



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

Optimización de Cobertura de Redes Inalámbricas: Predicción de Ubicaciones de Puntos de Acceso (APs) en Diferentes Planos Mediante Inteligencia Artificial

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

Jeferson Joseph Sandoval Díaz
Asesor: Marks Arturo Calderón Niquin

Lima, julio de 2024

Esta tesis denominada:

PREDICCIÓN DEL ESTADO DE FINANCIAMIENTO DE PROYECTOS DE
TECNOLOGÍA EN SITIO WEB DE CROWDFUNDING KICKSTARTER MEDIANTE
MODELO DE APRENDIZAJE PROFUNDO MULTIMODAL

ha sido aprobada.

.....
Eber Joseph Ballón Álvarez (Jurado Presidente)

.....
Wilfredo Mamani Ticona (Jurado)

.....
Pedro Nelson Shiguihara Juárez (Jurado)

Universidad ESAN
2021

PREDICCIÓN DEL ESTADO DE FINANCIAMIENTO DE PROYECTOS DE
TECNOLOGÍA EN SITIO WEB DE CROWDFUNDING KICKSTARTER MEDIANTE
MODELO DE APRENDIZAJE PROFUNDO MULTIMODAL

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 en el año 2017, y en el Summer School of Machine Learning en Rusia en el año 2020 luego de obtener mi grado de bachiller) y formar parte de la gran familia UE.

Por todos ellos, simplemente gracias.

Índice general

Resumen	1
Introducción	3
Capítulo I: Planteamiento del Problema	5
1.1 Descripción de la Realidad Problemática	5
1.2 Formulación del Problema	8
1.2.1 Problema General	8
1.2.2 Problemas Específicos	8
1.3 Objetivos de la Investigación	8
1.3.1 Objetivo General	8
1.3.2 Objetivos Específicos	8
1.4 Hipótesis	9
1.4.1 Hipótesis General	9
1.4.2 Hipótesis Específicas	9
1.5 Justificación de la Investigación	10
1.5.1 Teórica	10
1.5.2 Práctica	10
1.5.3 Metodológica	11
1.6 Delimitación del Estudio	11
1.6.1 Espacial	11
1.6.2 Temporal	11
1.6.3 Conceptual	11
Capítulo II: Marco Teórico	12
2.1 Antecedentes de la investigación	12
2.2 Bases Teóricas	39
2.2.1 Inteligencia Artificial	39
2.2.2 Aprendizaje Automático	40
2.2.3 Aprendizaje Profundo	45
2.2.4 Aprendizaje Profundo Multimodal	46
2.2.5 Modelo Predictivo	50
2.2.6 Minería de Datos	50
2.2.7 Metodologías de Minería de Datos	51
2.2.8 Técnicas de Minería de Datos	55
2.2.9 Procesamiento del Lenguaje Natural	75

2.3	Marco Conceptual	87
2.3.1	Crowdfunding	87
2.3.2	Kickstarter	89
2.3.3	Proyecto	90
2.3.4	Campaña	90
Capítulo III: Metodología de la Investigación		93
3.1	Diseño de la investigación	93
3.1.1	Tipo de la investigación	93
3.1.2	Enfoque de la investigación	93
3.1.3	Población	93
3.1.4	Muestra	93
3.1.5	Operacionalización de Variables	94
3.2	Técnicas de recolección de datos	95
3.3	Técnicas para el procesamiento y análisis de la información	96
3.3.1	Metodología de implementación de la solución	96
3.3.2	Metodología para la medición de resultados	120
3.4	Cronograma de actividades y presupuesto	126
Capítulo IV: Desarrollo del Experimento		128
4.1	Comprensión del negocio	128
4.2	Comprensión de los datos	129
4.3	Preparación de los datos	146
Capítulo V: Análisis y Discusión de Resultados		152
5.1	Modelamiento	152
5.2	Evaluación	159
5.3	Despliegue	169
Capítulo VI: Conclusiones y Recomendaciones		172
6.1	Conclusiones	172
6.2	Recomendaciones	174
Referencias		176
Anexos		186
A	Árbol de Problemas	187
B	Árbol de Objetivos	188
C	Matriz de Consistencia	189

D	Comparación de metodologías de antecedentes	192
E	Comparación de objetivos específicos de antecedentes	196
F	Parámetros para modelo predictivo de Metainformación	203
G	Parámetros para modelo predictivo de Descripción	204
H	Parámetros para modelo predictivo de Comentarios	205
I	Parámetros para modelo de Aprendizaje Profundo Multimodal	206

Índice de Figuras

Figura 1.	Quejas de Clientes en el foro de Cisco	7
Figura 2.	Problemas de uso de Access Points	7
Figura 3.	Extensión de Google Chrome creada para predecir el estado del proyecto .	13
Figura 4.	Ejemplos de frases del Top 100 y sus respectivos pesos que señalan si un proyecto será o no financiado	14
Figura 5.	Curvas ROC del modelo propuesto, el de base y el convencional	15
Figura 6.	Evaluación de la performance de los modelos construidos en distintas etapas de tiempo	17
Figura 7.	Curva ROC para cada modelo utilizado por el autor	18
Figura 8.	Comparación de resultados de conjuntos de datos de distintas características de proyectos considerando incluir o no proyectos fracasados	20
Figura 9.	Marco de trabajo de analítica textual de los autores	21
Figura 10.	Performance de ambas bases de datos con distintas métricas	21
Figura 11.	Exactitud del modelo SVM para cada categoría	23
Figura 12.	Performance estadística del modelo predictivo de los autores	24
Figura 13.	Gráfico de dispersión de cada modelo evaluado según su exactitud	26
Figura 14.	Gráfico de historial de entrenamiento para subconjuntos de entrenamiento y validación	27
Figura 15.	Promedio estimado de la exactitud de predicción vs tiempo estimado en días	29
Figura 16.	Arquitectura SMP de los autores	30
Figura 17.	Arquitectura de modelo de Aprendizaje Profundo Multimodal de los autores	32
Figura 18.	Palabras clave seleccionadas que afectan el ratio de éxito según modelo SVM-RFE	33
Figura 19.	Nube de palabras clave de 3 y 4 gramas usando el algoritmo TextRank . . .	35
Figura 20.	Arquitectura del sistema propuesto para predicción de temas y recomendaciones optimizadas	36
Figura 21.	Mapas de proyecciones de región de éxito o fracaso generado por K-means	38
Figura 22.	Distribución de clústers según ratio de éxito por categoría	38
Figura 23.	Ejemplo de algoritmos de regresión y clasificación	41
Figura 24.	Algoritmo de K Vecinos más cercanos con pesos ponderados	42
Figura 25.	Funcionamiento del algoritmo de K medias	43
Figura 26.	Componentes del Aprendizaje por Refuerzo	44
Figura 27.	Ambiente del videojuego Pacman	44
Figura 28.	Diferencia entre Aprendizaje Automático y Aprendizaje Profundo	45
Figura 29.	Ejemplo de modelo multimodal de imágenes y texto	46

Figura 30.	Ejemplo de modelo multimodal de señal de electroencefalografía y de ojo	47
Figura 31.	Ejemplo de aprendizaje ensamblado apilado	48
Figura 32.	Ejemplo de modelo apilado separado	48
Figura 33.	Ejemplo de modelo apilado integrado	49
Figura 34.	Fases de la metodología CRISP-DM	51
Figura 35.	Fases de la metodología SEMMA	52
Figura 36.	Fases de la metodología KDD	53
Figura 37.	Modelo para representar una neurona propuesto por McCulloch y Pitts (1943)	55
Figura 38.	Nodos con funciones de activación umbral en forma de puertas lógicas	57
Figura 39.	Función de activación sigmoide	57
Figura 40.	Ilustración del algoritmo gradiente descendiente	58
Figura 41.	Actualización de pesos W con el algoritmo	59
Figura 42.	Capa oculta simple MLP con propagación hacia atrás	60
Figura 43.	Redes neuronales de ejemplo	60
Figura 44.	Función de activación tangente hiperbólica	62
Figura 45.	Función de activación puramente lineal	62
Figura 46.	Función de activación ReLU	63
Figura 47.	Ejemplo de perceptrón simple	64
Figura 48.	Ejemplo de perceptrón multicapa	64
Figura 49.	Ejemplo de red neuronal convolucional	65
Figura 50.	Modelos de redes neuronales que inspiraron a la CNN	65
Figura 51.	Ejecución de la convolución en una entrada	66
Figura 52.	Generación de una nueva imagen a partir de filtros	67
Figura 53.	Secuencia de varias capas convolucionales	67
Figura 54.	Extracción de características a partir de convoluciones	68
Figura 55.	Ejemplo de matriz de imagen de entrada y un filtro	68
Figura 56.	Dimensiones de una entrada y un filtro	69
Figura 57.	Paso de 2 píxeles por parte de un filtro	69
Figura 58.	Aplanado de matrices luego de agrupar la capa	70
Figura 59.	Arquitectura completa de una CNN	71
Figura 60.	Ejemplo de red neuronal recurrente	71
Figura 61.	Hiperplano con dos clases separadas por una distancia m	72
Figura 62.	Ejemplo de separación de 2 clases	73
Figura 63.	Aplicación de un kernel para transformar el espacio de los datos	73
Figura 64.	Ejemplo del algoritmo de árbol de decisión	74
Figura 65.	Arquitectura de modelo CNN con 2 canales para una oración de ejemplo	76
Figura 66.	Diferencias entre convoluciones según su dimensión	77

Figura 67.	Arquitectura de modelo CNN para clasificación de oraciones	78
Figura 68.	Arquitectura de una LSTM	79
Figura 69.	Representación de arquitectura BiRNN de la palabra <i>jumped</i> en la oración .	80
Figura 70.	Comparación entre arquitecturas LSTM y GRU	81
Figura 71.	Arquitectura de un modelo Seq2seq	82
Figura 72.	Ejemplo de funcionamiento de una capa de incrustación	83
Figura 73.	Incrustaciones de palabras por Word2Vec	84
Figura 74.	Incrustaciones de palabras por GloVe	85
Figura 75.	Ejemplo de funcionamiento de bolsa de palabras	86
Figura 76.	Ejemplo de funcionamiento de TF-IDF	86
Figura 77.	Ejemplo de proyecto vigente en Kickstarter	89
Figura 78.	Flujograma de la recolección de conjuntos finales de datos	95
Figura 79.	Metodología de la investigación	99
Figura 80.	Marco de trabajo del prototipo final	100
Figura 81.	Proceso de obtención y preparación de datasets iniciales de Metainformación	104
Figura 82.	Proceso resumido de generación de conjunto final de Metainformación . .	104
Figura 83.	Proceso de extracción de la descripción de un proyecto	105
Figura 84.	Proceso de extracción de comentarios de un proyecto	106
Figura 85.	Proceso de limpieza de conjunto de datos de descripciones	108
Figura 86.	Procesos de vectorización y creación de matriz incrustaciones de palabras .	109
Figura 87.	Proceso de operación del prototipo del sistema	120
Figura 88.	Descripción de resultados de modelo descriptivo de ejemplo	122
Figura 89.	Comparación de tres resultados de la curva AUC en el modelo	123
Figura 90.	Cronograma de actividades de la investigación	126
Figura 91.	Vista del website Web Robots (visitado en agosto del 2019)	129
Figura 92.	Tamaño de conjunto de datos al corte de Julio 2019	130
Figura 93.	Visualización del archivo de metainformación subido a Kaggle	131
Figura 94.	Función para extraer textos de modalidad de descripción	132
Figura 95.	Ejecución de la función de extracción de descripciones y almacenamiento .	133
Figura 96.	Visualización del archivo de descripción subido a Kaggle	133
Figura 97.	Función para extraer textos de modalidad de comentarios	134
Figura 98.	Ejecución de la función de extracción de comentarios y almacenamiento .	135
Figura 99.	Instancias lanzadas en paralelo para la extracción de comentarios	135
Figura 100.	Visualización del archivo de comentarios subido a Kaggle	136
Figura 101.	Distribución de proyectos tecnológicos según su estado	137
Figura 102.	Evolución de cantidad de proyectos tecnológicos por año	137
Figura 103.	Evolución de proyectos tecnológicos, por su estado y año	138

Figura 104.	Distribución de las variables categóricas de Metainformación	138
Figura 105.	Diagrama de caja y bigote de patrocinadores	139
Figura 106.	Diagrama de caja y bigote de meta	140
Figura 107.	Diagrama de caja y bigote de monto patrocinado	140
Figura 108.	Diagrama de caja y bigote de duración	141
Figura 109.	Matriz de correlaciones entre variables independientes	141
Figura 110.	Gráfico de dispersión de correlaciones entre variables independientes . . .	142
Figura 111.	Distribución de proyectos por presencia de descripciones y estado final .	143
Figura 112.	Nube de palabras de descripciones	143
Figura 113.	Distribución de proyectos por presencia de comentarios y estado final .	144
Figura 114.	Distribución de comentarios en proyectos exitosos y fracasados	145
Figura 115.	Nube de palabras de comentarios más frecuentes	145
Figura 116.	Matriz de correlaciones entre variables independientes considerando adicionales	146
Figura 117.	Función para dividir base de datos en subconjuntos de entrenamiento y prueba	147
Figura 118.	Función para normalizar variables	148
Figura 119.	Nube de palabras de descripciones posterior a la limpieza de texto	149
Figura 120.	Proceso de representación de palabras en vectores codificados	149
Figura 121.	Proceso de creación de matriz de incrustaciones de palabras	150
Figura 122.	Nube de palabras de comentarios posterior a la limpieza de texto	151
Figura 123.	Arquitectura de modelo MLP para la metadata	153
Figura 124.	Arquitectura de modelo CNN para las descripciones	154
Figura 125.	Arquitectura de modelo RNN para los comentarios	156
Figura 126.	Arquitectura del modelo apilado final The Hydra	158
Figura 127.	Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo MLP de metadata con 100 épocas	160
Figura 128.	Matriz de confusión para el modelo de metadata	161
Figura 129.	Área bajo la curva ROC de modelo de metadata	162
Figura 130.	Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo CNN de descripciones con 100 épocas	162
Figura 131.	Matriz de confusión para el modelo de descripciones	163
Figura 132.	Área bajo la curva ROC de modelo de descripciones	164
Figura 133.	Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo RNN de comentarios con 50 épocas	164
Figura 134.	Matriz de confusión para el modelo de comentarios	165
Figura 135.	Área bajo la curva de modelo de comentarios	166

Figura 136. Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo apilado con 200 épocas	166
Figura 137. Matriz de confusión para el modelo apilado	167
Figura 138. Área bajo la curva de modelo apilado	168
Figura 139. Proyecto consultado para la demostración. Captura de pantalla: 15/02/21 . .	170
Figura 140. Campaña del proyecto consultado. Captura de pantalla: 15/02/21	170
Figura 141. Variables extraídas por modalidad del proyecto consultado	171
Figura 142. Resultado de predicción de The Hydra para el proyecto consultado	171

Índice de Tablas

Tabla 1.	Cuadro comparativo entre características de las tres metodologías	54
Tabla 2.	Matriz de operacionalización de variables	94
Tabla 3.	Cuadro comparativo para la selección de la metodología	96
Tabla 4.	Actividades de fase Comprensión del negocio	101
Tabla 5.	Actividades de fase Comprensión de los datos	102
Tabla 6.	Actividades de fase Preparación de los datos	107
Tabla 7.	Actividades de fase Modelamiento	110
Tabla 8.	Actividades de fase Evaluación	116
Tabla 9.	Actividades de fase Despliegue	119
Tabla 10.	Matriz de confusión	121
Tabla 11.	Presupuesto de los costos personales del autor	127
Tabla 12.	Presupuesto de los costos de las herramientas para el proyecto	127
Tabla 13.	Diccionario de datos del dataset final de Metainformación	131
Tabla 14.	Potenciales combinatorias de variables de metainformación	147
Tabla 15.	Diccionario de datos del conjunto final entrenado	157
Tabla 16.	Exactitud de los conjuntos de datos de validación para las 8 combinaciones .	160
Tabla 17.	Informe de clasificación para el modelo de metadata	161
Tabla 18.	Informe de clasificación para el modelo de descripciones	163
Tabla 19.	Informe de clasificación para el modelo de comentarios	165
Tabla 20.	Informe de clasificación para el modelo apilado	167
Tabla 21.	Comparación de resultados de modelos propuestos con antecedentes	168

Índice de Ecuaciones

Ecuación 1	Cálculo de pesos para K-NN mediante ponderación de sus distancias	42
Ecuación 2	Fórmula alternativa del algoritmo K-NN mediante sumatoria de pesos	42
Ecuación 3	Fórmula del algoritmo k-means	43
Ecuación 4	Fórmula del cálculo del valor de un nodo i	56
Ecuación 5	Fórmula de una función de activación g para la salida del nodo	56
Ecuación 6	Fórmula de la función de activación sigmoide	56
Ecuación 7	Fórmula de función de coste de una regresión logística	58
Ecuación 8	Actualización de pesos W mediante gradiente descendiente	59
Ecuación 9	Fórmula del algoritmo de propagación hacia atrás	60
Ecuación 10	Cálculo del error cometido en una red neuronal	61
Ecuación 11	Actualización de pesos mediante propagación hacia atrás	61
Ecuación 12	Cálculo de errores de nodos usando pesos actualizados	61
Ecuación 13	Fórmula de la función de activación tangente hiperbólica	62
Ecuación 14	Fórmula de la función de activación puramente lineal	62
Ecuación 15	Fórmula de la función de activación ReLU	63
Ecuación 16	Fórmula matemática de la convolución	66
Ecuación 17	Cálculo del volumen del mapa de activación	68
Ecuación 18	Cálculo del tamaño de la imagen reducida	69
Ecuación 19	Cálculo del tamaño de la imagen reducida con bordes rellenos con ceros	70
Ecuación 20	Ecuación del hiperplano para clasificar dos clases	73
Ecuación 21	Fórmula para determinar un estado en una S-RNN	79
Ecuación 22	Fórmula de TF-IDF	86
Ecuación 23	Fórmula para calcular la exactitud	122
Ecuación 24	Fórmula para calcular la precisión	122
Ecuación 25	Fórmula para calcular el área bajo la curva ROC	123
Ecuación 26	Fórmula para calcular la sensibilidad	124
Ecuación 27	Fórmula para calcular el puntaje F1	124
Ecuación 28	Fórmula del escalador Mínimo Máximo	148

Resumen

Desde la aparición del crowdfunding, muchas personas, especialmente emprendedoras, han tenido la oportunidad de presentar sus proyectos al público para conseguir su financiamiento. Durante el período 2009-2019, el 37 % de proyectos de Kickstarter, una de las plataformas de financiamiento colectivo más populares, alcanzó ser financiado exitosamente. En trabajos anteriores relacionados con el tema estudiado, se utilizaron distintas metodologías y técnicas de Inteligencia Artificial, considerando todas las categorías existentes en esta plataforma para crear modelos predictivos. Sin embargo, este ratio solo alcanza el 20 % para Tecnología. El objetivo de esta investigación fue predecir el estado de financiamiento de proyectos de tecnología en el sitio web de crowdfunding Kickstarter mediante un modelo de Aprendizaje Profundo Multimodal. Siguiendo la metodología CRISP-DM, se implementó un modelo ensamblado de otros modelos de Aprendizaje Profundo por cada modalidad considerada: un Perceptrón Multicapa para la Metainformación, una Red Neuronal Convolutacional para la descripción y un modelo LSTM Bidireccional para los comentarios de los patrocinadores. Se utilizó información de más de 27 mil proyectos de tecnología en Kickstarter entre 2009 y 2019. Al evaluarse por las métricas de clasificación seleccionadas y compararlo contra la línea de base y cada modalidad independiente, la propuesta superó a los otros modelos en cada métrica, alcanzando un valor de 93 % de AUC, la de mejor desempeño. Se logró no solamente desarrollar un modelo predictivo para un problema muy estudiado bajo una nueva perspectiva sino también contribuir con insights y un prototipo analítico para trabajos futuros y apoyo a creadores de proyectos.

Palabras claves: financiamiento colectivo, proyecto, Aprendizaje Profundo Multimodal, Perceptrón Multicapa (MLP), Red Neuronal Convolutacional (CNN), Red Neuronal Recurrente (RNN).

Abstract

Since the advent of crowdfunding, many people, especially entrepreneurs, have had the opportunity to present their projects to the public in order to fund them. During the 2009-2019 period, 37% of Kickstarter projects, one of the most popular crowdfunding platforms, were successfully funded. In previous works related to the subject studied, different Artificial Intelligence methodologies and techniques were used, considering all the existing categories in this platform to develop predictive models of this problem. However, this ratio only reaches 20% for Technology. The objective of this research was to predict the funding state of technology projects on the Kickstarter crowdfunding website using a Multimodal Deep Learning model. Following the CRISP-DM methodology, an assembled model of other Deep Learning models was implemented for each modal considered: a Multilayer Perceptron for Meta-information, a Convolutional Neural Network for the description and a Bidirectional LSTM model for the comments of the sponsors. Information from more than 27 thousand technology projects on Kickstarter between 2009 and 2019 was used. When evaluated by the selected classification metrics and compared against the baseline and each independent modal, the proposal surpassed the other models in each metric, reaching a value of 93% of AUC, the one with the best performance. It was possible not only to develop a predictive model for a well-studied problem from a new perspective, but also to contribute insights and an analytical prototype for future work and support to project creators.

Keywords: crowdfunding, project, Multimodal Deep Learning, Multilayer Perceptron (MLP), Convolutional Neural Network (CNN), Recurrent Neural Network (RNN).

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 nuevas perspectivas de estudios de estos casos para encontrar la clave que descifre las variables de éxito.

A diferencia de estudios previos, en donde casi la totalidad de antecedentes engloba todas las categorías existentes en Kickstarter, uno de los sitios web de crowdfunding más populares, la principal motivación para realizar esta investigación fue de proponer un nuevo enfoque para resolver el problema de predecir si un proyecto en esta plataforma será financiado o no durante su campaña, considerando solamente aquellos de la categoría Tecnología, cuyo ratio de 20 % de éxito representa la más baja de todas. Bajo este escenario, se buscó poder desarrollar un modelo predictivo con estas condiciones iniciales desfavorables sin que se afecte su desempeño por el comportamiento de otras categorías con mejores ratios de éxito. El nuevo aporte consistió en diseñar un modelo de Aprendizaje Multimodal Profundo utilizando información de las principales variables cuantitativas y el contenido textual de la campaña, tanto la descripción del proyecto redactada por el creador como los comentarios recibidos por los patrocinadores acerca del mismo, es decir, la interacción social directa entre los stakeholders.

En el Capítulo I, se describe la realidad problemática de la investigación y el entorno en que se desenvuelve. Asimismo, se formulan los problemas, objetivos, hipótesis, justificación y delimitación del estudio.

En el Capítulo II, se detallan los antecedentes principales que fueron considerados al estudiar el problema de clasificación desde distintas perspectivas y estrategias. A continuación, se expone la base teórica en donde son abordados los conceptos técnicos que fueron aplicados, desde los fundamentos de la Inteligencia Artificial hasta métodos que forman parte de ella. El capítulo cierra con el marco conceptual en donde se explican términos relacionados al financiamiento colectivo y cómo funciona su entorno.

En el Capítulo III, se describe el diseño, tipo, enfoque, población, muestra y la operacionalización de las variables de la investigación. Luego se expone la metodología de implementación de la solución, desde el criterio de su elección hasta la puesta en marcha de las fases y actividades incurridas, así como los entregables comprometidos. Después, se explaya la metodología para la medición de resultados de la implementación. Finalmente, el capítulo concluye con el detalle del cronograma de actividades y presupuesto.

En el Capítulo IV, se detalla el desarrollo de la solución, desde cada modelo construido según la modalidad correspondiente hasta el modelo apilado final, y las tareas ejecutadas que se plantearon efectuar en el anterior capítulo.

En el Capítulo V, se analizan y discuten los resultados obtenidos de los experimentos, desde el tiempo de ejecución hasta los valores calculados por cada métrica de clasificación aplicada. Aquí también se comparan los resultados del modelo propuesto con la línea de base.

En el Capítulo VI, se comentan las conclusiones principales de toda la investigación y se realizan recomendaciones en base a las fortalezas, oportunidades de mejora y trabajos que, en el futuro, podrían ser desarrollados.

La investigación concluye con las referencias utilizadas en el trabajo y los anexos que complementan con mayor información dentro de cada enunciado citado.

Capítulo I: Planteamiento del Problema

1.1 Descripción de la Realidad Problemática

En la actualidad, cada vez es más necesario el uso de la conectividad a internet como método de interacción entre los usuarios, empresas e instituciones educativas, convirtiéndose en un instrumento imprescindible para las actividades relacionadas a ella. En la mayoría de las empresas no se cuenta con una cobertura de la red Wi-Fi total que garantice el acceso a internet y es allí donde surgen muchos inconvenientes tanto para el envío de trabajos, realización de consultas, entre otras actividades en las que la red sería de gran ayuda.

Lograr encontrar la ubicación idónea de Aps Indoor para lograr la cobertura total de un lugar es un proceso iterativo que consume mucho tiempo, que requiere múltiples rondas de refinamientos. Un especialista de TI esboza un diseño, evalúa, ajusta y repite los ciclos hasta estar satisfecho con un diseño dentro de un presupuesto de tiempo dado. Desafortunadamente, diseñar un plano de cobertura de red efectivo solo es posible mediante especialistas de TI o Redes, donde gran parte de empresas hacen su propio diseño personalizado menos efectivo debido al costo. La generación automática de planos de cobertura de red con las ubicaciones de Aps Indoor tendrá un tremendo impacto en las industrias de bienes raíces/construcción, redes y TI de billones de dólares.

En los últimos años, ha habido un aumento en la demanda de redes inalámbricas entre los usuarios, gracias a los beneficios que ofrecen en términos de movilidad y costos de implementación más bajos. Dentro de las redes inalámbricas, se encuentran las WLAN (Redes de Área Local Inalámbricas), las cuales son comúnmente utilizadas en entornos cotidianos como hogares, oficinas y instituciones educativas, entre otros lugares. Estas redes suelen estar compuestas principalmente por dispositivos concentradores conocidos como Puntos de Acceso (AP), los cuales permiten a los usuarios conectarse de forma inalámbrica a la red, cumpliendo una función similar a la de un Switch en una red cableada. Sin embargo, a pesar de la capacidad de los AP para establecer conexiones inalámbricas, la distancia efectiva entre el usuario y el AP es limitada (generalmente inferior a 100 metros), debido a la potencia de la señal de transmisión y a posibles obstáculos en el entorno que puedan afectar la señal. Debido a la creciente necesidad de conectividad inalámbrica por parte de los usuarios, la cantidad de Puntos de Acceso en uso está en constante aumento.

Según una investigación realizada por ABI Research, compañía que asesora a fabricantes del mundo de los semiconductores sin cable, en 2026 se llegará al despliegue total de Wi-Fi 6, el cual se acelera rápidamente mucho más allá de los dispositivos Wi-Fi insignia, cada vez más dispositivos admiten la banda de 6 GHz está integrado en un número cada vez mayor de

dispositivos de consumo convencionales. (Zignani et al., 2022) Ello es algo que se debe tener, en cuenta debido al rápido avance de la tecnología y a las consecuencias que estas tendrán.

En un artículo del diario El País, se destaca un problema creciente relacionado con la conectividad Wi-Fi en los hogares. A medida que más dispositivos se conectan a las redes inalámbricas, la infraestructura existente se ve sometida a una presión cada vez mayor. Además, muchos hogares y empresas aún utilizan enrutadores y puntos de acceso antiguos que no pueden manejar la cantidad de dispositivos conectados, sumando la falta de actualización de la infraestructura contribuye al problema.(El País, 2023)

Según el Instituto Nacional de Estadística e Informática (INEI), en el primer trimestre de 2022, el 95,0 % de los hogares del país tenían al menos un servicio de Tecnología de Información y Comunicación (TIC). Este indicador muestra un crecimiento de 0,2 y 1,9 puntos porcentuales, al compararlo con el mismo trimestre de los años 2021 y 2019, respectivamente. (INEI, 2022)

En el ámbito de las redes inalámbricas, especialmente en entornos corporativos y de alta demanda, como los clientes de Cisco, las quejas sobre la conectividad y el rendimiento de las WLAN (Wireless Local Area Network) son una constante preocupación. A medida que aumenta la dependencia de estas redes para la comunicación, colaboración y operaciones empresariales, también lo hacen los desafíos técnicos y las expectativas de los usuarios. Uno de los problemas recurrentes es la cobertura inalámbrica insuficiente, que se manifiesta en áreas muertas donde la señal es débil o inexistente. Esto puede deberse a la ubicación subóptima de los puntos de acceso (AP) o a interferencias externas que obstaculizan la propagación de la señal. Los clientes de Cisco a menudo expresan su frustración por tener que moverse dentro de un espacio para obtener una señal sólida, lo que afecta negativamente la productividad y la experiencia del usuario. (Foro de Cisco, 2023)

Además de este problema, en muchos casos los usuarios optan por adquirir un nuevo punto de acceso para mejorar el medio ambiente, pero esto puede tener consecuencias negativas. La compatibilidad entre los dispositivos instalados no siempre está garantizada y también puede haber incompatibilidades a nivel de switch y VPN. Identificar y resolver estos problemas de una manera que tenga o tenga menos impacto en las operaciones requiere un sólido conocimiento técnico en el campo de las tecnologías de la información (TI), ya sea dentro del negocio o mediante el uso de servicios tecnológicos especializados. El procesamiento lento genera, entre otras desventajas, la necesidad de retrabajo, demoras en la emisión de informes y documentos y costos adicionales relacionados con las horas extras. (Napit, 2017)

A lo anterior, podemos sumar que los errores de configuración en switches, enrutadores y puntos de acceso también pueden tener un impacto negativo en la red inalámbrica, provo-



Figura 1. Quejas de Clientes en el foro de Cisco

Fuente: The Hustle (2019). *Cobertura de red ineficiente*

cando, por ejemplo, ralentizaciones y caídas de conexiones. Por lo tanto, reconfigurar estos dispositivos puede ayudar a abordar los desafíos de la red. Una solución eficaz a esta situación es realizar un análisis detallado del entorno y las estructuras existentes. Con esta evaluación, resulta más fácil determinar la ubicación adecuada para conectar los elementos, evaluar los requisitos de los nuevos componentes y garantizar que sigan funcionando de manera óptima. (Napit, 2017)

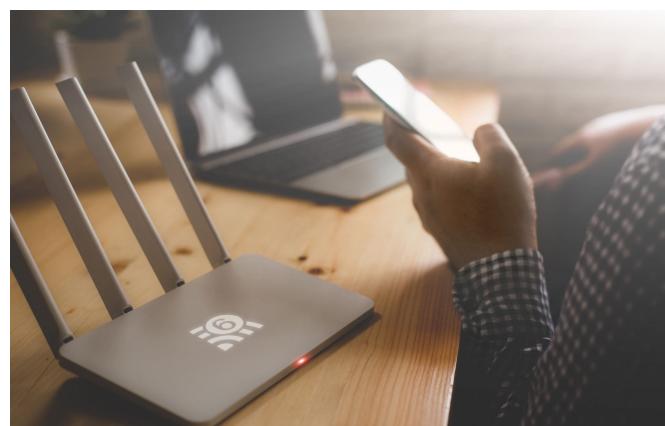


Figura 2. Problemas de uso de Access Points

Fuente: The Hustle (2019). *Problema con el funcionamiento ideal del Access Point*

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

PG: ¿Es posible predecir las ubicaciones de Puntos de Acceso (APs) Indoor en Diferentes Planos mediante Inteligencia Artificial Generativa?

1.2.2 Problemas Específicos

- PE1: ¿Cómo se puede modelar de manera efectiva la distribución espacial de usuarios y obstáculos en un entorno interior para predecir la cobertura inalámbrica?
- PE2: ¿Qué enfoques de inteligencia artificial generativa, como redes generativas adversarias (GANs) o modelos de generación de texto, son más adecuados para generar ubicaciones óptimas de APs en entornos interiores?
- PE3: ¿Qué estrategias de optimización se pueden aplicar para ajustar dinámicamente los recursos, como la potencia de transmisión y la capacidad de los APs, con el fin de mejorar la calidad de la cobertura inalámbrica?
- PE4: ¿Cómo se pueden equilibrar eficientemente los recursos asignados a diferentes APs para garantizar una cobertura uniforme y una distribución equitativa de la capacidad de red?

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

OG: Desarrollar y aplicar técnicas de inteligencia artificial generativa para predecir las ubicaciones óptimas de puntos de acceso (APs) en diferentes planos de un entorno interior, con el fin de optimizar la cobertura de redes inalámbricas y mejorar la conectividad y calidad del servicio para los usuarios finales.

1.3.2 Objetivos Específicos

- OE1: Desarrollar un algoritmo que pueda simular con precisión la cobertura de redes inalámbricas en interiores

- OE2: Implementar un modelo generativo capaz de proponer automáticamente ubicaciones eficientes para los Aps
- OE3: Desarrollar algoritmos de optimización que ajusten dinámicamente los parámetros de los APs para mejorar la eficiencia del sistema.
- OE4: Desarrollar e implementar un algoritmo de balanceo de carga dinámico para optimizar la asignación de recursos entre los diferentes puntos de acceso (APs) de una red inalámbrica.

1.4 Hipótesis

1.4.1 Hipótesis General

HG: La aplicación de técnicas de inteligencia artificial generativa para la predicción de la ubicación de puntos de acceso (Aps) en planes distribuidos en zonas interiores supone una mejora significativa en la cobertura y calidad del servicio de los recursos sin datos.

1.4.2 Hipótesis Específicas

- HE1: Los algoritmos de inteligencia artificial generativa, como las GAN, se pueden utilizar para predecir con precisión la ubicación óptima de los puntos de acceso (AP) en un entorno interior.
- HE2: La combinación de datos de sensores de señales inalámbricos con datos como planos arquitectónicos mejoran la precisión y predicción de la ubicación de AP mediante inteligencia artificial generativa.
- HE3: La optimización de la cobertura de la red inalámbrica mediante técnicas generativas reduce los costos operativos relacionados con la instalación y el mantenimiento de la infraestructura de red en entornos interiores.
- HE4: La implementación del sistema basado en funciones y el posicionamiento AP de IA generativa mejora la experiencia del usuario: conexiones estables y de alta calidad en diferentes planos complejos.

Los problemas, objetivos e hipótesis descritas anteriormente se encuentran alineados en la Matriz de Consistencia del Anexo C. Además, los objetivos específicos se formularon a partir de una lluvia de ideas luego de examinar los objetivos planteados en los antecedentes, cuyo detalle e item de referencia se encuentra en el Anexo E.

1.5 Justificación de la Investigación

1.5.1 Teórica

Los estudios teóricos sobre la optimización de la cobertura de redes inalámbricas mediante inteligencia artificial generativa se justifican por la necesidad de desarrollar soluciones eficaces, según N. Nauata et al. (2021), para entornos interiores donde se requieren muchas conexiones. La aplicación de algoritmos generativos brinda la oportunidad de mejorar la precisión del posicionamiento de los puntos de acceso (AP), reducir los costos operativos y garantizar una experiencia de usuario satisfactoria al minimizar las áreas de sombra y mantener una conexión estable. Estos avances son críticos en un contexto donde la conectividad inalámbrica es esencial para la productividad y disponibilidad de servicios digitales en planos complejos.

El objetivo del estudio no es sólo mejorar la cobertura y calidad del servicio de la red inalámbrica, sino también optimizar los recursos y el gasto relacionado en infraestructura nacional. Se espera que mapear las capacidades de la IA generativa mejore significativamente el rendimiento de la red inalámbrica, la satisfacción del usuario y la adaptabilidad en espacios donde la conectividad confiable es esencial para la vida cotidiana y los negocios, como lo menciona el antecedente de Alathari et al.(2023).

1.5.2 Práctica

La investigación práctica sobre la optimización de la cobertura inalámbrica mediante inteligencia artificial generativa es crucial para garantizar la viabilidad y eficacia de las soluciones propuestas en entornos reales. Se pueden utilizar pruebas y experimentos prácticos para evaluar el uso de algoritmos generativos para predecir la ubicación de puntos de acceso (AP) en diferentes planos. Estos estudios proporcionan datos empíricos sobre mejoras en la cobertura, la eficiencia de los recursos y la calidad del servicio, proporcionando una base sólida para la adopción y expansión de estas tecnologías en el campo inalámbrico.

También participa en el desarrollo de herramientas y métodos aplicables a la industria de las telecomunicaciones y la gestión de redes. Trabajando con casos reales y diferentes entornos, es posible identificar desafíos específicos, optimizar algoritmos y proponer mejores prácticas para diseñar y optimizar redes inalámbricas en entornos interiores. Estos resultados son útiles para los investigadores de telecomunicaciones, pero también tienen implicaciones directas para mejorar la conectividad y la experiencia del usuario en diversos sectores, como empresas, instituciones educativas y espacios públicos.

1.5.3 Metodológica

La aplicación del modelo de inteligencia artificial generativa propuesto en este estudio ayuda a optimizar la cobertura de redes inalámbricas en ambientes interiores. Este modelo utiliza técnicas avanzadas de inteligencia artificial generativa, como las redes generativas adversarias (GAN) para predecir con precisión los puntos de acceso (AP) óptimos en diferentes niveles. La combinación de sensores de señal inalámbricos y una estructura de datos estructurados mejora significativamente la calidad del servicio y reduce los costos operativos asociados con la instalación y mantenimiento de la infraestructura de red.

Es por ello que este estudio puede proporcionar soluciones de datos efectivas, mejorar la conectividad inalámbrica en entornos interiores complejos. Al confirmar la aplicación de algoritmos generativos en el mundo real, este estudio proporciona una base sólida para la adopción y expansión de estas técnicas en la industria de las telecomunicaciones. Esto beneficia tanto a los usuarios finales, que garantizan una experiencia de usuario superior, como a las empresas, reduciendo costes y optimizando la eficiencia de las infraestructuras de redes inalámbricas.

1.6 Delimitación del Estudio

1.6.1 Espacial

Esta investigación se centra en optimizar la cobertura de la red inalámbrica en entornos interiores específicos como entornos comerciales, institucionales o residenciales. La investigación se lleva a cabo en áreas edificadas o construidas donde la conectividad inalámbrica es esencial para el funcionamiento eficiente de dispositivos y servicios.

1.6.2 Temporal

El estudio cubre los últimos cinco años desde el año en curso 2024. Esto permite el uso de información actualizada y relevante para entrenar modelos creativos de IA y evaluar su efectividad para predecir la ubicación de puntos de acceso en diferentes niveles en un ambiente interior.

1.6.3 Conceptual

Esta investigación se orientará en técnicas de inteligencia artificial generativa, como las redes generativas adversarias (GAN) para optimizar la cobertura de la red inalámbrica. Se utilizan datos de sensores de señales inalámbricas, datos de construcción (por ejemplo, planos arquitectónicos) y técnicas de procesamiento de datos para mejorar la precisión de la predicción de la ubicación de la estación base y mejorar así la calidad del servicio de las redes inalámbricas en entornos interiores.

Capítulo II: Marco Teórico

2.1 Antecedentes de la investigación

A continuación, en esta sección se presentarán trabajos anteriores de investigación basados en la predicción de éxito o fracaso de campañas en Kickstarter o plataformas similares y el análisis de estas utilizando conjuntos de datos con estructuras similares al del presente trabajo. En cada uno se detalla el problema y su objetivo, la metodología implementada, las técnicas usadas y los resultados obtenidos.

K. Chen et al. (2013) realizaron la publicación del reporte técnico *KickPredict: Predicting Kickstarter Success* para el Departamento de Ciencias Computacionales y Matemáticas del Instituto de Tecnología de California, el cual traducido al español significa «KickPredict: Predicción del éxito de Kickstarter».

Ante la cantidad considerable de proyectos en Kickstarter que fracasaron en finanziarse debido a los datos asignados por sus creadores, los autores desarrollaron un sistema predictivo de estado de financiamiento de un proyecto, el cual captura información en tiempo real y estima su resultado a partir de ello.

De acuerdo a la metodología seguida por los autores, el primer paso fue la recolección de datos de 20,000 proyectos completados, es decir, cuyo estado de financiamiento fue exitoso o fracasado. De esta cantidad, el 95 % fue destinado al subconjunto de entrenamiento. El siguiente paso fue el desarrollo de un algoritmo clasificador binario estándar para identificar las variables más importantes y generar el modelo de predicción. Para el sistema propuesto, se utilizaron Máquinas de Vectores de Soporte (SVM) entrenadas con cada combinatoria posible de las variables del conjunto de datos. Además de recolectar registros de proyectos, se complementó utilizando data de YouTube (determinar si un proyecto no utiliza videos de este medio en su descripción) y Twitter (número de veces en que el enlace del proyecto fue compartida) para enriquecer la investigación.

