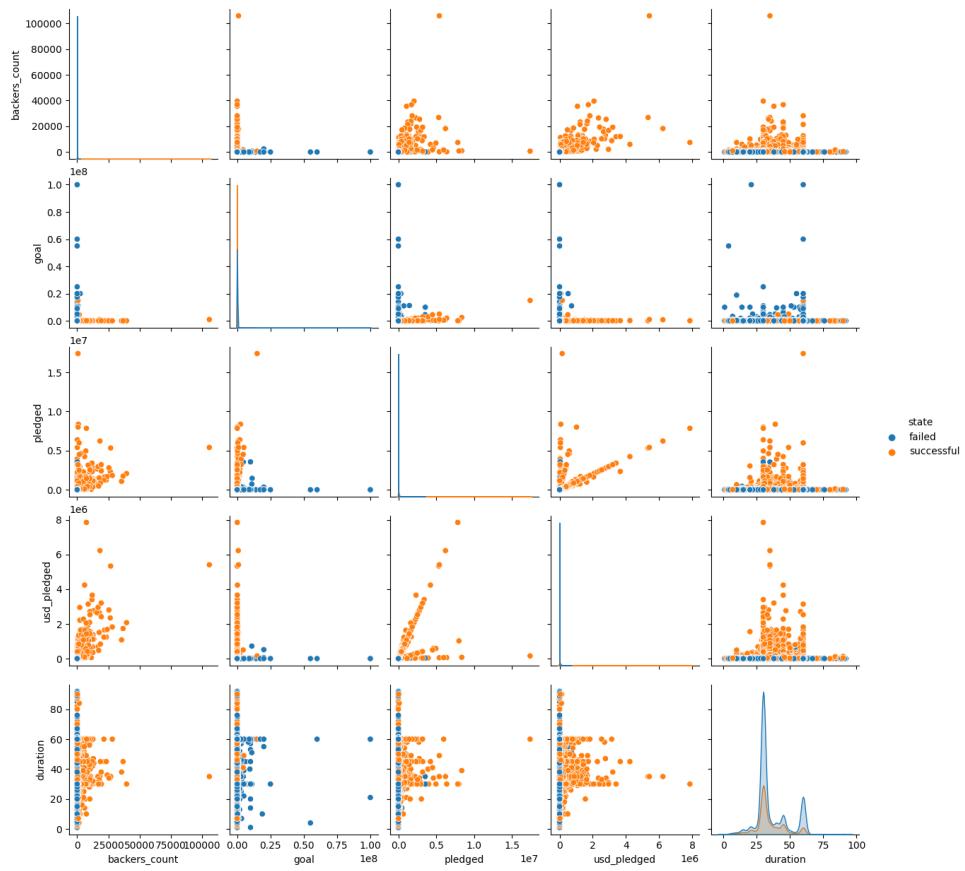


Figura 4.23: Gráfico alterno de dispersión de correlaciones entre variables independientes. Fuente: Elaboración propia.

4.2.2. Descripción

La descripción con mayor cantidad de palabras presenta 5,152 palabras y, a nivel total de proyectos, el vocabulario es de 165,683 palabras. La Figura 4.24 representa la nube de palabras del contenido de cada descripción, donde las más destacadas son aquellas que aparecen con mayor frecuencia.

4.2.3. Comentarios

Al analizar los proyectos exitosos y fracasados de acuerdo a la presencia de comentarios (Figura 4.25), se observa que hay una sólida relación entre aquellos que fracasaron y los que no cuentan con comentarios, mientras que para los proyectos exitosos, la asociación es menos contundente ya que la diferencia entre aquellos que presentan comentarios y los que no es menor.

Si se repite el ejercicio del análisis anterior, ahora partiendo de los comentarios por

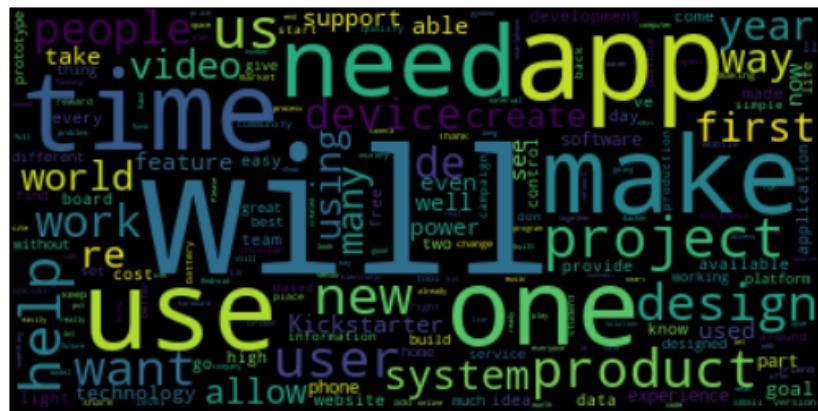


Figura 4.24: Nube de palabras de descripciones. Fuente: Elaboración propia.

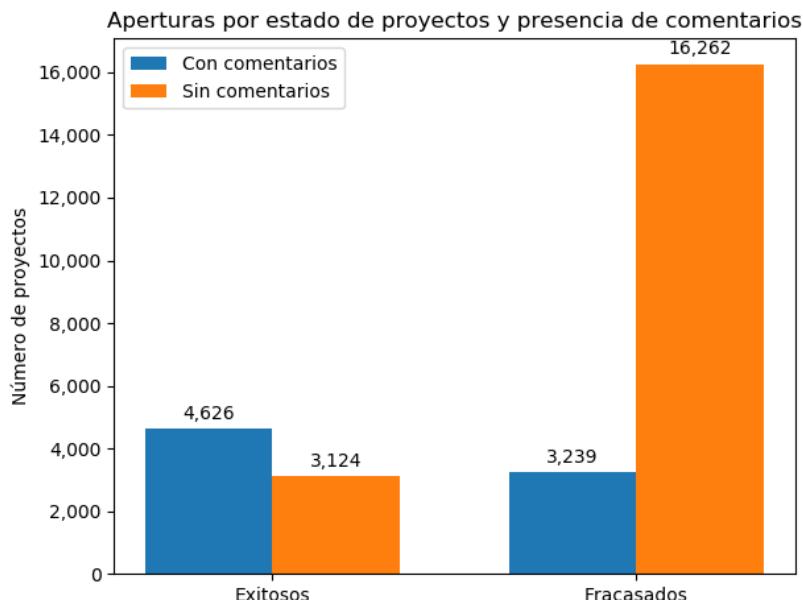


Figura 4.25: Aperturas de proyectos por estado de financiamiento y presencia de comentarios.
Fuente: Elaboración propia.

presencia de comentarios y abiertos por su estado de financiamiento (exitosos y fracasados), se visualiza en la Figura 4.26 una relación casi idéntica al anterior gráfico de barras.

Por último, del universo de proyectos tecnológicos que presentan comentarios, los más de 4 mil proyectos exitosos contienen casi la totalidad de comentarios registrados (97 %), representando más de 475 mil como se aprecia en la Figura 4.27. Estos datos confirman la correlación existe entre la presencia de un volumen considerable de comentarios y su éxito de financiación.

Respecto al contenido, en las Figura 4.28 y Figura 4.29 se ilustran las nubes de palabras agrupadas en unidades y/o parejas más frecuentes, respectivamente.

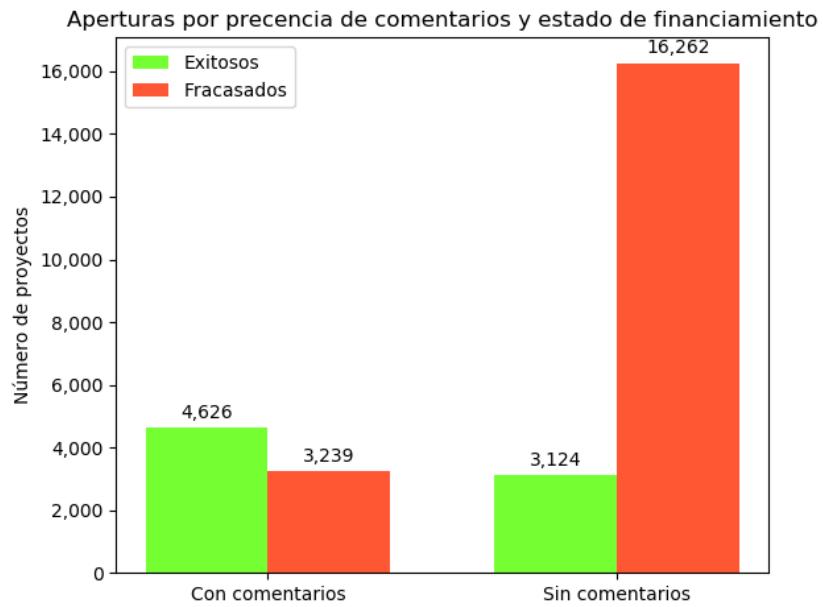


Figura 4.26: Aperturas de proyectos por presencia de comentarios y estado de financiamiento.

Fuente: Elaboración propia.



Figura 4.27: Distribución de comentarios en proyectos exitosos y fracasados. Fuente: Elaboración propia.

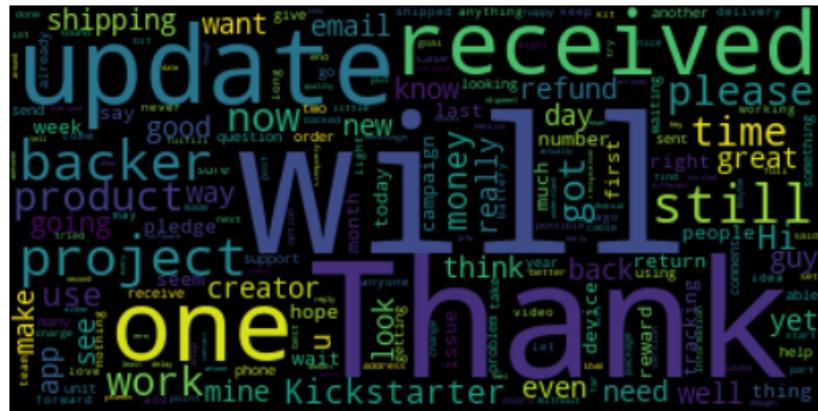


Figura 4.28: Nube de palabras de comentarios por unidades más frecuentes. Fuente: Elaboración propia.

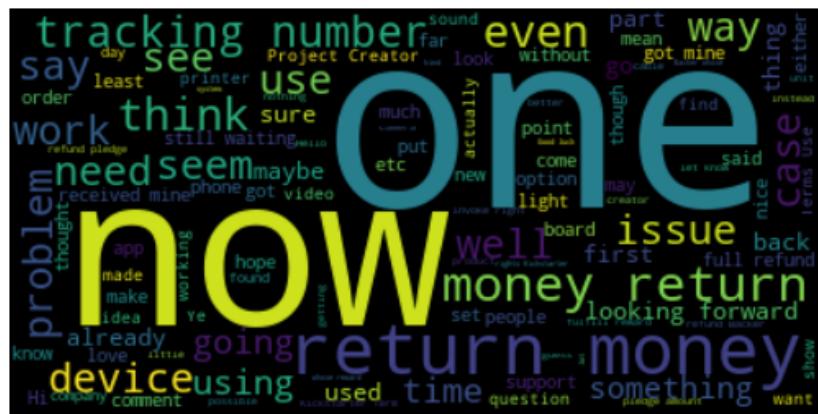


Figura 4.29: Nube de palabras de comentarios por unidades o parejas más frecuentes. Fuente: Elaboración propia.

4.3. Pre-procesamiento de los conjuntos de datos

4.3.1. Metainformación

De acuerdo al autor (citar), se consideran 7 variables adicionales basadas en la mediana (*pledges_median*), promedio (*pledges_mean*), valor máximo (*pledges_max*), valor mínimo (*pledges_min*), variación estándar (*pledges_std*), cantidad (*pledges_num*) y completitud (*completeness*) del monto prometido. La nueva matriz de correlación se observa en la Figura 4.30.

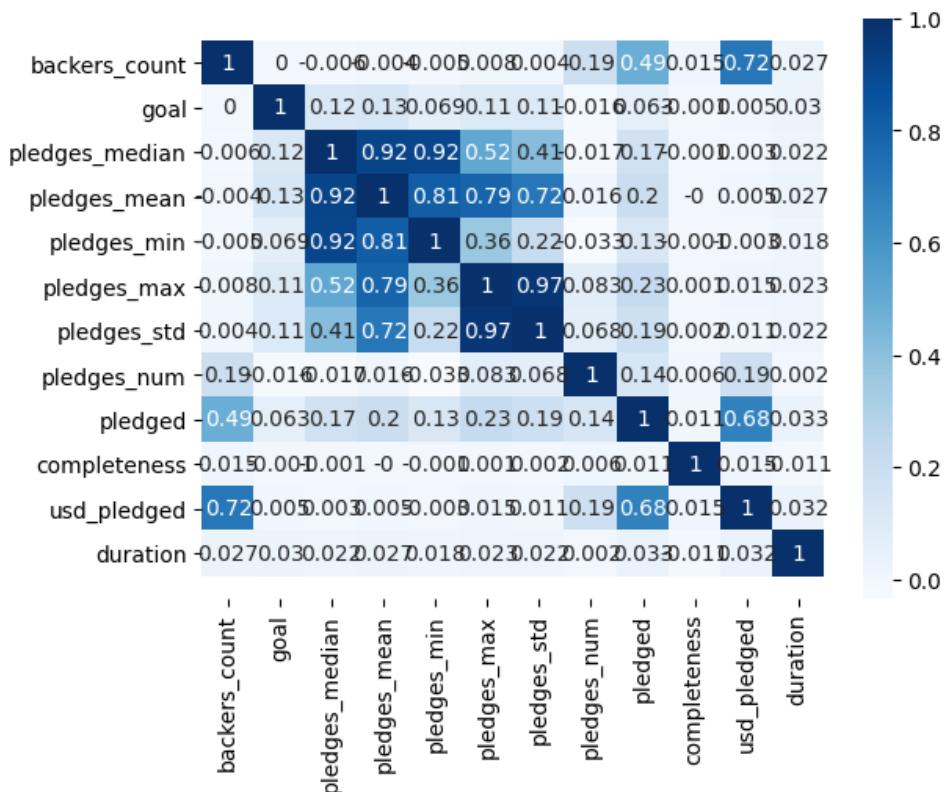


Figura 4.30: Matriz de correlaciones entre variables independientes considerando adicionales.

Fuente: Elaboración propia.

Como se puede observar en esta nueva matriz, el promedio presenta una alta correlación con el valor mínimo, valor máximo y la variación estándar del monto prometido. De igual manera, esta última presenta una correlación cerca a la unidad con el valor máximo. Por último, también se observa alta correlación entre el monto prometido y su conversión a dólares.

La selección de variables se considera bajo la regla de una correlación máxima de 0.30 (citar). Bajo esta norma, aquellas que cumplen son *goal*, *pledges_num*, *completeness* y *duration*. Para conservar la mayor cantidad posible de variables, se contabilizan las variables correlacionadas con cada una de las restantes. Así por ejemplo, de acuerdo a la figura anterior, tanto

pledges_median, *pledges_mean* y *pledges_max* sobrepasan el límite de la regla al ser tomadas en cuenta con otras 4; *pledges_min* y *pledges_std* con 3; y *pledged* y *usd_pledged* con 2.

4.3.2. Descripción

De acuerdo al autor (citar), para poder entrenar un modelo de clasificación binaria basado en texto, se debe pre-procesar siguiendo el flujo de Figura 4.31. Se remueven las contracciones, caracteres especiales, enlaces externos y contenidos en otros idiomas. Este resultado será tokenizado a palabras individuales con la intención de eliminar palabras de parada en inglés, lematizar las restantes y finalmente juntarlas en una lista, separadas por su proyecto correspondiente.

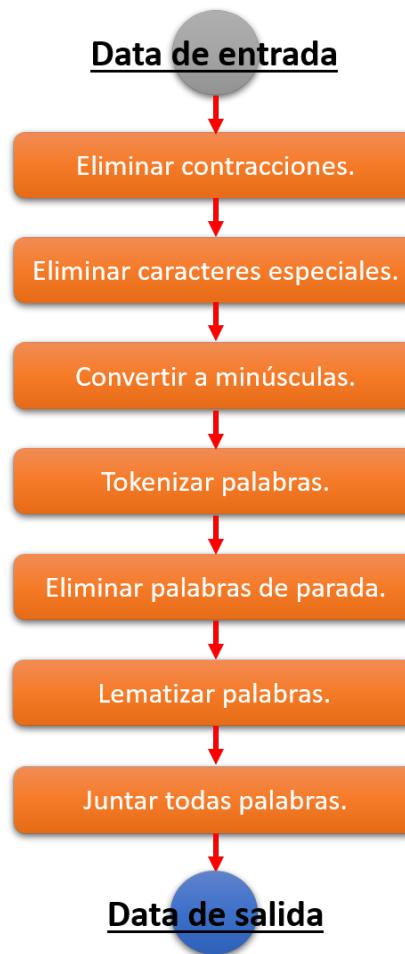


Figura 4.31: Flujograma de limpieza de conjunto de datos de descripciones. Fuente: Elaboración propia.