Las 4 variables más importantes encontradas del algoritmo clasificador fueron el número de proyectos patrocinados por el creador, número de proyectos creados por el creador, si presenta o no video, monto de la meta del proyecto. Evaluando el modelo con la métrica de exactitud, se alcanzó el valor de 0.90 al 40 % de transcurrida la duración de la campaña.

Finalmente, como último paso se implementó una aplicación en Android para la búsqueda de proyectos en Kickstarter y una extensión en Google Chrome para estimar el éxito y la exactitud de precisión de la predicción en tiempo real, mostrando la evolución de las cantidades patrocinadas en el tiempo transcurrido de la campaña, como se ilustra en la Figura 3.

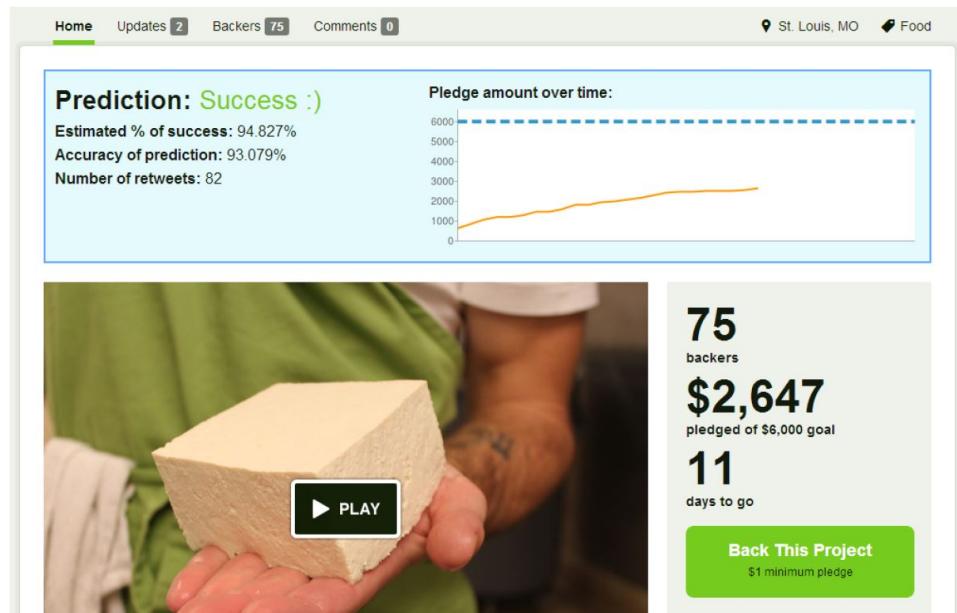


Figura 3. Extensión de Google Chrome creada para predecir el estado del proyecto.

Fuente: K. Chen et al. (2013). *KickPredict: Predicting Kickstarter Success.* (p. 4)

En el acta de la conferencia «The 17th ACM conference on Computer supported cooperative work & social computing (CSCW '14)», realizada del 15 al 19 de febrero del 2014 en Baltimore, Estados Unidos, Mitra y Gilbert (2014) publicaron el artículo titulado «The Language that Gets People to Give: Phrases that Predict Success on Kickstarter», el cual traducido al español significa «El lenguaje que hace que la gente dé: Frases que predicen el éxito en Kickstarter».

Debido a la existencia de poco conocimiento sobre los factores que impulsan a financiar proyectos a pesar del crecimiento en los últimos años de plataformas de financiamiento colectivo como Kickstarter, los autores elaboraron un modelo para predecir el éxito de financiamiento en proyectos crowdfunding a partir de frases textuales.

De acuerdo a la metodología seguida por los autores, el primer paso fue la recolección de 45,815 proyectos de Kickstarter del 2012. A continuación, se realizó limpieza de datos eliminando campañas inconclusas y con esto se procedió a extraer la información textual (descripción y recompensas del proyecto). El siguiente paso fue la generación del cuerpo de más de 9 millones de frases únicas, de las cuales quedaron 20,391 frases que aparecen al menos 50 veces en todos los proyectos. Finalmente, se creó un modelo de Regresión logística penalizada para proteger contra la colinealidad y escasez que prevalecen en el conjunto de datos de frases, de las cuales se pueda determinar cuáles son aquellas que ayudan que un proyecto sea financiado y cuáles lo contrario.

De las 59 variables de control que no son texto, 29 que tenían «valores aceptables» tenían pesos positivos. 15 variables fueron consideradas para proyectos que serán financiados, mientras que 14 para los que no. Se obtuvo un Top 100 de frases que tenían pesos positivos (proyectos que sí serán financiados), así como también para aquellas con pesos negativos (proyectos que no serán financiados) como se ilustra en la Figura 4. El ratio de error del modelo propuesto, considerando las variables de control más las frases, fue de 2.24 % frente al 17.03 % solo considerando las variables de control, y 48.47 % de la base comparada.

(F) phrases	β	(F) phrases	β	(NF) phrases	β	(NF) phrases	β
project will be	18.48	difference for	5.60	pledged	-7.12	dressed up	-4.64
has pledged	5.42	pledged will	4.01	not been able	-4.02	trusting	-3.91
pledged and	3.98	december of	3.21	all the good	-3.89	based in the	-3.87
we can afford	2.94	trip in	2.83	models of	-3.84	school that	-3.75
used in a	2.82	par	2.79	information at	-3.65	kids of all	-3.55
around new	2.78	trash	2.75	of the leading	-3.53	on a larger	-3.44
their creative	2.71	given the chance	2.69	new form of	-3.43	that uses	-3.42
mention your	2.69	your continued	2.65	we have lots	-3.24	to enjoy a	-3.20
to build this	2.65	cats	2.64	way for us	-3.18	room on	-3.18
option is	2.59	inspired me	2.57	an honorable mention	-3.17	panel of	-3.17
workshop and	2.56	project will allow	2.56	is time for	-3.14	even a dollar	-3.10

(a) Frases para proyectos exitosos

(b) Frases para proyectos fracasados

Figura 4. Ejemplos de frases del Top 100 y sus respectivos pesos que señalan si un proyecto será o no financiado.

Fuente: Mitra y Gilbert (2014). *The Language that Gets People to Give: Phrases that Predict Success on Kickstarter*. (p. 56)

En el acta de la conferencia «Twenty-first Americas Conference on Information Systems», realizada en Puerto Rico en el año 2015, M. Zhou et al. (2015) publicaron el artículo titulado «Money Talks: A Predictive Model on Crowdfunding Success Using Project Description», el cual traducido al español significa «Money Talks: un modelo predictivo sobre el éxito del crowdfunding utilizando la descripción del proyecto».

Las investigaciones existentes de crowdfunding se centran principalmente en las variables básicas del proyecto como la categoría y la meta; sin embargo, hay muy pocos estudios basados en el contenido de la información, es decir, en la descripción del mismo. Por ello, el objetivo de los autores fue estudiar la influencia y el impacto del uso de descripciones de proyectos en su éxito de financiación para predecirlo.

De acuerdo a la metodología seguida por los autores, el primer paso fue la operacionalización de los constructos, es decir, cómo se consideraría la información recolectada a partir de los antecedentes en la literatura. A continuación, se recolectó información de más de 154 mil proyectos de Kickstarter entre 2009 y 2014. Luego, se desarrolló un modelo de Regresión Logística usando variables previamente identificadas como la meta del proyecto, duración del

proyecto, si el creador presentaba conexión a Facebook, el número de sus amigos en Facebook, si el proyecto incluye una imagen, si el proyecto incluye un video, el número de recompensas, el año de lanzamiento del proyecto, y la categoría del proyecto. A éstas se añadieron el número de palabras en la descripción del proyecto, su legibilidad medido por Índice de niebla Gunning, ratio de positivo y negativo en descripción del proyecto, número de proyectos previamente creados y número de proyectos previamente patrocinados por el autor.

Finalmente, la propuesta se comparó contra otros modelos de la base y convencionales. Todos los modelos fueron evaluados por el valor-F probándose en validaciones cruzadas de 3, 5 y 10 iteraciones. Bajo esta métrica, el mejor modelo fue el propuesto, alcanzando un valor-F de 71.17 con 10 iteraciones. Asimismo, al ser comparado mediante la curva ROC, el modelo de los autores logró mejor performance frente al resto, superando el valor de 0.70 y alcanzando un ratio de verdaderos positivos frente al de falsos positivos como se aprecia en la Figura 5.

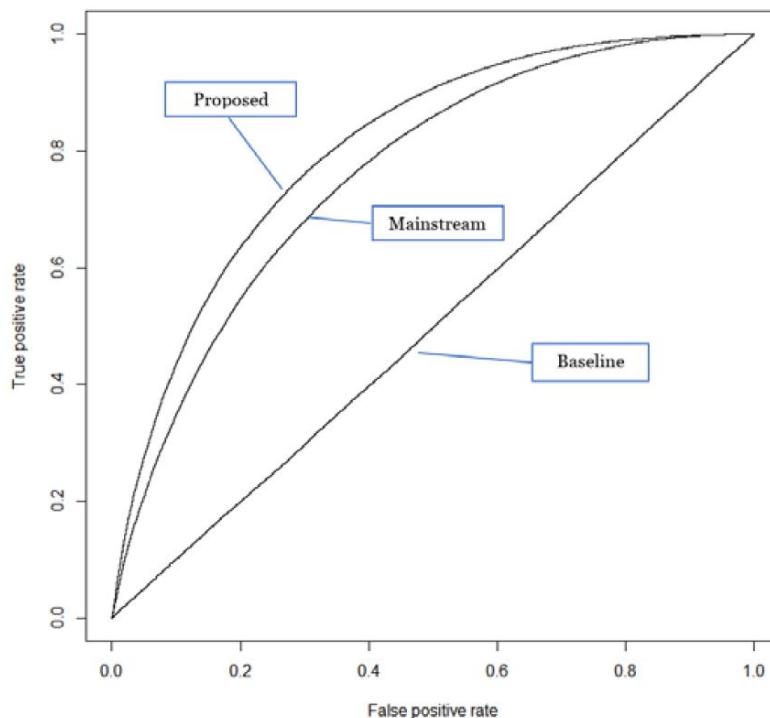


Figura 5. Curvas ROC del modelo propuesto de los autores, el de base y el convencional.

Fuente: M. Zhou et al. (2015). *Money Talks: A Predictive Model on Crowdfunding Success Using Project Description.* (p. 5)

En el acta de la conferencia «Pacific Asia Conference on Information Systems (PACIS) 2015» realizada en New York, Estados Unidos, en el año 2015, S.-Y. Chen et al. (2015) publicaron el artículo titulado «Will Your Project Get the Green Light? Predicting the Success of Crowdfunding Campaigns», el cual traducido al español significa «¿Tu proyecto obtendrá la luz verde? Prediciendo el éxito de campañas de financiamiento colectivo».

Desde el surgimiento de campañas de financiamiento colectivo en sitios web así como estudios de estos casos, la mayoría de estos presentan problemas en la predicción de éxito de una campaña por distintas casuísticas ya que suelen enfocarse en aquellos concluidos. Ante la interrogante de la posibilidad de desarrollar una técnica efectiva para predecir si una campaña será exitosa o no en diferentes momentos de tiempo de las campañas de crowdfunding, los autores desarrollaron una técnica efectiva para predecir si una campaña de crowdfunding tendrá éxito o fracasará a través de extracción y posterior uso de características estáticas y dinámicas.

De acuerdo a la metodología seguida por los autores, el primer paso fue recolectar más de 4 mil proyectos de Kickstarter obtenido tanto directamente desde el sitio como con ayuda de la página web Kickspy, entre el primer y último día de abril del 2014. Después de realizado el pre-procesamiento, se extrajeron las características para la tarea de predicción de objetivos. Luego, estas características se clasificaron en 5 categorías: características intrínsecas, mecanismo financiero, calidad y sentimiento del contenido, interacción social y efecto de progresión, en donde las 3 primeras + interacción social corresponden a un nuevo grupo asignado como «características estáticas», mientras que la última + interacción social se asignó como «características dinámicas». Ambos nuevos conjuntos agrupados finalmente fueron entrenados en una serie de 8 modelos de Bosques Aleatorios para diferentes etapas (puntos de tiempo) de la campaña desde el día 0 hasta el día 7.

Finalmente, se comparó el modelo propuesto considerando todas sus características contra el mismo alternando versiones que solo usaron algunas y un trabajo previo de otro autor. El resultado final de la evaluación medidos por la exactitud determinó que el modelo propuesto considerando todas sus características logró la mejor performance, con un valor de 0.8467. Asimismo, como se observa en la Figura 6, se evaluaron campañas exitosas y fracasadas con la precisión, sensibilidad y puntaje F1 en distintas etapas de tiempo (7 primeros días), en donde se determinó que, a mayor tiempo transcurrido, mejores son sus resultados.

Stage	Successful Campaigns			Failed Campaigns			Accuracy
	Precision	Recall	F1	Precision	Recall	F1	
T_0	70.44%	70.41%	70.43%	74.97%	75.00%	74.98%	72.89%
T_1	84.21%	87.34%	85.75%	89.61%	86.96%	88.27%	87.13%
T_2	85.02%	87.73%	86.35%	90.45%	88.26%	89.34%	88.03%
T_3	85.27%	87.73%	86.48%	90.65%	88.70%	89.66%	88.29%
T_4	85.11%	88.58%	86.81%	91.44%	88.73%	90.06%	88.67%
T_5	85.91%	87.93%	86.91%	91.32%	89.81%	90.56%	89.03%
T_6	86.69%	88.43%	87.55%	91.89%	90.61%	91.24%	89.72%
T_7	86.20%	88.23%	87.21%	92.00%	90.55%	91.27%	89.62%

Figura 6. Evaluación de la performance de los modelos construidos en distintas etapas de tiempo.

Fuente: S.-Y. Chen et al. (2015). *Will Your Project Get the Green Light? Predicting the Success of Crowdfunding Campaigns.* (p. 12)

Beckwith (2016) publicó la investigación de su tesis de grado titulada «Predicting Success in Equity Crowdfunding», traducida al español como «Prediciendo el éxito en el financiamiento colectivo de acciones», para el programa académico «Joseph Wharton Scholars» de la Universidad de Pensilvania en el año 2016.

El financiamiento colectivo o crowdfunding de inversión se vuelve cada vez más popular. Este concepto permite que los emprendedores ofrezcan algún tipo de producto o servicio como compensación por contribuciones financieras una vez que ya se tengan ventas reales o acuerdos comerciales. Los patrocinadores no necesitan contar con un capital muy alto ya que ellos recibirán algo a cambio una vez que el proyecto se realice. Este factor permite su mayor probabilidad de éxito de financiamiento al momento de darse una campaña de este tipo de crowdfunding ya que los patrocinadores pueden invertir en más de un proyecto a la vez. A partir de este punto, surge la interrogante por conocer los motivos de inversión y no inversión en ciertos start-ups ante la similitud de características observables. Por ello, el objetivo del autor fue determinar la relación entre las características de una compañía determinada y su capacidad para recaudar fondos en la plataforma de financiamiento colectivo de capital AngelList.

De acuerdo a la metodología seguida por el autor, el primer paso fue la recolección de más de 5 mil compañías de AngelList que solicitaron contribuciones financieras. De esta cantidad, se consideraron aquellas con datos completos (2,603 empresas). Luego del preprocesamiento del conjunto final de datos, el investigador desarrolló un modelo de Regresión logística usando las siguientes variables: si la empresa previamente había recibido o no financiación, si la compañía cuenta con perfil en Twitter, número de veces que la compañía había sido mencionada en alguna publicación, número de veces que la compañía había sido men-

cionada en TechCrunch, número de personas listadas como co-fundadoras en el perfil de la compañía AngelList, si AngelList lista entre 11 y 50 empleados como tamaño de la empresa, si la compañía está ubicada en San Francisco, si entre los fundadores de AngelList al menos uno de ellos cuenta con un MBA, si entre los fundadores de AngelList al menos uno de ellos realizó sus estudios en alguna de las mejores 20 universidades de Estados Unidos, y el número de cierre del S&P 500 el día anterior al lanzamiento de la campaña de crowdfunding de AngelList. La propuesta fue comparada con un Árbol de Decisión CART, un modelo de Naïve Bayes y una Máquina de Vectores de Soporte. Finalmente, se evaluó el modelo propuesto mediante la exactitud, precisión, sensibilidad, puntaje F1, y área bajo la curva (AUC).

Bajo las métricas mencionadas, el modelo del autor superó a los otros 3 modelos comparados en cada una de ellas. Sus resultados fueron 0.87 de exactitud, precisión promedio de 0.85, sensibilidad promedio de 0.88, puntaje F1 promedio de 0.86, y área bajo la curva de 0.74, este último como se observa en la Figura 7.

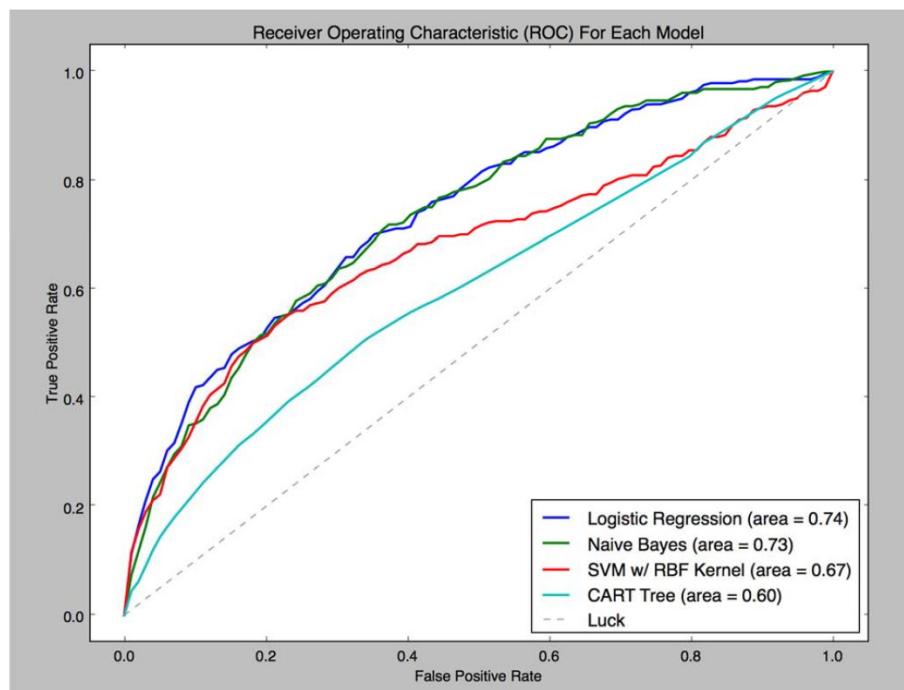


Figura 7. Curva ROC para cada modelo utilizado por el autor.

Fuente: Beckwith (2016). *Predicting Success in Equity Crowdfunding*. (p. 28)

En el acta de la conferencia «The Ninth ACM International Conference on Web Search and Data Mining (WSDM' 16)» realizado en San Francisco, Estados Unidos, en el año 2016, Y. Li et al. (2016) publicaron el artículo titulado «Project Success Prediction in Crowdfunding Environments», el cual traducido al español significa «Predicción de éxito de un proyecto en ambientes de financiamiento colectivo».

Durante la última década se han elaborado una serie de modelos de clasificación que permitan pronosticar con cierto nivel de exactitud si algunos proyectos de financiamiento colectivo tendrán éxito o no. Sin embargo, el hecho de estimar si esto ocurrirá durante el plazo dado de la campaña no puede proporcionar una guía adecuada a los patrocinadores que desean invertir en proyectos populares. Es entonces que se cuestiona la posibilidad de predecir el éxito de campañas en sitios web de crowdfunding como Kickstarter considerando características obtenidas durante la operación de la campaña. Para resolver esta interrogante, los autores formularon la predicción del éxito del proyecto como un problema de análisis de supervivencia y aplicar el enfoque de regresión censurada.

De acuerdo a la metodología seguida por los autores, el primer paso fue la recolección de más de 27 mil proyectos de Kickstarter entre diciembre del 2013 y junio del 2014 para la data estática, a los cuales se removieron aquellos que fueron cancelados, suspendidos o con menos de 1 patrocinador y monto prometido de \$100. Para la data dinámica se capturaron más de 106 mil tweets de Twitter que contenían el enlace rápido hacia la página de Kickstarter. Una vez obtenidos los conjuntos de datos, se realizó el pre-procesamiento. A continuación, se elaboraron 2 modelos predictivos para evaluar la regresión censurada: 1 modelo de distribución logística y 1 de distribución log-logística. Para poder compararlos, se elaboraron adicionalmente otros métodos para manejar observaciones censuradas, mencionados en la literatura como Cox, Regresión Tobit, estimación Buckley-James e Índice de Concordancia Boosting (BoostCI).

Finalmente, como resultado de las múltiples pruebas en los modelos, en los cuales se experimentaron distintas combinatorias con la data estática y dinámica, así como agregando y desagregando proyectos fracasados, se concluyó que el modelo de Regresión log-logística evaluado con el área bajo la curva (AUC) presentó mejor performance que el resto en 3 de las 4 combinatorias, alcanzando su mejor nivel en el conjunto de datos que combina data estática, dinámica, de características de proyectos transcurridos los primeros 3 días y con proyectos fracasados, con un puntaje Survival AUC de 0.9030, como se representa en la Figura 8.

	Static		Static+Social		Static+3days		Static+Social+3days	
	without failed	with failed						
Cox	0.7322 (0.0104)	0.7727 (0.0092)	0.7463 (0.0098)	0.7942 (0.0089)	0.7667 (0.0126)	0.7965 (0.0093)	0.7724 (0.0121)	0.8098 (0.0087)
Tobit	0.7281 (0.0108)	0.7755 (0.0100)	0.7381 (0.0099)	0.7960 (0.0082)	0.7833 (0.0124)	0.8226 (0.0096)	0.7841 (0.0121)	0.8309 (0.0084)
BJ	0.7097 (0.0130)	0.7313 (0.0114)	0.7235 (0.0128)	0.7587 (0.0080)	0.8016 (0.0127)	0.8157 (0.0102)	0.8016 (0.0127)	0.8201 (0.0089)
BoostCI	0.5919 (0.0140)	0.6649 (0.0288)	0.6128 (0.0380)	0.6796 (0.0212)	0.8135 (0.0430)	0.8668 (0.0229)	0.8141 (0.0421)	0.8671 (0.0231)
Logistic	0.7354 (0.0106)	0.7815 (0.0095)	0.7457 (0.0095)	0.8009 (0.0086)	0.8332 (0.0097)	0.8659 (0.0075)	0.8331 (0.0094)	0.8695 (0.0067)
Log-logistic	0.7277 (0.0111)	0.7826 (0.0099)	0.7411 (0.0096)	0.8029 (0.0081)	0.8800 (0.0057)	0.9010 (0.0056)	0.8774 (0.0060)	0.9030 (0.0057)

Figura 8. Comparación de resultados de conjuntos de datos de distintas características de proyectos considerando incluir o no proyectos fracasados.

Fuente: Y. Li et al. (2016). *Project Success Prediction in Crowdfunding Environments*. (p. 254)

H. Yuan et al. (2016) publicaron un artículo en la revista «Decision Support Systems», en el año 2016, titulado «The Determinants of Crowdfunding Success: A Semantic Text Analytics Approach», el cual traducido al español significa «Los determinantes del éxito del financiamiento colectivo: Un enfoque de analítica semántica de texto».

Actualmente existen diversos estudios de éxito de campañas de financiamiento colectivo, la mayoría usando modelos predictivos en proyectos crowdfunding basados en recompensas y análisis de características poco profundas del lenguaje en descripción de los proyectos (por ejemplo, número de palabras, errores ortográficos, etc) así como en la implementación de métodos de Aprendizaje Automático. Como propuesta alterna, los autores construyeron un marco de trabajo basado en análisis de texto que pueda extraer semánticas de descripciones textuales de proyectos para predecir sus resultados de recaudación de fondos.

De acuerdo a la metodología seguida por los autores, el primer paso consistió en recolectar 500 proyectos de 2 websites chinas sobre crowdfunding. Luego, se construyó el marco de trabajo del análisis textual ilustrado en la Figura 9. A continuación, se diseñó un modelo de Bosque Aleatorio para los datos numéricos, y POS Tag Stemming + DC-LDA + características tópicas para los datos textuales con la finalidad de realizar una extracción más efectiva de las características tópicas a partir de las descripciones. Esta arquitectura se comparó con otros modelos en la literatura de la publicación. Luego, se realizó un análisis empírico para identificar características discriminatorias que influye en el éxito de la recaudación de fondos.

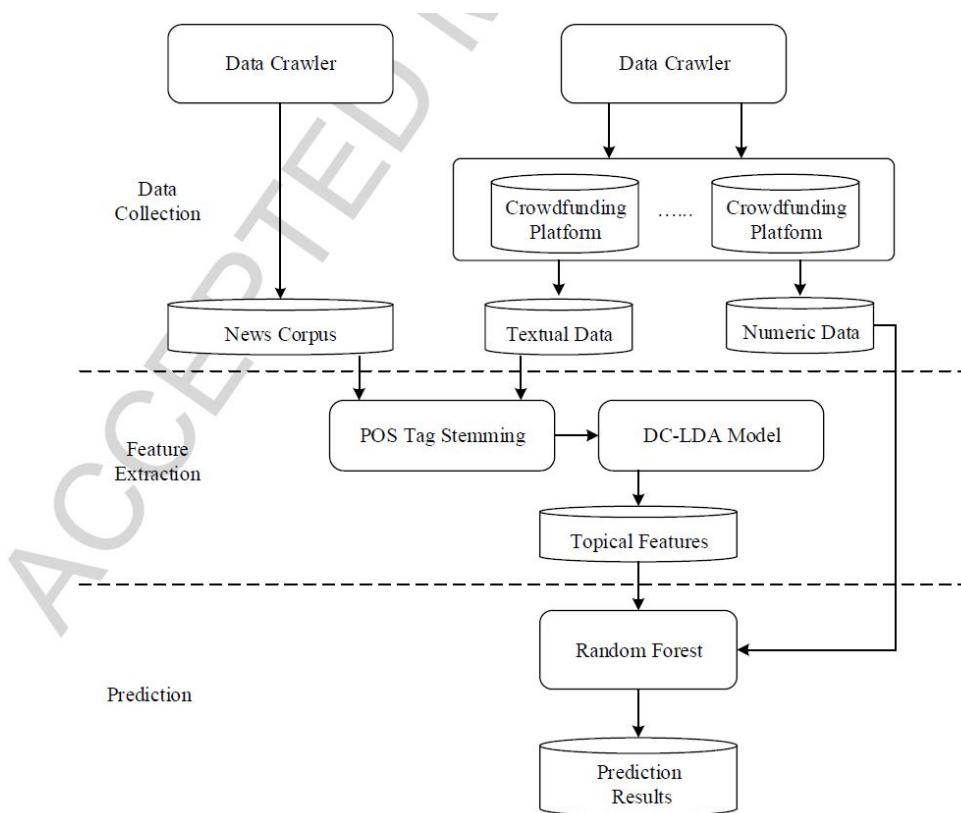


Figura 9. Marco de trabajo de analítica textual de los autores.

Fuente: H. Yuan et al. (2016). *The Determinants of Crowdfunding Success: A Semantic Text Analytics Approach.* (p. 74)

Finalmente, se evaluaron ambos conjuntos de datos. Para los datos numéricos, la mejor performance se obtuvo evaluando el modelo con la sensibilidad para el Bosque Aleatorio (0.98 en la segunda data); mientras que para los datos textuales, el modelo DC-LDA (al ser comparado con el LDA normal) obtuvo un puntaje F1 de 0.91 como se observa en la Figura 10.

Dataset	Dreamore				Zhongchou			
Classifier	RF	BPNN	SVM	ELM	RF	BPNN	SVM	ELM
Accuracy	77.00	57.00	53.00	59.00	95.00	94.00	80.00	85.00
Precision	73.68	54.93	51.72	58.18	92.45	90.74	71.43	87.23
Recall	84.00	78.00	90.00	64.00	98.00	98.00	100.00	82.00
F ₁	78.50	64.46	65.69	60.95	95.15	94.23	83.33	84.54

Figura 10. Performance de ambas bases de datos con distintas métricas.

Fuente: H. Yuan et al. (2016). *The Determinants of Crowdfunding Success: A Semantic Text Analytics Approach.* (p. 91)

Sawhney et al. (2016) publicaron el reporte técnico titulado *Using Language to Predict Kickstarter Success*, que traducido al español significa «Uso del lenguaje para predecir el éxito de Kickstarter», para el Departamento de Ciencia de la Computación de la Universidad Standford en el 2016.

Dada la considerable cantidad de investigaciones y trabajos para lograr la predicción de éxito financiamiento de proyectos basados en la evolución de variables estáticas, los autores decidieron optar por otra alternativa diseñando de un clasificador binario que logre predecir el éxito de una campaña a partir de su contenido, características lingüísticas y metadata.

De acuerdo a la metodología seguida por los autores, el primer paso consistió en recopilar más de 160 mil proyectos de Kickstarter, donde los proyectos etiquetados como exitosos se asignaron el valor de 1 y los fracasados como 0. Las variables utilizadas fueron el nombre del proyecto, el resumen breve, la meta de la campaña, su fecha de creación y su fecha de culminación. El conjunto total de datos se distribuyó en 80% y 20% para subconjuntos de entrenamiento y prueba respectivamente. A continuación, se recopilaron las características primarias y se realizó un análisis de sentimientos. Luego, asignaron las meta-características para utilizarlos como entradas durante el entrenamiento de los modelos. Para ellos, los autores utilizaron como modelo de base el clasificador de Naive Bayes para las características primarias, una Máquina de Vectores de Soporte (SVM) que considera ocurrencias unigramas basado en la cantidad de patrocinadores alcanza una campaña en su final, y un modelo SVM propuesto con meta-características asignadas luego de la extracción de características primarias obtenidas gracias a un Part-Of-Speech (POS). Finalmente, luego de calibrar los modelos, estos se probaron con el subconjunto de prueba para finalmente ser evaluados.

Considerando el modelo solo con las características primarias (base), el nivel de exactitud alcanzó 65 %. Sin embargo, utilizando el clasificador entrenado con el primer SVM, el valor de la exactitud logró llegar hasta 92 %. Adicionalmente, se probó el segundo modelo propuesto SVM y se compararon tanto sus niveles de exactitud en entrenamiento como en prueba, ilustrado en la Figura 11. En esta, se observa que en todas las categorías, la exactitud promedio de la prueba fue de 71 % con un rango entre 25 % a 100 %, mientras que para el entrenamiento fue como valor promedio de 88 % con un rango entre 81 % a 100 %.

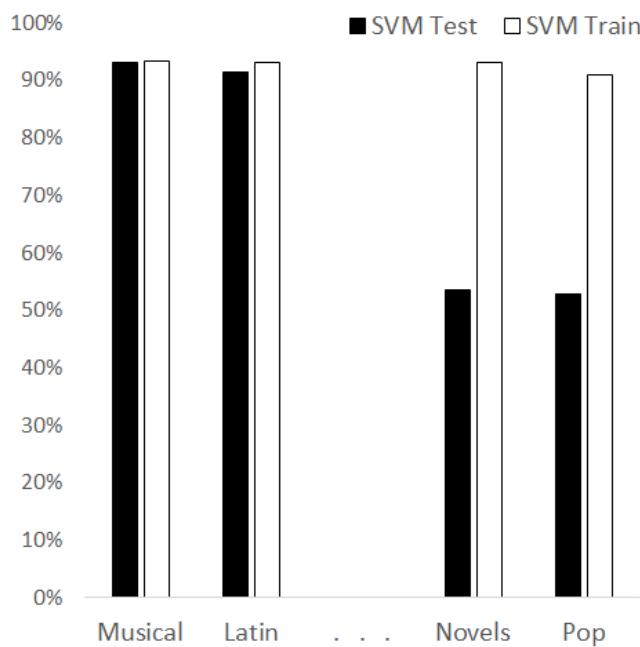


Figura 11. Exactitud del modelo SVM para cada categoría.

Fuente: Sawhney et al. (2016). *Using Language to Predict Kickstarter Success.* (p. 6)

Kaur y Gera (2017) publicaron un artículo titulado «Effect of Social Media Connectivity on Success of Crowdfunding Campaigns», el cual traducido al español significa «Efecto de conectividad en medios sociales en éxito de campañas de financiamiento colectivo», para la revista científica «Procedia Computer Science» en el año 2017.

Los proyectos de crowdfunding, así como los proyectos lanzados en sitios dedicados a esto, crecieron exponencialmente en los últimos años. Los medios sociales desempeñan un papel importante en la difusión de palabras sobre una campaña y en la obtención de fondos con éxito. Después de lanzarse una campaña, su éxito se define por la interacción social de los creadores en la plataforma y en las redes sociales. Además, el tamaño de la red social del creador y su presencia en línea motiva a los patrocinadores a participar y financiar el proyecto. Partiendo de esta premisa, se busca conocer el impacto de la conectividad con las redes sociales y las interacciones sociales en el rendimiento del crowdfunding. Para ello, los autores plantearon el desarrollo de un modelo predictivo que comprenda las características de la campaña e información sobre la conectividad con las diversas redes sociales como Facebook y Twitter.

De acuerdo a la metodología seguida por los autores, el primer paso consistió en recolectar 2 tipos de conjuntos de datos: el primero obtenido desde Kickstarter en el mes de Abril del 2014, con más de 4 mil campañas; mientras que el segundo a través del perfil del creador vinculado a redes sociales. El primer conjunto comprendió las variables de categoría, tipo de di-

visa, monto de la meta, monto prometido, número de recompensas, duración de la campaña en días, y el estado final de financiamiento (exitoso o fracasado). Por el lado del segundo conjunto, además de obtener marcas de hacia qué redes sociales el creador se conecta, mediante el enlace se extrajeron más de 19 mil tweets en el que se promocionaron las campañas gracias a la API de Twitter. Luego de armar los conjuntos de datos, se creó un modelo de Regresión Logística. Este usó las variables de categoría, meta de la campaña, número de recompensas, número de tweets publicados por stakeholders como patrocinadores, comunicadores, promotores y alguna compañía, si un proyecto es escogido como “Proyecto del día”, conexión a Facebook por parte del creador, número de cuentas creadas para la campaña tanto en Facebook como en Twitter, y sitios webs adicionales del creador. Las anteriores variables se determinaron luego de evaluarse todas en una matriz de correlación de Pearson. El conjunto final de datos fue fraccionado en 66 % para entrenamiento y 34 % para prueba. Por último, se comparó con las bases de referencia usando como métrica principal la exactitud.

El modelo de Regresión Logística alcanzó una exactitud de 76.7 %. Este fue comparado con algunos modelos mencionados en sus antecedentes como Naïve Bayes, J48 y Bosques Aleatorios. Bajo otras métricas, como se observa en la Figura 12, alcanzó los mayores niveles promedio en el Área bajo la curva ROC (AUC-ROC) y Área bajo la curva Precisión-Sensibilidad (AUC-PRC) con 84.7 % y 84.6 % respectivamente.

====Summary of Stratified cross-validation ===

Correctly Classified Instances	3162	76.7289 %
Incorrectly Classified Instances	959	23.2711 %
Total Number of Instances	4121	

==== Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0.828	0.304	0.763	0.828	0.794	0.530	0.847	0.868	failed
0.696	0.172	0.774	0.696	0.733	0.530	0.847	0.820	successful
Weighted Avg.								
0.767	0.244	0.768	0.767	0.766	0.530	0.847	0.846	

Figura 12. Performance estadística del modelo predictivo de los autores.

Fuente: Kaur y Gera (2017). *Effect of Social Media Connectivity on Success of Crowdfunding Campaigns.* (p. 772)

R. S. Kamath y Kamat (2018) publicaron un artículo titulado «Supervised Learning Model For Kickstarter Campaigns With R Mining», el cual traducido al español significa «Modelo de aprendizaje supervisado para campañas de Kickstarter con R Mining», para la revista científica «International Journal of Information Technology, Modeling and Computing» en el año 2018.

A pesar de que Kickstarter es la plataforma web de crowdfunding más grande existente, no todas las campañas que se encuentran son exitosas. Algunos trabajos previos citados en este artículo, incluyendo los anteriormente mencionados K. Chen et al. (2013) y Mitra y Gilbert (2014), citan que el lenguaje usado en las campañas alcanza el poder predictivo de 58.86 %, es decir, la descripción semántica de un proyecto ayuda considerablemente en la predicción de éxito debido a que muchos patrocinadores se fijan en la calidad presente en una campaña antes de invertir. Sin embargo, es dejada de lado en muchos trabajos de investigación destinados a la predicción de éxito de financiamiento, así como también las interacciones de un proyecto en redes sociales. Para el tiempo en que este artículo fue publicado, el nivel de exactitud de los modelos predictivos ya existentes bordeaba por el 68 % como el descrito anteriormente. Ante esto, los autores desarrollaron un sistema con técnicas de Aprendizaje Automático usando el entorno R aplicadas a un conjunto de datos de campañas en Kickstarter para alcanzar su objetivo de clasificar proyectos entrenando diferentes clasificadores y predecir con un alto nivel de exactitud.

De acuerdo a la metodología seguida por los autores, en primer lugar se analizó y definió el problema de la investigación. A continuación, se recolectaron 120 proyectos de Kickstarter a través de su URL. Luego, las variables categóricas fueron transformadas en numéricas como parte del pre-procesamiento. Esto permitió realizar extracción y exploración del conjunto final mediante estadísticas descriptivas con paquetes de R. Luego, se implementaron 5 modelos clasificadores para la investigación: un Árbol de decisión, un Bosque Aleatorio, una Red Neuronal Artificial, el algoritmo del Vecino K más cercano, y el modelo Naïve Bayes. Se consideraron como variables para estos modelos al nombre del proyecto, URL del proyecto, descripción del proyecto, categoría, sub-categoría, número de patrocinadores, monto total patrocinado, meta de financiación, fecha de lanzamiento del proyecto, duración del proyecto, número de actualizaciones, número de recompensas, si el proyecto cuenta con video, localización del proyecto, y creador del proyecto.

Por último, con la matriz de confusión, se escogió la métrica de exactitud para evaluar los modelos, en donde el mejor de ellos fue la Red Neuronal, con un valor de 0.94. En la Figura 13, se representa en un gráfico de dispersión cada modelo (con excepción del algoritmo del vecino K más cercano) escalado según el valor de su exactitud.

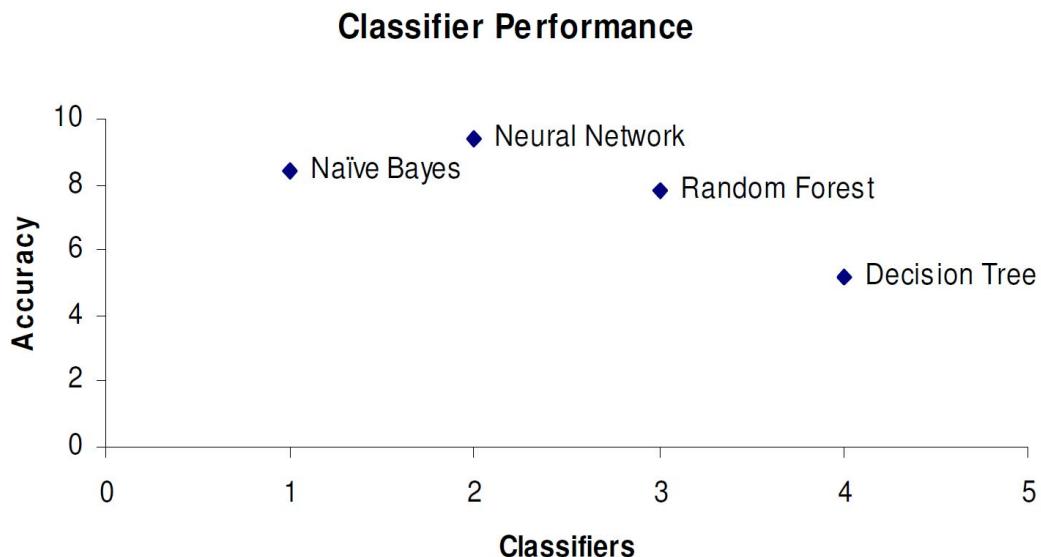


Figura 13. Gráfico de dispersión de cada modelo evaluado según su exactitud.

Fuente: R. S. Kamath y Kamat (2018). *Supervised Learning Model For Kickstarter Campaigns With R Mining*. (p. 26)

En el acta de la conferencia «2018 IEEE 15th International Conference on e-Business Engineering (ICEBE)» realizado en Xi'an, China, del 12 al 14 de octubre del 2018, P.-F. Yu et al. (2018) publicaron el artículo titulado «Prediction of Crowdfunding Project Success with Deep Learning», el cual traducido al español significa «Predicción del éxito de proyecto de financiamiento colectivo con Aprendizaje Profundo».