Para el presente trabajo, se omitió el paso de lematización ya que según el antecedente (citar) y corroborando con los experimentos, los resultados fueron más favorables sin el anterior

mencionado. Con el pre-procesamiento detallado en el párrafo anterior, la descripción de mayor longitud pasó a presentar 3,671 palabras y a nivel general de proyectos, el nuevo vocabulario tuvo 165,526 palabras.

Las nubes de palabras reflejan las palabras más frecuentes dentro de un conjunto de datos. La Figura 4.32 representa aquellas palabras que más aparecen en las descripciones de proyectos de tecnología en Kickstarter.

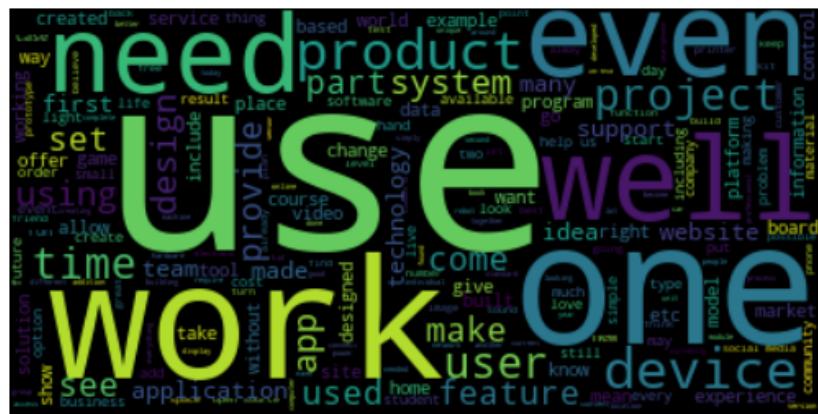


Figura 4.32: Nube de palabras de descripciones posterior al pre-procesamiento. Fuente: Elaboración propia.

4.3.3. Comentarios

La base de datos de comentarios está conformada por un proyecto asignado a un conjunto de comentarios de patrocinadores, separados entre sí. Antes de realizar el pre-procesamiento, se agregó un término aleatorio no relacionado con las temáticas para llenar aquellos proyectos sin comentarios: "kuwagatabaizan".

El siguiente paso fue separar cada comentario de un mismo proyecto, manteniendo el id y estado de financiamiento correspondiente, es decir, existirán registros duplicados por la columna *id*. Así, la proporción original de las etiquetas de la variable dependiente, el estado de financiamiento, será ahora de 91 % de proyectos exitosos (341,943) y 9 % fracasados (32,777).

Finalmente, se removieron caracteres especiales, espacios múltiples, signos de puntuación y números. La nueva nube de palabras se presenta en la Figura 4.33.

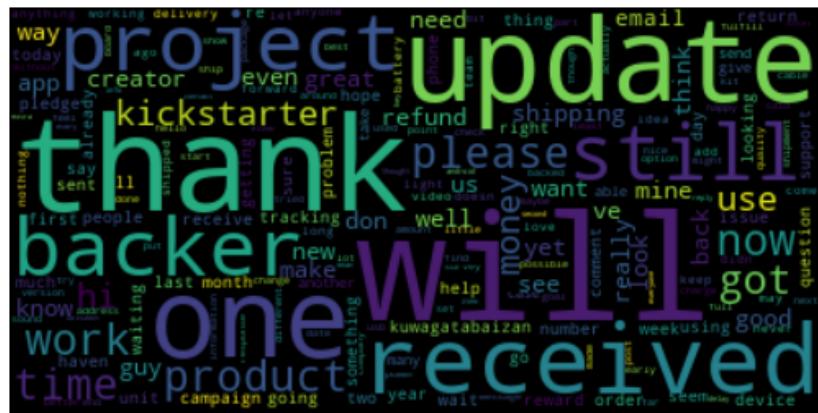


Figura 4.33: Nube de palabras de comentarios posterior al pre-procesamiento. Fuente: Elaboración propia.

4.4. Creación de los modelos predictivos

Una vez generado las variables independiente (observación o conjunto de observaciones, X) y dependiente (etiquetas del estado de financiamiento, Y), cada uno es separado en subconjuntos de entrenamiento y prueba, con una proporción de 80% y 20% respectivamente según el autor (citar) y asignándose un valor fijo de aleatoriedad. Asimismo, dentro de los parámetros de separación, se establece el argumento de estratificación según la variable Y dentro de la función `train_test_split`, de la siguiente manera:

```
train_test_split(X, Y, test_size = 0.20, stratify = Y, random_state=0)
```

Antes de crear los modelos correspondientes, y después de definir los valores de entrada y parámetros (las subsecciones que se detallarán a continuación), se asigna una semilla inicial con un valor fijado por el usuario con el fin de evitar resultados aleatorios para futuras iteraciones. Se establece, además, una ruta local en donde se almacena cada punto de control basado en la mejora de la pérdida del subconjunto de validación con respecto a su iteración anterior. En caso de un estancamiento de esta última durante 10 épocas, es decir, si el valor de la pérdida no decrementa, el modelo dejará de entrenar. A esta regla se le añade la reducción de la tasa de aprendizaje luego de 5 épocas en caso el valor de la exactitud del subconjunto de validación no refleje un incremento. El objetivo de estas condiciones es evitar el sobreajuste en los modelos.

Por último, es importante asignar un peso distinto para cada una de las dos clases de la variable dependiente *status*. Con el fin de evitar un mal entrenamiento, los pesos de ambas clases se balancean y se almacenan en un diccionario con su etiqueta correspondiente.

4.4.1. Metainformación

Con las entradas listas para los parámetros, se diseñó el modelo de descripciones basada en un Perceptrón Multicapa (MLP por sus siglas en inglés) bajo la arquitectura de la Figura 4.34. La cantidad de épocas fue de 100 y el número de lotes fue de 32.

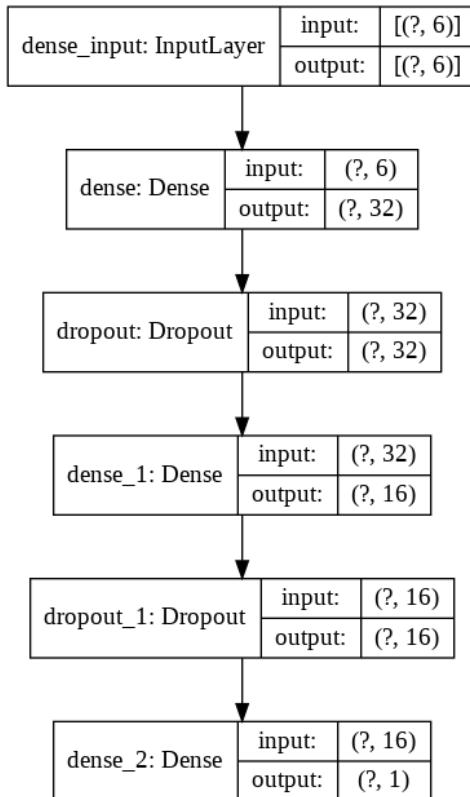


Figura 4.34: Arquitectura de modelo MLP para la metadata. Fuente: Elaboración propia.

La arquitectura comienza con la capa de entrada alimentadas por las 6 variables consideradas, la cual asimismo representará la cantidad de neuronas, tanto de entrada como de salida. Si bien no existe alguna regla general para definir el número de capas óptimas, así como los hiperparámetros que se deben configurar en ellas, se puede utilizar como referencia algunas metodologías como las Reglas del Pulgar ([Ranjan, 2019](#)).

De acuerdo a algunas de estas, el número de capas ocultas comienza con 2 sin contar la última. La primera capa densa continúa a la capa de entrada, mientras que la segunda aparece después de la primera capa de desactivación.

Otro punto considerado fue el número de nodos o neuronas de las capas intermedias. Estas deben seguir una progresión geométrica de 2, donde la primera capa debe ser la mitad del número de variables en la capa de entrada. Dado que la mitad de 6 es un valor que no cumple, un número potencial puede ser 4.

El autor de algunas de estas reglas también menciona tener en consideración utilizar la función de activación **relu** para las capas intermedias, una tasa de abandono de por lo menos 0.5 para las capas de desactivación, tamaño de salida de 1 neurona y función de activación **sigmoide** por tratarse de un problema de clasificación binaria, utilizar el optimizador **adam**, comenzar con 20 épocas en adelante de acuerdo al progreso de los resultados y fijar un tamaño de lote bajo progresión geométrica de 2; además de otros requerimientos previamente establecidos como la ponderación de clases para la variable dependiente en caso de datos desbalanceados y escalado de datos antes del entrenamiento.

Todas estas opciones fueron probadas en el modelo y evaluadas con las métricas correspondientes. Sin embargo, al calibrar el modelo y comparar distintos resultados, se obtuvo que la mejor cantidad de neuronas para la primera capa densa era de 32. De este modo, la siguiente capa intermedia se le asignó la mitad (16). La función de activación **tanh** para la segunda capa oculta presentó mejores resultados, así como tasas de abandono entre 0.25 y 0.3 para las capas de desactivación. Por último, además de *adam*, se realizaron experimentos con otros optimizadores como por ejemplo *RMSprop* siendo este el resultado más cercano. Al final, *adam* fue escogido pero con una tasa de aprendizaje baja como 0.0005 debido a que el modelo tenía a aprender muy rápido durante el transcurso de las épocas. En conjunto, el ratio de decaimiento se asignó un valor más bajo, 0.00005, y para evaluar los subconjuntos de entrenamiento y prueba se eligió precisión como métrica.

4.4.2. Descripción

Para crear la capa de incrustación de palabras, se usa la función *Tokenizer* de la librería **tensorflow.keras.preprocessing.text**, creando una función para originar un diccionario de palabras a índice al subconjunto de entrenamiento, asignándose a cada una un código. La función creada asimismo permite colocar un valor (en este caso, el término *;OOV;*) a aquellos términos no entrenados y serán parte de la data de prueba (Figura 4.35). A continuación, los arreglos de estos textos codificados son llenados con ceros a la derecha, hasta uniformizar con la longitud de la descripción más larga, ya que todas presentan distintos tamaños (Figura 4.36).

Una vez obtenido el vocabulario de palabras únicas y asignado el tamaño de cada arreglo (en este caso, se asignó el de la descripción de mayor longitud), se procedió a elaborar la matriz de características usando incrustaciones de GloVe (Figura 4.37), un algoritmo de aprendizaje no supervisado para obtener representaciones vectoriales de palabras ([Pennington y col., s.f.](#)). Para la presente investigación, se seleccionó la opción más usada, GloVe pre-entrenado con 6 billones de tokens, vocabulario de más de 400 mil palabras de Wikipedia (2014) y Gigaword (5ta edición) de la Universidad de Pensilvania, y matriz de 100 columnas, donde cada

```

oov_tok = "<OOV>"

# fit a tokenizer
def create_tokenizer(lines):
    tokenizer = Tokenizer(oov_token=oov_tok)
    tokenizer.fit_on_texts(lines)
    return tokenizer

tokenizer = create_tokenizer(training_sentences)

```

Figura 4.35: Función de tokenización de palabras de descripciones. Fuente: Elaboración propia.

```

# encode a list of lines
def encode_text(tokenizer, lines, length):
    # integer encode
    encoded = tokenizer.texts_to_sequences(lines)
    # pad encoded sequences
    padded = pad_sequences(encoded, maxlen=length, padding='post')
    return padded

# encode data
training_sentences = encode_text(tokenizer, training_sentences, length_long_sentence)
testing_sentences = encode_text(tokenizer, testing_sentences, length_long_sentence)

```

Figura 4.36: Función de codificación de palabras y relleno de arreglos. Fuente: Elaboración propia.

una contendrá las incrustaciones de palabras GloVe para las palabras del corpus ([Malik, 2019](#)).

```

embeddings_dictionary = dict()
glove_file = open('Glove/glove.6B.100d.txt', encoding="utf8")

for line in glove_file:
    records = line.split()
    word = records[0]
    vector_dimensions = asarray(records[1:], dtype='float32')
    embeddings_dictionary [word] = vector_dimensions

glove_file.close()

embedding_matrix = zeros((vocab_size, embedding_dim))
for word, index in tokenizer.word_index.items():
    embedding_vector = embeddings_dictionary.get(word)
    if embedding_vector is not None:
        embedding_matrix[index] = embedding_vector

```

Figura 4.37: Elaboración de matriz de incrustaciones de palabras. Fuente: Elaboración propia.

Con las entradas listas para los parámetros, se diseñó el modelo de descripciones basada en una Red Neuronal Convolutacional (CNN por sus siglas en inglés) bajo la arquitectura de la Figura 4.38. La cantidad de épocas fue de 100 y el número de lotes fue de 128.

4.4.3. Comentarios

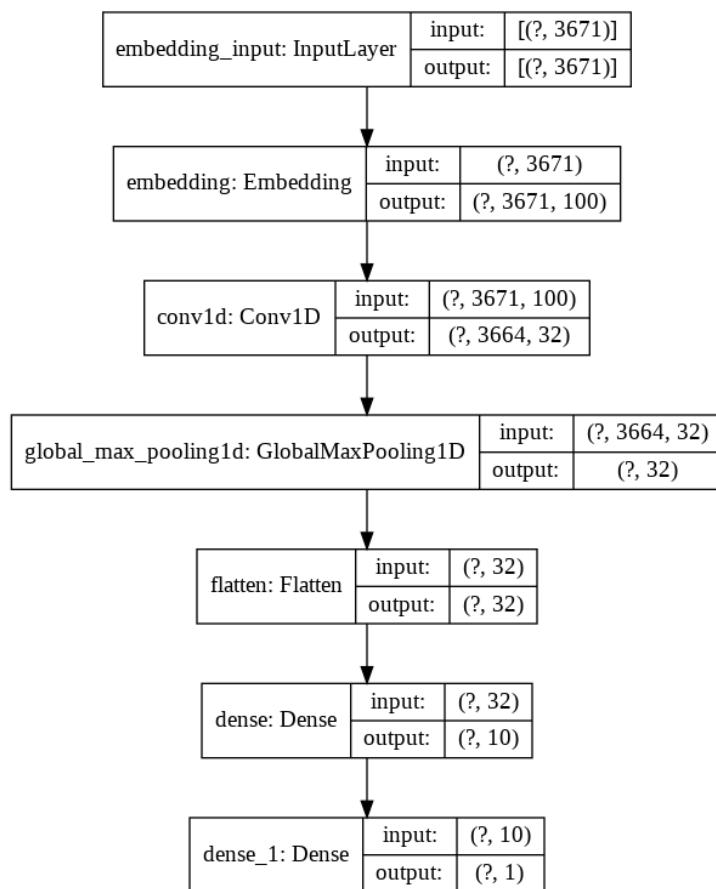


Figura 4.38: Arquitectura de modelo CNN para las descripciones. Fuente: Elaboración propia.

Capítulo 5

ANÁLISIS Y DISCUSIÓN DE RESULTADOS

5.1. Metadata

Luego de 27 épocas, con un rango entre 2 y 3 segundos de entrenamiento cada una, el modelo dejó de entrenar dado que durante 7 épocas no registró una reducción en el valor de la pérdida del subconjunto de validación, por más que 3 épocas antes se había reducido su tasa de aprendizaje.

Así, de acuerdo a las Figuras [5.1](#) y [5.2](#), en la época 20 se registran los mejores valores de exactitud y pérdida para el subconjunto de validación, alcanzando 0.9523 y 0.1246 respectivamente.

La matriz de confusión resultante se representa en la Figura [5.3](#).

De esta matriz, derivan los siguientes resultados evaluados según las métricas seleccionadas (citar).

5.2. Descripción

Luego de 78 épocas, con un rango entre 102 y 114 segundos de entrenamiento cada una, el modelo dejó de entrenar dado que durante 10 épocas no registró una reducción en el valor de la pérdida del subconjunto de validación, por más que 5 épocas antes se había reducido su tasa de aprendizaje.

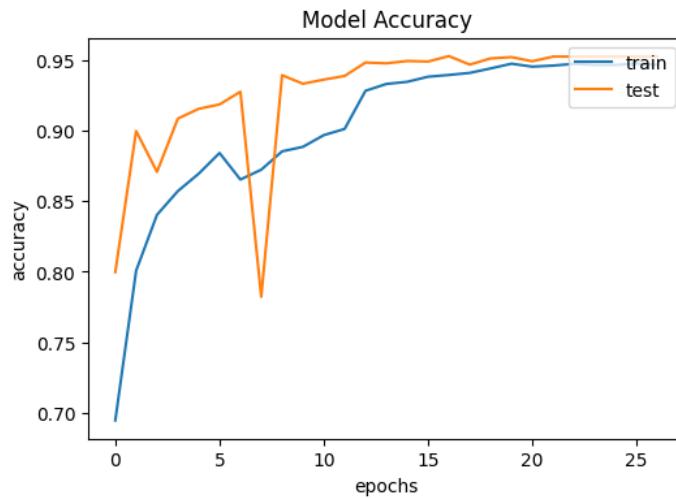


Figura 5.1: Evolución de la exactitud de los subconjuntos de entrenamiento y validación para el modelo MLP de metadata para un entrenamiento de 100 épocas. Fuente: Elaboración propia.