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. La incógnita que rodea a esta situación es la posibilidad de elaborar un modelo predictivo de éxito en proyectos de Kickstarter utilizando distintas técnicas al Aprendizaje Automático. Basándose en investigaciones previas que incluyeron a los ya mencionados K. Chen et al. (2013), Y. Li et al. (2016), Sawhney et al. (2016) y R. S. Kamath y Kamat (2018), los autores plantean como objetivo el desarrollo de un modelo que prediga el éxito del proyecto de crowdfunding con Aprendizaje Profundo usando registros históricos de campañas en Kickstarter.

De acuerdo a la metodología seguida por los autores, en primer lugar se recolectaron más de 378 mil campañas de Kickstarter entre Mayo 2009 y Marzo 2018, recopilados retrospectivamente desde Kaggle. A continuación, realizaron análisis exploratorio de los datos para

entender la data y seleccionar variables. Estas últimas fueron el nombre del proyecto, categoría del proyecto, subcategoría, divisa en que se encuentra la meta de la campaña, fecha de lanzamiento, fecha de culminación, meta de la campaña, monto prometido alcanzado, número de patrocinadores, país del proyecto, monto prometido alcanzado en dólares y meta de la campaña en dólares. Los autores elaboraron un modelo Perceptrón Multicapa (MLP), alimentado por 6 variables visibles de la campaña (metadata), entrenado con 100 épocas con tamaño de lote de 30, como se observa en la Figura 14.

Finalmente, el modelo propuesto se comparó con los de la literatura, entre ellos Bosques aleatorios, AdaBoost, Máquina de vectores de soporte, Árboles de decisión, Regresión logística y Naïve Bayes. Se tomaron en cuenta como métricas la exactitud y el área bajo la curva ROC (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.

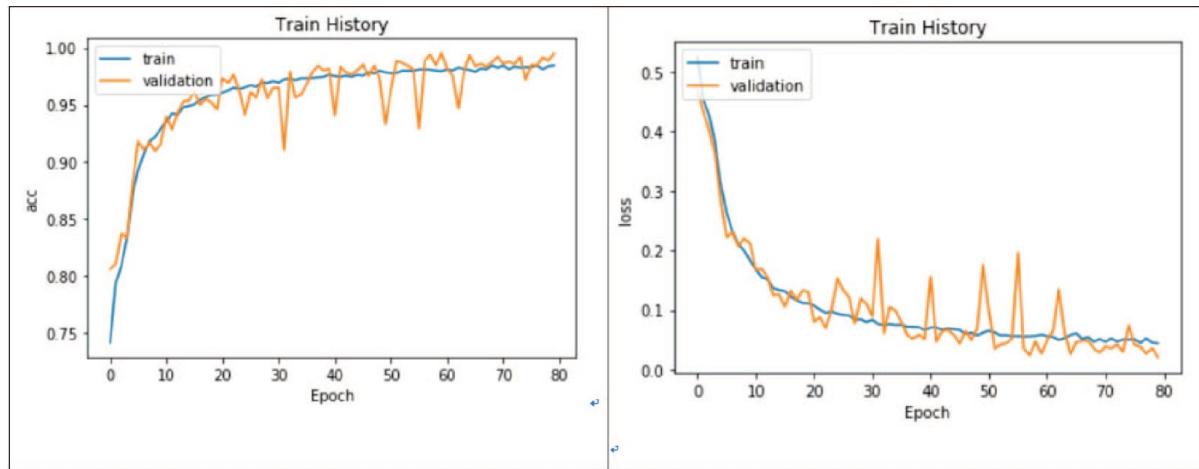


Figura 14. Gráfico de historial de entrenamiento para subconjuntos de entrenamiento y validación.

Fuente: P.-F. Yu et al. (2018). *Prediction of Crowdfunding Project Success with Deep Learning.* (p. 6)

En el acta de la conferencia «Companion of the 2018 ACM Conference on Computer Supported Cooperative Work and Social Computing», realizada del 3 al 7 de noviembre del 2018 en Jersey City, Estados Unidos, S. Lee et al. (2018) publicaron el artículo titulado «Content-based Success Prediction of Crowdfunding Campaigns: A Deep Learning Approach», el cual traducido al español significa «Predicción de éxito basada en contenido de campañas de crowdfunding: un enfoque de Aprendizaje Profundo».

Debido a que el ratio promedio de éxito de financiamiento colectivo ha ido decreciendo en los últimos años a pesar del éxito de plataformas basadas en este tipo, así como la llegada de desafíos como descubrir las razones de su éxito y lograr predecirlo, los autores plantearon utilizar solo el contenido textual de los proyectos (descripción de campaña, actualizaciones, comentarios y palabras obtenidas del video principal) para predecir el estado de financiamiento de proyectos de Tecnología (la categoría con ratio más bajo de éxito de financiamiento y a la vez, con los mayores montos financiados en la plataforma) a través de la creación de una Red Neuronal Profunda.

De acuerdo a la metodología seguida por los autores, en primer lugar se recolectaron 216,136 proyectos de crowdfunding en Kickstarter, lanzados entre abril del 2009 y agosto del 2017. De este grupo, conformado por 15 categorías, se filtraron aquellos de la categoría Tecnología (22,982 registros) para luego separar el conjunto de datos final en 0.80 (entrenamiento), 0.10 (validación) y 0.1 (prueba). Luego, se diseñaron 2 modelos de NLP: 1 Modelo de Secuencia a Secuencia (Seq2seq) con atención a nivel de oración y 1 Modelo de Atención Jerárquica (HAN).

Finalmente, luego de ser entrenados, los modelos fueron evaluados bajo la métrica de exactitud, bajo enfoques de distintos experimentos que alternaban la presencia de las variables. De todos ellos, basado en aquellos que utilizaron todas en conjunto, el modelo de Seq2seq + atención logró una exactitud de 0.80, mientras que el modelo HAN alcanzó el valor de 0.60, como se aprecia en la Figura 15.

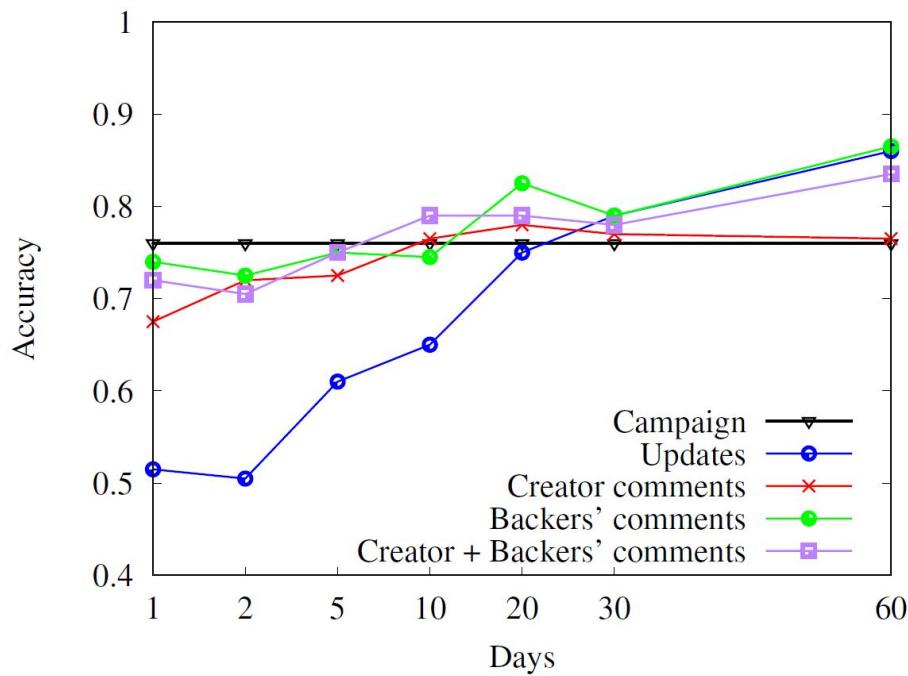


Figura 15. Promedio estimado de la exactitud de predicción vs tiempo estimado en días.

Fuente: S. Lee et al. (2018). *Content-based Success Prediction of Crowdfunding Campaigns: A Deep Learning Approach*. (p. 196)

En el acta de la conferencia «The 33rd AAAI Conference on Artificial Intelligence (AAAI'2019)», realizada del 27 de enero al 1 de febrero del 2019 en Honolulu, Hawái, Jin et al. (2019) publicaron un artículo titulado «Estimating the Days to Success of Campaigns in Crowdfunding: A Deep Survival Perspective», el cual traducido al español significa «Estimación de días de éxito de campañas en financiamiento colectivo: Una perspectiva de supervivencia profunda».

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. El reto de predecir el tiempo de duración es más complicado porque se ve afectada por la variabilidad de la metadata y los comentarios del proyecto, resultando un alto nivel de fracaso (60 %) en la financiación de proyectos. Por ello, 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.

De acuerdo a la metodología seguida por los autores, en primer lugar se recolectaron más de 14 mil proyectos de Indiegogo con información estática (descripción de campaña, descripción de las recompensas, categoría de la campaña, tipo del creador, duración declarada de

financiación, meta declarada prometida, número de recompensas, precio máximo/mínimo/promedio de recompensas, plazo máximo/mínimo/promedio de entrega) y dinámica (comentarios). Luego se definirse el problema estudiado, se introdujo los detalles técnicos de SMP, la arquitectura que se usó. A continuación, el conjunto de datos de entrada fue pre-procesado, se elaboró un modelo Seq2seq (encoder + decoder) con previos 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, ilustrado en la Figura 16. Después se designó el método de entrenamiento para el modelo.

Finalmente, los modelos fueron evaluados por 3 métricas: Divergencia de Kullback-Leibler (KL), la Raíz del Error Cuadrático Medio (RMSE) y el Error Absoluto Medio (MAE). Respecto a la predicción de distribución de patrocinios, y evaluando mejor con RMSE, los modelos propuestos de SMP fueron los más efectivos (para 70 % de ratio de partición, se obtuvo 0.135 con el previo evolutivo y 0.137 sin el previo evolutivo, frente a 0.142 del Perceptrón Multicapa y 0.144 de la Regresión Logística Multinomial) para el Encoder. Respecto a la predicción del tiempo de éxito, los modelos propuestos de SMP (para 50 % de ratio de partición, se obtuvieron 0.960, 0.930 y 0.729) fueron mejores que modelos de referencia (para 50 % de ratio de partición, el mejor de los 8 modelos de referencia fue el BoostCOX con 0.776) para el Decoder.

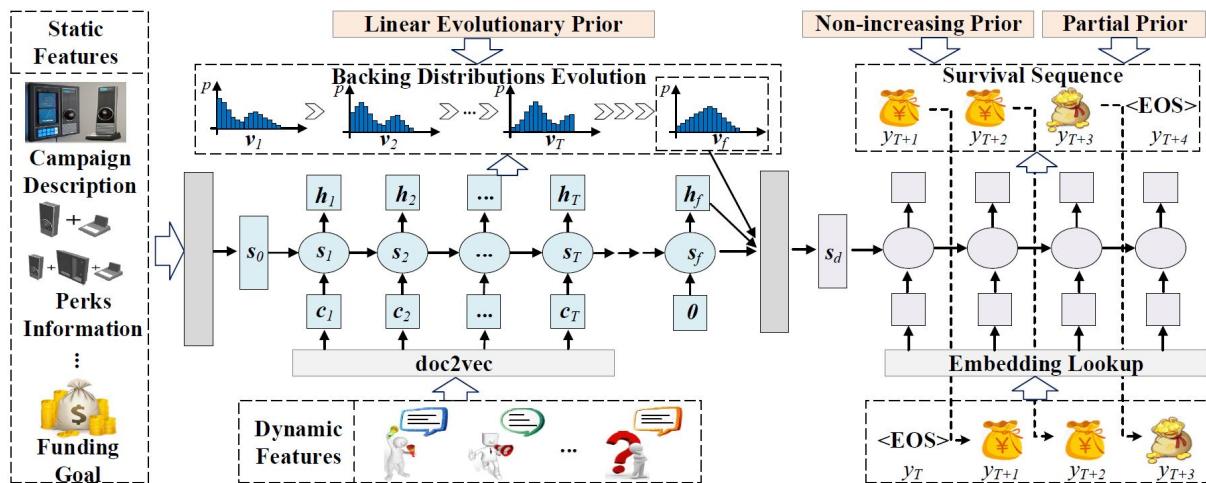


Figura 16. Arquitectura SMP de los autores.

Fuente: Jin et al. (2019). *Estimating the Days to Success of Campaigns in Crowdfunding: A Deep Survival Perspective.* (p. 3)

En el acta de la conferencia «The Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19)», realizada del 10 al 16 de agosto del 2019 en Macao, China, Cheng et al. (2019) publicaron el artículo titulado «Success Prediction on Crowdfunding with Multimodal Deep Learning», el cual traducido al español significa «Predicción del éxito en financiamiento colectivo con Aprendizaje Profundo Multimodal».

La mayoría de los enfoques de predicción existentes aprovechan solo la modalidad dominada por el texto a pesar de existir más información en los perfiles de los proyectos, como por ejemplo, metadatos e imágenes. A estos últimos se han realizado poco trabajo para evaluar sus efectos hacia la predicción del éxito. Además, la metainformación ha sido explotada en muchos enfoques existentes para mejorar la precisión de la predicción. Sin embargo, esta generalmente se limita a la dinámica después de la publicación de los proyectos, haciendo que tanto los creadores del proyecto como las plataformas no puedan predecir el resultado de manera oportuna. Se carecen estudios basados en distintas modalidades más allá de la metainformación. El objetivo de los autores es construir esquemas avanzados de redes neuronales que combinan información de diferentes modalidades para predecir el estado de financiamiento del proyecto usando solo información pre-publicación.

De acuerdo a la metodología seguida por los autores, en primer lugar se recolectaron más de 20 mil campañas finalizadas (exitosas o fracasadas) en Kickstarter gracias a un algoritmo creado para capturar datos en la página por 2 semanas. Se descartaron aquellos proyectos previos al 2015, quedando proyectos entre este año y el 2018. Asimismo, entre el contenido obtenido se registraron perfil textual (título, resumen, descripción del financiamiento, y riesgos/retos), imágenes (en formato jpg), categorías, meta de financiamiento, fecha de inicio y de finalización de campaña. Luego, se realizó pre-procesamiento a los subconjuntos de datos y se dividieron en 3 subconjuntos: entrenamiento (proyectos del 2015 y 2016), validación (2017) y prueba (2018). A continuación, se configuró la arquitectura del marco de trabajo del modelo Aprendizaje Profundo Multimodal y finalmente se realizaron distintos experimentos elaborando combinaciones de distintas modalidades para compararlas entre sí mediante las métricas de sensibilidad, precisión, puntaje F1 y área bajo la curva (AUC). Debido a que los autores construyeron un modelo apilado de 3 modelos para cada modalidad, ilustrada en la Figura 17, cada una se distribuyó de la siguiente manera: un modelo de Máquina de Vectores de Soporte (SVM) para la metainformación; la Red Neuronal Convolutacional (CNN) pre-entrenada de 16 capas VGG16, trabajando junto con la Bolsa de Palabras Visuales (BoVW) para las imágenes; y la Bolsa de Palabras (BoW) trabajando junto con la Frecuencia de Términos - Frecuencia Inversa de Documentos (TF-IDF) e incrustaciones de palabras GloVe de 300 dimensiones.

Finalmente, para evaluar los resultados, se comparó el modelo propuesto (MDL-TIM) por los autores contra la base de referencia de un modelo SVM con texto, imágenes y metadata

(SVM-TIM). Así, se observó que bajo todas las métricas, con excepción de la precisión, el marco de trabajo Multimodal obtuvo mejores resultados, con 0.7505 en sensibilidad, 0.7534 en puntaje F1 y 0.8326 en área bajo la curva, frente a 0.7411, 0.7483 y 0.7411 respectivamente.

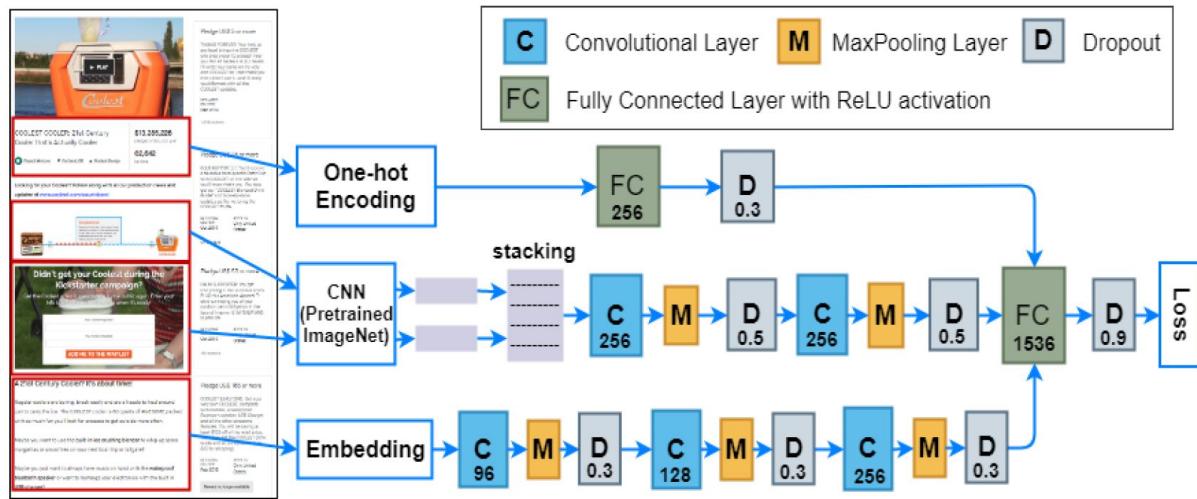


Figura 17. Arquitectura de modelo de Aprendizaje Profundo Multimodal de los autores.

Fuente: Cheng et al. (2019). *Success Prediction on Crowdfunding with Multimodal Deep Learning*. (p. 2161)

En el acta de la conferencia «2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA)», realizada del 12 al 15 de abril del 2019 en Tokyo, Japón, L.-S. Chen y Shen (2019) publicaron el artículo titulado «Finding the Keywords Affecting the Success of Crowdfunding Projects», el cual traducida al español significa «Encontrar las palabras clave que influyen en el éxito de los proyectos de financiación colectiva».

A la fecha de esta publicación, existían pocas investigaciones basadas en el impacto de la descripción y palabras para predecir el ratio de éxito de financiamiento de un proyecto crowdfunding. Ante esta situación, los autores tuvieron como objetivo analizar el impacto del contenido textual en un proyecto de Kickstarter a partir del análisis de sus palabras clave que determinen el ratio de éxito de su financiamiento.

De acuerdo a la metodología seguida por los autores, en primer lugar se recolectaron las descripciones de proyectos de Kickstarter de la categoría “Juegos”, desde el 1 de julio hasta el 30 de agosto del 2018, de los cuales la distribución quedó en 50 proyectos que alcanzaron el 100 % de su meta y otros 50 proyectos que no alcanzaron por lo menos el 75 %. A continuación, se pre-procesó la data eliminando caracteres especiales, términos distintos al idioma inglés y palabras de parada en inglés para poder armar una matriz de Término-Documento que usará los pesos TF-IDF. Los autores propusieron 2 modelos de Máquinas de Vectores de Soporte con

Eliminación de Características Recursivas (SVM-RFE), usando las herramientas “Weka 3.8” y “libSVM”, los cuales fueron comparados. Luego, se seleccionaron las características realizando experimentos de validación cruzada antes de la construcción del modelo predictivo.

Finalmente, se evaluaron los resultados con la métrica de exactitud. Se realizó hasta dos experimentos manipulando los conjuntos de datos para el SVM-RFE y el mejor resultado fue una exactitud de 78.90 % para predecir el ratio de éxito de un proyecto crowdfunding a partir de 50 palabras claves importantes que se encuentren dentro de su contenido, las cuales se detallan en la Figura 18.

1	RUN	18	INCLUDE	35	BASE
2	KICKSTARTER	19	PLEASE	36	BOOK
3	PLEDGE	20	RULES	37	FINAL
4	ALWAYS	21	CHECK	38	ART
5	STRETCH	22	BOX	39	GOALS
6	HAND	23	ADDITIONAL	40	KEEP
7	REWARDS	24	LITTLE	41	PROJECTS
8	CAMPAIGN	25	CARDS	42	SHIP
9	LOOK	26	POSSIBLE	43	NUMBER
10	BACKERS	27	ADD	44	CARD
11	SHIPPING	28	PRICE	45	DELIVERY
12	INCLUDED	29	COMES	46	PRINT
13	INCLUDING	30	RECEIVE	47	TOTAL
14	RISKS	31	PAGE	48	FULL
15	PRINTING	32	COPY	49	ADVENTURE
16	PRODUCT	33	PRODUCTS	50	FAMILY
17	QUALITY	34	COPIES		

Figura 18. Palabras clave seleccionadas que afectan el ratio de éxito según modelo SVM-RFE.

Fuente: L.-S. Chen y Shen (2019). *Finding the Keywords Affecting the Success of Crowdfunding Projects.* (p. 571)

En el acta de la conferencia «2019 Portland International Conference on Management of Engineering and Technology (PICMET)», realizada del 25 al 29 de agosto del 2019 en Portland, Estados Unidos, Chaichi y Anderson (2019) publicaron el artículo titulado «Deploying Natural Language Processing to Extract Key Product Features of Crowdfunding Campaigns: The Case of 3D Printing Technologies on Kickstarter», el cual traducido al español significa «Implementación del procesamiento del lenguaje natural para extraer características clave del producto de las campañas de crowdfunding: el caso de las tecnologías de impresión 3D en Kickstarter».

Debido a la dificultad para predecir éxito de campañas en Kickstarter a partir de información textual (data no estructurada), los autores propusieron predecir éxito de campañas de proyectos de impresión 3D en Kickstarter a partir de la extracción de características de producto a partir de información textual como título, resumen y descripción.

De acuerdo a la metodología seguida por los autores, en primer lugar se extrajo el contenido textual (incluyendo título, resumen y descripción de campaña) de 528 proyectos de tecnología que contenían las palabras «3d printer» (impresora 3D) recopilados en Septiembre del 2017 usando la herramienta Selenium, de los cuales al eliminar aquellos que no contenían estas 3 variables textuales y algunos duplicados, quedaron finalmente 246 proyectos. El siguiente paso consistió en pre-procesar la data con algunas librerías de Python, incluyendo el paquete de R UDPipe, para armar el corpus final. UDPipe permitió generar segmentación de oraciones, tokenización, etiquetado POS (Part Of Speech), lematización, árboles de análisis de dependencia y colocaciones (palabras que continúan a otra). A continuación, se usó el algoritmo basado en grafos TextRank para extraer palabras claves, así como la técnica no supervisada RAKE (Extracción automática rápida de palabras clave) para extraer frases clave. Se probaron, además, otras tres técnicas más. Finalmente, los resultados de las 6 técnicas para extraer frases claves fueron comparados y se llegaron a conclusiones basadas en las similitudes entre estas.

Finalmente, se encontraron palabras con mayor frecuencia según cada una de las 6 técnicas de extracción de palabras clave (que incluyen frecuencia de términos por n gramas según cada algoritmo mencionado anteriormente), ilustradas en nubes de palabras, como por ejemplo relacionadas a disponibilidad, buenas características y precios cómodos de impresoras 3D. En la Figura 19 se representa la nube de palabras de las palabras clave de 3 y 4 gramas usando el algoritmo TextRank. También se observaron 2 problemas: no todas las palabras con mayor frecuencia son características del producto, y ambigüedad de algunas palabras.

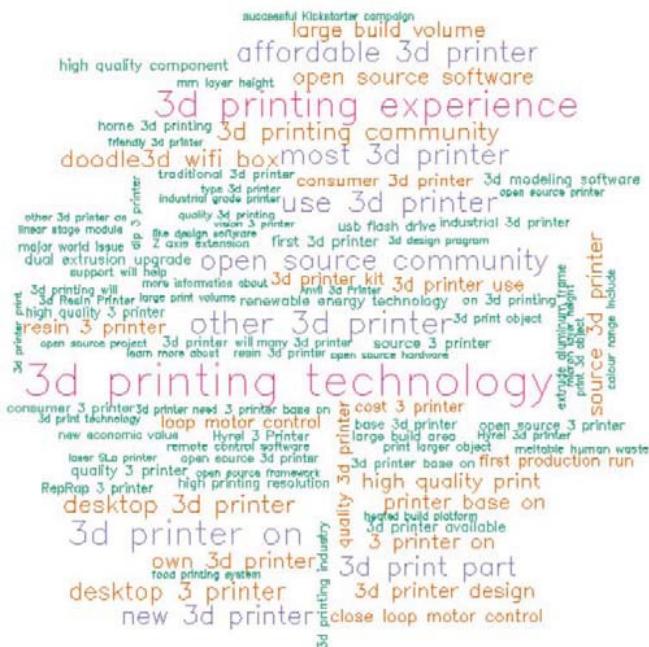


Figura 19. Nube de palabras clave de 3 y 4 gramas usando el algoritmo TextRank.

Fuente: Chaichi y Anderson (2019). *Deploying Natural Language Processing to Extract Key Product Features of Crowdfunding Campaigns: The Case of 3D Printing Technologies on Kickstarter*. (p. 6)

Shafqat y Byun (2019) publicaron un artículo titulado «Topic Predictions and Optimized Recommendation Mechanism Based on Integrated Topic Modeling and Deep Neural Networks in Crowdfunding Platforms», el cual traducido al español significa «Predicciones de temas y mecanismo de recomendación optimizado basado en modelos integrados de temas y redes neuronales profundas en plataformas de financiación colectiva», para la revista científica «Applied Sciences» en el año 2019.

Los comentarios de los usuarios al momento de evaluar algún producto de su interés en el comercio electrónico pasan desapercibidos. Estas reseñas suelen ser importantes en las plataformas de financiamiento colectivo dado que influye en otros usuarios en patrocinar un proyecto. Ante ello, los autores plantearon fusionar técnicas de modelado de lenguaje con redes neuronales para identificar patrones de discusión ocultos en los comentarios.

De acuerdo a la metodología seguida por los autores, el primer paso fue elaborar un flujo para la creación del conjunto final de datos, excluyendo potenciales proyectos fraudulentos en Kickstarter según distintas fuentes en la web, análisis de contenido y otros criterios para la lista restante de enlaces de campañas, y finalmente almacenando los comentarios en archivos.

Como resultado, se obtuvieron inicialmente más de 645 mil comentarios provenientes de 600 proyectos. Sin embargo, estos se redujeron a 504,184 comentarios (841 en promedio por proyecto) cuyo contenido era visible públicamente. El 70% fue destinado al subconjunto de entrenamiento, mientras que el resto, al de prueba. A continuación, se elaboró la arquitectura del modelo híbrido, ilustrado en la Figura 20, la cual comienza recibiendo de entrada a los comentarios para crear los vectores de palabras en el tiempo t de una campaña. El híbrido está basado en una red LSTM, alimentados con la distribución de temas latentes aprendidos del modelo pre-entrenado LDA. Además, se utilizó el algoritmo Optimización por Enjambre de Partículas (PSO) para optimizar las recomendaciones. Como resultado del proceso LDA, estos generan vectores latentes de temas, que alimentan al modelo LSTM con el fin de que logre predecir la clase del tema de la siguiente palabra en el documento. Con estas salidas, el módulo de recomendación sugiere proyectos basados en los temas descubiertos por el patrocinador.

Finalmente, tanto el modelo propuesto (LSTM-LDA) como los modelos de referencia fueron evaluados con la métrica de exactitud para 300 épocas de entrenamiento. El modelo de los autores RNN-LDA logró la mejor performance, alcanzando una exactitud de aproximadamente 0.96, frente al 0.94 de una red NN-LDA y 0.92 de una red NN.

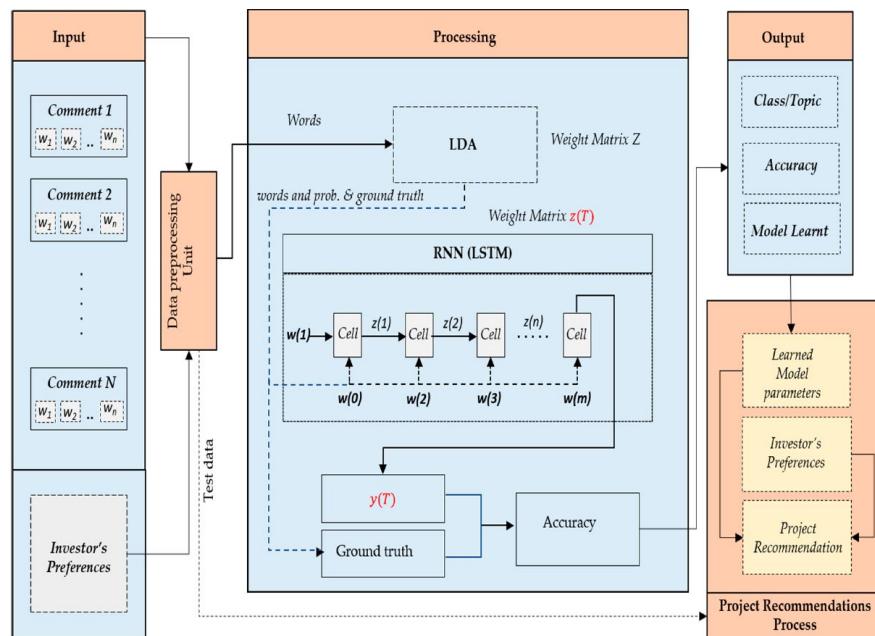


Figura 20. Arquitectura del sistema propuesto para predicción de temas y recomendaciones optimizadas.

Fuente: Shafqat y Byun (2019). *Topic Predictions and Optimized Recommendation Mechanism Based on Integrated Topic Modeling and Deep Neural Networks in Crowdfunding Platforms.* (p. 8)

Fernandez-Blanco et al. (2020) publicaron un artículo titulado «Key Factors for Project Crowdfunding Success: An Empirical Study», el cual traducido al español significa «Factores clave para el éxito del crowdfunding en proyectos: un estudio empírico», para la revista internacional académica «Sustainability» en el año 2020.

Desde sus inicios, el crowdfunding ha servido como respuesta al problema de financiamiento para proyectos innovadores. Sin embargo, la repentina popularidad del crowdfunding atribuida al impacto de los proyectos más exitosos ha llevado a generalizar esta mecánica a cualquier proyecto nuevo con riesgo de desajuste. Por ello, la mayor incertidumbre para un creador de proyectos es desconocer la garantía de financiamiento hacia el suyo. El objetivo de los autores fue identificar los factores clave y atributos que caracterizan a un proyecto, agrupados en clústers por su similitud, que influyen en el éxito o fracaso de un proyecto y definir estereotipos de comportamiento que pueden estar asociados a nuevos proyectos para ayudar a orientar la campaña hacia el logro de su objetivo.

Los autores escogieron la metodología CRISP-DM debido a su flexibilidad y su suficiencia para lograr la conversión de datos en conocimiento de forma organizada. De acuerdo a la metodología seguida, el primer paso realizado fue el entendimiento del problema, en donde se formularon las necesidades y objetivos del estudio. A continuación, se recolectaron más de 45 mil proyectos entre Marzo del 2009 y Enero del 2012, distribuidos en 12 categorías. De esta cantidad, solamente se trabajó con 23,941 proyectos concluidos, es decir, cuyo estado de financiamiento fue exitoso o fracasado. Luego se seleccionaron 13 variables numéricas, de las cuales según su atributo se agruparon en 3 tipos: variables asignadas antes de la campaña (meta, mes, niveles, duración, contribución mínima, contribución máxima, diferencia entre contribución máxima y mínima), variables que son modificadas en el transcurso de la campaña (número de comentarios de patrocinadores, número de actualizaciones en el proyecto, número de patrocinadores), y variables que dependen del anterior tipo (monto contribuido, porcentaje de financiamiento de la meta, monto promedio contribuido por patrocinador). Luego, a través del análisis de correlaciones entre ellas, se corroboró las hipótesis de trabajos previos respecto a la correlación entre las variables *backers* (número de patrocinadores), *pledged* (monto contribuido) y *comments* (número de comentarios). Para determinar los factores clave, se generaron 6 clústers a partir de los datos utilizando el algoritmo *K-means*. Este método de Minería de Datos permitió identificar qué zonas dentro del mapa generado de los grupos se asociaban más a proyectos exitosos o fracasados como se ilustra en la Figura 21. Cada uno de los grupos fue distribuido según el ratio de éxito y bajo qué variable fue estudiada.

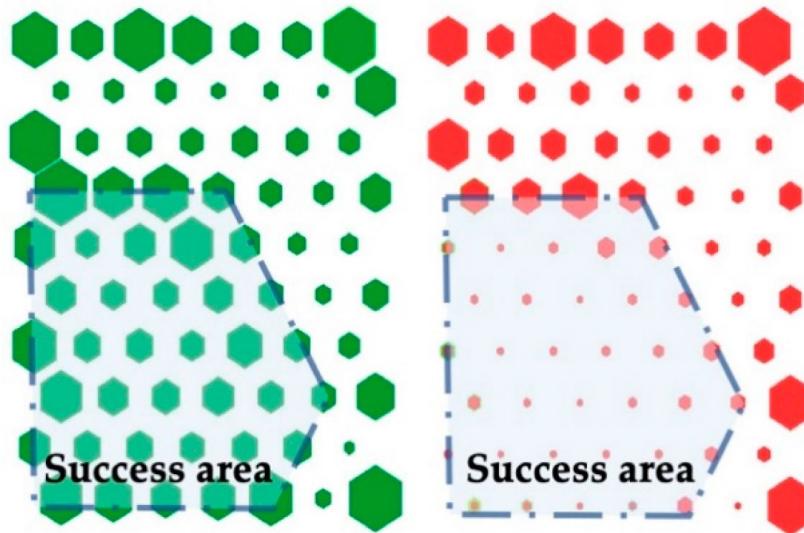


Figura 21. Mapas de proyecciones de región de éxito o fracaso generado por K-means.

Fuente: Fernandez-Blanco et al. (2020). *Key Factors for Project Crowdfunding Success: An Empirical Study*. (p. 9)

De esta manera, finalmente, se logró crear 6 clústers con características definidas cada uno y asignándose un nombre asociado a ellas. Siguiendo el orden ascendente, estos fueron nombrados respectivamente «Agujero ancho», «Colecciones principales», «Patrocinadores comprometidos», «Advertencia», «Agujero profundo» y «Meta épica», donde el tercero fue el mejor del grupo, como se observa en la Figura 22.

	Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5	Cluster 6
Art	43.65%	72.95%	91.53%	64.41%	46.76%	44.64%
Comics	28.44%	66.47%	92.20%	42.67%	34.07%	39.34%
Dance	63.95%	88.10%	90.91%	86.05%	64.74%	44.19%
Design	22.22%	52.22%	91.54%	48.12%	29.29%	42.17%
Fashion	19.69%	41.74%	82.76%	53.76%	16.23%	33.60%
Food	33.33%	67.42%	91.23%	58.38%	30.05%	42.40%
Games	10.71%	42.62%	83.96%	30.14%	16.03%	24.07%
Music	52.29%	79.97%	96.79%	77.00%	46.26%	64.74%
Photography	32.14%	59.20%	88.11%	51.56%	30.24%	38.98%
Publishing	26.08%	57.47%	87.84%	48.42%	27.28%	29.95%
Technology	20.00%	50.00%	93.46%	38.00%	16.36%	30.22%
Theater	61.98%	83.68%	92.73%	76.19%	67.40%	52.88%

Figura 22. Distribución de clústers según ratio de éxito por categoría.

Fuente: Fernandez-Blanco et al. (2020). *Key Factors for Project Crowdfunding Success: An Empirical Study*. (p. 11)

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 et al., 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 (ver Figura 23). El primero es la regresión, que consiste en obtener como resultado un número específico a partir de un conjunto de variables de las características; 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 (Zambrano, 2018).

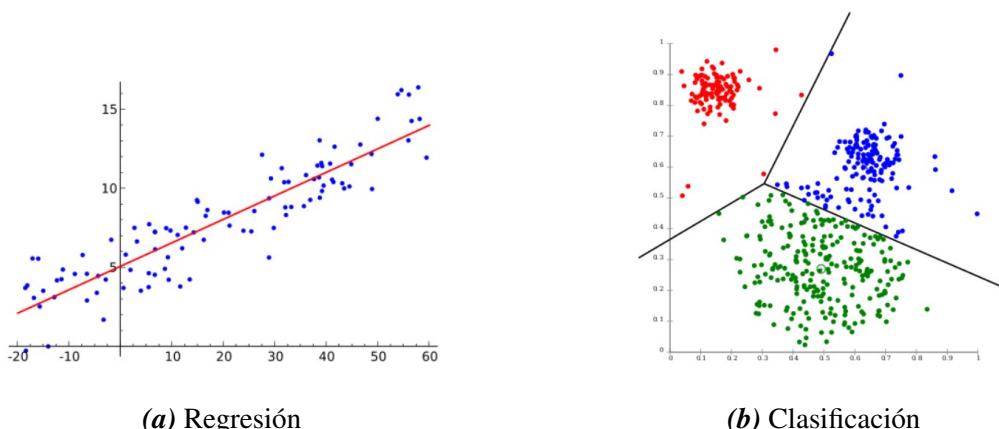


Figura 23. Ejemplo de algoritmos de regresión y clasificación.

Fuente: Zambrano (2018). *¿Aprendizaje supervisado o no supervisado? Conoce sus diferencias dentro del machine learning y la automatización inteligente.*

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 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 1 y su variante en la Ecuación 2:

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

$$\operatorname{argmax}_{v \in V} \sum_{i=1 \dots k, x_i \in v} W_i \quad (2)$$

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 24:

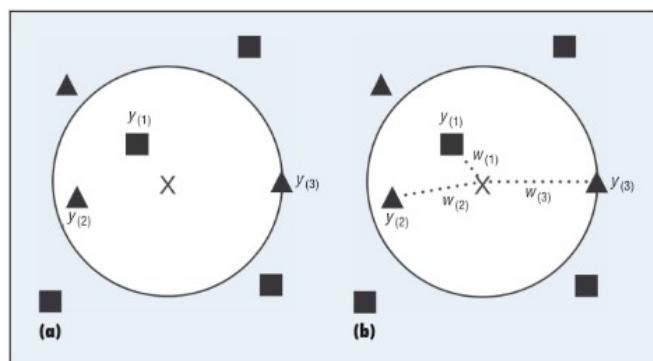


Figura 24. Algoritmo de K Vecinos más cercanos con pesos ponderados.

Fuente: Sancho Caparrini (2018). *Clasificación Supervisada y No Supervisada*.

- **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 3:

$$\sum_i \sum_j d(x_j^i, c_i)^2 \quad (3)$$

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 25, 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.

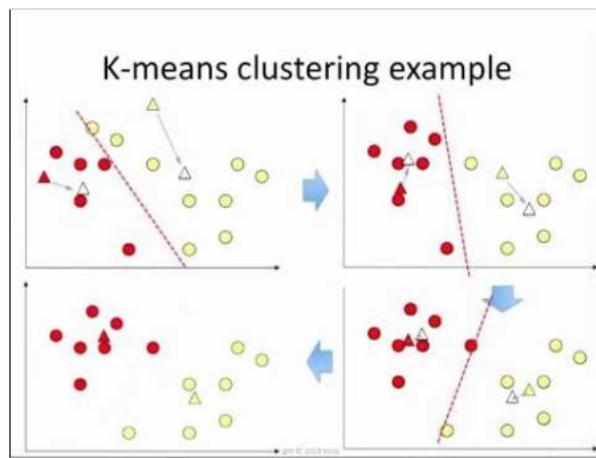


Figura 25. Funcionamiento del algoritmo de K medias.

Fuente: Sancho Caparrini (2018). *Clasificación Supervisada y No Supervisada*.

- **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).

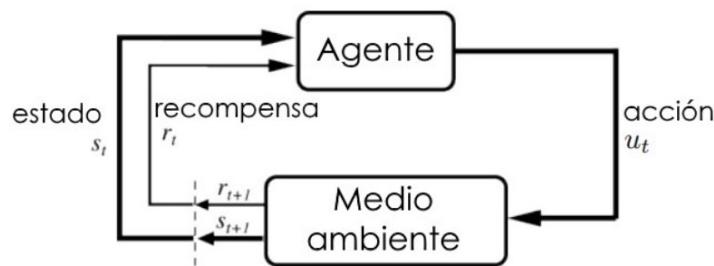


Figura 26. Componentes del Aprendizaje por Refuerzo.