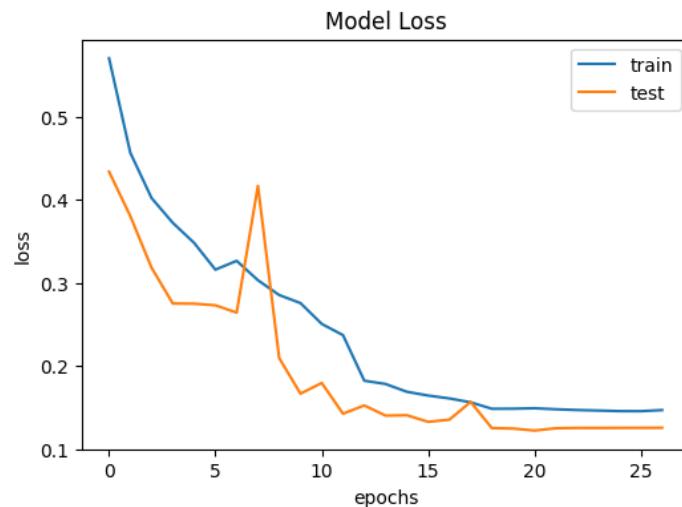


Figura 5.2: Evolución de la pérdida de los subconjuntos de entrenamiento y validación para el modelo MLP de metadata para un entrenamiento de 100 épocas. Fuente: Elaboración propia.

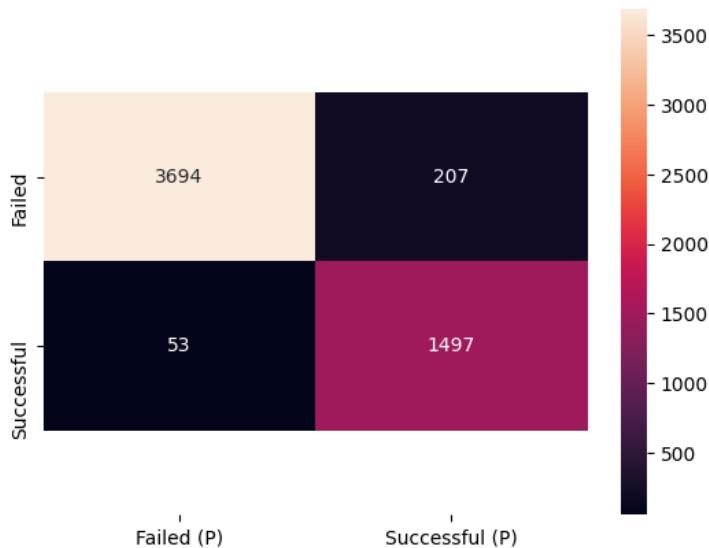


Figura 5.3: Matriz de confusión para el modelo MLP de metadata. Fuente: Elaboración propia.

Así, de acuerdo a las Figuras 5.4 y 5.5, en la época 68 se registran los mejores valores de exactitud y pérdida para el subconjunto de validación, alcanzando 0.7665 y 0.4922 respectivamente.

La matriz de confusión resultante se representa en la Figura 5.6.

De esta matriz, derivan los siguientes resultados evaluados según las métricas seleccionadas (citar).

5.3. Comentarios

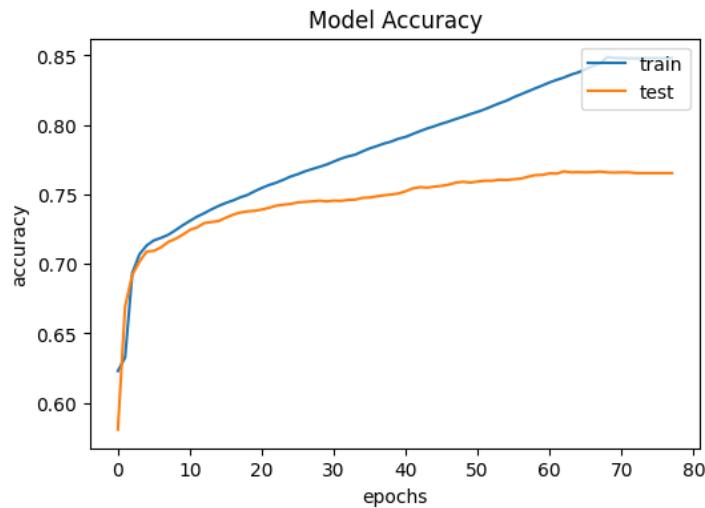


Figura 5.4: Evolución de la exactitud de los subconjuntos de entrenamiento y validación para el modelo CNN de descripciones para un entrenamiento de 100 épocas. Fuente: Elaboración propia.

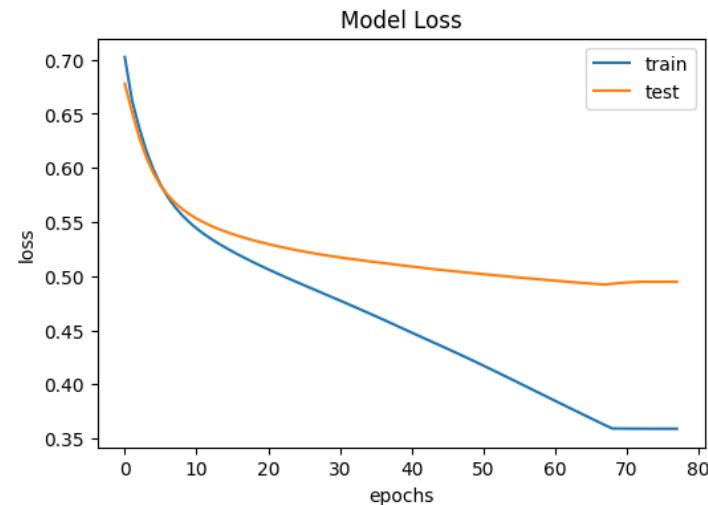


Figura 5.5: Evolución de la pérdida de los subconjuntos de entrenamiento y validación para el modelo CNN de descripciones para un entrenamiento de 100 épocas. Fuente: Elaboración propia.

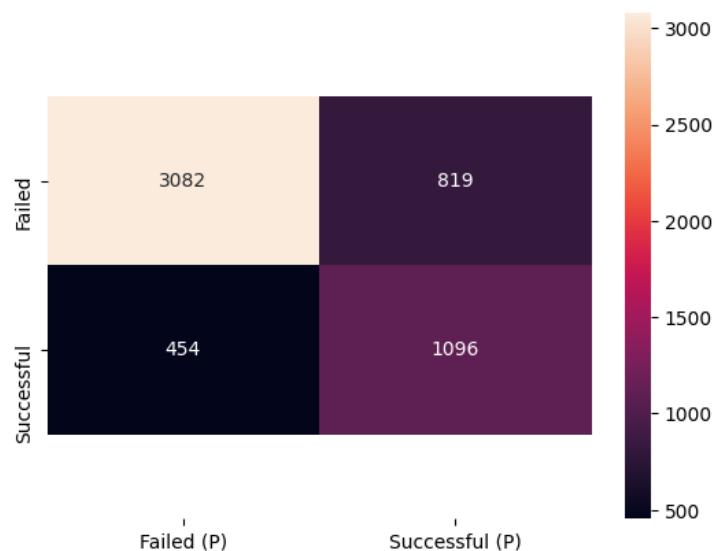


Figura 5.6: Matriz de confusión para el modelo CNN de descripciones. Fuente: Elaboración propia.

Capítulo 6

CONCLUSIONES Y RECOMENDACIONES

6.1. Conclusiones

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn? Kjift ”not at all!...

6.2. Recomendaciones

Nisi porta lorem mollis aliquam ut porttitor leo. Aenean pharetra magna ac placerat vestibulum. Est placerat in egestas erat imperdiet sed euismod. Velit euismod in pellentesque massa placerat. Enim praesent elementum facilisis leo vel fringilla. Ante in nibh mauris cursus mattis molestie a iaculis. Erat pellentesque adipiscing commodo elit at imperdiet dui accumsan sit. Porttitor lacus luctus accumsan tortor posuere ac ut. Tortor at auctor urna nunc id. A iaculis at erat pellentesque adipiscing commodo elit.

BIBLIOGRAFÍA

- A Not So Random Walk. (2019). Backpropagation Example With Numbers Step by Step. <https://www.anotsorandomwalk.com/backpropagation-example-with-numbers-step-by-step/>
- Alpaydin, E. (2014). *Introduction to Machine Learning* (Tercera edición). MIT Press.
- Asociación de Emprendedores de Perú. (2018). Avances y limitaciones del emprendimiento peruano. <https://asep.pe/index.php/avances-limitaciones-emprendimiento-peruano/>
- BBVA OpenMind. (2019). ¿Qué es el aprendizaje Profundo? (A. Banafa, Ed.). <https://www.bbvaopenmind.com/tecnologia/mundo-digital/que-es-el-aprendizaje-profundo/>
- Beckwith, J. (2016). Predicting Success in Equity Crowdfunding. *Joseph Wharton Scholars*. http://repository.upenn.edu/joseph_wharton_scholars/25
- Bertona, L. F. (2005). *Entrenamiento de Redes Neuronales basado en Algoritmos Evolutivos* (Tesis de grado). Universidad de Buenos Aires. Buenos Aires. <http://laboratorios.fi.uba.ar/lxi/bertona-tesisingenieriainformatica.pdf>
- Betancourt, G. A. (2005). Las Máquinas de Soporte Vectorial (SVMs). *Scientia et Technica*. <https://revistas.utp.edu.co/index.php/revistaciencia/article/view/6895/4139>
- Braulio Gil, N. & Curto Díaz, J. (2015). *Customer Analytics: Mejorando la inteligencia del cliente a través de los datos* (Publicación). Universitat Oberta de Catalunya. Barcelona. <https://docplayer.es/17897069-Customer-analytics-mejorando-la-inteligencia-del-cliente-a-traves-de-los-datos-jordi-conesa-i-caralt-coordinador-nuria-braulio-gil-josep-curto-diaz.html>
- Calvo, D. (2017). Clasificación de redes neuronales artificiales. <http://www.diegocalvo.es/clasificacion-de-redes-neuronales-artificiales/>
- Calvo, D. (2018a). Definición de Red Neuronal Recurrente. <http://www.diegocalvo.es/red-neuronal-recurrente/>
- Calvo, D. (2018b). Función de activación - Redes neuronales. <http://www.diegocalvo.es/funcion-de-activacion-redes-neuronales/>

- Chen, S.-Y., Chen, C.-N., Chen, Y.-R., Yang, C.-W. & Lin, W.-C. (2015). Will Your Project Get the Green Light? Predicting the Success of Crowdfunding Campaigns. <http://aisel.aisnet.org/pacis2015/79>
- Cheng, C., Tan, F., Hou, X. & Wei, Z. (2019). Success Prediction on Crowdfunding with Multimodal Deep Learning, 2158-2164. <https://www.ijcai.org/proceedings/2019/0299.pdf>
- Colgren, D. (2014). The rise of crowdfunding: Social media, big data, cloud technologies. *Strategic Finance*, 95(10), 56.
- Collins, L. (2014). *Crowdfunding: Innovative access to finance and regulatory challenges*.
- Cyberclic. (s.f.). ¿Qué es una campaña publicitaria? <https://www.cyberclick.es/publicidad/campana-publicitaria>
- Deng, L. & Liu, Y. (2018). *Deep Learning in Natural Language Processing*. Springer. <https://doi.org/10.1007/978-981-10-5209-5>
- Dorofki, M., Elshafie, A. H., Jaafar, O., Karim, O. A. & Mastura, S. (2012). Comparison of Artificial Neural Network Transfer Functions Abilities to Simulate Extreme Runoff Data. En IPCBEE (Ed.). IACSIT Press. <http://www.ipcbee.com/vol33/008-ICEEB2012-B021.pdf>
- Fernández-Bedoya, V. H., Gago-Chávez, J. d. J. S., Meneses-la-Riva, M. E. & Suyo-Vega, J. A. (2020). Collaborative Economy in Peru: Past, Present and Future. *Path of Science*, 6(5), 7001-7006. <https://doi.org/10.22178/pos.58-5>
- Figueroa M., G. (s.f.). Convolución y transformadas. *Revista digital Matemática*. <https://tecdigital.tec.ac.cr/revistamatematica/cursos-linea/EcuacionesDiferenciales/EDO-Geo/edo-cap5-geo/laplace/node7.html>
- Gandhi, R. (2018). Support Vector Machine — Introduction to Machine Learning Algorithms. *Towards Data Science*. <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>
- Gartner. (2019a). Gartner IT Glossary - Machine Learning. <https://www.gartner.com/it-glossary/machine-learning/>
- Gartner. (2019b). Gartner IT Glossary - Predictive Modeling. <https://www.gartner.com/it-glossary/predictive-modeling/>
- Gupta, P. (2017). Decision Trees in Machine Learning. *Towards Data Science*. <https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052>
- Hollas, J. (2013). Is crowdfunding now a threat to traditional finance? *Corporate Finance Review*, 18(1), 27.
- House of Bots. (2018). Most Popular 20 Free Online Courses to Learn Deep Learning (K. Cook, Ed.). <https://www.houseofbots.com/news-detail/3620-4-most-popular-20-free-online-courses-to-learn-deep-learning>

- IArtificial.net. (2019a). Matemáticas de la Regresión Logística. <https://iartificial.net/regresion-logistica-para-clasificacion/>
- IArtificial.net. (2019b). Método del Gradiente Descendiente. <https://iartificial.net/gradiente-descendiente-para-aprendizaje-automatico/>
- IBM. (2019). Regresión Logística. https://www.ibm.com/support/knowledgecenter/es/SSLVMB_sub/statistics_mainhelp_ddita/spss/regression/idh_lreg.html
- Inzaugarat, E. (2018). Understanding Neural Networks: What, How and Why? *Towards Data Science*. <https://towardsdatascience.com/understanding-neural-networks-what-how-and-why-18ec703ebd31>
- Jin, B., Zhao, H., Chen, E., Liu, Q. & Ge, Y. (2019). Estimating the Days to Success of Campaigns in Crowdfunding: A Deep Survival Perspective. http://staff.ustc.edu.cn/~cheneh/paper_pdf/2019/Binbin-Jin-AAAI.pdf
- Kamath, R. S. & Kamat, R. K. (2018). Supervised Learning Model For Kickstarter Campaigns With R Mining. *International Journal of Information Technology, Modeling and Computing (IJITMC)*, 4(1). <https://doi.org/10.5281/zenodo.1228716>
- Kaur, H. & Gera, J. (2017). Effect of Social Media Connectivity on Success of Crowdfunding Campaigns. *Procedia Computer Science*, 122, 767-774. <https://doi.org/10.1016/j.procs.2017.11.435>
- Kickstarter. (s.f.-a). Acerca de nosotros: Kickstarter. <https://www.kickstarter.com/about?ref=global-footer>
- Kickstarter. (s.f.-b). Create something to share with others. Intro to Kickstarter for designers, makers and technologists (Diapositivas de PowerPoint).
- Kickstarter. (s.f.-c). Empieza tu proyecto. <https://www.kickstarter.com/learn?lang=es>
- Kickstarter. (s.f.-d). Financiamiento: Kickstarter. <https://www.kickstarter.com/help/handbook/funding?lang=es>
- Kickstarter. (s.f.-e). Nuestras normas. https://www.kickstarter.com/rules?ref=learn_faq
- Kickstarter. (s.f.-f). Prensa: Kickstarter. <https://www.kickstarter.com/press?ref=hello>
- Lee, H.-D. (2019). Factors affecting successful crowdfunding. *ACM International Conference Proceeding Series*, 379-382. <https://doi.org/10.1145/3306500.3306524>
- Lee, I. & Shin, Y. (2018). Fintech: Ecosystem, business models, investment decisions, and challenges. *Business Horizons*, 61(1), 35-46.
- Li, F.-F., Johnson, J. & Yeung, S. (2019). *Convolutional Neural Networks* (Publicación). Universidad Standford. http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture05.pdf
- Li, Y., Rakesh, V. & Reddy, C. K. (2016). Project Success Prediction in Crowdfunding Environments, 247-256. <https://doi.org/10.1145/2835776.2835791>