Fuente: Sutton y Barto (2018). *Finite Markov Decision Processes*. (p. 48)

En resumen, y mencionando otro ejemplo, el aprendizaje por refuerzo está compuesto por un agente (Pacman) en un estado determinado (su ubicación o posición actual) dentro 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).

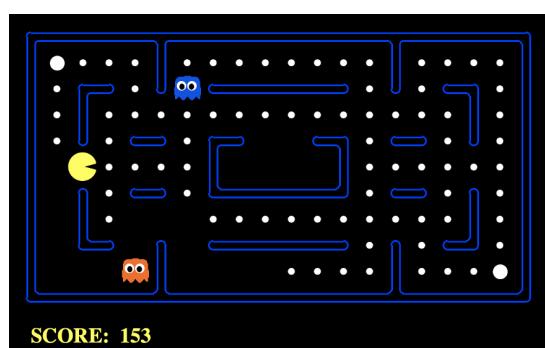


Figura 27. Ambiente del videojuego Pacman.

Fuente: Das (2020). *Reinforced Pac-man*.

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 realizar tareas hechas por los seres humanos, desde la identificación de imágenes hasta predecir 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 28.

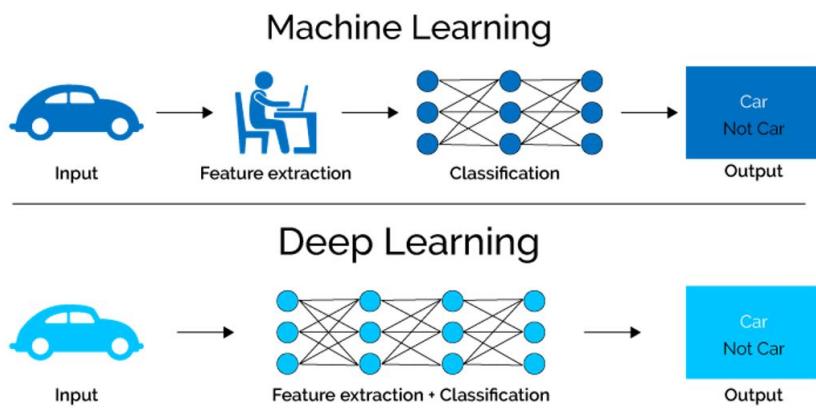


Figura 28. Diferencia entre Aprendizaje Automático y Aprendizaje Profundo.

Fuente: House of Bots (2018). *Most Popular 20 Free Online Courses to Learn Deep Learning*.

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 Aprendizaje Profundo Multimodal

El Aprendizaje Profundo Multimodal y Multitarea (*Multimodal and Multitask Deep Learning*) son un tipo de aprendizaje basado en la explotación de representaciones latentes en las redes neuronales profundas agrupando distintas modalidades (por ejemplo, audio, imágenes, texto, videos, etc) o múltiples tareas (predicción, clasificación, series de tiempo, entre otros) como en la Figura 29 (Deng & Liu, 2018).

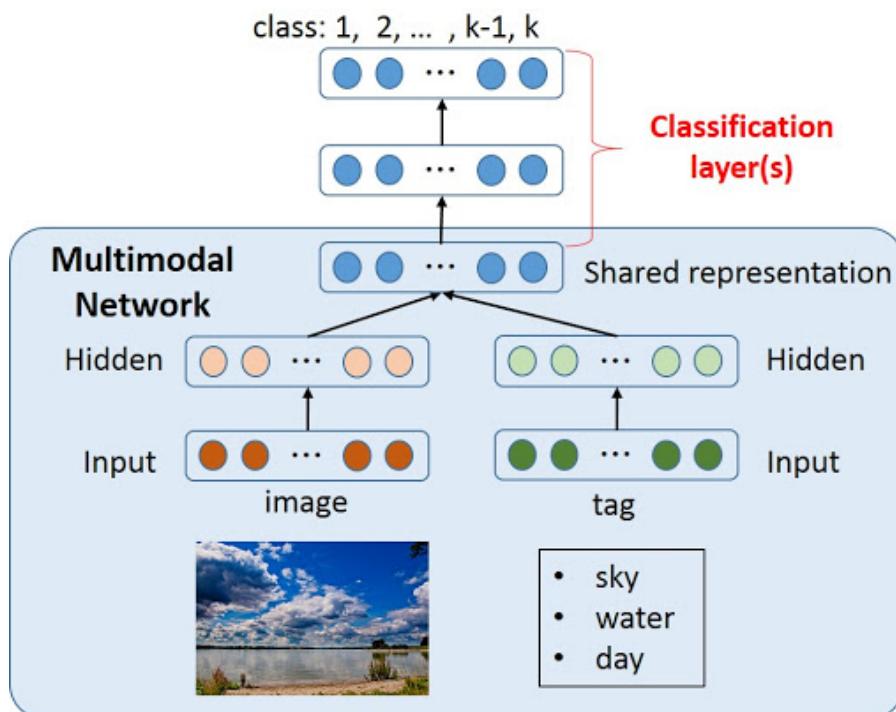


Figura 29. Ejemplo de modelo multimodal de imágenes y texto.

Fuente: Nishida y Nakayama (2015). *Multimodal gesture recognition using multi-stream recurrent neural network*.

Dado que las señales de diferentes modalidades llevan información complementaria sobre diferentes aspectos de un objeto, evento o actividad, los modelos de este tipo son capaces de realizar inferencias más sólidas. Algunas de las técnicas multimodales existentes son la fusión temprana y tardía, fusión de modelos, ensamblado de modelos y Redes Neuronales Profundas. Las características unidas que se modelan en grupo para tomar una decisión son llamadas «enfoques aditivos» ya que pueden recopilar información útil y logran realizar predicciones colectivamente (Liu et al., 2018).

Según Baheti (2020), trabajar con modelos multimodales no solo mejora las redes neuronales, sino que permite lograr una mejor extracción de características de todas las fuentes para realizar predicciones a mayor escala. Entre los principales beneficios se menciona que los datos, al provenir de distintas fuentes, como el ejemplo de la Figura 30, brinda información que se complementa entre sí y revela patrones ocultos que no son observables al trabajarse las modalidades de manera individual, mejorando así la predicción.

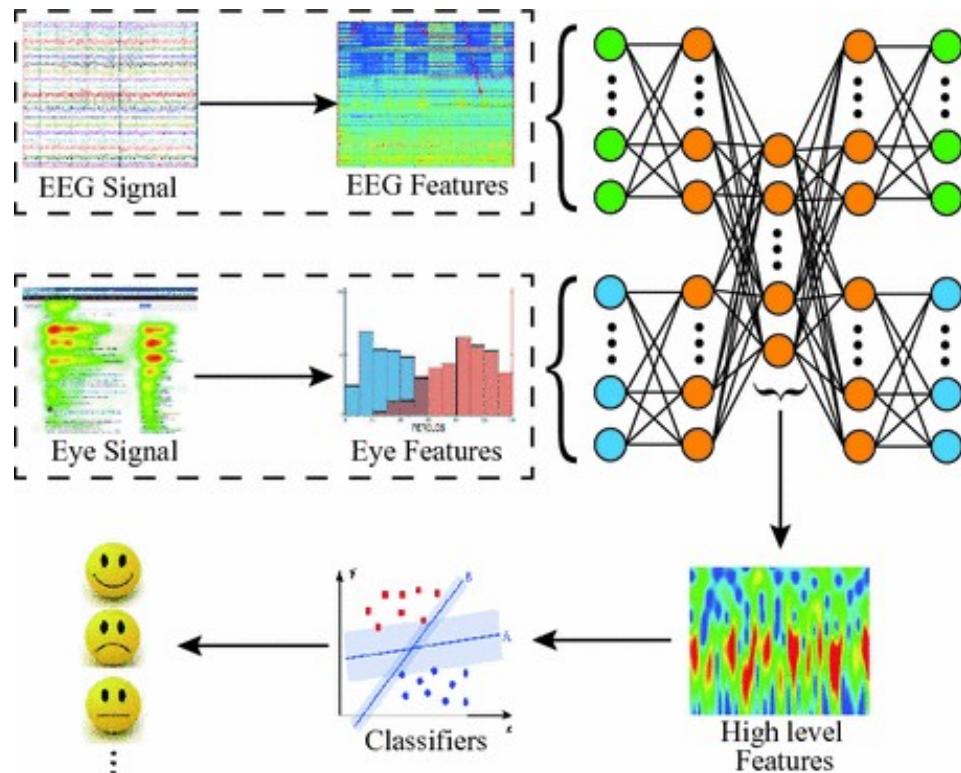


Figura 30. Ejemplo de modelo multimodal de señal de electroencefalografía y de ojo.

Fuente: Baheti (2020). *Introduction to Multimodal Deep Learning*.

Brownlee (2018) afirma que el promedio del modelo puede mejorarse ponderando las contribuciones de cada submodelo a la predicción combinada por el rendimiento esperado del submodelo, el cual se denomina «generalización apilada». Este enfoque presenta 2 niveles, como se ilustra en la Figura 31:

- Nivel 0: los datos de este nivel son las entradas del conjunto de datos de entrenamiento y sirven para enseñar a predecir a los modelos de este nivel.
- Nivel 1: los datos de este nivel toman las salidas de los modelos del anterior nivel como entrada y enseñan a predecir a los modelos de este nivel (*meta-learner* o generalizador).

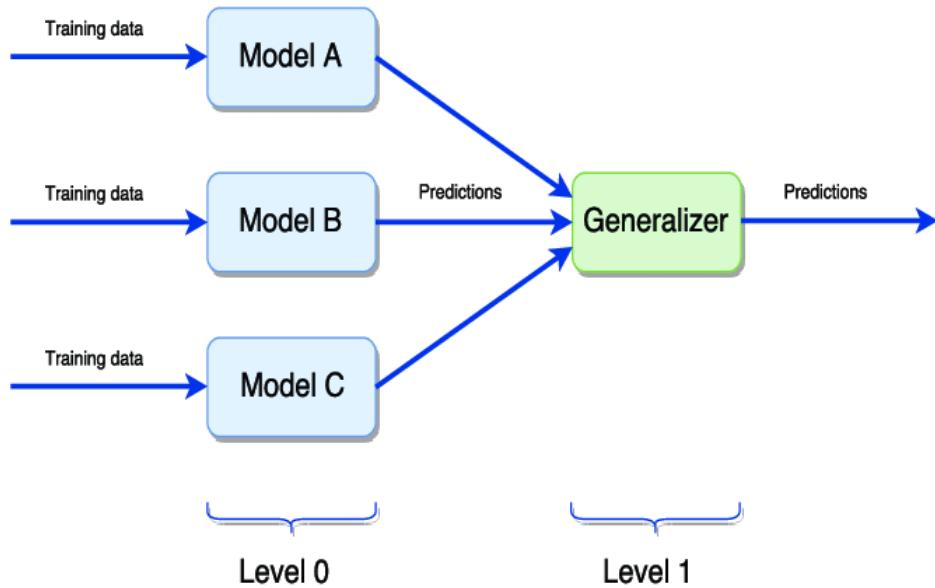


Figura 31. Ejemplo de aprendizaje ensamblado apilado.

Fuente: Divina et al. (2018). *Stacking Ensemble Learning for Short-Term Electricity Consumption Forecasting*. (p. 7)

Dentro de los modelos ensamblados apilados, se pueden utilizar 2 enfoques:

- **Modelo apilado separado:** Consiste en crear un conjunto de datos de entrada que resultan de las predicciones ensambladas de otros modelos por separado, los cuales servirán para entrenar un nuevo modelo y poder realizar la predicción.

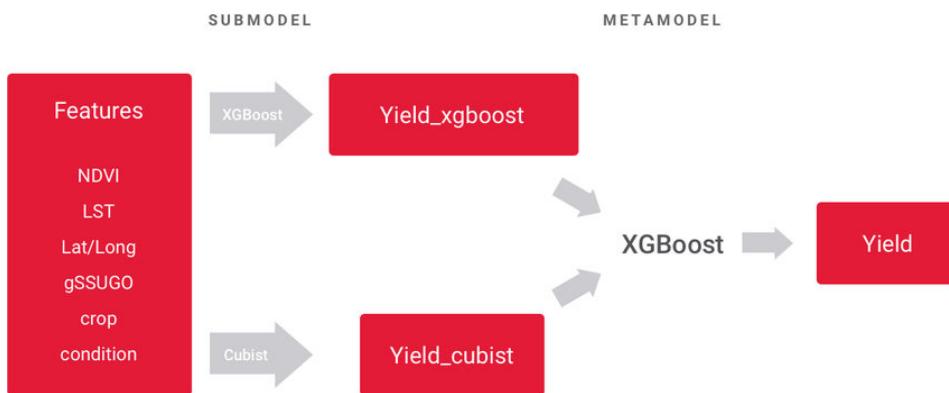


Figura 32. Ejemplo de modelo apilado separado.

Fuente: Cai et al. (2017). *Crop yield predictions - high resolution statistical model for intra-season forecasts applied to corn in the US*. (p. 8)

- **Modelo apilado integrado:** Las diferencias con el tipo anterior son el poder utilizar redes neuronales, implementar un modelo multicéfalo compuesto de otros modelos cargados que no necesariamente deben mantener la misma estructura gracias a asignar sus capas como «no entrenables» para evitar actualizar sus pesos antes de generar sus predicciones, y los resultados de los submodelos se implementan directamente al generalizador.

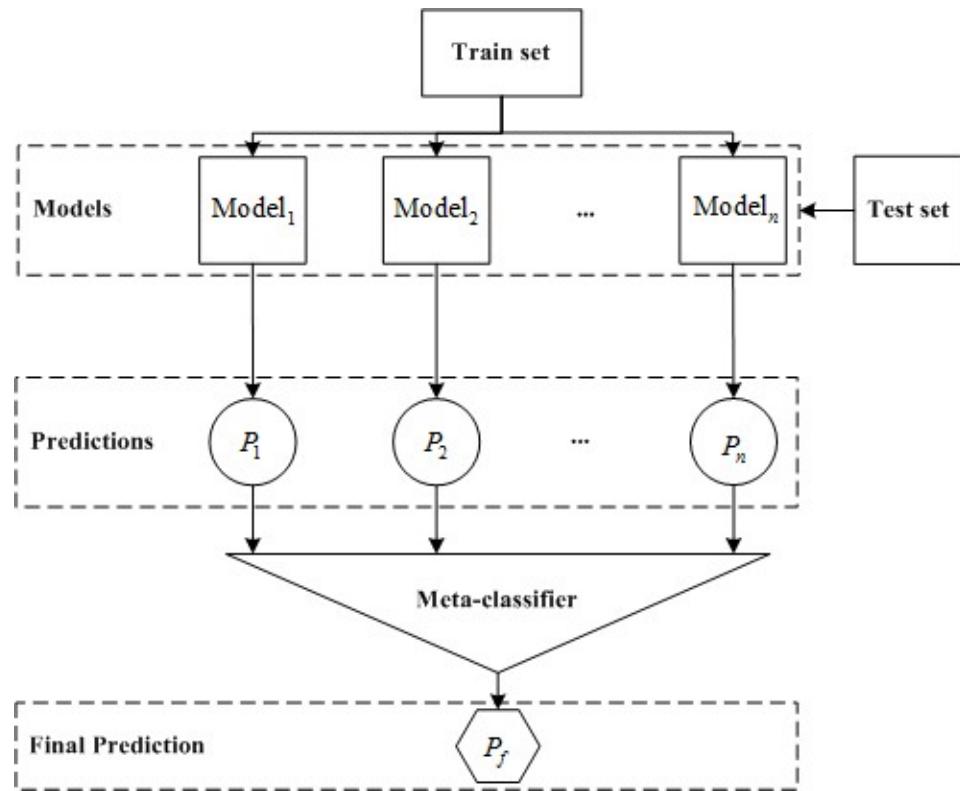


Figura 33. Ejemplo de modelo apilado integrado.

Fuente: Jiang et al. (2019). *SSEM: A Novel Self-Adaptive Stacking Ensemble Model for Classification.* (p. 2)

Para esta investigación, la propuesta consistió en un modelo de apilamiento integrado compuesto por diferentes modalidades, estructuras y conjuntos de datos de entrenamiento, para aprender a buscar mejor cómo combinar las predicciones de cada submodelo de entrada.

Sin embargo, estas técnicas de combinar distintas modalidades presentan algunas dificultades. De acuerdo con Liu et al. (2018), el gran desafío para diseñar modelos con distintas modalidades recae en el rendimiento de cada modalidad individual. Si una de ellas presenta una performance pobre, ya sea por la calidad de sus datos o los parámetros con la que fue configurada, afectará negativamente la performance del modelo ensamblado. Los autores mencionan esta desventaja con el siguiente ejemplo:

En el ejemplo de la creación de perfiles de usuario, el sexo y la edad de algunos usuarios, se pueden predecir con precisión mediante una foto de perfil clara, mientras que otros con una foto de perfil ruidosa o inútil (por ejemplo, dibujos animados) pueden tener la información más relevante codificada en su participación en las redes sociales, como publicaciones e interacciones con amigos, etc. En tal escenario, nos referimos a la modalidad afectada, en este caso la modalidad de imagen, como una modalidad débil. Hacemos hincapié en que esta debilidad puede depender de la muestra y, por lo tanto, no se controla fácilmente con algunos parámetros de sesgo global. Un algoritmo ideal debe ser robusto al ruido de esas modalidades débiles y seleccionar la información relevante de las modalidades fuertes por muestra, mientras que al mismo tiempo captura la posible complementariedad entre las modalidades. (p. 2).

2.2.5 Modelo Predictivo

Son modelos de datos estadísticos utilizados para predecir el comportamiento futuro. Se recopilan datos históricos y actuales, se formula un modelo estadístico, se realizan predicciones y se valida a medida que se dispone de datos adicionales. Los modelos predictivos analizan el rendimiento pasado para evaluar la probabilidad de que un agente muestre un comportamiento específico en el futuro. También se abarca la búsqueda de patrones ocultos (Gartner, 2019b).

2.2.6 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.

2.2.7 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 34 a continuación.

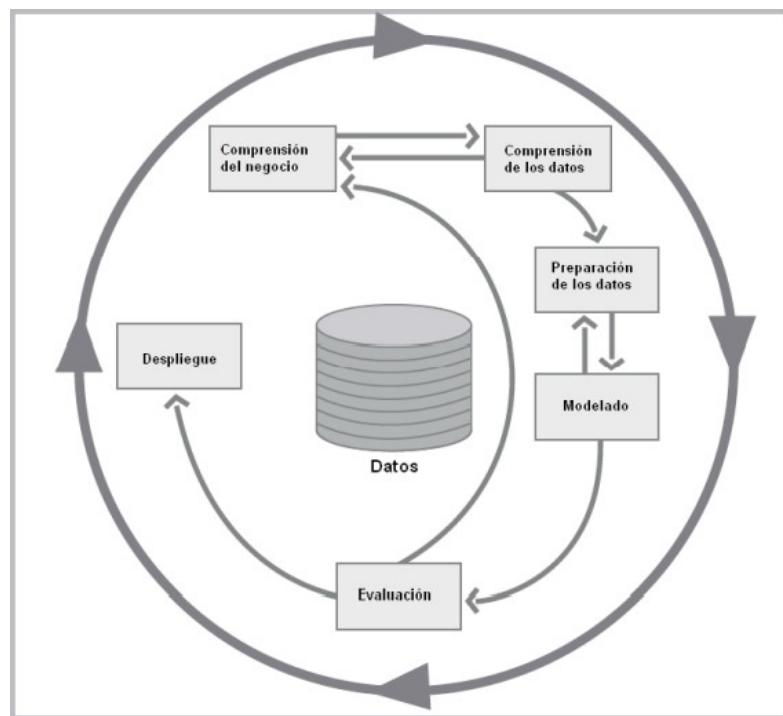


Figura 34. Fases de la metodología CRISP-DM.

Fuente: Braulio Gil y Curto Díaz (2015). *Customer Analytics: Mejorando la inteligencia del cliente a través de los datos.* (p. 19)

- En la comprensión del negocio se determinan los objetivos y requerimientos desde el lado del negocio, así como generar plan del proyecto.
- 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 35. A diferencia de la anterior, esta metodología se enfoca más en el modelado.

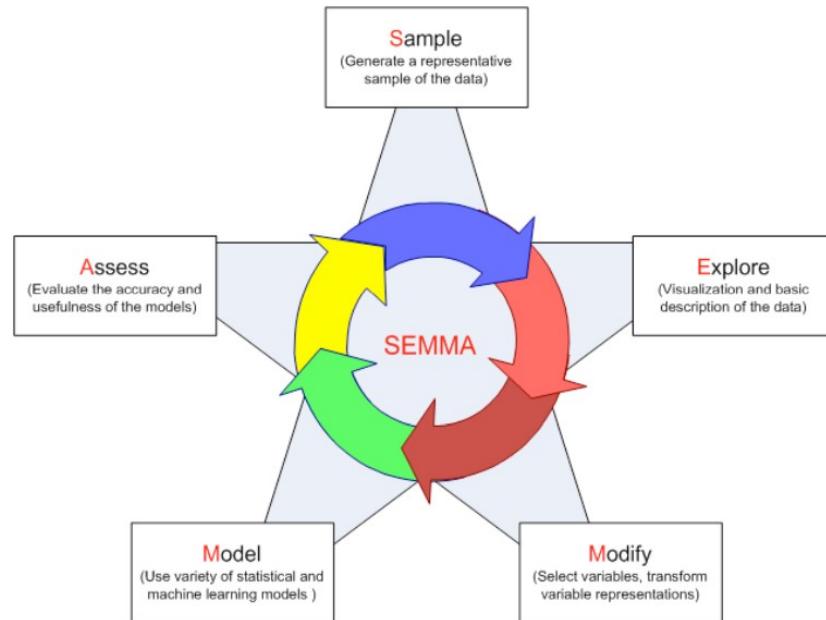


Figura 35. Fases de la metodología SEMMA.

Fuente: Braulio Gil y Curto Díaz (2015). *Customer Analytics: Mejorando la inteligencia del cliente a través de los datos.* (p. 20)

- En la Muestra (*Sample*) se crea una muestra significativa.

- 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 36.

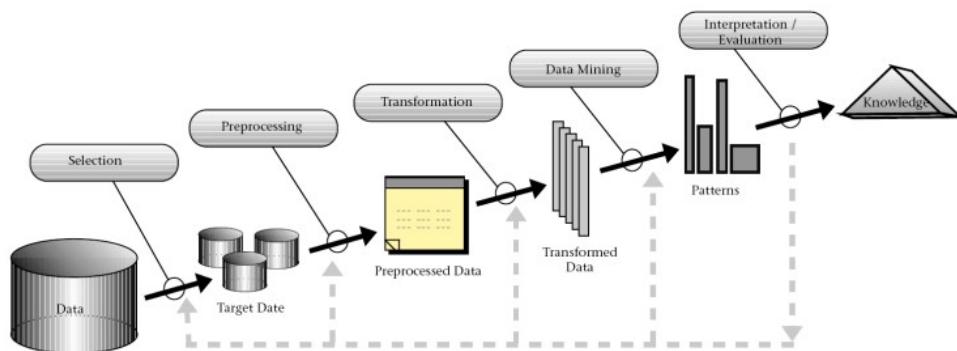


Figura 36. Fases de la metodología KDD.

Fuente: Braulio Gil y Curto Díaz (2015). *Customer Analytics: Mejorando la inteligencia del cliente a través de los datos.* (p. 21)

- 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.
- 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 1.

Tabla 1
Cuadro comparativo entre características de las tres metodologías.

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 Creación de un conjunto de datos de destino Limpieza de datos y pre-procesamiento Transformación de datos Elección de la tarea adecuada de Minería de datos Elección del algoritmo adecuado de Minería de datos Implementación del algoritmo de Minería de datos Interpretación de patrones minados Uso de conocimiento descubierto	Entendimiento del negocio Entendimiento de los datos Preparación de los datos Modelamiento Evaluación Despliegue	- Muestreo Exploración Modificación Modelo Evaluación -

Fuente: Shafique y Qaiser (2014). *A Comparative Study of Data Mining Process Models (KDD, CRISP-DM and SEMMA)*. (p. 221)

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.

2.2.8 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 37, por los ya mencionados anteriormente McCulloch y Pitts.

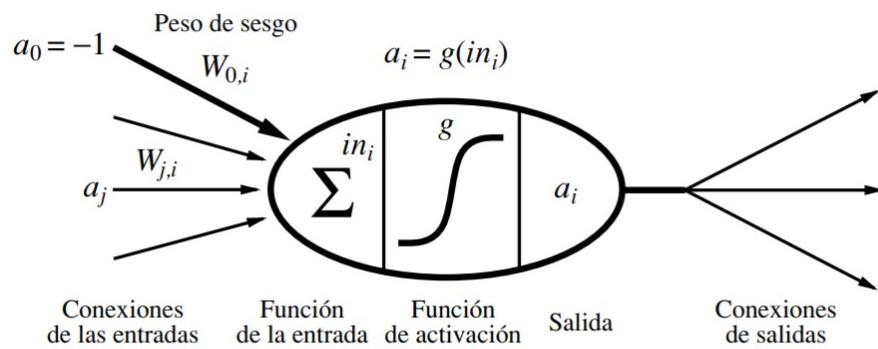


Figura 37. Modelo para representar una neurona propuesto por McCulloch y Pitts (1943).

Fuente: Russell y Norvig (2004). *Inteligencia Artificial: Un Enfoque Moderno*. (segunda edición). (p. 839)

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. La última figura descrita muestra, además de los pesos, funciones de activación tanto para la entrada (a_j) como para la salida (a_i).

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).

$$in_i = \sum_{j=0}^n W_{j,i} * a_j + \theta_i \quad (4)$$

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

$$a_i = g(in_i) = g\left(\sum_{j=0}^n W_{j,i} * a_j + \theta_i\right) \quad (5)$$

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 (Figura 38).

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 39 y su fórmula para calcular su nuevo valor es la Ecuación 6:

$$a = Logsig(n) = \frac{1}{1 + e^{-n}} \quad (6)$$

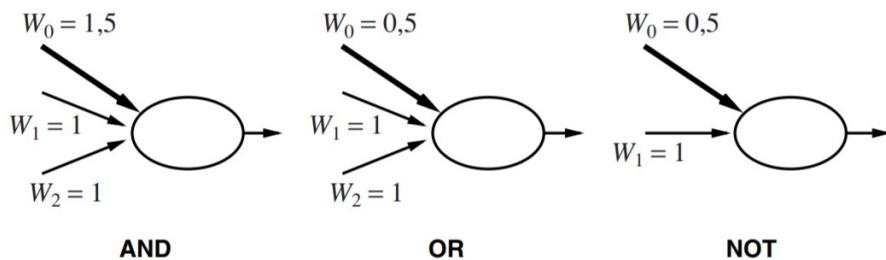


Figura 38. Nodos con funciones de activación umbral en forma de puertas lógicas.

Fuente: Russell y Norvig (2004). *Inteligencia Artificial: Un Enfoque Moderno*. (segunda edición). (p. 840)

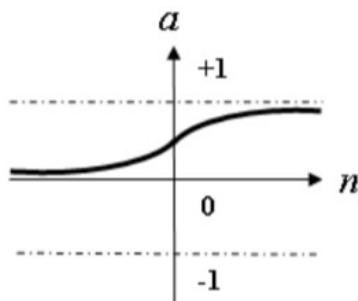


Figura 39. Función de activación sigmoide.

Fuente: Dorofki et al. (2012). *Comparison of Artificial Neural Network Transfer Functions Abilities to Simulate Extreme Runoff Data*. (p. 40)

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 7:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^m [-y_i * \log(h_\theta(x_i)) - (1 - y_i) * \log(1 - h_\theta(x_i))] \quad (7)$$

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. 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 40 (Sancho Caparrini, 2017).

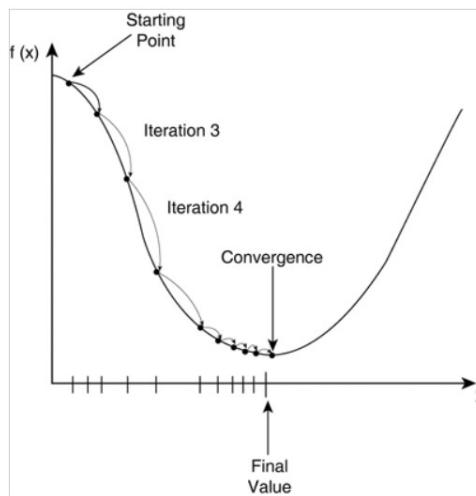


Figura 40. Ilustración del algoritmo gradiente descendiente.

Fuente: Sancho Caparrini (2017). *Entrenamiento de Redes Neuronales: mejorando el Gradiente Descendiente.*

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 8 (IArtificial.net, 2019b):

$$W_{(t+1)} = W_{(t)} - \alpha \left(\frac{\partial MSE}{\partial W} \right) \quad (8)$$

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 41.

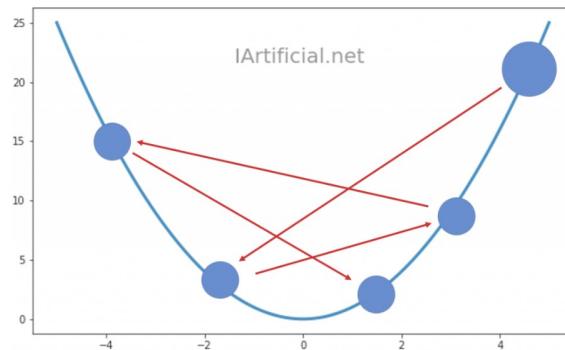


Figura 41. Actualización de pesos W con el algoritmo.

Fuente: IArtificial.net (2019b). *Método del Gradiente Descendiente*.

- **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 9:

$$W_{ji}(t+1) = W_{ji}(t) + [\alpha \delta_{pj} y_{pj} + \beta \Delta W_{ji}(t)] \quad (9)$$

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

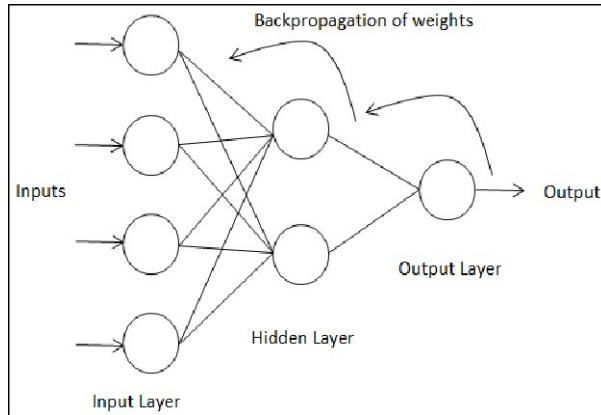


Figura 42. Capa oculta simple MLP con propagación hacia atrás.

Fuente: IArtificial.net (2019b). *Método del Gradiente Descendiente*.

Para entender mejor la teoría y la fórmula de actualización de pesos, se seguirá el siguiente ejemplo del conjunto de redes de la Figura 43.

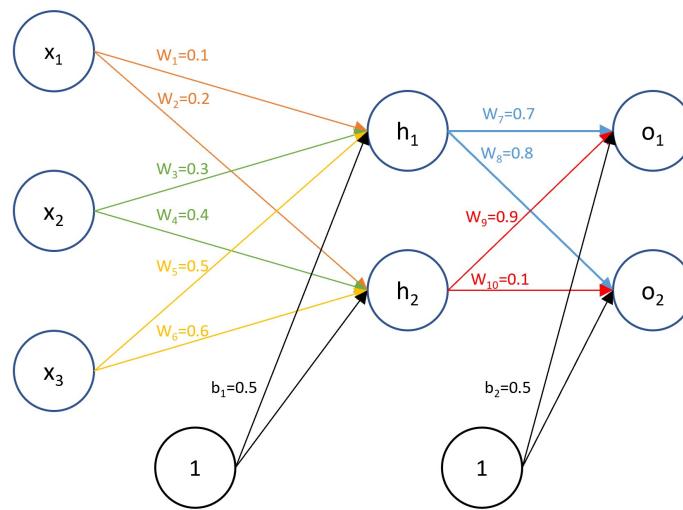


Figura 43. Redes neuronales de ejemplo.

Fuente: A Not So Random Walk (2019). *Backpropagation Example With Numbers Step by Step*.

Se tiene una red neuronal con tres nodos de entradas ($x_1=1, x_2=4$ 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 (Viera Balanta, 2013). Este último se calcula mediante la Ecuación 10:

$$E_k = (T_k - O_k) * O_k * (1 - O_k) \quad (10)$$

Donde:

T_k = salida correcta de cada nodo de salida

O_k = salida actual que cada nodo genera

Con estos errores, se retrocede hacia la capa oculta y se procede a calcular los nuevos pesos para sus nodos. Esto se realiza mediante la Ecuación 11:

$$W_{jk} = W_{jk} + L * E_k * O_j \quad (11)$$

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 12):

$$E_j = O_j * (1 - O_j) * \sum E_k * W_{jk} \quad (12)$$

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 sigmoidal 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 44 y su fórmula para calcular su nuevo valor es la Ecuación 13:

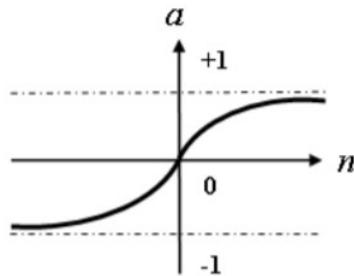


Figura 44. Función de activación tangente hiperbólica.

Fuente: Dorofki et al. (2012). *Comparison of Artificial Neural Network Transfer Functions Abilities to Simulate Extreme Runoff Data.* (p. 40)

$$a = \text{Tansig}(n) = \frac{2}{1 + e^{-2n}} - 1 \quad (13)$$

- **Función puramente lineal (purelin):** Se caracteriza porque su salida es igual a su entrada debido a su linealidad. Normalmente se usa para obtener los mismos valores de entrada. Se representa en la Figura 45 y se calcula mediante la Ecuación 14:

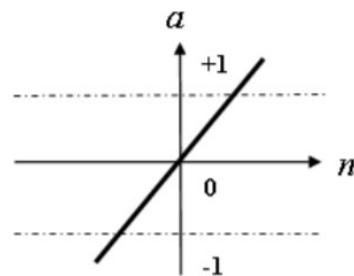


Figura 45. Función de activación puramente lineal.

Fuente: Dorofki et al. (2012). *Comparison of Artificial Neural Network Transfer Functions Abilities to Simulate Extreme Runoff Data.* (p. 40)

$$a = n \quad (14)$$

- **Función Unidad Lineal Rectificada (ReLU):** Se caracteriza por conservar los valores positivos y convertir los negativos de entrada en 0, con la finalidad de no considerarlos en la siguiente capa de convolución (SitioBigData.com, 2019b). Si bien tiene un buen desempeño en redes convolucionales y es muy usada para procesar imágenes, al no estar acotada pueden morirse demasiadas neuronas (Calvo, 2018b). Se representa en la Figura 46 y se calcula mediante la Ecuación 15:

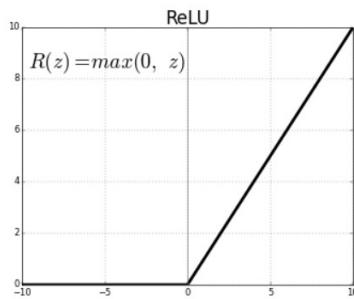


Figura 46. Función de activación ReLU.

Fuente: Machine Learning for Artists (2019). *Redes Neuronales*.

$$f(x) = \max(0, x) = \begin{cases} 0 & \text{para } x < 0 \\ x & \text{para } x \geq 0 \end{cases} \quad (15)$$

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 47. 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).

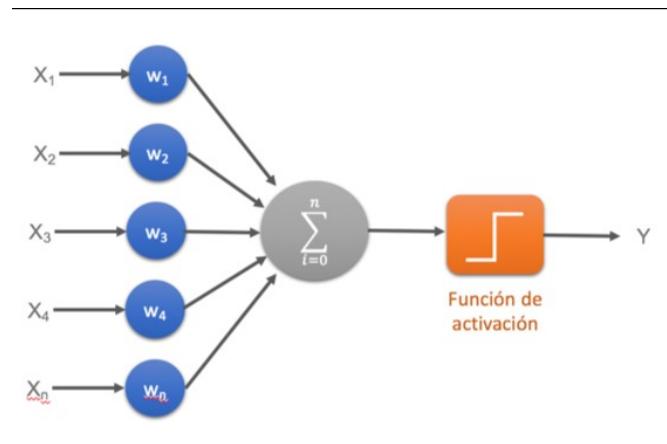


Figura 47. Ejemplo de perceptrón simple.

Fuente: Calvo (2017). *Clasificación de redes neuronales artificiales*.

- **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 48.

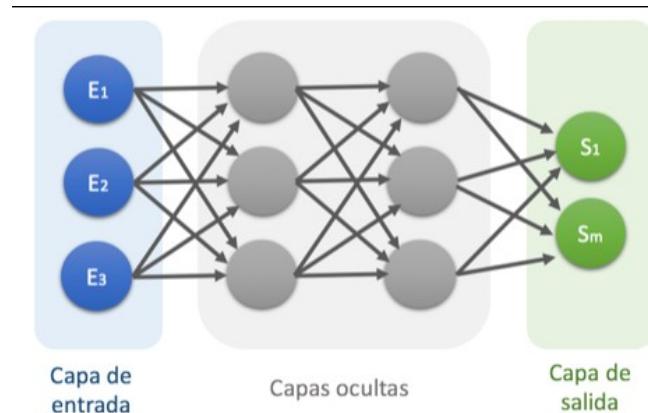


Figura 48. Ejemplo de perceptrón multicapa.

Fuente: Calvo (2017). *Clasificación de redes neuronales artificiales*.

- **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 49 (Calvo, 2017).

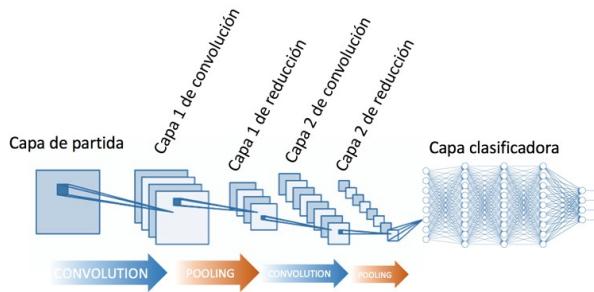
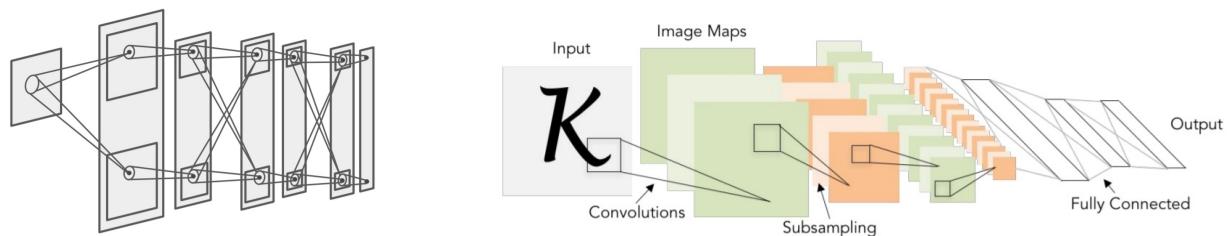


Figura 49. Ejemplo de red neuronal convolucional.

Fuente: Calvo (2017). *Clasificación de redes neuronales artificiales*.

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 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 et al., 2019).

En la Figura 50, se ilustran redes neuronales que dieron origen a la CNN. La primera imagen representa 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”, 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 y representado en la segunda imagen (F.-F. Li et al., 2019).



(a) Modelo Neocognitron de Fukushima (1980)

(b) Modelo LeNet-5 de LeCun (1998)

Figura 50. Modelos de redes neuronales que inspiraron a la CNN.

Fuente: F.-F. Li et al. (2019). *Convolutional Neural Networks*. (pp. 6, 15)

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 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 16 (Figueroa M., s.f.):

$$(f * g)(t) = \int_0^t f(t - \tau)g(\tau)d\tau \quad (16)$$

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

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 51.

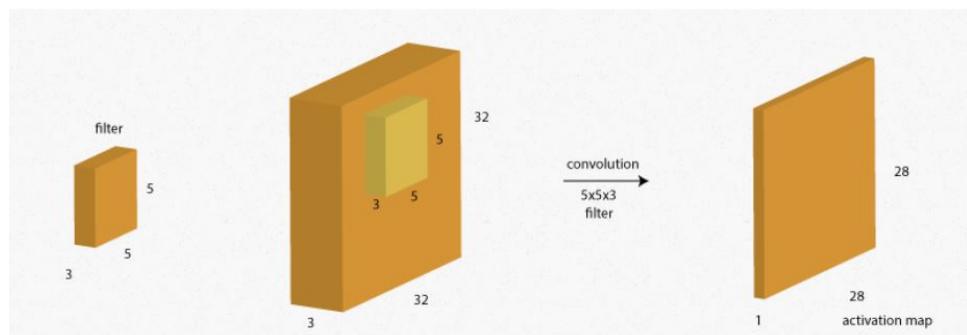


Figura 51. Ejecución de la convolución en una entrada.