- Lichtig, B. (2015). Crowdfunding Success: The Short Story - Analyzing the Mix of Crowd-funded Ventures. *Wharton Research Scholars*, 121. https://repository.upenn.edu/wharton_research_scholars/121/
- López Briega, R. E. (2016). Redes neuronales convolucionales con TensorFlow. <https://relopezriegua.github.io/blog/2016/08/02/redes-neuronales-convolucionales-con-tensorflow/>
- López-Golán, M., Vaca Tapia, A. C., Benavides García, N. & Coronado Otavallo, X. M. (2017). Crowdfunding campaigns. Its effectiveness in audiovisual projects in the Latin American context — Las campañas de crowdfunding. Su eficacia en proyectos audiovisuales en el contexto latinoamericano. *Iberian Conference on Information Systems and Technologies, CISTI*, 1-6. <https://doi.org/10.23919/CISTI.2017.7976016>
- Machine Learning for Artists. (2019). Redes Neuronales. https://ml4a.github.io/ml4a/es/neural_networks/
- Maimon, O. & Rokach, L. (2010). *Data Mining and Knowledge Discovery Handbook* (Primera edición). Springer.
- Malik, U. (2019). *Python for NLP: Movie Sentiment Analysis using Deep Learning in Keras*. <https://stackabuse.com/python-for-nlp-movie-sentiment-analysis-using-deep-learning-in-keras/>
- MathWorks. (s.f.). Máquina de vectores de soporte (SVM). <https://la.mathworks.com/discovery/support-vector-machine.html>
- Merino, M. (2019). Conceptos de inteligencia artificial: qué es el aprendizaje por refuerzo. <https://www.xataka.com/inteligencia-artificial/conceptos-inteligencia-artificial-que-aprendizaje-refuerzo>
- Microsoft. (2018). Algoritmos de minería de datos (Analysis Services: Minería de datos). <https://docs.microsoft.com/es-mx/sql/analysis-services/data-mining/data-mining-algorithms-analysis-services-data-mining?view=sql-server-2017>
- Microsoft. (2019). Conceptos de minería de datos. <https://docs.microsoft.com/es-es/sql/analysis-services/data-mining/data-mining-concepts?view=sql-server-2017>
- Pardo, C. J. (s.f.). MACHINE LEARNING – Regresión Logística. <https://carlosjuliopardoblog.wordpress.com/2017/12/31/regresion-logistica/>
- Pennington, J., Socher, R. & Manning, C. D. (s.f.). GloVe: Global Vectors for Word Representation. <https://nlp.stanford.edu/projects/glove/>
- Poole, D., Mackworth, A. & Goebel, R. (1998). Computational Intelligence: A Logical Approach. *Oxford University Press*.
- Prabhu, R. (2018). Understanding of Convolutional Neural Network (CNN) — Deep Learning. Medium. <https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148>

- Project Management Institute. (2017). *A guide to the project management body of knowledge (PMBOK guide)* (Sexta edición). PMBOK.
- Ranjan, C. (2019). Rules-of-thumb for building a Neural Network. *Towards Data Science*.
<https://towardsdatascience.com/17-rules-of-thumb-for-building-a-neural-network-93356f9930af>
- Real Academia Española. (2020). Diccionario de la lengua española (Vigesimotercera edición).
<https://dle.rae.es/>
- Redacción Gestión. (2015). Emprendimiento en el Perú se origina más por oportunidades de negocio que por desempleo. *Diario Gestión*. <https://gestion.pe/economia/emprendimiento-peru-origina-oportunidad-negocio-desempleo-80578>
- Redacción Gestión. (2018). Perú es el tercer país con mayor cantidad de emprendimientos en fase temprana a nivel mundial. *Diario Gestión*. <https://gestion.pe/economia/peru-tercer-pais-mayor-cantidad-emprendimientos-fase-temprana-nivel-mundial-240264>
- Russell, S. & Norvig, P. (2004). *Inteligencia Artificial: Un Enfoque Moderno* (J. M. Corchado Rodríguez, F. Martín Rubio, J. M. Cadenas Figueredo, L. D. Hernández Molinero, E. Paniagua Arís, R. Fuentetaja Pinzán, M. Robledo de los Santos & R. Rizo Aldeguer, Trad.; Segunda edición). Pearson Educación, S.A. <https://luismejias21.files.wordpress.com/2017/09/inteligencia-artificial-un-enfoque-moderno-stuart-j-russell.pdf>
- Russell, S. & Norvig, P. (2009). *Inteligencia Artificial: Un Enfoque Moderno* (Tercera edición). Prentice Hall.
- Sancho Caparrini, F. (2017). *Entrenamiento de Redes Neuronales: mejorando el Gradiente Descendiente* (Publicación). Universidad de Sevilla. Sevilla. <http://www.cs.us.es/~fsancho/?e=165>
- Sancho Caparrini, F. (2018). *Clasificación Supervisada y No Supervisada* (Publicación). Universidad de Sevilla. Sevilla. <http://www.cs.us.es/~fsancho/?e=77>
- Sandoval, L. (s.f.). Barreras del Emprendedor ¿Por qué cuesta tanto hacerlo? <https://www.emprender-facil.com/es/barreras-del-emprendedor/>
- SAS Institute. (s.f.). ¿Qué es Deep Learning? https://www.sas.com/es_pe/insights/analytics/deep-learning.html
- Shafique, U. & Qaiser, H. (2014). A Comparative Study of Data Mining Process Models (KDD, CRISP-DM and SEMMA). *International Journal of Innovation and Scientific Research*, 12(1), 217-222. <http://www.ijisr.issr-journals.org/abstract.php?article=IJISR-14-281-04>
- SitioBigData.com. (2019). ReLU: Funciones de activación. <http://sitiobigdata.com/2019/06/22/relu-funciones-activacion/>

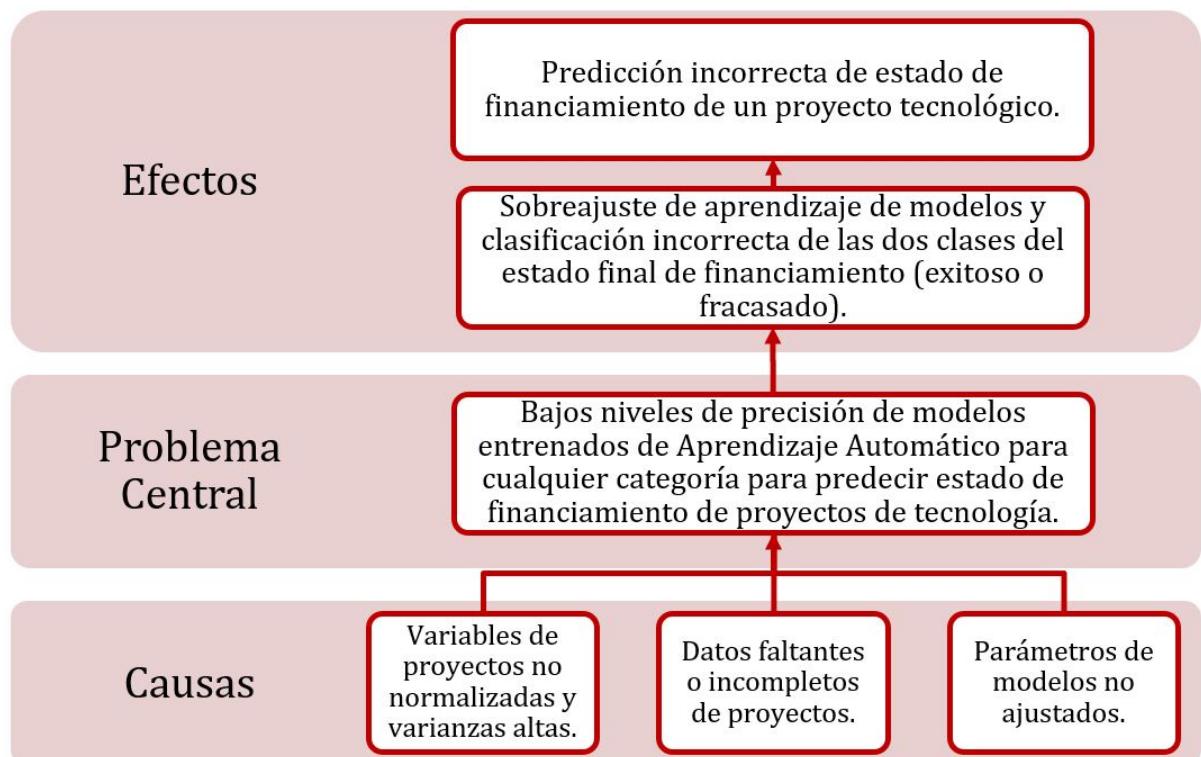
- Smola, A. J. & Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, 14(3), 199-222. <https://link.springer.com/article/10.1023/B:STCO.0000035301.49549.88>
- Solidaridad Latina. (s.f.). ¿Cómo funciona el crowdfunding en Latinoamérica? <https://solidaridadlatina.com/actualizacion/como-funciona-crowdfunding-latinoamerica/>
- Stokes, K., Clarence, E., Anderson, L. & Rinne, A. (2014). *Making sense of the UK collaborative economy*. http://collaboriamo.org/media/2014/10/making_sense_of_the_uk_collaborative_economy_14.pdf
- Sutton, R. S. & Barto, A. G. (2018). Finite Markov Decision Processes. *Reinforcement Learning: An Introduction* (Segunda edición, p. 48). MIT Press. <http://incompleteideas.net/book/RLbook2018.pdf>
- The Hustle. (2019). What are your chances of successfully raising money on Kickstarter? <https://thehustle.co/crowdfunding-success-rate>
- Tung, F.-W. & Liu, X.-Y. (2018). Understanding backers' motivations and perceptions of information on product-based crowdfunding platforms, 84-88. <https://doi.org/10.1109/ISCBI.2018.00026>
- Ullah, S. & Zhou, Y. (2020). Gender, Anonymity and Team: What Determines Crowdfunding Success on Kickstarter. *Journal of Risk and Financial Management*, 13(4), 80. <https://doi.org/10.3390/jrfm13040080>
- Universo Crowdfunding. (s.f.). ¿Qué es el crowdfunding? <https://www.universocrowdfunding.com/que-es-el-crowdfunding/>
- Viera Balanta, V. (2013). *Backpropagation explicación*. <https://www.youtube.com/watch?v=0odQ286nsIY>
- Web Robots. (2019). Kickstarter Datasets. <https://webrobots.io/kickstarter-datasets/>
- Xuefeng, L. & Zhao, W. (2018). Using Crowdfunding in an Innovative Way: A Case Study from a Chinese Crowdfunding Platform, 1-9. <https://doi.org/10.23919/PICMET.2018.8481838>
- Yu, P.-F., Huang, F.-M., Yang, C., Liu, Y.-H., Li, Z.-Y. & Tsai, C.-H. (2018). Prediction of Crowdfunding Project Success with Deep Learning, 1-8. <https://doi.org/10.1109/ICEBE.2018.00012>
- Yuan, H., Lau, R. Y. & Xu, W. (2016). The Determinants of Crowdfunding Success: A Semantic Text Analytics Approach. *Decision Support Systems*, 91, 67-76. <https://doi.org/10.1016/j.dss.2016.08.001>
- Zambrano, J. (2018). ¿Aprendizaje supervisado o no supervisado? Conoce sus diferencias dentro del machine learning y la automatización inteligente. *Medium*. <https://medium.com/@juanzambrano/aprendizaje-supervisado-o-no-supervisado-39ccf1fd6e7b>

Zhou, M., Zhang, X., Wang, A. G., Du, Q., Qiao, Z. & Fan, W. (2018). Money Talks: A Predictive Model on Crowdfunding Success Using Project Description. 20(2), 259-274.
<https://doi.org/10.1007/s10796-016-9723-1>

Anexos

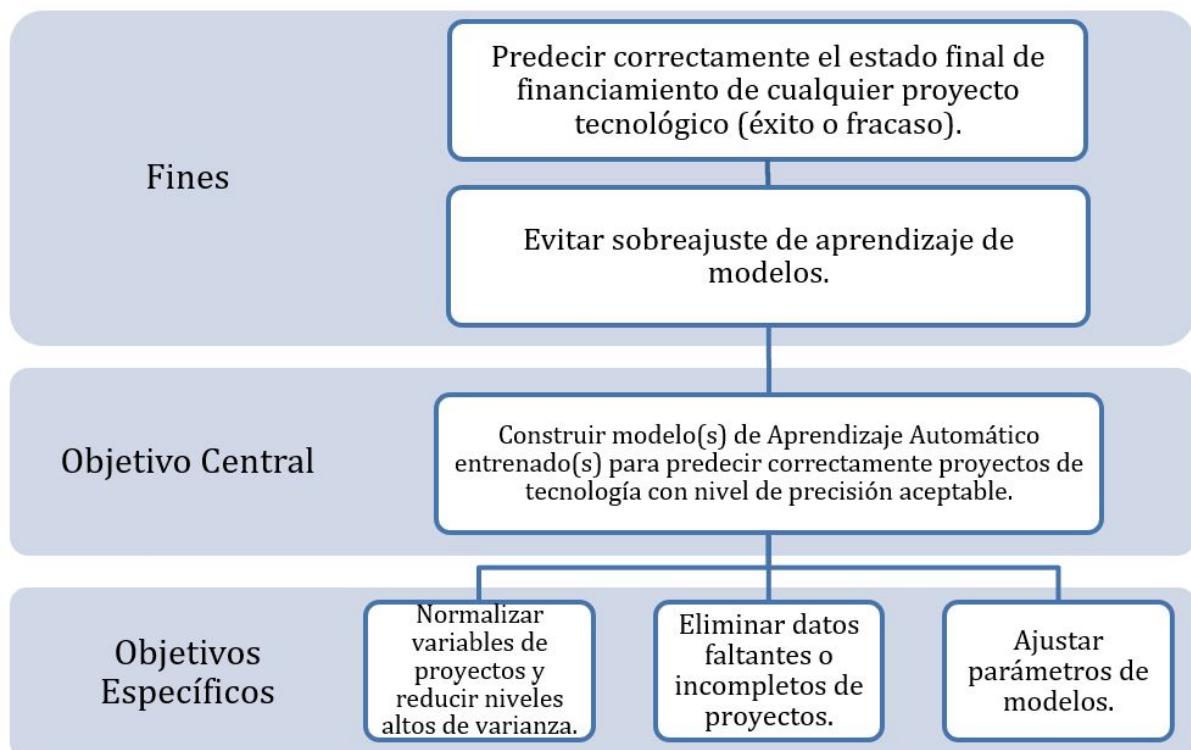
Anexo A

Árbol de Problemas



Anexo B

Árbol de Objetivos



Anexo C

Matriz de Consistencia

PROBLEMAS	OBJETIVOS	HIPÓTESIS
Problema General	Objetivo General	Hipótesis General
Bajos niveles de precisión de modelos entrenados de Aprendizaje Automático para cualquier categoría para predecir estado de financiamiento de proyectos de tecnología.	Construir modelo(s) de Aprendizaje Automático entrenado(s) para predecir correctamente proyectos de tecnología con nivel de precisión aceptable.	El modelo entrenado de Aprendizaje Automático logrará predecir correctamente proyectos de tecnología con nivel de precisión aceptable.
Problemas Específicos	Objetivos Específicos	Hipótesis Específicas
Variables de proyectos no normalizadas y varianzas altas.	Normalizar variables de proyectos y reducir niveles altos de varianza.	Las variables de los proyectos descargados se normalizarán y se reducirán los niveles altos de varianza.
Datos faltantes o incompletos de proyectos.	Eliminar datos faltantes o incompletos de proyectos.	Los datos faltantes o incompletos de los proyectos serán eliminados.
Parámetros de modelos no ajustados.	Ajustar parámetros de modelos.	Los parámetros de los modelos usados serán ajustados.
Sobreajuste de aprendizaje de modelos y clasificación incorrecta de las dos clases del estado final de financiamiento (exitoso o fracasado).	Evitar sobreajuste de aprendizaje de modelos.	Se evitará el sobreajuste de aprendizaje de modelos para clasificar correctamente las dos clases del estado final de financiamiento.
Predictión incorrecta de estado de financiamiento de un proyecto tecnológico.	Predecir correctamente el estado final de financiamiento de cualquier proyecto tecnológico (éxito o fracaso).	El estado final de financiamiento de cualquier proyecto tecnológico será predicho correctamente.

Anexo D

Resumen de Papers investigados

Tipo	Nº	Título	Autor	Año	País	Fuente
Problema	1	Copper price estimation using bat algorithm	Dehghani Bogdanovic	2018	United Kingdom	Resources Policy
	2	Alternative techniques for forecasting mineral commodity prices	Cortez, Say-dam, Coulton, Sammut	2018	Netherlands	International Journal of Mining Science and Technology
Propuesta	3	Prediction of the crude oil price thanks to natural language processing applied to newspapers	Trastour, Ge-nin, Morlot	2016	USA	Standfort University ML repository
	4	Stock Price Prediction Using Deep Learning	Tipirisetty	2018	USA	Master's Theses San Jose State University
	5	Deep Learning for Stock Prediction Using Numerical and Textual Information	Akita, R., Yoshihara, A., Matsubara, T., Uehara, K.	2016	USA	2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)
Técnica	6	Stock Prices Prediction using the Title of Newspaper Articles with Korean Natural Language Processing	Yun, Sim, Seok	2019	Japan	2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)
	7	A Method of Optimizing LDA Result Purity Based on Semantic Similarity	Jingrui, Z., Qinglin, W., Yu, L., Yuan, L.	2017	China	2017 32nd Youth Academic Annual Conference of Chinese Association of Automation (YAC)
	8	Qualitative Stock Market Predicting with Common Knowledge Based Nature Language Processing: A Unified View and Procedure	Rao, D., Deng, F., Jiang, Z., Zhao, G.	2015	USA	2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics
	9	Fuzzy Bag-of-Words Model for Document Representation	Zhao, R., Mao, K.	2018	USA	IEEE Transactions on Fuzzy Systems (Volume: 26 , Issue: 2 , April 2018)