Fuente: López Briega (2016). *Redes neuronales convolucionales con TensorFlow*.

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 ($32 \times 32 \times 3$). A ella se le aplica un filtro de dimensiones ($5 \times 5 \times 3$) 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 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 52.

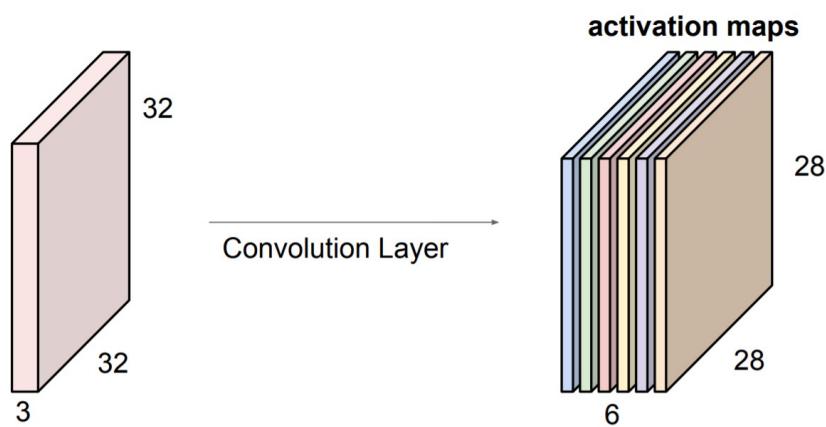


Figura 52. Generación de una nueva imagen a partir de filtros.

Fuente: F.-F. Li et al. (2019). *Convolutional Neural Networks*. (p. 36)

Asimismo, cada vez que se aplica una convolución a una imagen, se aplicará una función de activación como en la secuencia de la Figura 53.

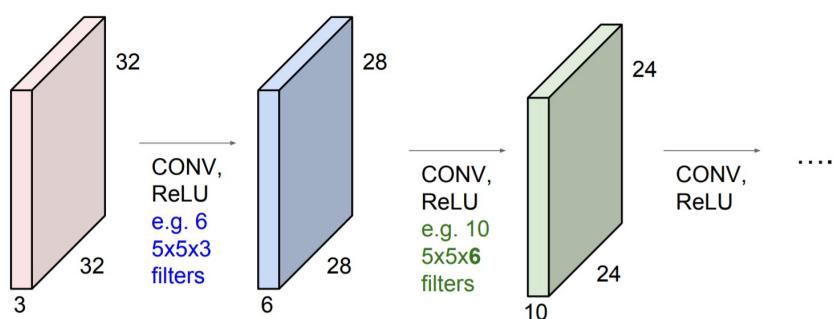


Figura 53. Secuencia de varias capas convolucionales.

Fuente: F.-F. Li et al. (2019). *Convolutional Neural Networks*. (p. 38)

A nivel visual, en la Figura 54 se aprecia un ejemplo de los resultados de aplicar varias convoluciones a una imagen.

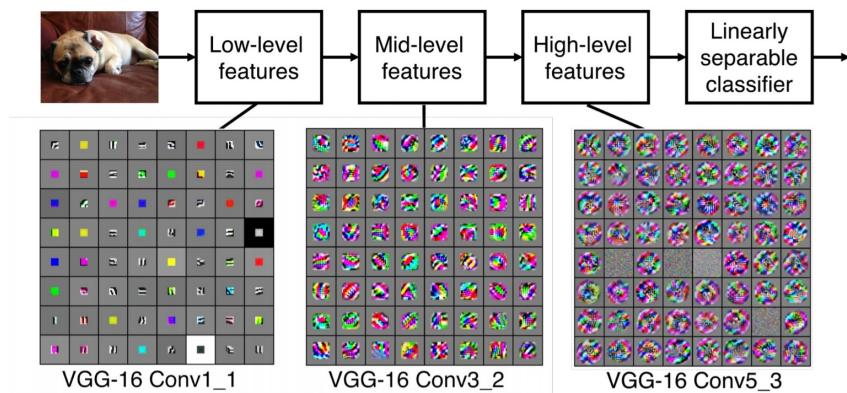


Figura 54. Extracción de características a partir de convoluciones.

Fuente: F.-F. Li et al. (2019). *Convolutional Neural Networks*. (p. 39)

Finalmente, se calcula el volumen de la dimensión de la salida de la Figura 55 mediante la Ecuación 17:

- ◊ Se tiene una entrada de dimensiones ($h * w * d$).
- ◊ Se tiene un filtro de dimensiones ($f_h * f_w * d$).

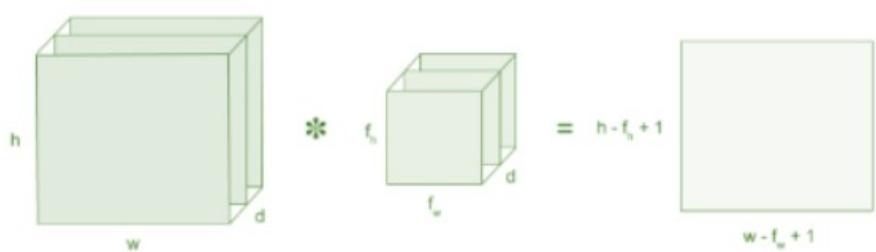


Figura 55. Ejemplo de matriz de imagen de entrada y un filtro.

Fuente: Prabhu (2018). *Understanding of Convolutional Neural Network (CNN) — Deep Learning*.

$$\text{Volumen} = (h - f_h + 1) * (w - f_w + 1) * 1 \quad (17)$$

- ◊ **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) 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 como en la Figura 56) con esta capa, se aplica la Ecuación 18:

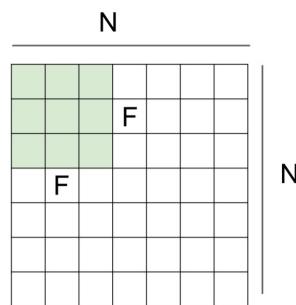


Figura 56. Dimensiones de una entrada y un filtro.

Fuente: F.-F. Li et al. (2019). *Convolutional Neural Networks*. (p. 54)

$$\text{Salida} = \frac{N - F}{S} + 1 \quad (18)$$

Donde:

N = tamaño del lado de la imagen de entrada

F = tamaño del lado del filtro

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

Si el paso es 1, los filtros se mueven a 1 pixel por vez, cuando si es 2, se mueven a 2 píxeles (como en la Figura 57) y así sucesivamente (Prabhu, 2018).

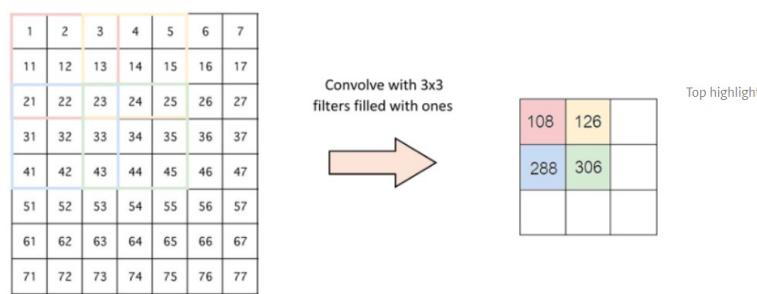


Figura 57. Paso de 2 píxeles por parte de un filtro.

Fuente: Prabhu (2018). *Understanding of Convolutional Neural Network (CNN) — Deep Learning*.

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 fórmula:

$$\text{Salida} = \frac{N - F + 2 * \text{Relleno}}{S} + 1 \quad (19)$$

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 58).

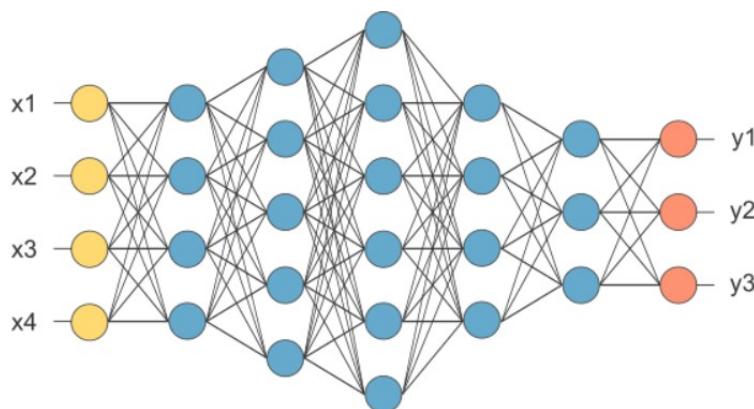


Figura 58. Aplanado de matrices luego de agrupar la capa.

Fuente: Prabhu (2018). *Understanding of Convolutional Neural Network (CNN) — Deep Learning*.

Para concluir, en la Figura 59 se representa la arquitectura completa de una Red Neuronal Convolucional resumiendo los conceptos anteriores.

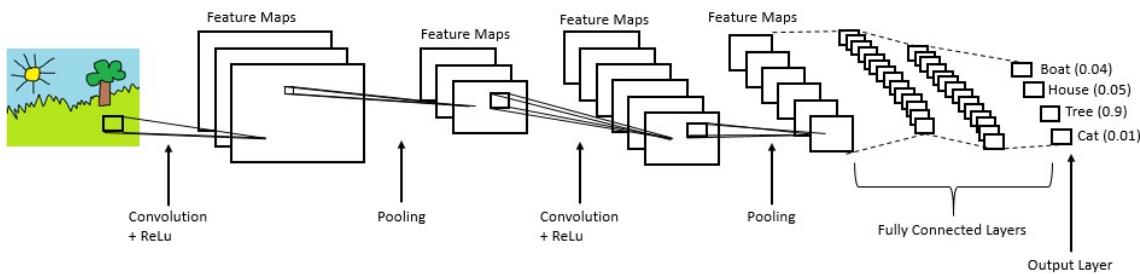


Figura 59. Arquitectura completa de una CNN.

Fuente: Prabhu (2018). *Understanding of Convolutional Neural Network (CNN) — Deep Learning*.

- **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 60, 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).

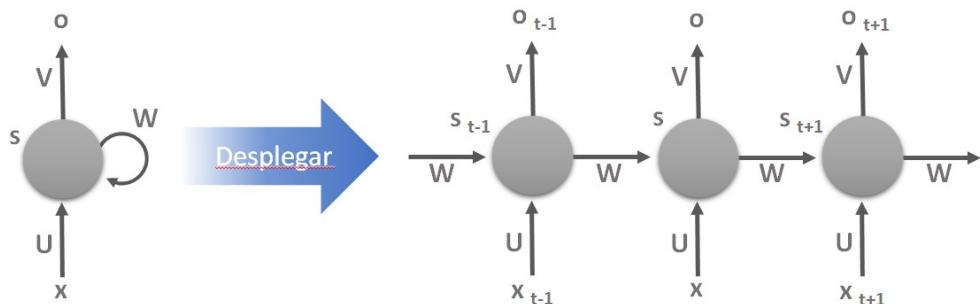


Figura 60. Ejemplo de red neuronal recurrente.

Fuente: Calvo (2018a). *Definición de Red Neuronal Recurrente*.

- **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, permitiendo distinguir 2 clases, como se muestra en la Figura 61 (Gandhi, 2018).

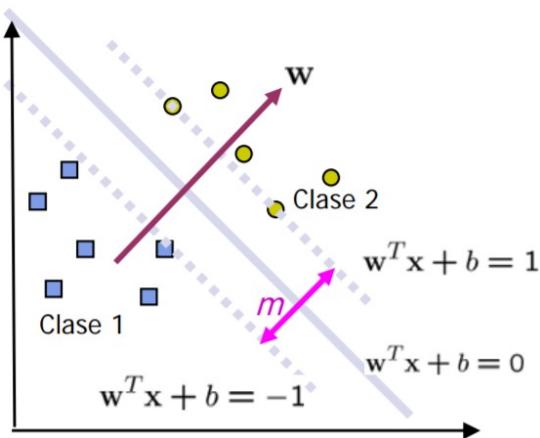


Figura 61. Hiperplano con dos clases separadas por una distancia m.

Fuente: Betancourt (2005). *Las Máquinas de Soporte Vectorial (SVMs)*.

Este algoritmo tiene sus orígenes en la década de los años 60 en Rusia, desarrollados por 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 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 la Figura 62 se observan casos linealmente y no linealmente separables, respectivamente.

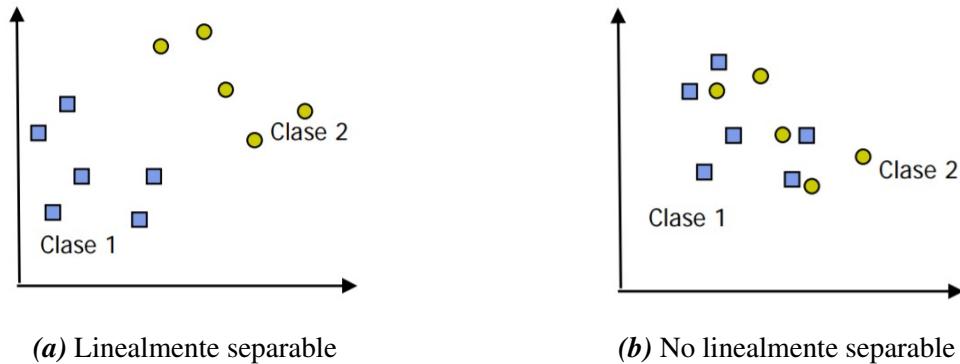


Figura 62. Ejemplo de separación de 2 clases.

Fuente: Betancourt (2005). *Las Máquinas de Soporte Vectorial (SVMs)*.

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 20:

$$f(x_i) = \text{sign}(w * z + b) = \begin{cases} 1 & y_i = 1 \\ -1 & y_i = -1 \end{cases} \quad (20)$$

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 63.

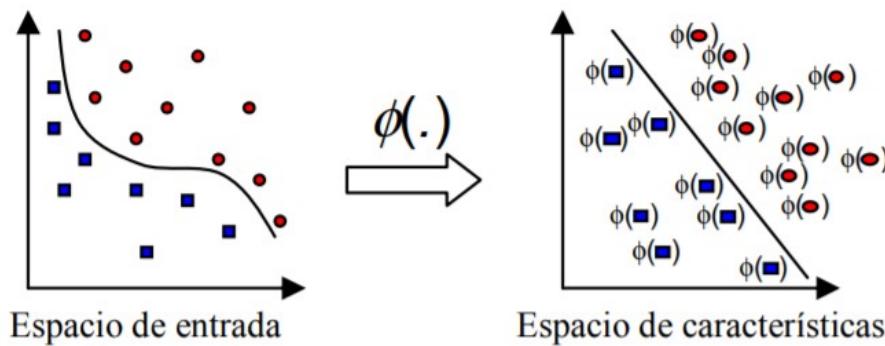


Figura 63. Aplicación de un kernel para transformar el espacio de los datos.

Fuente: Betancourt (2005). *Las Máquinas de Soporte Vectorial (SVMs)*.

- **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 64 para decidir si se debe esperar por una mesa en un restaurante.

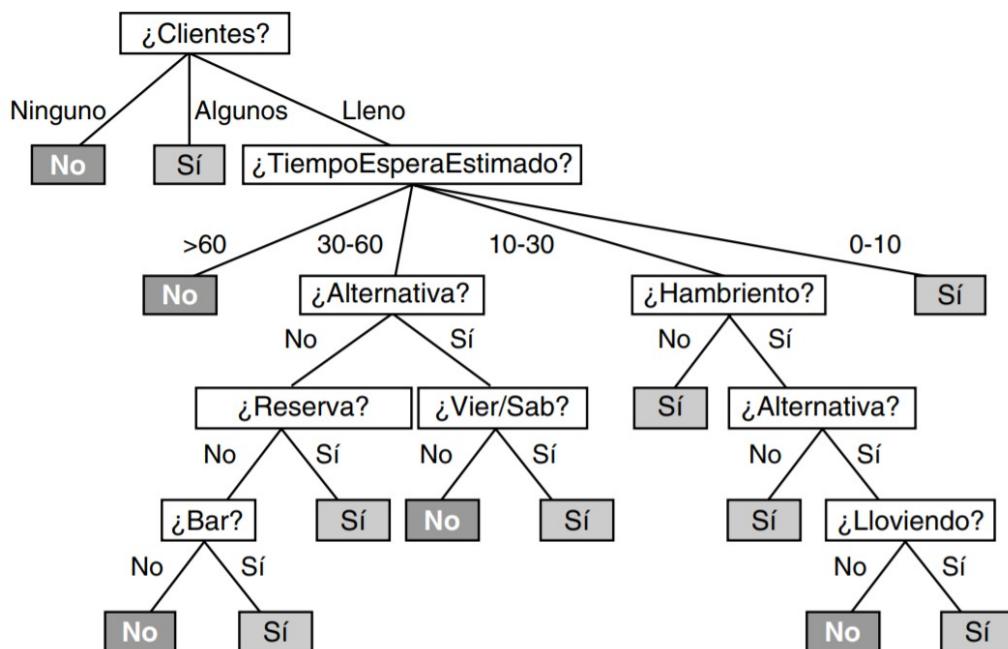


Figura 64. Ejemplo del algoritmo de árbol de decisión.

Fuente: Russell y Norvig (2004). *Inteligencia Artificial: Un Enfoque Moderno*. (segunda edición). (p. 745)

2.2.9 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. Durante las décadas de los 70s y 80s, los sistemas de entendimiento de lenguaje hablado y sistemas de diálogo se basaron en conjuntos de reglas, desarrollados por la ingeniería de conocimiento experto.

El Empirismo, la segunda ola, se caracterizó por la explotación del cuerpo de texto (*data corpora*) y su uso con Aprendizaje Automático. Esta idea plantea que la mente humana comienza con operaciones de asociación, patrones de reconocimiento y generalización. Algunos ejemplos son el Modelo Oculto de Markov (HMM) y los modelos de traducción de IBM.

El Aprendizaje Profundo, la tercera y última ola, se basa en el uso de estas técnicas para resolver problemas de Lenguaje Natural que el Aprendizaje Automático no puede lograr para entrenar con grandes cantidades de datos, por ejemplo. Las más comunes son las Redes Neuronales Artificiales, debido a que su configuración permite personalizar la arquitectura basada en múltiples capas e hiperparámetros para soportar el entrenamiento con volúmenes considerables. Sin embargo, pese a sus ventajas, algunas de las limitaciones que presenta son justamente este último detalle para lograr resultados estadísticos impresionantes, mucha capacidad computacional, o habilidades pobres para entender las relaciones de inter-oracional como frases o palabras progresivas dentro de una oración.

Algunas técnicas de Aprendizaje Profundo utilizados actualmente para resolver problemas de NLP comprenden modelos anteriormente mencionados como las Redes Neuronales Convolucionales (CNN) o las Redes Neuronales Recurrentes (RNN). A continuación, se detallarán algunas de las más usadas, así como las principales características y diferencias de ejemplos ya explicados bajo este contexto.

- **Redes Neuronales Convolucionales (Convolutional Neural Networks):** Entre las técnicas de minería de datos, se explicaron los conceptos y tipos de Redes Neuronales Artificiales, entre ellas, la actual mencionada. Se comentó que entre sus mayores usos se dan actualmente en el tratamiento de imágenes, problemas de clasificación a partir de estas y visión por computador. El ejemplo más popular fue ImageNet desarrollado por Yann LeCun, informático reconocido por ser el fundador de este tipo de redes, para reconocer objetos dentro de imágenes.

Sin embargo, también son utilizadas para problemas de clasificación de texto. Ronan Collobert y Jason Weston fueron los pioneros de la aplicación de las CNN en tareas de procesamiento de lenguaje natural, modificando y adaptando su arquitectura y parámetros internos (U. Kamath et al., 2019). En la Figura 65 se ilustra la arquitectura de una CNN para problemas de NLP.

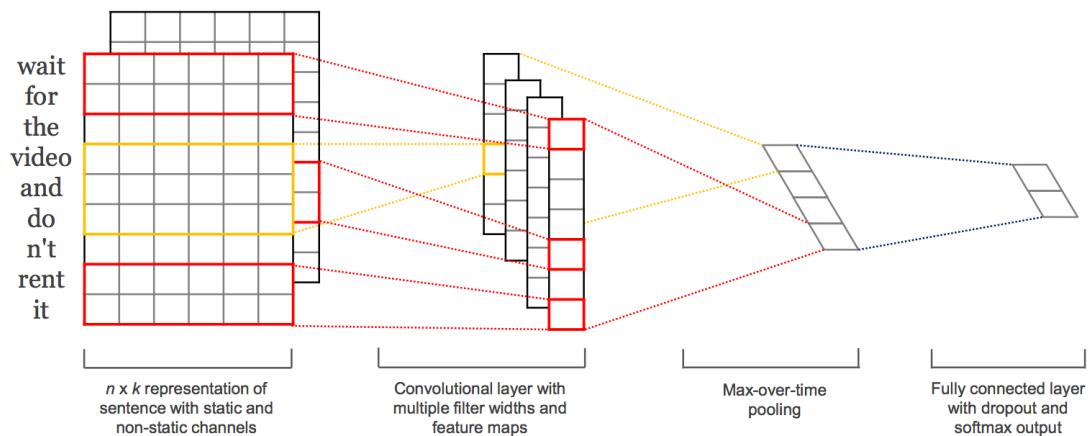


Figura 65. Arquitectura de modelo CNN con 2 canales para una oración de ejemplo.

Fuente: Kim (2014). *Convolutional Neural Networks for Sentence Classification*. (p. 1747)

Una de las principales diferencias entre ambos tipos de aplicación es que las convoluciones para imágenes son bidimensionales (2d) debido a que se desplazan a través de matrices de 2 dimensiones (largo y ancho). Mientras que las convoluciones unidimensionales (1d) son muy útiles para series de tiempo y operaciones de NLP por estar conformadas

por vectores, aprendiendo así patrones en la dimensión de secuencia (Rao & McMahan, 2019). En la Figura 66 se observa una comparación entre ambos tipos de convolución según el tamaño de dimensión.

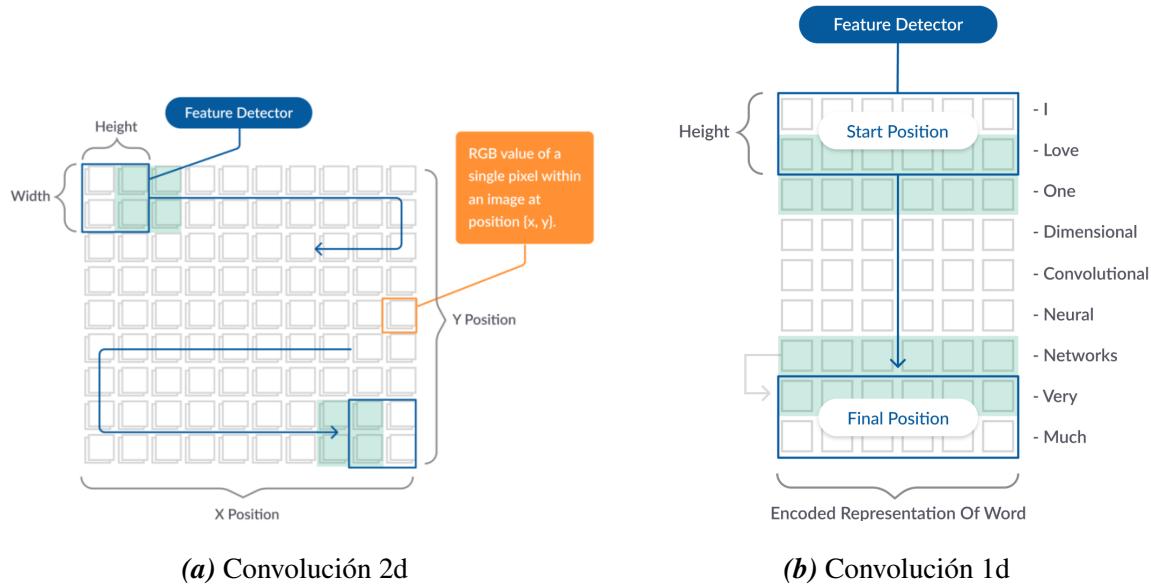


Figura 66. Diferencias entre convoluciones según su dimensión.

Fuente: MissingLink.ai (s.f.). *Keras Conv1D: Working with 1D Convolutional Neural Networks in Keras*.

En la anterior imagen, donde cada palabra codificada se representa por un vector, un kernel de convolución de tamaño de 2 bloques (*kernel_size*) recorre toda la oración con paso (*stride*) de 1 bloque.

Para problemas de clasificación de texto como los que se considera en esta investigación y en algunos antecedentes con contenido textual, el proceso de la arquitectura CNN de manera general se basa en la Figura 67.

La idea general es básicamente formar vectores de palabras codificadas para generar una matriz, la cual al ser recorrida por filtros de una dimensión determinada, se obtengan mapas de características para la siguiente entrada. De cada capa resultante, se agruparán (*pooling*) según el criterio del usuario (puede ser valor máximo, mínimo, promedio, etc) para generar nuevos vectores univariantes que serán concatenados y luego el valor final de predicción regularizado entre un rango dependiendo de la función de activación asignada para el problema (*sigmoid* para clasificación de 2 clases o *softmax* a partir de 3 clases).

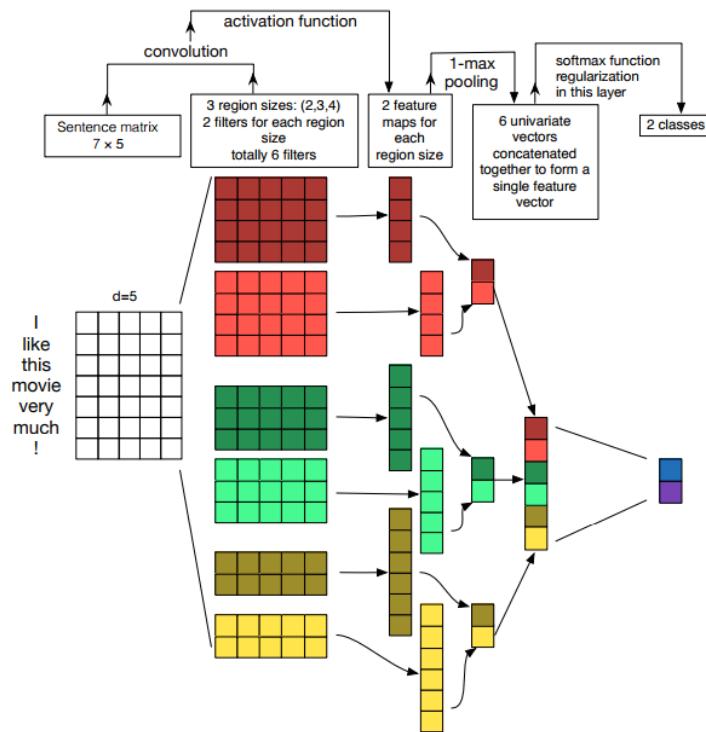


Figura 67. Arquitectura de modelo CNN para clasificación de oraciones.

Fuente: Zhang y Wallace (2017). *A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification.* (p. 256)

- **Redes Neuronales Recurrentes (Recurrent Neural Networks):** Este tipo de redes también fue comentada brevemente en los tipos de redes neuronales más conocidas. Las RNN son muy usadas incluso también en series de tiempo, ya que los datos para estas casuísticas son secuencias, es decir, una colección ordenada de elementos. Para el caso del lenguaje humano, en donde el habla es un conjunto de secuencia de palabras llamadas fonemas, se busca predecir la siguiente palabra en una oración dada a partir de un elemento dependiente, en este caso, las palabras previas (Rao & McMahan, 2019). Como ejemplos de estos casos de uso más comunes se mencionan a los motores de búsqueda en navegadores o sitios web, traductores, entre otros.

Según Brownlee (2017a), las RNN generalmente más usadas son los siguientes 3 tipos:

- **RNN Simple (Elman Network o S-RNN):** Son el tipo de redes neuronales recurrentes más básicas, propuesto por Jeffrey L. Elman en 1990, cuya arquitectura se caracteriza por la secuencia de elementos, como en la Figura 59. Las S-RNN proporcionan resultados sólidos para el etiquetado de secuencias, así como para el modelado del lenguaje.

La ecuación para representar un estado determinado en una RNN toma la siguiente forma:

$$s_i = R_{SRNN}(x_i, s_{i-1}) = g([s_{i-1}; x_i]W + b) \quad (21)$$

Donde se observa que un estado depende de la información del estado previo.

- **Memoria Larga a Corto Plazo (Long-Short Term Memory o LSTM):** Este modelo fue desarrollado para encargarse del problema de los gradientes que desaparecen de la RNN Simple, que limitaba más adelante el entrenamiento de las RNN profundas. Según el mismo autor, este problema surge debido a que los gradientes incluyen la multiplicación repetida de la matriz W (de la Ecuación 21), haciendo que los valores desaparezcan.

Esta arquitectura divide el vector de un estado observado s_i en dos mitades, donde una es tratada como “celdas de memoria” y la otra es la memoria de trabajo. Las del primer grupo tienen como función preservar la memoria, así como también los gradientes de error, a lo largo del tiempo. Son controladas mediante componentes de puerta diferenciables, que son funciones matemáticas suaves simuladoras puertas lógicas. En cada estado de entrada, se utiliza una puerta para decidir qué cantidad de la nueva entrada se debe escribir en la memoria.

La arquitectura LSTM es actualmente la más exitosa dentro de las RNN. Un ejemplo de su representación puede ilustrarse en la Figura 68.

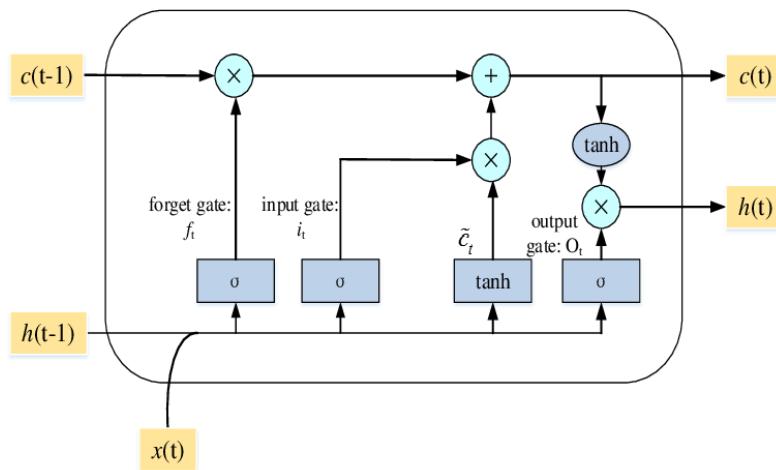


Figura 68. Arquitectura de una LSTM.

Fuente: X. Yuan et al. (2019). *Nonlinear Dynamic Soft Sensor Modeling With Supervised Long Short-Term Memory Network*. (p. 2)

En donde x son las entradas, c_{t-1} y c_t los estados de la celda, y h_{t-1} y h_t las salidas.

Una variante muy útil de las RNN son las RNN Bidireccionales (*Bidirectional RNN* o *BiRNN*). Mientras una red neuronal recurrente toma una palabra del pasado para predecir el siguiente, las bidireccionales permiten mirar arbitrariamente lejos tanto al pasado como al futuro dentro de una secuencia (Goldberg, 2017). La ventaja que se logra a partir de estas características es que, por ejemplo, el modelo puede identificar y discernir mejor en la predicción de la siguiente palabra en el caso de existir 2 o más secuencias idénticas (problema de reconocimiento de entidades nombradas) al lograr conocer más información.

Ng (2018), fundador de Coursera, DeepLearning.AI, Landing AI y Google Brain, explica lo anterior con el siguiente ejemplo como parte del curso Redes Neuronales Recurrentes:

Sean las oraciones *He said, “Teddy bears are on sale!”* y *He said, “Teddy Roosevelt was a great President!”*, se desea saber si *Teddy* es parte del nombre de una persona. El problema es que se cuenta con 2 oraciones cuya secuencia inicial de 3 palabras resultan ser las mismas, pero con distinto panorama posterior a estas. Por ello, la red es duplicada pero con orden de secuencia invertida y colocada en paralelo con la original para que cada una de las nuevas capas conecten sus salidas con las preexistentes y originen una nueva predicción que toma información previa y posterior.

Bajo la misma premisa, pero con la oración de ejemplo *The brown fox jumped over the dog*, se representa la arquitectura de una BiRNN en la Figura 69.

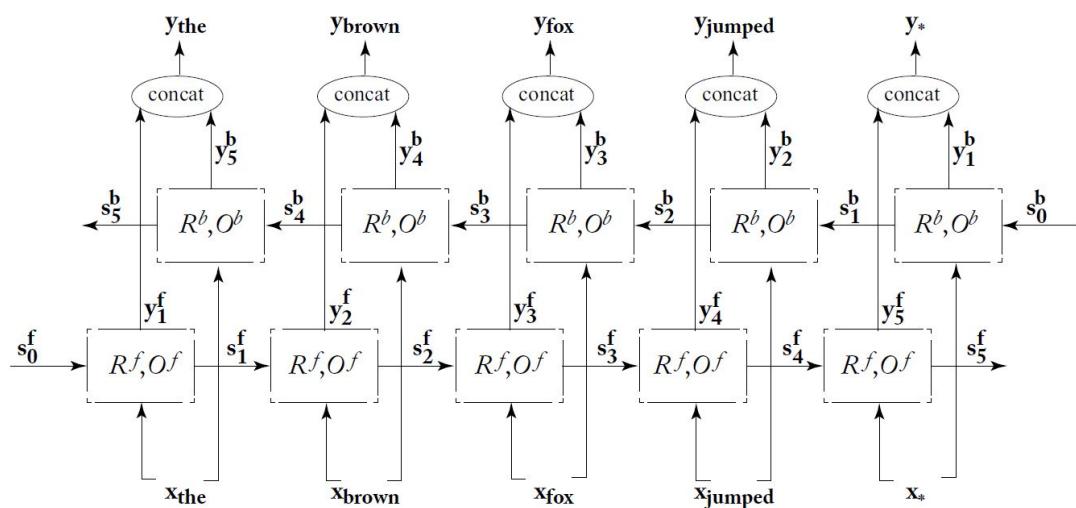


Figura 69. Representación de arquitectura BiRNN de la palabra *jumped* en la oración.

Fuente: Goldberg (2017). *Neural Network Methods for Natural Language Processing*. (p. 171)

Como parte de las BiRNN, se encuentra la LSTM Bidireccional (*Bidirectional LSTM o BiLSTM*), mejora de la LSTM, que sobresale particularmente en la representación de palabras en la secuencia junto con sus contextos, capturando la palabra y las incontables posibilidades existentes a su alrededor (Deng & Liu, 2018).

- **Unidad Recurrente Cerrada (Gated Recurrent Unit o GRU):** Este modelo se desarrolló con la intención de simplificar la LSTM debido a su complejidad. Al igual que esta, consiste en un mecanismo de puertas pero en menor cantidad y sin un componente de memoria separada (ver Figura 70). La red GRU muestra ser efectiva en modelamiento de lenguaje y máquina traductora (Goldberg, 2017).

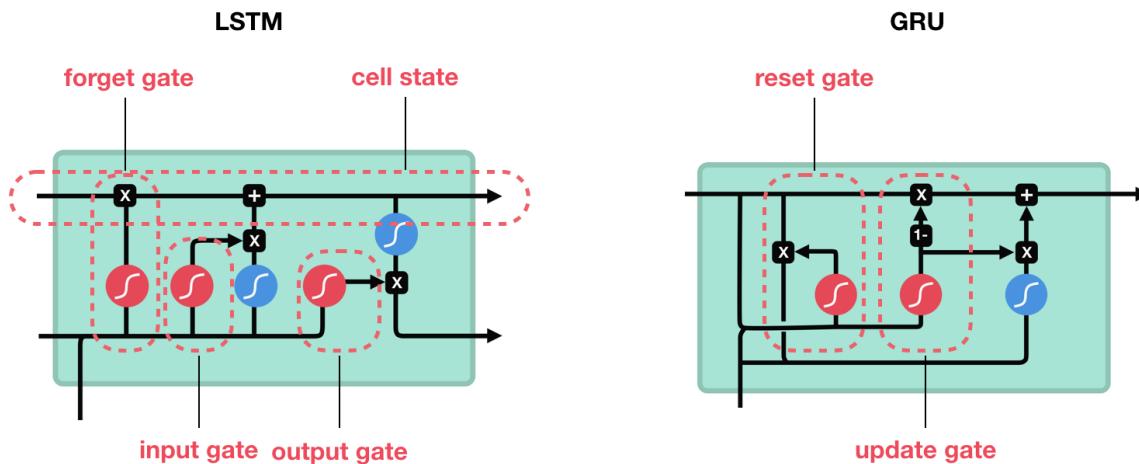


Figura 70. Comparación entre arquitecturas LSTM y GRU.

Fuente: Phi (2018). *Illustrated Guide to LSTM's and GRU's: A step by step explanation*.

Sin embargo, al compararse los resultados entre estos 3 tipos de modelos, por lo general el mejor desempeño tiene la LSTM (Brownlee, 2017a).

- **Modelo Secuencia a Secuencia (Sequence-to-Sequence Model o Seq2seq Model):** También conocida como *Encoder-Decoder* (Codificador-Decodificador por su traducción al español), es un tipo de generador de lenguaje natural (*Natural Language Generation o NLG*) usado comúnmente para traducción (por ejemplo, Google Traductor). Se basa en 2 capas LSTM, en donde la primera es usada para codificar la oración de entrada en un “vector de pensamiento”, y la otra para decodificar en una respuesta (ver Figura 71) (Deng & Liu, 2018).

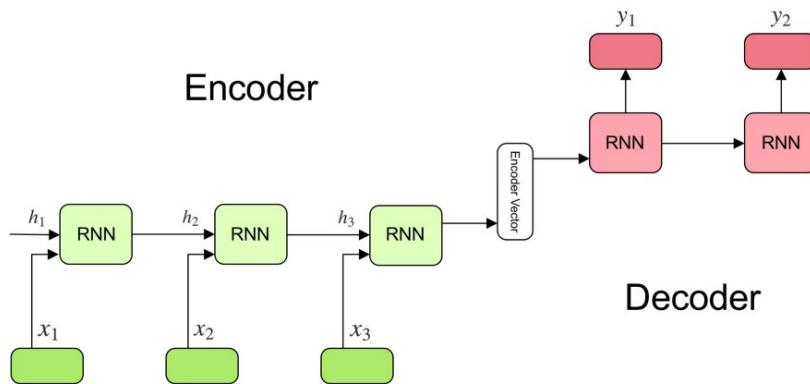


Figura 71. Arquitectura de un modelo Seq2seq.

Fuente: Kostadinov (2019). *Understanding Encoder-Decoder Sequence to Sequence Model*.

El contenido textual que se usa en los modelos anteriores debe ser pre-procesada previamente. El texto en sí no puede ser incluido tal cual en los modelos de Aprendizaje Automático o Aprendizaje Profundo sin antes ser limpiado (Brownlee, 2017a). Python ofrece una librería para trabajar y modelar texto llamada Natural Language Toolkit o NLTK. Algunas de las funciones disponibles se encuentran separación de texto en oraciones, separación en palabras o tokenización, eliminación de signos de puntuación, eliminación de palabras de parada (*stop words*), reducción de palabras hacia su forma raíz (*stemming*), retorno de palabras hacia su base o forma diccionario (*lemmatization*).

Suprimir contenido como signos de puntuación, caracteres especiales, palabras de parada, enlaces web, entre otros, ayuda a reducir la cantidad de vectores innecesarios para las representaciones de palabras que se utilizan en la fase de entrenamiento. Para ello, la secuencia de entrada debe dividirse en *tokens*, que pueden ser una palabra, oración, párrafo.

NLTK ofrece más de una alternativa para la tokenización, como por ejemplo, *WhitespaceTokenizer* (elimina espacios en blanco para separar palabras y signos de puntuación en una oración, estos últimos no son independientes), *TreebankWordTokenizer* (separa palabras y signos de puntuación en una oración de manera independiente) y *WordPunctTokenizer* (funciona igual que TreebankWordTokenizer pero no distingue contracciones en idiomas como el inglés). La elección de alguna de estas depende del objetivo que busque el usuario. Asimismo, la reducción de palabras hacia su raíz (llamada *stem*) permite suprimir los prefijos o sufijos agregados a una palabra. Sin embargo, presenta problemas en formas irregulares, generando “No palabras”. El retorno de palabras hacia su forma base (llamada *lemma*), por su lado, convertir palabras conjugadas en distintas variantes de tiempo. Aún así, no todas las formas pueden ser reducidas.

La finalidad de estos 2 últimos es reducir una palabra a su forma más primitiva (expresiones regulares) para generar un diccionario homogéneo (Zimovnov, 2018).

Luego de la limpieza de texto, el nuevo conjunto de datos debe representarse como vectores. Dentro de las modalidades más usadas para representación de palabras se encuentran:

- **Incrustación de palabra (Word Embedding)**: Es una representación aprendida para texto en donde las palabras con el mismo significado tienen una similar representación. La principal ventaja que presenta es la baja dimensionalidad de vectores a nivel computacional, ya que una palabra individual se representa como vectores de valor real en un espacio vectorial predefinido, a diferencia de otros métodos de muy alta dimensionalidad como la codificación en caliente (*one hot encoding*) o la bolsa de palabras (*bag-of-words*), en donde distintas palabras presentan diferentes representaciones (Brownlee, 2017a).

Algunos de los algoritmos destacados de este grupo son:

- **Capa de incrustación (Embedding Layer)**: Consiste en una incrustación de palabras que aprende con un modelo de red neuronal. En esta capa, se especifica el tamaño del espacio vectorial (dimensiones), donde los vectores inicializan con pequeños valores aleatorios. La capa se utiliza en el extremo frontal de la red, es decir, luego de la capa de entrada, y es ajustada de forma supervisada mediante el algoritmo de propagación hacia atrás. Puede recibir palabras codificadas, en donde cada una se representa por un código, o codificación en caliente, la cual presenta mayor dimensión. Si se utiliza un Perceptrón Multicapa (MLP), los vectores de palabras son concatenados antes de entrar al modelo. En el caso se use una RNN, cada palabra será tomada como una entrada en una secuencia (Brownlee, 2017a). Como se observa en la Figura 72, la capa de incrustación toma como entrada a la matriz de incrustación y la transforma en un vector tridimensional de n registros.

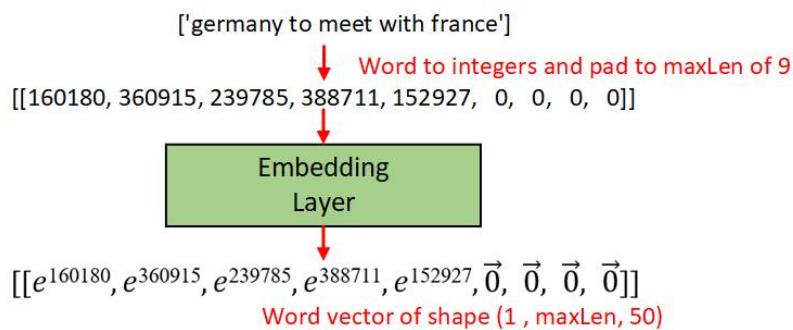


Figura 72. Ejemplo de funcionamiento de una capa de incrustación.

Fuente: Chengwei (2018). *Simple Stock Sentiment Analysis with news data in Keras*.

- **Palabra a vector (Word2Vec):** Se trata de un método estadístico, desarrollado en el 2013 por Tomas Mikolov, cuya finalidad es de aprender eficientemente una incrustación de palabras independientes de un corpus de texto, incluso si este y sus dimensiones son más grandes. En este trabajo se involucró el análisis de los vectores aprendidos y la exploración de la matemática vectorial para representar palabras. Para entender estos conceptos, se ilustra bajo el ejemplo de la analogía “*Rey es a Reina como hombre es a mujer*”, en donde se evidencia la relación sintáctica y semántica capturada por su proximidad vectorial (Brownlee, 2017a). La Figura 73 grafica las palabras relacionadas en un espacio vectorial gracias al algoritmo.

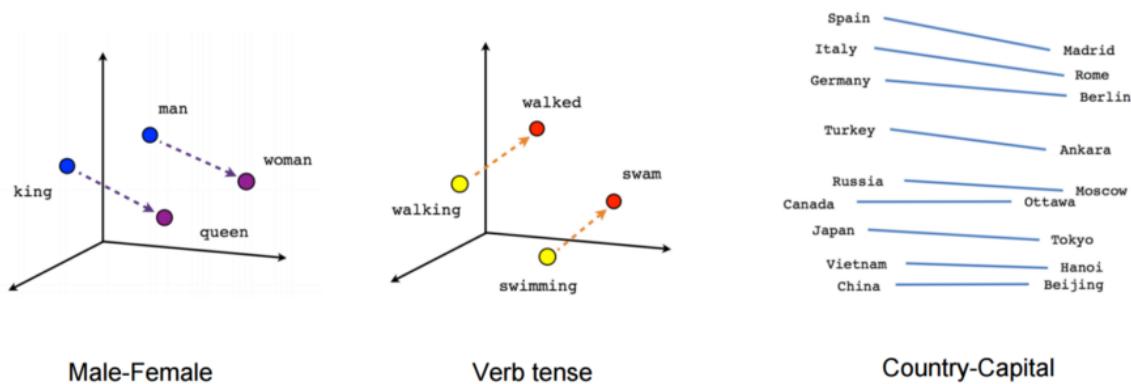


Figura 73. Incrustaciones de palabras por Word2Vec.

Fuente: Bujokas (2020). *Creating Word Embeddings: Coding the Word2Vec Algorithm in Python using Deep Learning*.

- **Vectores globales para representación de palabra (GloVe):** Es un algoritmo de aprendizaje no supervisado para obtener representaciones vectoriales de palabras. Es una extensión del método Word2Vec, desarrollado en 2014 por Jeffrey Pennington como proyecto de código abierto en la Universidad Stanford. Las representaciones de palabras del modelo de espacio vectorial clásico se desarrollaron utilizando técnicas de factorización matricial como el Análisis semántico latente (LSA), con el cual combina sus características de aprendizaje adoptadas de Word2Vec, logrando un modelo de aprendizaje con mejor performance para incrustación de palabras (Pennington et al., 2014a). GloVe dispone en su web distintos vectores de palabras pre-entrenados, entre ellas, un vocabulario de más de 400 mil palabras, hasta 4 opciones de matrices con distintas dimensiones, y 6 billones de tokens extraídos de Wikipedia (2014) y Gigaword (5ta edición) de la Universidad de Pensilvania, así como datos públicos extraídos de mayor tamaño. La Figura 74 ilustra ejemplos de palabras relacionadas en un espacio vectorial gracias al algoritmo. La relación

semántica (palabras clasificadas en un mismo grupo) entre ellas se aprecia por la cercanía en su eje Y. Mientras que aquellos señalados con líneas horizontales en su eje X se encuentran asociadas por analogías, como por ejemplo, género, geolocalización, compañía, grados del adjetivo, entre otros.

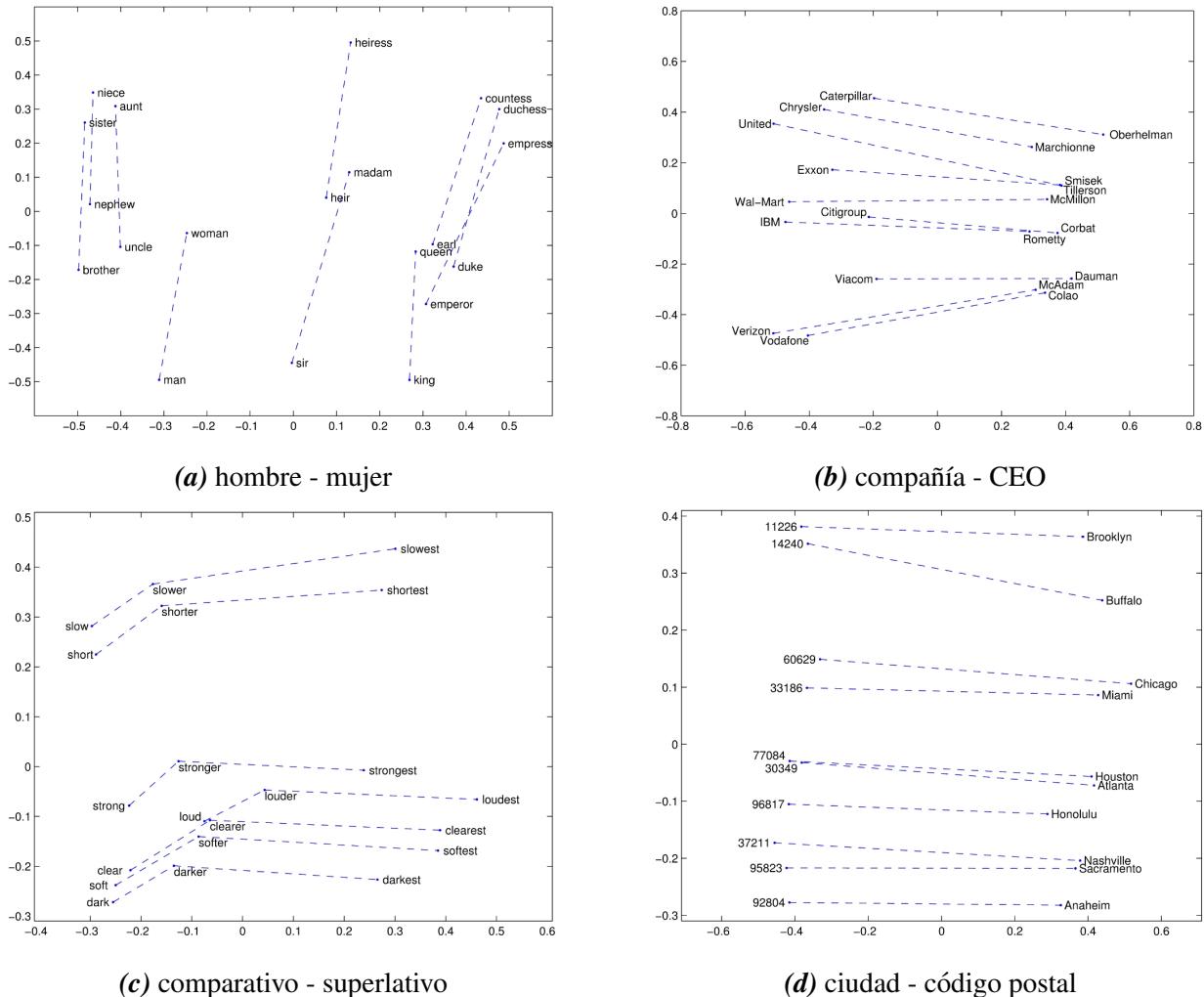


Figura 74. Incrustaciones de palabras por GloVe.

Fuente: Pennington et al. (2014b). *GloVe: Global Vectors for Word Representation*.

- **Bolsa de palabras (Bag-of-Words o BoW):** Consiste en una forma de extraer características de textos implicando un vocabulario de palabras conocidas y una métrica de presencia de éstas. Se le denomina Bolsa de palabras ya que no considera el orden o la estructura de las palabras dentro del documento, sólo se enfoca en conocer su presencia. El algoritmo relaciona la similitud entre documentos si tienen contenido similar (Brownlee, 2017a). Por ejemplo, en la Figura 75 el algoritmo cuantifica la frecuencia cada palabra individual dentro de un documento sin importar su orden. De esta manera, los relaciona a partir de su recurrencia en todos ellos.

Document	the	cat	sat	in	hat	with
<i>the cat sat</i>	1	1	1	0	0	0
<i>the cat sat in the hat</i>	2	1	1	1	1	0
<i>the cat with the hat</i>	2	1	0	0	1	1

Figura 75. Ejemplo de funcionamiento de bolsa de palabras.

Fuente: V. Zhou (2019). *A Simple Explanation of the Bag-of-Words Model*.

- **Frecuencia del término - Frecuencia inversa del documento (TF-IDF):** Es un algoritmo que también contabiliza palabras pero que resaltan aquellas más significativas para calcular sus pesos correspondientes. Se compone por la frecuencia que una palabra aparece en un documento, y la frecuencia inversa del documento, el cual reduce la escala de palabras que aparecen mucho en los documentos (Brownlee, 2017a).

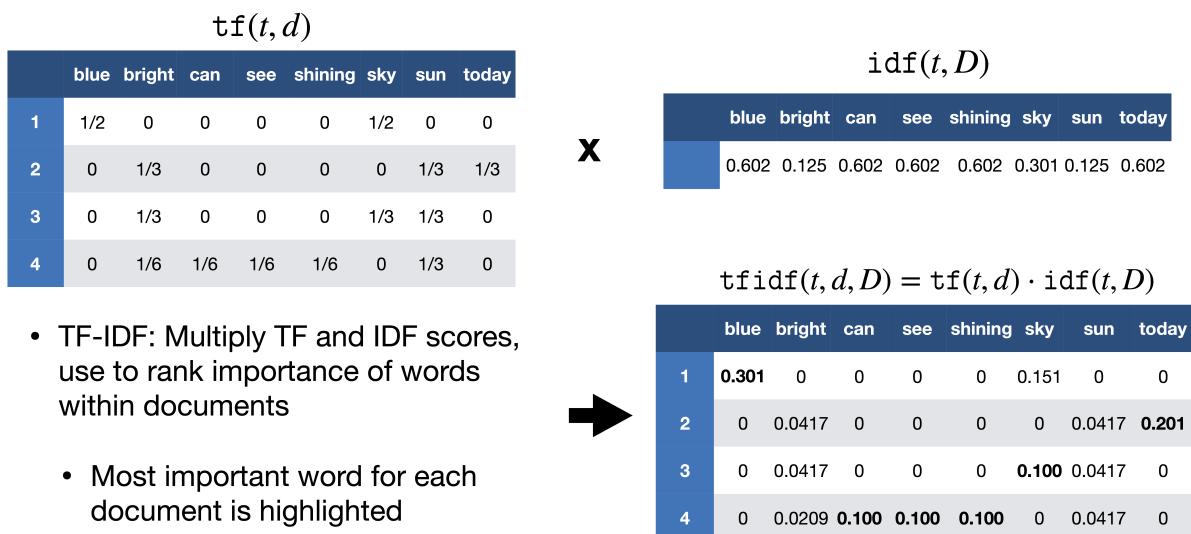


Figura 76. Ejemplo de funcionamiento de TF-IDF.

Fuente: Brinton y Inouye (2020). *Python for Data Science: n-grams and basic natural language processing*.

Su peso se calcula bajo la siguiente fórmula:

$$TF-IDF_{(n,d)} = TF_{(n,d)} * IDF_{(n)} \quad (22)$$

Donde:

$TF_{(n,d)}$ = número de veces que el término t aparece en un documento d

$IDF_{(n)} = \log \frac{1+n}{1+df_{(d,n)}} + 1$

n = número de documentos

$df_{(d,n)}$ = frecuencia de documentos d que contienen el término t

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 et al., 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).

El modelo basado en donaciones se refiere a la recaudación de fondos a través de la Web 2.0, en el cual los patrocinadores no esperan recompensas materiales a cambio sino más bien una recompensa social. Por el contrario, el modelo basado en recompensas ofrece compensación tanto material como inmaterial y representa hoy en día el modelo de crowdfunding más frecuente. Los finanziadores, por un lado, pueden verse beneficiados de la venta anticipada, recibiendo el proyecto o producto financiado antes de su publicación o entrada al mercado al mejor precio. Los proyectos que pertenecen a esta categoría a menudo son organizaciones sin fines de lucro (Kraus et al., 2016).

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 complementa 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).

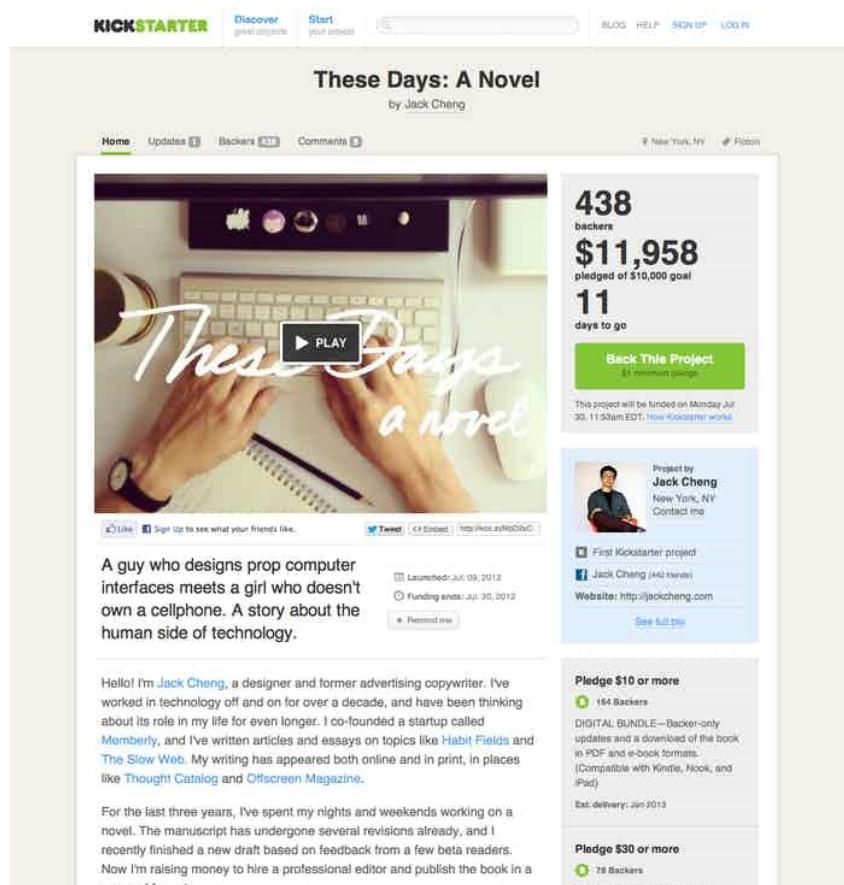


Figura 77. Ejemplo de proyecto vigente en Kickstarter.

Fuente: P. Chen (2012). *The New Project Page*.

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.
- Atrae 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.

El autor A. Yu (2017) también complementa lo anterior con las siguientes estrategias para empezar a realizar la campaña:

- Además de estimar los costos, se debe contar con un buffer en caso ocurra un imprevisto. Se puede determinar la meta a través de la fórmula (Gastos + Buffer) x Alcance.
- Un proyecto con video tiene 50 % de chance de tener éxito. El promedio de su duración debería ser de 3.38 minutos, aunque el público presta más atención hacia aquellos que duran poco. El vídeo debe contener y responder las siguientes preguntas:
 - ¿Qué hace memorable al proyecto?
 - ¿Cómo se verá el proyecto?
 - ¿Por qué se está creando el producto?
 - ¿Por qué se necesita?
 - ¿Por qué es importante que el proyecto exista?
 - ¿Por qué se debe ser creador?
 - ¿Por qué deben escoger la compañía?
 - ¿Cómo se proveerá una solución?

- ¿Cómo funciona la solución?

Es importante destacar las estadísticas del problema que se piensa resolver al inicio del video ya que el primer minuto resulta ser el lapso de tiempo que más capta la atención del público. Se recomienda también tener buen audio e iluminación.

- La descripción deberá ser un resumen de los puntos más relevantes del producto. Se debe mencionar una breve introducción del proyecto, cómo funciona, quiénes son el público objetivo, el proceso que se siguió en el desarrollo, detalles y especificaciones, y un cronograma e hitos.
- Las imágenes deben comunicar cómo lucirá el proyecto. Esta prueba convencerá a más de una persona en querer invertir en el proyecto. Las imágenes a veces atraen más a las personas que no gustan leer la descripción completa del proyecto.
- Al basarse en un modelo de crowdfunding en recompensas, los patrocinadores esperan un beneficio por parte del proyecto una vez este logre alcanzar su financiamiento. El número ideal de recompensas puede ser entre 5 y 7. Una recompensa en promedio de \$100 es la que genera más dinero. Las recompensas pueden ser de cuatro tipos: el producto en sí mismo entregado hacia los patrocinadores antes que este salga al mercado, reconocimientos en la página web y acceso hacia actualizaciones, recuerdos y certificados de membresía, y nuevas experiencias como visita a los desarrolladores en sus estudios.

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 III: Metodología de la Investigación

3.1 Diseño de la investigación

En esta sección del documento se explica cuál fue el diseño, el tipo y el enfoque del trabajo de investigación, así como también la población y la muestra.

3.1.1 Tipo de la investigación

Para determinar el tipo de la investigación, primero fue necesario definir el actual trabajo como Diseño Experimental ya que, según Hernández Sampieri et al. (2014) en su libro *Metodología de la Investigación*, se pretende establecer el posible efecto de una causa que se manipula. Dentro de esta categoría se clasifica como Diseño Experimental Puro, ya que se manipuló intencionalmente más de una variable independiente (fueron agregadas y/o removidas) con la finalidad de medir el efecto que estas generan en la variable dependiente, el estado final de financiamiento de un proyecto de tecnología en Kickstarter.

3.1.2 Enfoque de la investigación

El presente trabajo tuvo un enfoque cuantitativo ya que, según Hernández Sampieri et al. (2014) en su libro *Metodología de la Investigación*, este enfoque utiliza la recolección de datos para probar hipótesis con base en la medición numérica y el análisis estadístico, con el fin de establecer pautas de comportamiento y probar teorías. Esto se refleja en los 10 pasos del proceso cuantitativo descritos por el anterior autor y que fueron aplicados en la investigación, desde la concepción de la idea hasta la elaboración del reporte de resultados.

3.1.3 Población

La población fueron todos los proyectos de la web de crowdfunding Kickstarter.

3.1.4 Muestra

La muestra fueron 27,251 proyectos, incluyendo exitosos y fracasados, de la categoría Tecnología, de la web de crowdfunding Kickstarter, entre los períodos 2009 y 2019. Como se mencionó en el Capítulo I, el criterio de la elección de esta categoría se debe a su bajo ratio de éxito de financiamiento en promedio durante los períodos mencionados, los cuales se consideraron hasta el 2019 dado que tomar el año 2020 incurría en la distorsión del ratio por la coyuntura de la pandemia del Covid-19.

3.1.5 Operacionalización de Variables

En la Tabla 2 se presenta la operacionalización de variables para la investigación.

Tabla 2

Matriz de operacionalización de variables.

VARIABLE	DIMENSIÓN	INDICADOR	CÁLCULO
Independiente: Aprendizaje Profundo Multimodal	Modelo de Aprendizaje Profundo	Cantidad de modelos de Aprendizaje Profundo	
		Efectividad de modelos de Aprendizaje Profundo	Exactitud
		Aprendizaje Profundo	Precisión
		Profundo	Área bajo la curva ROC
		Profundo	Sensibilidad
		Profundo	Puntaje F1
Dependiente: Financiamiento de un proyecto	Estructura del modelo de Aprendizaje Profundo	Complejidad de la estructura de modelos de Aprendizaje Profundo	
			$Exito = Meta \leq Contribución$
			$Fracasado = Meta > Contribución$
Campaña del proyecto en Kickstarter	Estado de financiamiento de un proyecto	$Exito = Meta \leq Contribución$ $Fracasado = Meta > Contribución$	
	Metainformación	Cantidad de patrocinadores Duración en días Completitud de la meta Cantidad de palabras Cantidad de comentarios Cantidad de palabras	

Fuente: Elaboración propia.

El trabajo busca resolver el problema de clasificación del estado final de financiamiento de un proyecto implementando un modelo de Aprendizaje Profundo Multimodal.

Como se explica en la subsección 2.3.3, un proyecto de Kickstarter presenta 5 categorías de acuerdo al logro del objetivo de su meta. Dado que se busca predecir si llegará a ser financiado o no, el estudio se delimita a exitoso o fracasado. Para determinar este valor, y según la política “Todo o Nada” de la plataforma explicado en la subsección 2.3.2, un proyecto es determinado como exitoso cuando el monto contribuido al culminar el tiempo de vida de su campaña logra ser mayor o igual a su meta estimada. De lo contrario, es un proyecto fracasado.

El modelo propuesto, alimentado de 3 modalidades independientes presentes en cada campaña de Kickstarter, comprende el ensamblaje de 3 modelos de Aprendizaje Profundo: un modelo Perceptrón Multicapa (MLP), una Red Neuronal Convolucional (CNN) y un modelo LSTM Bidireccional. Para estos 2 últimos, se trabajaron sus datos de entrada con técnicas de Procesamiento de Lenguaje Natural que incluye desde el pre-procesamiento de los textos hasta su conversión en vectores que finalmente fueron entrenados en sus respectivos modelos.

Por último, para poder evaluar la efectividad que este registrará al ser entrenado, su desempeño será evaluado a través de métricas de clasificación de Minería de Datos. Para la investigación, las 5 métricas utilizadas fueron la exactitud, la precisión, el área bajo la curva ROC, la sensibilidad y el puntaje F1, las cuales se describen en la Sección 3.3.2.

3.2 Técnicas de recolección de datos

Los conjuntos de datos recolectados para la investigación se componen de variables cuantitativas (características del proyecto como la meta de financiamiento, montos prometidos, duración de la campaña y otros indicadores financieros) y cualitativas (descripción y comentarios del proyecto). Para obtener los 3 principales datasets, se siguió el flujo de la Figura 78. La base de datos de la metainformación fue construida luego de descargar un histórico público de 10 años de la página web “Web Robots” y posteriormente pre-procesarla. Por el lado de la descripción y comentarios de los patrocinadores, se usaron técnicas y herramientas de *web scraping* a partir de los URLs de cada proyecto disponibles en la Metainformación.

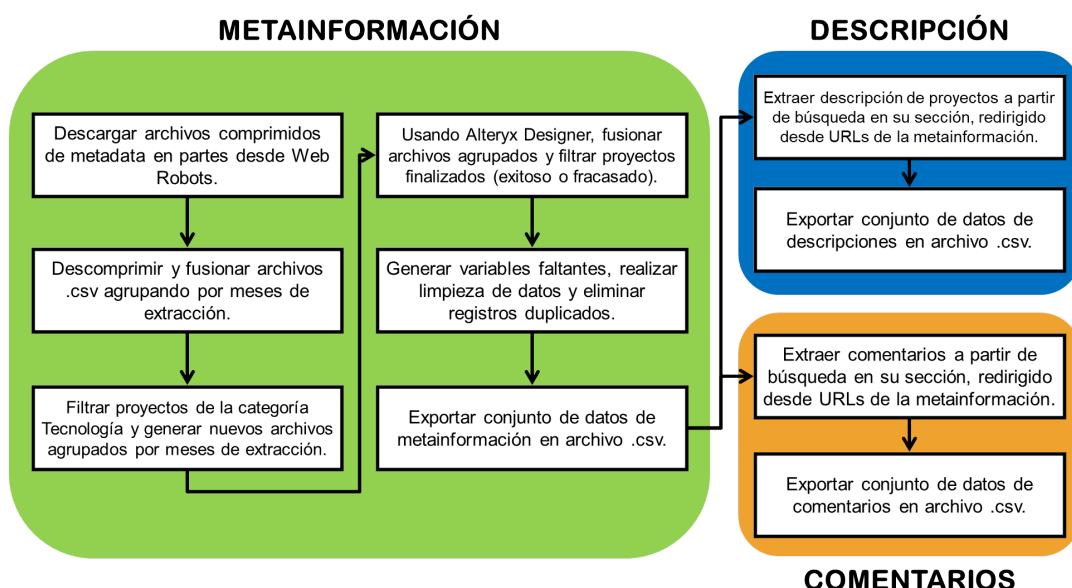


Figura 78. Flujograma de la recolección de conjuntos finales de datos.

Fuente: Elaboración propia.

Asimismo, 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*.

3.3 Técnicas para el procesamiento y análisis de la información

3.3.1 Metodología de implementación de la solución

Como se explicó en el punto 2.2.6 del Marco Teórico del presente trabajo, según Braulio Gil y Curto Díaz (2015), dentro de los sistemas de analítica de negocio, Big Data y Minería de Datos, tres de las metodologías más usadas son CRISP-DM, SEMMA y KDD. Los mismos autores detallan en la Tabla 1 las características que cada una presenta al compararse.

Para escoger la metodología, se elaboró la Tabla 3 con el fin de comparar las utilizadas por los autores en los antecedentes. 17 de los 18 autores implementaron sus propias metodologías, las cuales de acuerdo a su similitud se agruparon en 8 grupos.

Tabla 3
Cuadro comparativo para la selección de la metodología.

Metodología	Cantidad de referencias	Número de pasos	Nombre de los pasos
CRISP-DM	1	6	<ul style="list-style-type: none"> – Comprensión del negocio. – Comprensión de los datos. – Preparación de los datos. – Modelado. – Evaluación. – Despliegue.
Grupo A	2	6	<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Pre-procesamiento de datos. – Modelado. – Evaluación. – Despliegue.

				<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Selección de características. – Modelado. – Evaluación. – Despliegue.
Grupo B	1	6		<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Selección de características. – Modelado. – Evaluación.
Grupo C	2	5		<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Pre-procesamiento de datos. – Modelado. – Evaluación.
Grupo D	3	5		<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Modelado. – Evaluación.
Grupo E	3	5		<ul style="list-style-type: none"> – Modelado. – Evaluación. – Despliegue.
Grupo F	3	4		<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Modelado. – Evaluación.
Grupo G	1	4		<ul style="list-style-type: none"> – Recolección de datos. – Modelado. – Evaluación. – Despliegue.
Grupo H	2	3		<ul style="list-style-type: none"> – Recolección de datos. – Modelado. – Evaluación.

Fuente: Elaboración propia

Si bien es cierto que de las 3 metodologías mencionadas por Braulio Gil y Curto Díaz, solamente 1 autor especifica en su investigación el uso de una de ellas, en este caso CRISP-DM (Fernandez-Blanco et al., 2020), otros 3 autores (grupos A y B) en sus propias metodologías siguieron secuencias de actividades que se encuentran comprendidas también dentro de esta metodología. Por su parte, los grupos C, F y H guardan cierta relación de semejanza con la metodología SEMMA por comprender enfocarse más en el modelado, así como tener de primera actividad el Muestreo y culminar con la evaluación de resultados, obviando la actividad de formulación del problema. En el Anexo D se puede observar con más detalle las metodologías de la Tabla 3 asignadas a la investigación correspondiente.

Además de lo anterior expuesto, algunas de las siguientes características de acuerdo a la literatura también influyeron en la elección de la metodología CRISP-DM:

- La metodología seleccionada contempla entre sus fases la comprensión del negocio además de la parte técnica que incluye el modelado y análisis de resultados. Esta guarda un rol importante ya que en ella se define el inicio de todo el proceso para dar el alcance del proyecto y definir objetivos que se buscan a partir de la Minería de Datos y Big Data. La formulación del problema se encuentra presente en el actual trabajo.
- Ayuda a los responsables del proyecto y/o investigación en la planeación y toma de decisiones (fase Despliegue) a partir de los resultados obtenidos, reportando y convirtiéndolos en oportunidades a considerar en los objetivos.
- Evalúa en todo el proceso los datos y variables usadas con el fin de crear el mejor modelo. Estas serán importantes para interpretar los resultados y tomar decisiones.
- Respecto a las otras dos metodologías, ambas omiten en su primera iteración la formulación del problema, la cual se encuentra presente en el actual trabajo. Por el lado de KDD, si bien esta contempla 9 pasos durante su proceso, el objetivo de KDD en cada uno de estos resulta ser más técnico, es decir, trabajar, seleccionar e interpretar métricas, variables, modelos, entre otros para obtener los mejores resultados más allá de considerar el contexto y comprensión del negocio. De hecho, no existe alguna fase dedicada al entendimiento del mismo.
- Por otra parte, la metodología SEMMA se basa, como su nombre lo indica, en la selección, exploración y modelado de grandes cantidades de datos para descubrir patrones de negocio desconocidos. Sin embargo, al limitarse a 5 fases comenzando con la fase de

muestreo, no hace hincapié en la comprensión del negocio, sino más bien comienza con el procesamiento de datos para la construcción del modelo.

Cada una de las fases de la metodología seleccionada se detalla en la Figura 79.

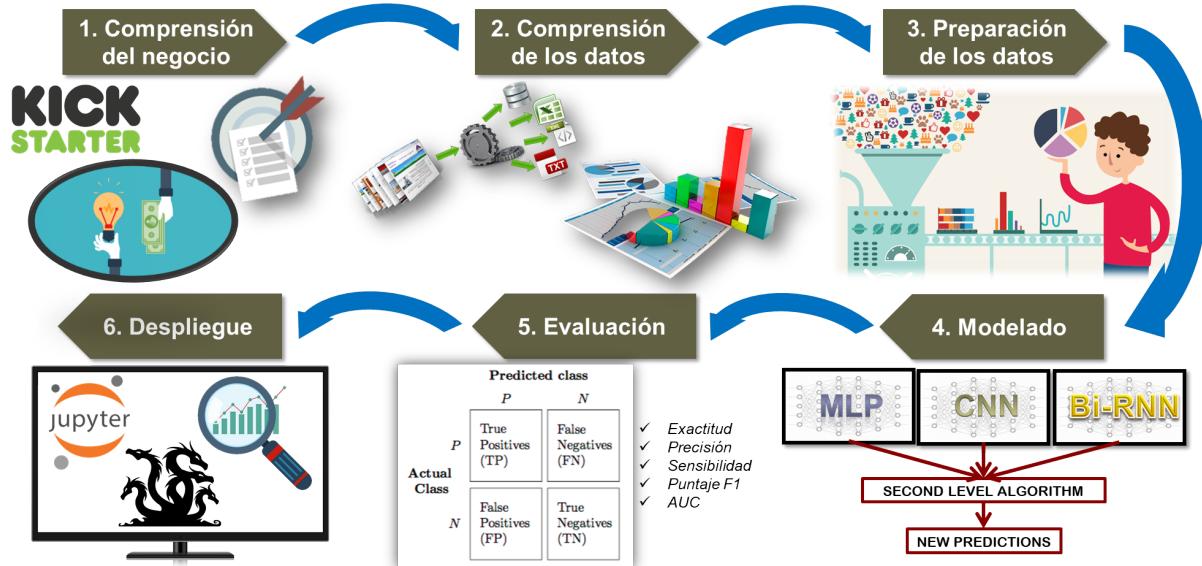


Figura 79. Metodología de la investigación.

Fuente: Elaboración propia

Durante el proceso de la implementación de la metodología, en donde se decidió construir un modelo de Aprendizaje Profundo Multimodal (utilizado por R. S. Kamath y Kamat (2018), Jin et al. (2019), y Cheng et al. (2019)), de acuerdo a la literatura, se analizaron las siguientes posibles modalidades que se tomarían en cuenta para la investigación:

- **Metainformación:** K. Chen et al. (2013), Mitra y Gilbert (2014), M. Zhou et al. (2015), S.-Y. Chen et al. (2015), Beckwith (2016), Y. Li et al. (2016), H. Yuan et al. (2016), Sawhney et al. (2016), Kaur y Gera (2017), R. S. Kamath y Kamat (2018), P.-F. Yu et al. (2018), Jin et al. (2019), Cheng et al. (2019), Fernandez-Blanco et al. (2020).
- **Descripción del proyecto:** Mitra y Gilbert (2014), M. Zhou et al. (2015), H. Yuan et al. (2016), Sawhney et al. (2016), R. S. Kamath y Kamat (2018), S. Lee et al. (2018), Jin et al. (2019), Cheng et al. (2019), L.-S. Chen y Shen (2019), Chaichi y Anderson (2019).
- **Comentarios en la campaña:** S.-Y. Chen et al. (2015), Y. Li et al. (2016), S. Lee et al. (2018), Jin et al. (2019), Shafqat y Byun (2019).
- **Actualizaciones y/o recompensas:** M. Zhou et al. (2015), S. Lee et al. (2018).

■ **Contenido audiovisual (imagen y/o video del proyecto):** Cheng et al. (2019).

De las anteriores modalidades, las actualizaciones no representan un cambio significativo ya que consisten en agregar o desagregar contenido textual o modificar recompensas en la campaña. Por el lado del contenido audiovisual, además de no presentarse en todas las campañas, para el caso particular de la categoría Tecnología no se encontró un patrón entre sus imágenes que pueda ser de utilidad, como se presenta en el trabajo de tesis de pregrado del presente investigador (Puente, 2019).

Finalmente, fueron seleccionadas las modalidades de metainformación, descripción del proyecto y comentarios, para este último considerando solamente expresados por patrocinadores. Las 2 primeras se localizan en la sección principal de la campaña, mientras que la tercera se ubica en su sección respectiva, como se puede apreciar en la Figura 80.

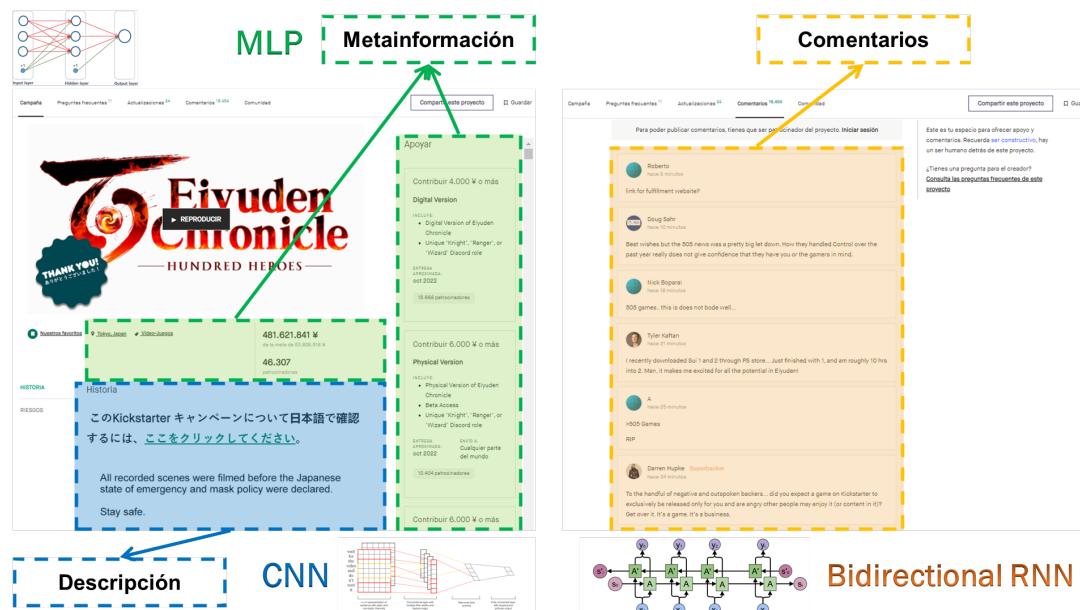


Figura 80. Marco de trabajo del prototipo final.

Fuente: Elaboración propia

3.3.1.1. Comprensión del negocio

En la primera fase, se definió el problema a partir de comprender el contexto de la investigación. A partir de aquí, se formularon los objetivos y requerimientos que se espera lograr en el proyecto. La lista de actividades y tareas se presenta en la Tabla 4.

Tabla 4

Actividades de fase Comprensión del negocio.

Actividades	Descripción	Tareas
Definir problemas, objetivos e hipótesis.	Definición de los problemas, objetivos e hipótesis generales y específicos de la investigación.	<ul style="list-style-type: none"> – Identificar la realidad problemática del contexto. – Definir los objetivos de la investigación. – Formular las hipótesis del trabajo.
Desarrollar la literatura de la investigación.	Selección de los antecedentes y trabajos previos relacionados a la investigación.	<ul style="list-style-type: none"> – Buscar material académico sobre las variables de la investigación. – Buscar material académico relacionado a la solución de la realidad problemática.
Definir metodología de la investigación.	Definición de la metodología a implementarse en el estudio.	<ul style="list-style-type: none"> – Evaluar metodologías de la literatura. – Seleccionar metodología a implementar en el trabajo.

Fuente: Elaboración propia

A continuación, cada actividad de la anterior tabla es detallada y se indican sus entregables respectivos.

Actividad 1: Definir problemas, objetivos e hipótesis

En esta actividad se detalla cómo se definieron los problemas, objetivos e hipótesis generales y específicos de la investigación a partir del estudio de la coyuntura en que se desenvuelve la realidad problemática.

Entregable: Problemas, objetivos e hipótesis definidos.

Actividad 2: Desarrollar la literatura de la investigación

A partir de la definición de los objetivos, se realizó la búsqueda de fuentes, publicaciones, libros, artículos y material académico relacionado al contexto del financiamiento colectivo, sus características y qué métodos se utilizaron para resolver problemas dentro del entorno. Asimismo, se elaboró el marco teórico abordado en el estudio.

Entregable: Material académico y conocimientos obtenidos sobre el ambiente del crowdfunding y la Inteligencia Artificial.

Actividad 3: Definir metodología de la investigación

Luego de conseguir los recursos intelectuales suficientes y los trabajos previos para el desarrollo de la investigación, se analizó cada antecedente y se compararon todas las metodologías implementadas por cada autor. Al culminar con esta tarea, en base a la literatura y los procesos más alineados al presente trabajo, se seleccionó la metodología CRISP-DM como la mejor opción.

Entregable: Metodología de implementación de la solución.

3.3.1.2. Comprensión de los datos

La segunda fase abarcó tanto la recolección de los datos como el análisis estadístico de estos para conocer un poco más sus características. 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.

Para ello, se elaboró una lista de actividades y tareas presentadas en la Tabla 5.

Tabla 5
Actividades de fase Comprensión de los datos.

Actividades	Descripción	Tareas
-------------	-------------	--------

Construir base de datos de Metainformación.	Construcción de base de datos que contenga las variables de la campaña con excepción del contenido textual (descripción, actualizaciones y comentarios acerca del proyecto).	<ul style="list-style-type: none">– Descargar conjuntos de datos comprimidos de la página Web Robots.– Descomprimir archivos en partes y fusionarlos en uno solo según su mes.– Fusionar archivos por mes, realizar limpieza de datos y generar base final.
Construir base de datos de Descripción.	Construcción de base de datos que contenga la descripción del proyecto en la página principal de la campaña.	<ul style="list-style-type: none">– Identificar sección de descripción a partir de la búsqueda del URL desde la base de datos de Metainformación.– Capturar información de sección identificada.– Generar base final.
Construir base de datos de Comentarios.	Construcción de base de datos que contenga los comentarios de los patrocinadores del proyecto en su sección correspondiente.	<ul style="list-style-type: none">– Identificar sección de comentarios a partir de la búsqueda del URL desde la base de datos de Metainformación.– Capturar información de sección identificada, excluyendo comentarios y respuestas de los creadores del proyecto.– Generar base final.
Realizar análisis exploratorio y estadístico de variables considerados.	Análisis exploratorio y estadístico de las variables disponibles para cada modalidad.	<ul style="list-style-type: none">– Describir distribución de los datos y tendencias.– Analizar existencia de correlación entre variables.

Fuente: Elaboración propia

A continuación, se detalla cada actividad y su entregable respectivo.

Actividad 1: Construir base de datos de Metainformación

De acuerdo a la Figura 78, en esta actividad se enfocó en la construcción de la base de datos

de la Metainformación desde la descarga de los conjuntos de datos comprimidos de la página Web Robots hasta la generación del archivo final luego de limpiar los datos.

Las 2 primeras tareas de la anterior tabla se resumen en el flujo de la Figura 81.



Figura 81. Proceso de obtención y preparación de datasets iniciales de Metainformación.

Fuente: Elaboración propia.

La siguiente fase de la actividad fue la selección de variables y limpieza de datos utilizando el software Alteryx Designer, el cual permite desarrollar flujos de trabajo para preparar, unir y analizar volúmenes de datos complejos de distintas fuentes. Se ejecutó el flujograma representado de forma resumida en la Figura 82.



Figura 82. Proceso resumido de generación de conjunto final de Metainformación.

Fuente: Elaboración propia.

Entregable: Conjunto de datos de Metainformación, que contiene 19 variables cuantitativas y cualitativas para la muestra considerada.

Actividad 2: Construir base de datos de Descripción

De acuerdo a la Figura 78, una vez generado el conjunto de datos de Metainformación descrito

en la anterior actividad, a partir de la variable *urls* se puede extraer la descripción de cada proyecto, siguiendo el proceso del diagrama de flujo representado en la Figura 83.

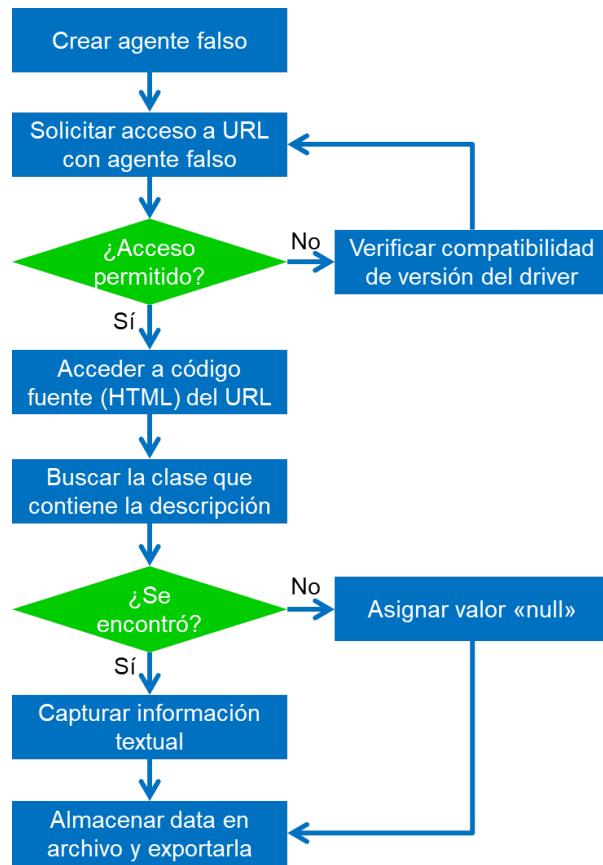


Figura 83. Proceso de extracción de la descripción de un proyecto.

Fuente: Elaboración propia.

Entregable: Conjunto de datos de Descripción, que contiene la variable textual de descripción del proyecto para la muestra considerada.

Actividad 3: Construir base de datos de Comentarios

De acuerdo a la Figura 78, y realizando el mismo ejercicio con Descripción, una vez generado el conjunto de datos de Metainformación, a partir de la variable *urls* se pueden extraer los comentarios de los patrocinadores por cada proyecto, siguiendo el proceso del diagrama de flujo representado en la Figura 84.

Entregable: Conjunto de datos de Comentarios, que contiene la variable textual de comentarios de los patrocinadores, cada uno separado por autor y almacenados en conjunto en una lista por proyecto, para la muestra considerada.

Actividad 4: Realizar análisis exploratorio y estadístico de variables considerados

En esta actividad, las variables de los 3 conjuntos finales de datos se describen en distribución

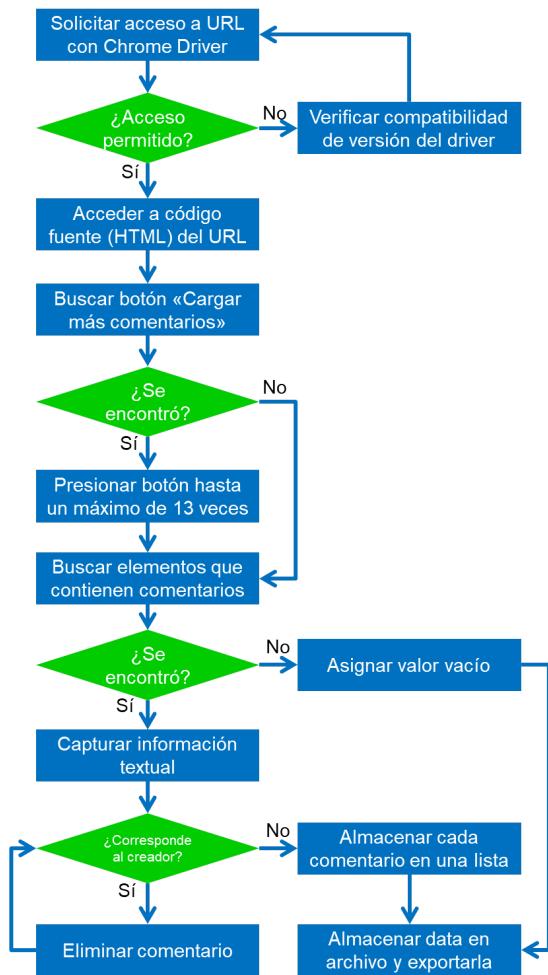


Figura 84. Proceso de extracción de comentarios de un proyecto.

Fuente: Elaboración propia.

de gráficos de barras, diagramas de cajas y bigotes y análisis de correlaciones de variables para la Metainformación; y nubes de palabras para la Descripción y Comentarios.

Entregable: Gráficos estadísticos de Metainformación, Descripción y Comentarios.

3.3.1.3. Preparación de los datos

En el anterior paso, se logró analizar estadísticamente las tendencias y distribuciones de cada variable según su respectiva modalidad. En esta etapa, como se detalla en la Tabla 6, las variables observadas fueron pre-procesadas con la finalidad de incluir datos homologados que no afecten negativamente el rendimiento de los modelos.

Tabla 6
Actividades de fase Preparación de los datos.

Actividades	Descripción	Tareas
Pre-procesar base de datos de Metainformación.	Pre-procesamiento de las variables actuales y adicionales luego de evaluación de su comportamiento entre ellas.	<ul style="list-style-type: none"> – Añadir más variables cuantitativas según literatura. – Evaluar comportamiento de variables en conjunto utilizando matriz de correlaciones. – Armar grupos combinatorias potenciales de variables. – Escalar variables a un mismo rango.
Pre-procesar base de datos de Descripción.	Pre-procesamiento de la variable <i>description</i> .	<ul style="list-style-type: none"> – Realizar limpieza de datos de los textos. – Armar vocabulario de palabras únicas. – Transformar textos en vectores de palabras.
Pre-procesar base de datos de Comentarios.	Pre-procesamiento de la variable <i>comments</i> .	<ul style="list-style-type: none"> – Realizar limpieza de datos de los textos. – Armar vocabulario de palabras únicas. – Transformar textos en vectores de palabras.

Fuente: Elaboración propia

A continuación, se detalla cada actividad y su entregable respectivo.

Actividad 1: Pre-procesar base de datos de Metainformación

En esta actividad, antes de realizar el pre-procesamiento de las variables finales, se agregaron algunas cuantitativas sugeridas en la literatura que no estaban consideradas inicialmente. Para evaluar correctamente el nuevo conjunto de variables, se realizó un nuevo análisis de correlación y se armaron grupos de combinatorias de variables cuyos rendimientos serán medidos en los siguientes pasos. Previamente a la evaluación, los subconjuntos de entrenamiento y prueba

divididos en 80 % y 20 % respectivamente (según H. Yuan et al. (2016), P.-F. Yu et al. (2018), L.-S. Chen y Shen (2019), Mitra y Gilbert (2014) y Sawhney et al. (2016)) estratificados según la variable *state*, se escalaron a un mismo rango para evitar un entrenamiento incorrecto.

Entregable: Conjuntos de datos pre-procesados de Metainformación.

Actividad 2: Pre-procesar base de datos de Descripción

Esta actividad se resume en la Figura 85, en donde se realizó desde la limpieza de los datos textuales, eliminando palabras y partes de ellas que no aportan para el diccionario final que se usará en la siguiente etapa, hasta la elaboración del vocabulario de palabras únicas, mayormente sustantivos y verbos en su forma original, los cuales se les asignaron un código para poder formar vectores de palabras y crear más adelante, las incrustaciones de palabras.

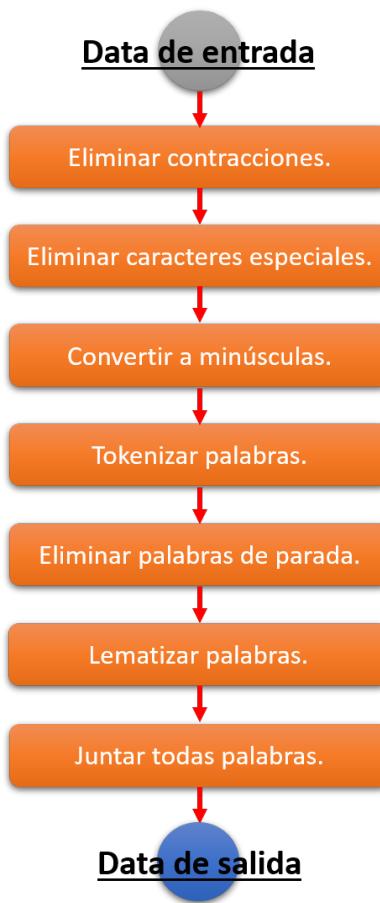


Figura 85. Proceso de limpieza de conjunto de datos de descripciones.

Fuente: Elaboración propia.

Luego de separar, al igual que la Metainformación, los subconjuntos de entrenamiento y prueba en 80 % y 20 % respectivamente estratificados según la variable *state*, se ejecutó el proceso de la Figura 86 dividido en 2 partes.

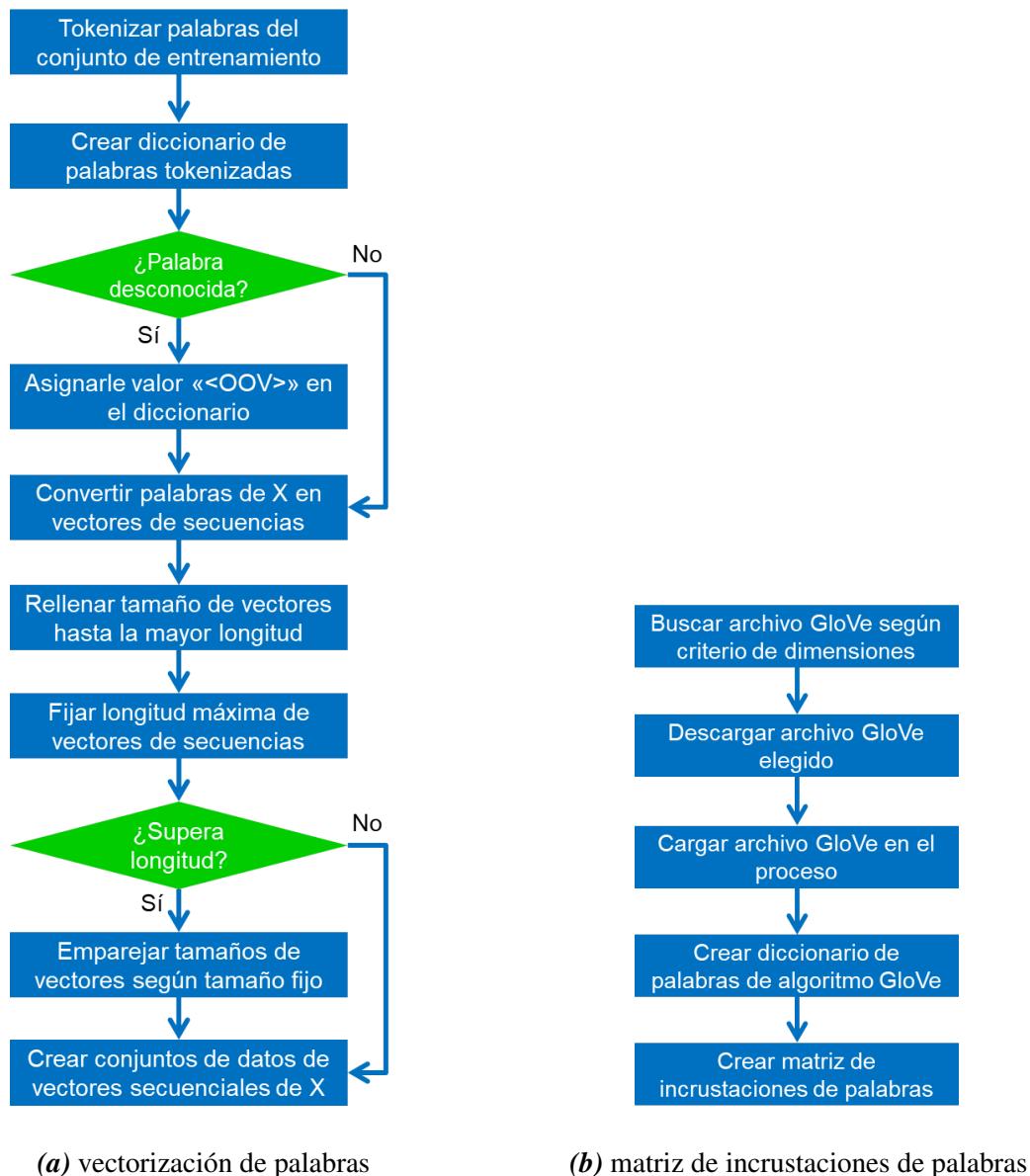


Figura 86. Procesos de vectorización y creación de matriz incrustaciones de palabras.

Fuente: Elaboración propia.

La primera mitad corresponde a la transformación del texto, para los subconjuntos de entrenamiento y prueba, de las descripciones en representaciones de vectores numéricos para la fase de entrenamiento del modelo. De forma paralela, se creó una matriz de incrustaciones de palabras que será parte de la capa de incrustaciones en la arquitectura del modelo.

Entregable: Descripciones representadas en secuencias de vectores numéricos y matriz de incrustaciones de palabras.

Actividad 3: Pre-procesar base de datos de Comentarios

Al igual que en el proceso de Descripción, se realizó la limpieza de los datos textuales si-

guiendo la misma secuencia de la Figura 84 para generar la representación de palabras de los comentarios en vectores numéricos, con la diferencia de acotar el tamaño final de los vectores debido a su extensa longitud al agruparse todos aquellos separados por autor. De igual manera, para generar la matriz de incrustaciones de palabras, se repitió el proceso de la Figura 86.

Entregable: Comentarios representadas en secuencias de vectores numéricos y matriz de incrustaciones de palabras.

3.3.1.4. Modelamiento

Al concluir la etapa del pre-procesamiento, las variables pudieron ser utilizadas para los modelos de cada modalidad, que fueron diseñados siguiendo las actividades de la Tabla 7.

Tabla 7
Actividades de fase Modelamiento.

Actividades	Descripción	Tareas
Desarrollar modelo predictivo de Metainformación.	Implementación del modelo Perceptrón Multicapa (MLP) para la modalidad Metainformación.	<ul style="list-style-type: none"> - Realizar benchmarking de modelos utilizados en trabajos previos. - Diseñar arquitectura del modelo predictivo de acuerdo a opción escogida.
Desarrollar modelo predictivo de Descripción.	Implementación de la Red Neuronal Convolucional (CNN) para la modalidad Descripción.	<ul style="list-style-type: none"> - Realizar benchmarking de modelos utilizados en trabajos previos. - Diseñar arquitectura del modelo predictivo de acuerdo a opción escogida.
Desarrollar modelo predictivo de Comentarios.	Implementación del modelo LSTM Bidireccional para la modalidad Comentarios.	<ul style="list-style-type: none"> - Realizar benchmarking de modelos utilizados en trabajos previos. - Diseñar arquitectura del modelo predictivo de acuerdo a opción escogida.

Desarrollar modelo ensamblado apilado.	Implementación del modelo de Aprendizaje Profundo Multimodal.	<ul style="list-style-type: none">- Realizar benchmarking de modalidades utilizadas en trabajos previos.- Diseñar arquitectura del modelo ensamblado apilado.
----------------------------------------	---------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Fuente: Elaboración propia

Actividad 1: Desarrollar modelo predictivo de Metainformación

En esta actividad, se diseñó la arquitectura que se usó para la modalidad Metainformación. Para ello, se realizó benchmarking sobre las propuestas de los autores enunciadas a continuación.

- **Perceptrón Multicapa (MLP)**: R. S. Kamath y Kamat (2018), P.-F. Yu et al. (2018), Cheng et al. (2019).
- **Máquina de Vectores de Soporte (SVM)**: K. Chen et al. (2013), Beckwith (2016), Sawhney et al. (2016).
- **Regresión Logística**: Mitra y Gilbert (2014), M. Zhou et al. (2015), Beckwith (2016), Y. Li et al. (2016), Kaur y Gera (2017).
- **Regresión Log-logística**: Y. Li et al. (2016).
- **Bosques Aleatorios**: S.-Y. Chen et al. (2015), H. Yuan et al. (2016), R. S. Kamath y Kamat (2018).
- **Árboles de Decisión**: Beckwith (2016), R. S. Kamath y Kamat (2018).
- **Naïve Bayes**: Beckwith (2016), R. S. Kamath y Kamat (2018).
- **Modelo Seq2seq**: Jin et al. (2019).

De acuerdo a los resultados de los autores citados, los modelos de Redes Neuronales (MLP), SVM y Regresión Logística tuvieron los mejores rendimientos en sus experimentos. Dado que la investigación consistió en implementar un modelo de Aprendizaje Profundo Multimodal, en donde se ensamblan modelos de Aprendizaje Profundo, se optó por el modelo Perceptrón Multicapa para la Metainformación.

Para desarrollar la arquitectura del modelo Perceptrón Multicapa, se evaluaron metodologías y estrategias utilizadas en la literatura, como la investigación de los autores P.-F. Yu et al. (2018) y las 17 Reglas del Pulgar según Ranjan (2019).

De estas últimas reglas, se consideraron las siguientes 13:

- El número de capas ocultas debe comenzar con 2, sin considerar la última capa.
- El número de neuronas en capas intermedias debe seguir progresión geométrica de 2.
- El número de nodos para la capa de salida de clasificación debe ser 1 si es clasificación binaria, y el número de clases si se trata de una clasificación multiclase.
- Usar la función de activación *relu* para las capas intermedias.
- La función de activación para la capa de salida debe ser *sigmoid* para clasificación binaria y *softmax* para clasificación multiclase.
- Las capas de desactivación deben de ir después de cada capa, excepto luego de la capa de entrada, y su ratio debe ser recomendable 0.5, aunque podría usarse un menor valor.
- Usar escaladores como *MinMaxScaler*, o *StandardScaler* si el anterior no funciona bien, como método para pre-procesar los datos cuantitativos.
- Usar la función *train_test_split* para separar la data en subconjuntos de entrenamiento y prueba. En caso la data se encuentre desbalanceada, utilizar parámetro *stratify* con Y.
- Balancear los pesos de las clases en caso la data se encuentre desbalanceada.
- Usar *Adam* como optimizador con sus ratios por defecto de preferencia.
- Utilizar *binary_crossentropy* como función de pérdida para clasificación binaria y *categorical_crossentropy* para clasificación multiclase.
- Utilizar la métrica de exactitud (*accuracy*) para clasificación. En caso se tenga data desbalanceada, se sugiere usar sensibilidad (*recall*) o ratio de falsos positivos.
- Comenzar con 20 épocas y aumentar gradualmente en caso se perciba mejora de la exactitud y decrecimiento de la pérdida. Se recomienda entrenar con hasta 100 épocas.

Por el lado de la metodología seguida por P.-F. Yu et al., los investigadores determinaron el número de neuronas de la capa de entrada como el número de variables del conjunto de datos, el número de neuronas en la capa de salida como el número de clases, y agregar una o más capas lineales como capas ocultas entre la capa de entrada y la capa de salida. Se encontraron

puntos de coincidencias con las Reglas del Pulgar en la función de activación como *relu* para las capas ocultas, la función de activación *sigmoid* para la capa de salida en caso se presente un problema de clasificación binario y *softmax* para un problema de clasificación multiclas, utilizar de preferencia el optimizador *Adam* y aumentar el número de épocas de entrenamiento en caso darse mejoras del rendimiento del modelo.

Entregable: Modelo Perceptrón Multicapa para modalidad Metainformación.

Actividad 2: Desarrollar modelo predictivo de Descripción

En esta actividad, se diseñó la arquitectura que se usó para la modalidad Descripción. Para ello, se realizó benchmarking sobre las propuestas de los autores enunciadas a continuación.

- **CNN:** Cheng et al. (2019).
- **LSTM:** Jin et al. (2019).
- **Modelo Seq2seq:** S. Lee et al. (2018).
- **Máquina de Vectores de Soporte o variantes:** Sawhney et al. (2016), L.-S. Chen y Shen (2019).
- **Regresión Logística:** Mitra y Gilbert (2014), M. Zhou et al. (2015).
- **LDA o variantes:** H. Yuan et al. (2016), Sawhney et al. (2016).

De acuerdo a la anterior lista, los modelos como el LDA o Seq2seq requirieron procesadores de gama alta o pocos proyectos para ser entrenados debido a la complejidad de su arquitectura. Por el lado de los modelos SVM y Regresión Logística, al ser métodos de Aprendizaje Automático, se descartaron para la investigación dado a la imposibilidad de ser anexadas a modelos de Aprendizaje Profundo. Finalmente, entre el modelo CNN y LSTM, se optó por elegir el primero para la modalidad de descripción dado que este, el cual bajo una arquitectura unidimensional puede ser aplicado para textos, solo necesita usar representaciones de textos en vectores para clasificar un target, a diferencia del segundo método, el cual fue utilizado para la modalidad de comentarios al tratarse estos de secuencias de palabras.

Para desarrollar el submodelo basado en una CNN, se siguieron los pasos utilizados por Malik (2019) para diseñar la arquitectura, asignando el tamaño máximo de palabras de las descripciones como el número de neuronas de la capa de entrada, en este caso 3,671, fijando por defecto 128 filtros para la capa de convolución unidimensional y tamaño de kernel 5, agregando una capa de reducción (*GlobalMaxPooling*) luego de la capa de convolución y una capa densa antes de la capa de salida de clasificación. La dimensión de la capa de incrustación fue 100

ya que la matriz asignada en este parámetro fue entrenada con el algoritmo GloVe de 100 dimensiones. Los parámetros establecidos para el número de neuronas de las capas intermedias y funciones de activación de cada capa derivaron de las mismas estrategias empleadas para la construcción del submodelo de Metainformación, siendo 64 el número de neuronas de la única capa densa y utilizó la función *tanh* en vez de *relu* por presentar una mejor performance en el entrenamiento. Ambos son opciones válidas para el problema de la desaparición de gradientes según Brownlee (2019). Por último, la función de activación de la capa de salida, al igual que el anterior modelo, fue *sigmoid* por ser clasificación binaria.

Ardi (2020), por su lado, coincide con el párrafo anterior, agregando una capa de aplanaamiento después de la capa de reducción y hasta 3 capas de desactivación entre la capa de reducción y la capa de salida. Para el presente trabajo, las capas de desactivación no fueron tomadas en cuenta ya que su inclusión no afectó notablemente el rendimiento del modelo entrenado, como se explica más adelante en el Capítulo V.

Entregable: Red Neuronal Convolutacional para modalidad Descripción.

Actividad 3: Desarrollar modelo predictivo de Comentarios

En esta actividad, se diseñó la arquitectura que se usó para la modalidad Comentarios. Para ello, se realizó benchmarking sobre las propuestas de los autores enunciadas a continuación.

- **LSTM:** Jin et al. (2019), Shafqat y Byun (2019).
- **LDA o variantes:** Shafqat y Byun (2019).
- **Modelo Seq2seq:** S. Lee et al. (2018), Jin et al. (2019).
- **HAN:** S. Lee et al. (2018).
- **Regresión Logística:** Y. Li et al. (2016), Kaur y Gera (2017).
- **Regresión Log-logística:** Y. Li et al. (2016).

Como se menciona en la anterior actividad, para el modelo predictivo de comentarios se implementó un modelo de LSTM dado el comportamiento de esta modalidad basada en secuencias de palabras. Además, se definió esta red como Bidireccional para que cada comentario que se busque predecir pueda ser entrenado en ambas direcciones de la secuencia, es decir con palabras previas y posteriores en una oración, logrando así encaminar de forma más efectiva hacia su clasificación de estado de financiamiento.

En una primera instancia, se planeó considerar desarrollar un modelo LDA teniendo como base el trabajo de Shafqat y Byun (2019). Sin embargo, tal y como los autores mencionan, la complejidad de su arquitectura sumado a la poca cantidad de registros utilizados para el entrenamiento (solo 600 proyectos) contrastan con las características de la variable de comentarios en esta investigación. Además, este tipo de redes neuronales recurrentes representa una mejor opción para tratar problemas de secuencias de palabras ya que almacenan estados previos para decidir el siguiente en un texto.

Para desarrollar el submodelo basado en una LSTM Bidireccional, al tratarse de un modelo más complejo que un MLP o una CNN ya que está compuesto por compuertas y funciones de activación pre-establecidas, según Malik (2019) se debe diseñar la misma arquitectura que se utilizó para una CNN, con la diferencia de reemplazar desde la capa de convolución unidimensional hasta la capa densa previa a la capa de salida por una capa LSTM con un número de neuronas asignada por el usuario. Para mantener la relación con el modelo anterior también basado en contenido textual, se asignaron 128 neuronas. Según Brownlee (2017b), con la finalidad de lograr mejores resultados luego del entrenamiento, se debe establecer la capa como Bidireccional. Como resultado de este valor agregado, el número de neuronas se duplicó a 256 ya que la característica de las RNN Bidireccionales es la búsqueda de un estado guardado previa y posteriormente al estado actual, es decir, en 2 sentidos (ver ejemplo de Figura 69). La longitud de la capa de entrada fue de 5,000, un valor fijado por el usuario ante el enorme costo computacional que supone utilizar la longitud máxima de comentarios, como se definió en el modelo de descripción. Estos detalles se explican más adelante en el Capítulo V. Por último, el ratio de desactivación de su capa respectiva fue de 0.50 según una de las Reglas del Pulgar, la dimensión de la capa de incrustación fue de 100 al igual que el anterior modelo y la función de activación de la capa de activación continuó siendo *sigmoid* por ser una clasificación binaria.

Entregable: Modelo LSTM Bidireccional para modalidad Comentarios.

Actividad 4: Desarrollar modelo ensamblado apilado

En esta actividad, los modelos desarrollados por cada modalidad se ensamblaron luego de ajustarse sus hiperparámetros a partir de los mejores resultados obtenidos. La apilación consistió en cargar cada uno simultáneamente, y unir las predicciones de salida en una nueva capa. Debido a esta acción, para entrenar el modelo de Aprendizaje Profundo Multimodal desarrollado se necesitó unir en una lista los datos de entrada utilizados para las 3 modalidades, ya que estos ingresan en simultáneo.

De los 2 enfoques de modelos ensamblados de Aprendizaje Profundo explicados por Brownlee (2018) en el Capítulo II, se implementó un modelo apilado integrado, ya que la otra alternativa se basa en juntar las predicciones de distintos modelos para entrenar la unión en

un nuevo modelo. Además, la primera estrategia consiste en asignar todas las capas de los modelos cargados como «no entrenables», con el fin de evitar que los pesos establecidos de cada submodelo no se actualicen durante la fase de entrenamiento del modelo propuesto y solamente ocurra para los pesos de la nueva capa oculta y la de salida, consiguiendo así una mejor clasificación a partir de la búsqueda de la mejor combinación de las predicciones de cada modalidad. El autor también agrega una capa densa luego de la capa concatenada para profundizar un poco más la red creada y lograr un mejor rendimiento del modelo. El número de neuronas definidas en esta capa oculta no sigue una regla como el caso de los anteriores modelos, ya que el autor sugirió utilizar 10 neuronas. Sin embargo, sí se mantiene aplicando la definición de la función de activación *relu* para la capa oculta y *sigmoid* para la capa de salida.

Los autores Cheng et al. (2019), asimismo, son los únicos de los 18 antecedentes mencionados que implementaron un modelo de Aprendizaje Profundo Multimodal. Su trabajo consistió en ensamblar 3 modalidades presentes en la sección principal de la campaña: la metainformación, la imagen y la descripción del proyecto. Para ello, utilizaron los modelos SVM, un VGG16 pre-entrenado con ImageNet y BoVW, y SVMs usando BoW y TF-IDF respectivamente.

Entregable: Modelo de Aprendizaje Profundo Multimodal

3.3.1.5. Evaluación

Esta fase comprende la evaluación de los modelos desarrollados en la investigación a través de las métricas de clasificación seleccionadas y explicadas en la sección 3.3.2. En la Tabla 8 se indican las actividades y tareas seguidas para esta etapa.

Tabla 8
Actividades de fase Evaluación.

Actividades	Descripción	Tareas
Evaluar el modelo predictivo de Metainformación.	Evaluación del desempeño del modelo Perceptrón Multicapa para la modalidad Metainformación.	<ul style="list-style-type: none"> - Ejecutar experimentos para definir hiperparámetros. - Comparar resultados por cada grupo de combinatorias de variables.

Evaluar el modelo predictivo de Descripción.	Evaluación del desempeño de la Red Neuronal Convolucional para la modalidad Descripción.	<ul style="list-style-type: none"> - Ejecutar experimentos para definir hiperparámetros. - Comparar resultados de cada experimento.
Evaluar el modelo predictivo de Comentarios.	Evaluación del desempeño de la red LSTM Bidireccional para la modalidad Comentarios.	<ul style="list-style-type: none"> - Ejecutar experimentos para definir hiperparámetros. - Comparar resultados de cada experimento.
Evaluar el modelo ensamblado apilado.	Evaluación del desempeño del modelo de Aprendizaje Profundo Multimodal.	<ul style="list-style-type: none"> - Entrenar modelo combinando cada modalidad. - Ejecutar experimentos para definir hiperparámetros. - Comparar resultados de cada experimento.

Fuente: Elaboración propia

Actividad 1: Evaluar desempeño del modelo predictivo de Metainformación

En esta actividad, primero se entrenó el submodelo durante 100 épocas alternando con cada una de las 8 combinaciones de variables potenciales. Se determinó así al mejor grupo a partir del mejor valor de exactitud obtenido con el conjunto de datos de validación. Para obtener los mejores hiperparámetros, se ejecutaron entre 3 y 5 experimentos del modelo diseñado de Metainformación con las nuevas variables. Al compararse los resultados no solo se definieron los hiperparámetros de mejor performance, sino también las variables finales del modelo que superaron al resto a nivel general. Luego de definir el modelo mejor calibrado, este fue entrenado con las clases balanceadas de la variable dependiente y una configuración personalizada de elementos para su ejecución. Para ello, se utilizó un acelerador GPU de máximo 38 GB de RAM con tiempo de ejecución estándar. Al final de la acción, se calcularon los valores de las 5 métricas por el cual fue evaluado a partir de su matriz de confusión.

Entregable: Resultados de métricas de clasificación para modelo de Metainformación.

Actividad 2: Evaluar desempeño del modelo predictivo de Descripción

En esta actividad, para obtener los mejores hiperparámetros, se ejecutaron igualmente entre 3 y 5 experimentos del modelo diseñado de Descripción, comparando cada resultado con su

sucesor. Si este, con distintos valores para sus hiperparámetros, generaba una mayor exactitud y menor pérdida, tanto en entrenamiento como en prueba, se le consideraba como el modelo mejor calibrado. El ejercicio se repitió hasta que un experimento no logre superar en performance a su antecesor. Luego de definir el modelo mejor calibrado, este fue entrenado con las clases balanceadas de la variable dependiente y una configuración personalizada de elementos para su ejecución. Para ello, se utilizó un acelerador GPU. Al final de la acción, se calcularon los valores de las 5 métricas por el cual fue evaluado a partir de su matriz de confusión.

Entregable: Resultados de métricas de clasificación para modelo de Descripción.

Actividad 3: Evaluar desempeño del modelo predictivo de Comentarios

Repetiendo el proceso para la modalidad de Descripción, para obtener los mejores hiperparámetros, se ejecutaron entre 3 y 5 experimentos del modelo diseñado de Comentarios, comparando cada resultado con su sucesor. Si este, con distintos valores para sus hiperparámetros, generaba una mayor exactitud y menor pérdida, tanto en entrenamiento como en prueba, se le consideraba como el modelo mejor calibrado. El ejercicio se repitió hasta que un experimento no logre superar en performance a su antecesor. Luego de definir el modelo mejor calibrado, este fue entrenado con las clases balanceadas de la variable dependiente y una configuración personalizada de elementos para su ejecución. A diferencia de los 2 modelos anteriores, se utilizó un acelerador TPU ya que se requirió mayor rendimiento de RAM. Al final de la acción, se calcularon los valores de las 5 métricas por el cual fue evaluado a partir de su matriz de confusión.

Entregable: Resultados de métricas de clasificación para modelo de Comentarios.

Actividad 4: Evaluar desempeño del modelo ensamblado apilado

Finalmente, en esta actividad se entrenó el modelo ensamblado por las 3 modalidades que comprende el trabajo. Para determinar los mejores hiperparámetros, también se ejecutaron entre 3 y 5 experimentos siguiendo las metodologías anteriores. Para el modelo propuesto, se utilizó un acelerador GPU ya que fue entrenado con modelos pre-entrenados que solo requirió unir sus datos de entrada para esta acción. Para concluir la actividad, se calcularon los valores de las 5 métricas por el cual fue evaluado a partir de su matriz de confusión.

Entregable: Resultados de métricas de clasificación para modelo de Aprendizaje Profundo Multimodal.

3.3.1.6. Despliegue

En la última fase de la metodología seguida, se desplegó el modelo propuesto una vez terminada su entrenamiento y evaluación correspondiente. Las actividades que se llevaron a cabo son detalladas en la Tabla 9.

Tabla 9
Actividades de fase Despliegue.

Actividades	Descripción	Tareas
Diseñar prototipo de sistema con modelo propuesto.	Diseño del prototipo del sistema que integrará la captura de datos con el modelo propuesto para predecir el estado de financiamiento de un proyecto.	<ul style="list-style-type: none"> – Diseñar estructura de prototipo de sistema conformado por captura de datos y modelo propuesto. – Cargar complementos del modelo.
Ejecutar prototipo con proyectos tecnológicos vigentes.	Ejecución del sistema prototipo en tiempo real con proyectos de tecnología de campañas vigentes.	<ul style="list-style-type: none"> – Ejecutar prototipo con dirección web de proyecto consultado. – Clasificar el estado de financiamiento a partir de resultado predicho.

Fuente: Elaboración propia

Luego de analizar el desempeño de cada modelo, tanto individual como el modelo final, se compararon los resultados con los antecedentes y se desarrolló una prueba piloto en la cual el modelo apilado entrenado fue ejecutado usando como entrada la URL de un proyecto aleatorio de Kickstarter para, luego de obtener sus características, predecir su estado de financiamiento. Esta acción también se encuentra detallada en el Capítulo V.

Actividad 1: Diseñar prototipo de sistema con modelo propuesto

En esta actividad, se diseñó el prototipo del sistema que integra tanto su fase inicial de captura de datos del URL dado por el usuario como el modelo de Aprendizaje Profundo Multimodal denominado «The Hydra». Antes de comenzar el proceso seguido en la Figura 87, se cargaron los complementos necesarios para el correcto funcionamiento del mismo.

Entregable: Arquitectura del prototipo del sistema integrado.

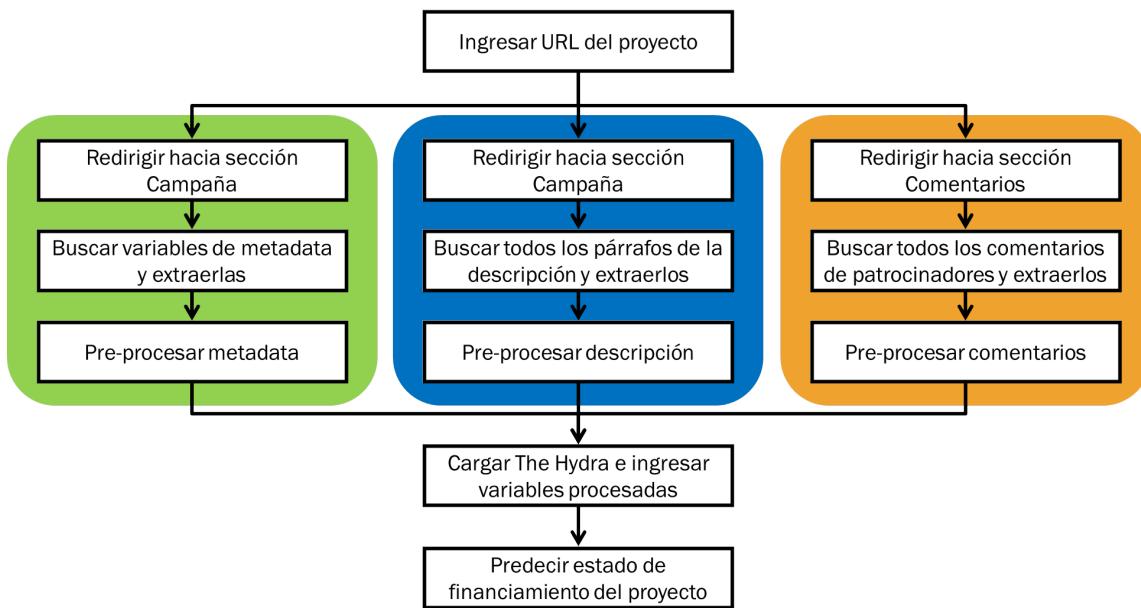


Figura 87. Proceso de operación del prototipo del sistema.

Fuente: Elaboración propia.

Actividad 2: Ejecutar prototipo con proyectos tecnológicos vigentes

En esta actividad, se ejecutan las acciones de la arquitectura del prototipo mencionadas en la figura anterior. Comienza desde la captura de datos del proyecto consultado a través de su dirección web y culmina con la predicción de su estado de financiamiento.

Entregable: Predicción del proyecto consultado.

3.3.2 Metodología para la medición de resultados

Para evaluar la performance de un modelo, que representa la quinta fase de la metodología CRISP-DM, se utilizan diversas métricas como instrumentos de medición de desempeño a partir de los resultados arrojados en la Matriz de confusión. A continuación, se detalla su concepto y sus elementos.

- **Matriz de confusión:** Es una tabla de NxN que resume el nivel de éxito de las predicciones de un modelo de clasificación; es decir, la correlación que existe entre la etiqueta y la clasificación del modelo. Un eje de una matriz de confusión es la etiqueta que el modelo predijo; el otro es la etiqueta real. N representa el número de clases. Es un problema de clasificación binaria, N=2 (Kohavi & Provost, 1998). Su principal objetivo es describir el rendimiento de un modelo supervisado de Machine Learning en los datos de prueba, donde se desconocen los verdaderos valores. Se le llama “matriz de confusión” porque hace que sea fácil detectar dónde el sistema está confundiendo dos clases (SitioBigData.com,

2019a). Se representa en la Tabla 10.

Tabla 10
Matriz de confusión.

		Valores Actuales	
		Positivos (1)	Negativos (0)
Valores Predichos	Positivos (1)	Verdaderos Positivos (VP)	Falsos Positivos (FP)
	Negativos (0)	Falsos Negativos (FN)	Verdaderos Negativos (VN)

Fuente: Izco (2018)

- **Verdadero positivo** (TP o *True Positive*): Es el ejemplo en el que el modelo predijo de manera correcta la clase positiva. Por ejemplo, el modelo infirió correctamente que un paciente con determinadas características descritas en las variables sufre de cáncer (Google Developers, 2018).
- **Verdadero negativo** (TN o *True Negative*): Es el ejemplo en el que el modelo predijo de manera correcta la clase negativa. Por ejemplo, el modelo infirió correctamente que una determinada especie animal de acuerdo a sus características no era un mamífero (Google Developers, 2018).
- **Falso positivo** (FP, *False Positive* o Error del Tipo I): Es el ejemplo en el que el modelo predijo de manera incorrecta la clase positiva. Por ejemplo, el modelo infirió que un paciente varón presentaba embarazo (clase positiva) cuando en realidad no era así (Google Developers, 2018).
- **Falso negativo** (FN, *False Negative* o Error del Tipo II): Es el ejemplo en el que el modelo predijo de manera incorrecta la clase negativa. Por ejemplo, el modelo infirió que un mensaje de correo electrónico en particular no era spam (clase negativa), pero ese mensaje en realidad sí era spam (Google Developers, 2018).

Explicado los conceptos anteriores, se derivan las siguientes métricas de clasificación usadas comúnmente, de las cuales serán usadas solo las 3 primeras tomando como referencia los papers de los antecedentes:

- **Exactitud** (*accuracy*): Representa la fracción de predicciones que se realizaron correctamente sobre el total de ejemplos en un modelo de clasificación. Se determina mediante la siguiente fórmula (Kohavi & Provost, 1998):

$$\text{Exactitud} = \frac{V.P. + V.N.}{V.P. + V.N. + F.P. + F.N.} \quad (23)$$

Esta métrica responde a la pregunta ¿Cuál es la proporción de predicciones que se realizaron correctamente? (Izco, 2018)

- **Precisión (precision):** Representa el número de elementos identificados correctamente como positivo de un total de elementos identificados como positivos (SitioBigData.com, 2019a). Se calcula mediante la siguiente fórmula:

$$\text{Precisión} = \frac{V.P.}{V.P. + F.P.} \quad (24)$$

Esta métrica responde a la pregunta ¿Qué proporción de predicciones positivas es correcta? (Izco, 2018)

- **Área bajo la curva ROC (AUC):** Considera todos los umbrales de clasificación posibles. Representa la probabilidad de que un clasificador tenga más seguridad de que un ejemplo resulte ser un verdadero positivo con respecto a que sea un falso positivo (Google Developers, 2018). Para entender el concepto del área, se necesita entender qué es la curva ROC y para qué sirve en primer lugar.

La curva ROC permite cuantificar la performance de distinción entre dos cosas del modelo como, por ejemplo, si un paciente tiene cáncer o no.

Siguiendo el anterior ejemplo, se tiene un modelo que predice si un paciente sufre de cáncer o no, cuyo resultado es el siguiente (Figura 88)

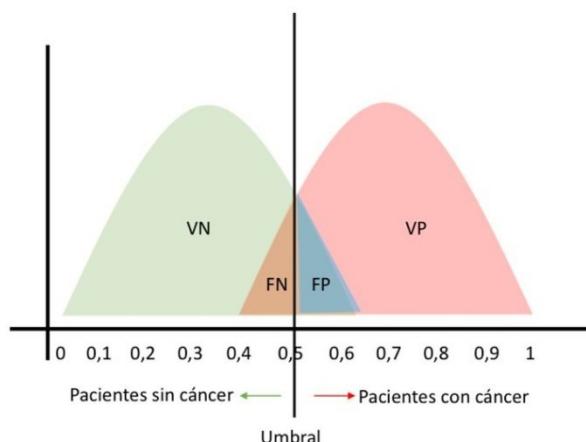


Figura 88. Descripción de resultados de modelo descriptivo de ejemplo.

Fuente: González (2019). *Curvas ROC y Área bajo la curva (AUC)*.

En esta imagen, se puede observar que el área de borde verde (que contiene a los Falsos Positivos y el total de Negativos) representa a todos los pacientes que no tienen cáncer, mientras que el área de borde rojo (que contiene a los Falsos Negativos y el total de Positivos) representa a todos los pacientes que sí tienen cáncer. El umbral, que está establecido con valor 0.5, representa el punto de corte en el que el modelo clasificará a todos los pacientes por encima de ese valor como positivos, es decir, que sí tienen cáncer; mientras que aquellos por debajo del valor del umbral serán clasificados como negativos, es decir, que no tienen cáncer.

Cuando el umbral se desplaza hacia la izquierda, es decir, cuando la sensibilidad aumenta, la especificidad disminuirá. Por el contrario, cuando el umbral se desplaza hacia la derecha, la sensibilidad disminuirá y la especificidad aumentará. Se concluye entonces que existe una relación inversa entre la sensibilidad y la especificidad. En la curva ROC se representa la sensitividad (1-especificidad) (González, 2019).

Ahora bien, el área que se grafica bajo esta curva explicará qué tan bien funciona el modelo. Este tendrá un mejor desempeño si la curva se aleja de la diagonal principal como se observa en la Figura 89 y se calcula con la siguiente fórmula:

$$P(score(x^+) > score(x^-)) \quad (25)$$

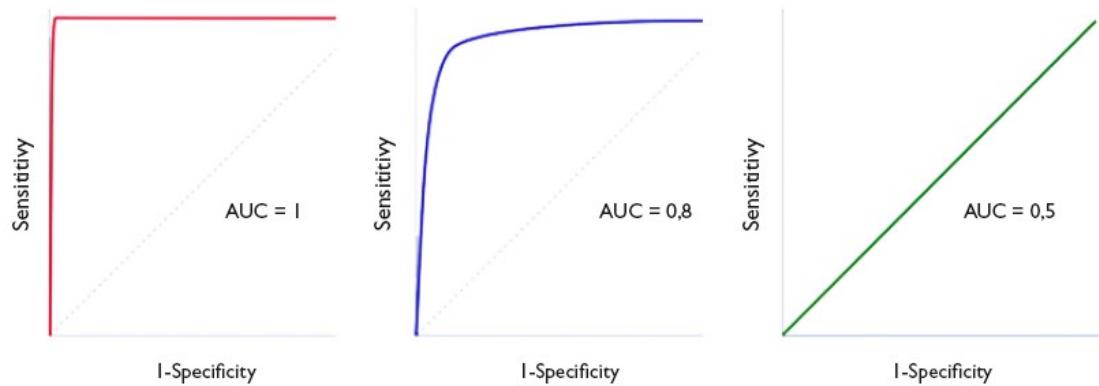


Figura 89. Comparación de tres resultados de la curva AUC en el modelo.

Fuente: Molina Arias y Ochoa Sangrador (2017). *Pruebas diagnósticas con resultados continuos o politómicos. Curvas ROC.* (p. 4)

Una interpretación básica del área bajo la curva ROC respecto del poder discriminante del modelo se muestra a continuación (Britos et al., 2006):

- Si el área bajo la curva $\text{ROC} = 0.5$, entonces el poder discriminante del modelo es nulo.
 - Si el área bajo la curva $0.5 < \text{ROC} < 0.7$, entonces el poder discriminante del modelo no es aceptable.
 - Si el área bajo la curva $0.7 \leq \text{ROC} < 0.8$, entonces el poder discriminante del modelo es aceptable.
 - Si el área bajo la curva $0.8 \leq \text{ROC} < 0.9$, entonces el poder discriminante del modelo es excelente.
 - Si el área bajo la curva $\text{ROC} \geq 0.9$, entonces el poder discriminante del modelo es excepcionalmente bueno.
- **Sensibilidad** (*recall, sensitivity* o *True Positive Rate*): Representa el número de elementos correctamente identificados como positivos del total de positivos verdaderos (Sitio-BigData.com, 2019a). Se calcula mediante la siguiente fórmula:

$$\text{Sensibilidad} = \frac{\text{V.P.}}{\text{V.P.} + \text{F.N.}} \quad (26)$$

- **Puntaje F1 (F1-Score)**: Representa la media armónica de la precisión y la sensibilidad. Normalmente, se usa cuando uno difiere mucho del otro y no es posible realizar una conclusión determinante ya que solo es posible predecir bien una clase (SitioBigData.com, 2019a). Se calcula mediante la siguiente fórmula:

$$\text{PuntajeF1} = \frac{2 * \text{Precisión} * \text{Sensibilidad}}{\text{Precisión} + \text{Sensibilidad}} \quad (27)$$

La elección de las anteriores métricas para evaluar cada modelo por modalidad y el modelo ensamblado propuesto se dio luego de realizar benchmarking sobre aquellas que fueron utilizadas por los autores en los antecedentes de la investigación.

- **Exactitud**: K. Chen et al. (2013), S.-Y. Chen et al. (2015), Beckwith (2016), H. Yuan et al. (2016), Sawhney et al. (2016), Kaur y Gera (2017), R. S. Kamath y Kamat (2018), P.-F. Yu et al. (2018), S. Lee et al. (2018), Cheng et al. (2019), L.-S. Chen y Shen (2019), Shafqat y Byun (2019).
- **Precisión**: Beckwith (2016), H. Yuan et al. (2016), Kaur y Gera (2017), Cheng et al. (2019).

- **Sensibilidad:** Beckwith (2016), H. Yuan et al. (2016), Kaur y Gera (2017), Cheng et al. (2019), L.-S. Chen y Shen (2019).
- **Especificidad:** L.-S. Chen y Shen (2019).
- **Ratio de Falsa Alarma:** Kaur y Gera (2017).
- **Media Geométrica (G-Mean):** L.-S. Chen y Shen (2019).
- **Puntaje F1:** M. Zhou et al. (2015), Beckwith (2016), H. Yuan et al. (2016), Kaur y Gera (2017), Cheng et al. (2019), L.-S. Chen y Shen (2019).
- **Curva ROC:** M. Zhou et al. (2015), Beckwith (2016).
- **Área bajo la Curva ROC (AUC):** Beckwith (2016), Y. Li et al. (2016), Kaur y Gera (2017), P.-F. Yu et al. (2018), Cheng et al. (2019).
- **Área bajo la Curva Precisión-Sensibilidad (PRC):** Kaur y Gera (2017).
- **Error de Validación Cruzada:** Mitra y Gilbert (2014).
- **Coeficiente de Correlación de Matthew (MCC):** Kaur y Gera (2017).
- **Divergencia de Kullback-Leibler (KL):** Jin et al. (2019).
- **Raíz del Error Cuadrático Medio (RMSE):** Jin et al. (2019).
- **Error Absoluto Medio (MAE):** Jin et al. (2019).
- **Índice de Concordancia (CI):** Jin et al. (2019).

3.4 Cronograma de actividades y presupuesto

Se elaboró un cronograma de actividades de toda la investigación, mostrada en la Figura 90, contemplando desde el inicio de la misma a mediados del 2020 hasta la sustentación del trabajo estimado para finales de mayo del 2021.

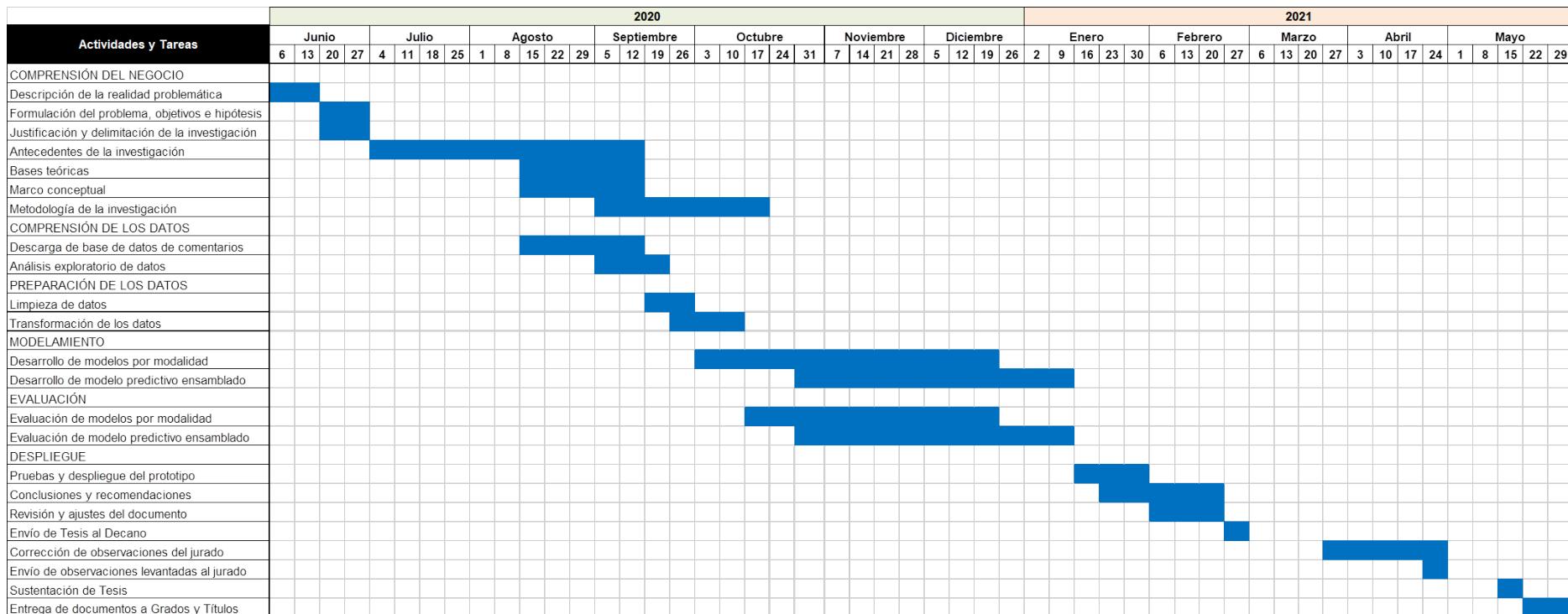


Figura 90. Cronograma de actividades de la investigación.

Fuente: Elaboración propia.

Los costos personales del autor de la investigación se muestran en la Tabla 11. Estos incluyen las herramientas adquiridas antes del inicio de la investigación como la laptop, así como pagos de servicios generales y del trámite de elaboración y sustentación pública de Tesis.

Tabla 11

Presupuesto de los costos personales del autor.

Item	Tiempo usado (horas)	Costo (soles)	Subtotal
Recursos materiales			
Laptop Lenovo ideapad 330 Core i7 8va Gen		S/.4,500.00	S/.4,500.00
Pagos del trámite de elaboración y sustentación pública de Tesis			
Derecho de inscripción de tema de investigación		S/.800.00	S/.800.00
Reserva del tema de tesis		S/.2,700.00	S/.2,700.00
Derecho de sustentación		S/.1,500.00	S/.1,500.00
Recursos humanos			
Avance de tesis	900	Incalculable	-
Servicios generales			
Internet + luz (7 meses)	110	S/.80.00	S/.560.00
Total			S/.10,060.00

Fuente: Elaboración propia.

Asimismo, también se contemplan los costos por uso de servicios en la nube como parte del proceso extracción de comentarios desde servidores en Google Cloud Platform (GCP) y desarrollo de modelos en Google Colab Pro en la Tabla 12.

Tabla 12

Presupuesto de los costos de las herramientas para el proyecto.

Item	Unidades	Costo (dólares)	Horas	Subtotal
Google Cloud Platform				-\$98.56
Instancia VM Ubuntu (g1-small, 12GB RAM)	1	\$0.021	310	\$6.51
Imagen de instancia Ubuntu	8	\$0.02	306	\$48.96
Instancia VM de Windows Server 2019 (4GB RAM)	1	\$0.086	300	\$25.80
Costo por instancias encendidas	10			\$3.41
Uso de instancias posteriormente eliminadas				\$95.88
Pago mensual (luego de aceptar mejora de plan)	4	\$5.22		\$20.88
Crédito de \$300.00 por 12 meses	1	-\$300.00		-\$300.00
Google Colab Pro				\$49.95
Pago mensual (luego de aceptar mejora de plan)	5	\$9.99		\$49.95
Total a pagar				\$49.95

Fuente: Elaboración propia.

Los montos por el uso de instancias creadas en GCP son descontados de los \$300.00 en crédito gratuito válidos por 12 meses (Google Cloud, s.f.), recibidos inicialmente desde el 12 de agosto del 2020 para poder ser usados como versión de prueba. A partir del siguiente mes, se mejoró el plan para acceder a más servicios y comenzó a facturarse \$5.22 por mes.

Capítulo IV: Desarrollo del Experimento

En este capítulo se detalla el proceso explicado en el capítulo anterior para cada actividad de la metodología aplicada, así como los entregables comprometidas.

4.1 Comprensión del negocio

Actividad 1: Definir problemas, objetivos e hipótesis

El inicio de la implementación de la primera fase de la metodología CRISP-DM fue la identificación del problema general a partir del estudio de la realidad problemática abordada en los primeros capítulos. El objetivo, por lo tanto, busca resolver el problema y se encuentra alineado con el título de la investigación. Asimismo, la hipótesis resulta la proposición planteada para explicar el logro del objetivo. Los objetivos específicos del trabajo, que responde a cada problema específico, se definieron a partir de una lluvia de ideas en base a la asociación de objetivos específicos de los antecedentes con la actual investigación. Estos se explican en la siguiente actividad.

Actividad 2: Desarrollar la literatura de la investigación

Se buscaron desde libros y artículos online para la comprensión teórica del financiamiento colectivo e Inteligencia Artificial, hasta papers publicados en conferencias, revistas internacionales y científicas, reportes técnicos y tesis de grado acerca de propuestas para resolver el problema estudiado en la investigación.

Para ello, como se describió en la sección 3.2, a través de búsqueda de palabras clave como *crowdfunding*, *Machine Learning*, *Deep Learning*, *prediction*, *Kickstarter*, *accuracy* y *projects*, y el uso del buscador Google Académico, se encontraron estos papers publicados entre el 2013 y 2020.

A continuación, se realizó un resumen de cada antecedente en una hoja de cálculo de Excel con el fin de comparar sus objetivos y metodologías implementadas, como se puede observar el detalle en el Anexo E.

Actividad 3: Definir metodología de la investigación

Luego de elaborar el detalle del anterior anexo, a excepción de la investigación del autor Fernandez-Blanco et al. (2020) que utilizó la metodología CRISP-DM, cada antecedente fue agrupado con otro similar de acuerdo a los pasos seguidos en su propia metodología. El resultado fue la Tabla 1 explicada en la sección 3.3.1.

4.2 Comprensión de los datos

Actividad 1: Construir base de datos de Metainformación

El punto de partida para la construcción de los conjuntos de datos que se usaron más adelante en cada modelo de acuerdo a su modalidad fue la adquisición de bases de datos capturadas mensualmente desde finales del 2015 hasta 2019 por la página Web Robots (<https://webrabots.io/kickstarter-datasets/>), fundada por los ex corporativos de TI Tomás Vitulskis y Paulius Jonaitis, como se aprecia en la Figura 91 Web Robots (2019). Para el presente trabajo, se optó por descargar en archivos de valores separados por comas (.csv). De acuerdo sus creadores, 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.

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 91. Vista del website Web Robots (visitado en agosto del 2019).

Fuente: Elaboración propia.

A continuación, los archivos descargados que se encontraban fraccionados en varios archivos .csv, donde luego de ser descomprimidos, fueron unidos por mes de captura y almacenados en carpetas independientes por mes, cuyo peso individual osciló entre 1 y 5 gigabytes (GB). Con el fin de ahorrar espacio en la computadora, las partes originales fueron eliminadas.

En la Figura 92 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 habían 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 92. Tamaño de conjunto de datos al corte de Julio 2019.

Fuente: Elaboración propia.

A cada conjunto de datos generado se filtraron los que pertenecen a la categoría **Technology**. Al no contar con la información de la columna *main_category*, este proceso se logró utilizando la variable *source_url* seleccionando aquellos registros que contengan la cadena de caracteres “<https://www.kickstarter.com/discover/categories/technology>”.

Cuando se repitió este procedimiento con cada conjunto generado, se observó que la proporción de proyectos tecnológicos en Kickstarter representa el 10% de la totalidad, aproximadamente más de 21 mil registros por mes. Esto se calculó al comparar el tamaño de cada conjunto generado con el total.

A continuación, se unieron los 45 archivos separados por coma (.csv) capturados mensualmente desde noviembre del 2015 hasta agosto del 2019. Se realizaron 2 uniones internamente ya que, a partir de marzo del 2018, algunas de las variables y valores presentan diferente estructura a la de sus predecesoras.

Luego, 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 calcular 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. Los proyectos cancelados o suspendidos no aparecieron. El último paso del flujo consistió en generar y exportar el archivo final en formato .csv. En la Figura 93 se visualiza el conjunto final de Metainformación subido públicamente a la plataforma Kaggle. Cada variable se detalla en la Tabla 13.

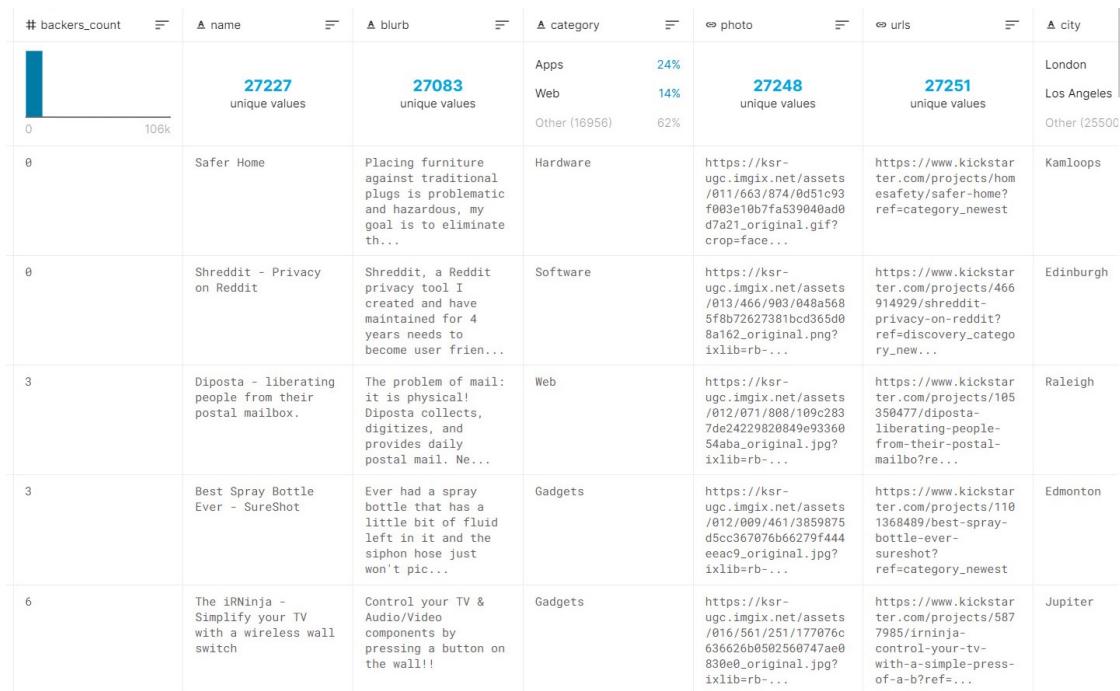


Figura 93. Visualización del archivo de metainformación subido a Kaggle.

Fuente: Elaboración propia.

Tabla 13

Diccionario de datos del dataset final de Metainformación.

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

Fuente: Elaboración propia.

Actividad 2: Construir base de datos de Descripción

La variable *description* se obtuvo utilizando web scraping en cada proyecto gracias a la variable *urls*. Para ello, se elaboró un algoritmo usando la librería BeautifulSoup que, mediante acceso y 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, uniendo previamente todos los párrafos y eliminando caracteres especiales, para posteriormente asignarle la id de su proyecto y guardarla en un archivo de extensión .csv y exportarlo. En caso el algoritmo no encuentre esta clase dentro de las páginas (*IndexError*), el vector almacena con el valor “null”.

```
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)
        # Juntar 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 94. Función para extraer textos de modalidad de descripción.

Fuente: Elaboración propia.

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, como se observa en la Figura 95.

```
# Asignar urls de búsqueda (origen)
urls_primer_parte = urls_list[0:9083]
urls_segunda_parte = urls_list[9084:18168]
urls_tercera_parte = urls_list[18169:27251]
#ids
ids_primer_parte = data_final["id"][0:9083]
ids_segunda_parte = data_final["id"][9084:18168]
ids_tercera_parte = data_final["id"][18169:27251]
description_txt = [getPageText(url) for url in urls_primer_parte]
df = {"id":ids_primer_parte, "description":description_txt}
export_csv = data_final.to_csv (r'D:\TTT\DB\dataset_descripciones\data_final_parte1.csv', index = None, header=True)
# Hacemos lo mismo con la segunda parte
description_txt = [getPageText(url) for url in urls_segunda_parte]
df = {"id":ids_segunda_parte, "description":description_txt}
data_final = pd.DataFrame(df)
export_csv = data_final.to_csv (r'D:\TTT\DB\dataset_descripciones\data_final_parte2.csv', index = None, header=True)
# Hacemos lo mismo con la tercera parte
description_txt = [getPageText(url) for url in urls_tercera_parte]
df = {"id":ids_tercera_parte, "description":description_txt}
data_final = pd.DataFrame(df)
export_csv = data_final.to_csv (r'D:\TTT\DB\dataset_descripciones\data_final_parte3.csv', index = None, header=True)
# Borramos variables para liberar memoria
del description_txt, df, data_final, export_csv
# Me redirijo a la nueva ruta de los archivos generados
os.chdir('D:/TTT/DB/dataset_descripciones')
#Las siguientes líneas son para combinar todos los archivos .csv dentro de la carpeta
extension = '.csv'
all_filenames = [i for i in glob.glob('*.format(extension)))]
#combinar todos los archivos
data_description = pd.concat([pd.read_csv(f) for f in all_filenames ])
#exportar a csv
data_description.to_csv( "data_description.csv", index=False, encoding='utf-8-sig')
```

Figura 95. Ejecución de la función de extracción de descripciones y almacenamiento.

Fuente: Elaboración propia.

El archivo generado fue subido a la plataforma Kaggle de manera pública para que pueda ser descargada a través del API de la web, como se aprecia en la Figura 96.

id	description
	[null] 2%
	Galen Framework is... 0%
	Other (2660) 98%
48.1k	2.15b
1000245024	Purpose Safer Home ensures that electrical plugs will not become a fire hazard. The invention comes ...
1000256230	I once had a Reddit account that was four years old with thousands of comments forever stored in Red...
1000261018	Every day you go home to a mail box filled with junk and even worse, if you are a traveler it is ove...

Figura 96. Visualización del archivo de descripción subido a Kaggle.

Fuente: Elaboración propia.

Actividad 3: Construir base de datos de Comentarios

Al igual que la descripción, los comentarios se obtuvieron utilizando la variable *urls* para web scraping, pero reemplazando caracteres que contengan desde “**?ref=**” en adelante, por “**/comments**” para redireccionarse a la sección de comentarios de cada proyecto. Para lograrlo, se codificó la función de la Figura 97.

```
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 97. Función para extraer textos de modalidad de comentarios.

Fuente: Elaboración propia.

Los comentarios, al ser dinámicos, no podían ser extraídos mediante la librería BeautifulSoup como en el caso de las descripciones, por lo que se utilizó la librería Selenium para extraerlos al iniciar una sesión desde Google Chrome. Una vez permitido el acceso al navegador, el algoritmo se redirecciona a la sección de comentarios del proyecto, espera un máximo de 30 segundos de carga de la página, busca el elemento Xpath “**//*[@id=react-project-comments]/div/button**” para hacer clic en el botón “Load More” (Cargar más) hasta un máxi-

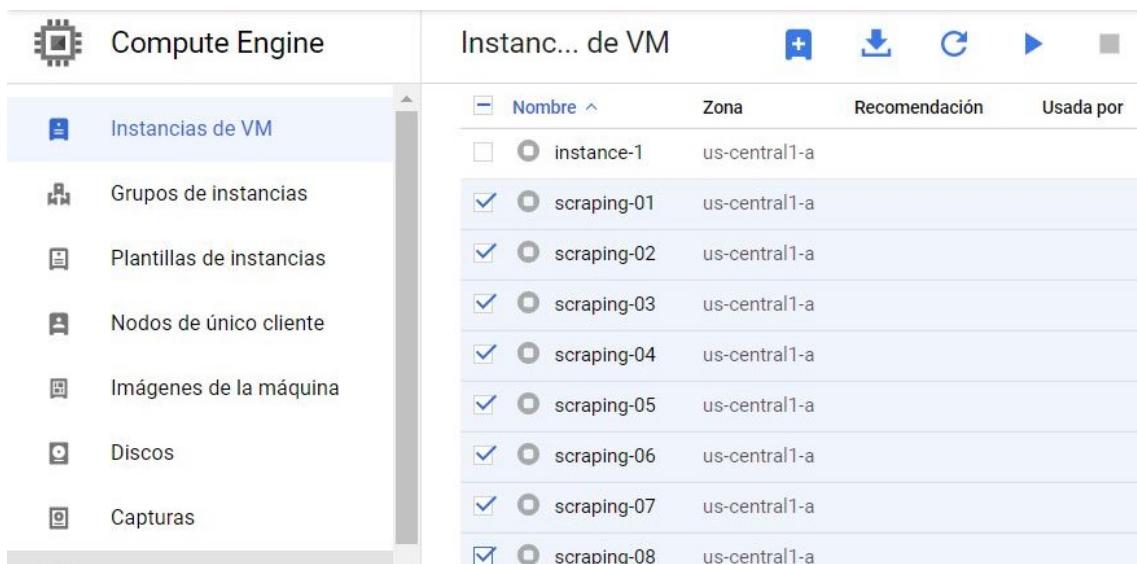
mo de 13 veces esperando 2 segundos entre cada clic, busca todos los comentarios bajo el nombre del elemento CSS “**div.w100p**”, elimina aquellos que pertenezcan al creador del proyecto identificados con etiqueta verde en el extremo superior derecho del recuadro del comentario con el nombre del elemento “**span.bg-ksr-green-700.white.px1.type-14.mr1**” (asignados con el valor de «Creator»), y almacena los restantes que pertenecen a los patrocinadores. En caso no encuentre ningún ítem de comentarios en la sección, se asignará un valor aleatorio no relacionado con proyectos. Luego de eliminar mensajes de la página acerca de comentarios ocultos o eliminados, el algoritmo culmina creando la lista de comentarios separados por autor y cerrando el web driver. Esta función fue ejecutada en 16 partes, asignando el id respectivo a los comentarios extraídos por cada proyecto en un archivo .csv, como se observa en la Figura 98.

```
count_proj=0
for id_proj, url_proj in zip(data_final["id"][:], urls_list[:]):
    comments_txt = getPageText(url_proj)
    df_comments = {"ids": [id_proj], "comments": [comments_txt]}
    data_comments = pd.DataFrame(df_comments)
    data_comments.to_csv(r'data_comentarios.csv', mode = 'a', sep = ',', index = False, header=False)
    count_proj += 1
    del data_comments, df_comments, comments_txt
    print(count_proj)
print('Ya terminó el scraping!')
```

Figura 98. Ejecución de la función de extracción de comentarios y almacenamiento.

Fuente: Elaboración propia.

Para optimizar la descarga, se crearon 8 instancias en Google Cloud (Figura 99).



The screenshot shows the Google Cloud Platform Compute Engine interface. On the left, there is a sidebar with icons for Compute Engine, VM instances, groups, templates, nodes, images, disks, and snapshots. The main area is titled 'Instanc... de VM' and displays a list of 8 instances. The table has columns for Nombre, Zona, Recomendación, and Usada por. All instances are in the 'us-central1-a' zone. The instances are listed as follows:

Nombre	Zona	Recomendación	Usada por
instance-1	us-central1-a		
scraping-01	us-central1-a		
scraping-02	us-central1-a		
scraping-03	us-central1-a		
scraping-04	us-central1-a		
scraping-05	us-central1-a		
scraping-06	us-central1-a		
scraping-07	us-central1-a		
scraping-08	us-central1-a		

Figura 99. Instancias lanzadas en paralelo para la extracción de comentarios.

Fuente: Elaboración propia.

Cada instancia 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. Este se encuentra disponible públicamente en Kaggle y se visualiza en la Figura 100.

id	comments
48.1k	71% [@Creator: What wi... 0% Other (7857) 29%
1001265769	['Waste of money ', 'Battery is Sound Quality ', "What's with the squares?", 'My battery started to...
1001502333	["First to market, last to deliver MVP. Just here to put my monthly DO NOT BUY PIMAX. Nothing but re...
1001565620	[]

Figura 100. Visualización del archivo de comentarios subido a Kaggle.

Fuente: Elaboración propia.

Actividad 4: Realizar análisis exploratorio y estadístico de variables considerados

En esta sección, se analizaron estadísticamente las variables de cada modalidad, tanto las distribuciones de sus datos para la metainformación y contenido textual, así como estadísticos para las variables cuantitativas de la metainformación, entre ellos el rango de sus valores, la media, la mediana, la moda, la desviación estándar y la varianza. En la variable dependiente estado de financiamiento, los 27,251 proyectos se distribuyen mediante el gráfico de pie de la Figura 101. Se observa que casi 20 mil proyectos entre 2009 y 2019 no llegaron a ser financiados, es decir, aproximadamente el 72 % del total fracasaron.

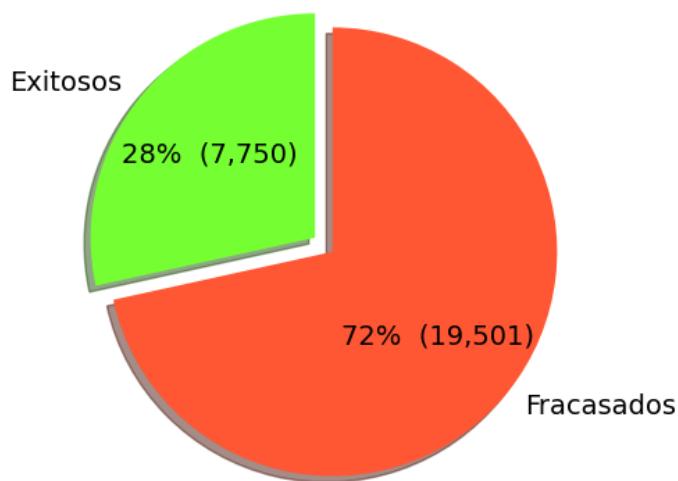


Figura 101. Distribución de proyectos tecnológicos según su estado.

Fuente: Elaboración propia.

De acuerdo a la distribución por año de la Figura 102, 2015 fue el periodo en donde se registraron más campañas de proyectos tecnológicos en la plataforma.

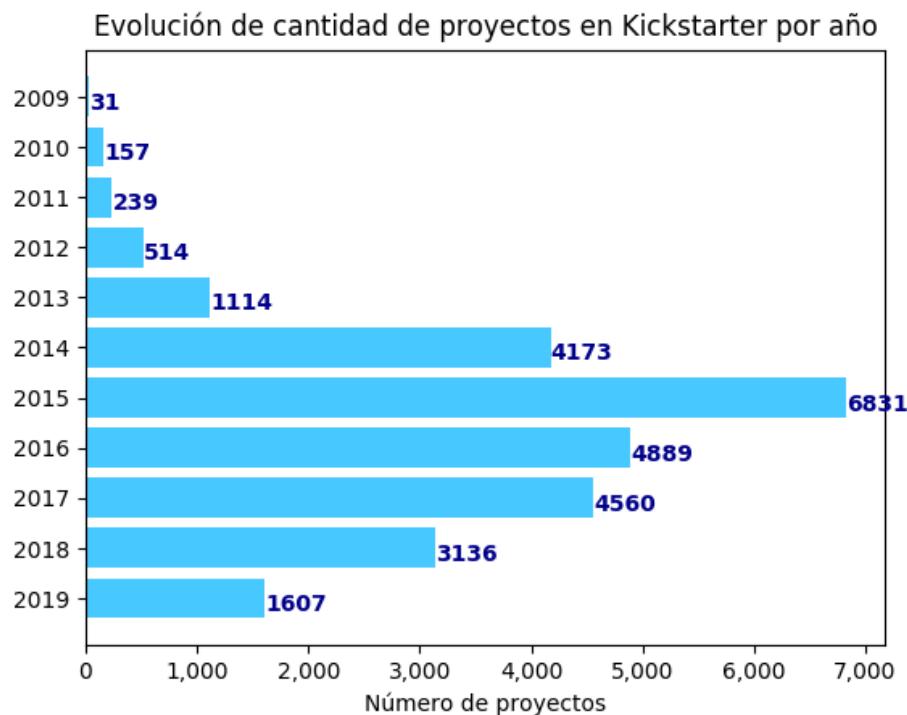


Figura 102. Evolución de cantidad de proyectos tecnológicos por año.

Fuente: Elaboración propia.

El anterior gráfico abierto por estado de financiamiento, como se representa en la Figura 103, muestra que el 2015 resultó ser el año más disparo, donde el 78 % fueron fracasados.

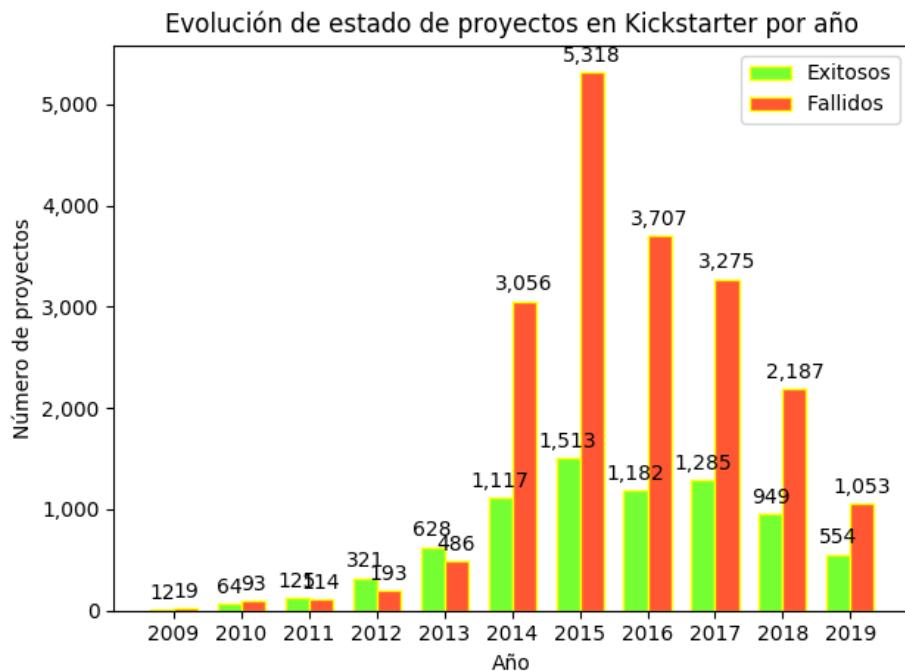


Figura 103. Evolución de proyectos tecnológicos, por su estado y año.

Fuente: Elaboración propia.

Por el lado de Metainformación, de las 19 variables de la Tabla 13, se consideraron como potenciales variables independientes a 3 categóricas, 5 numéricas y 1 lista compuesta por números (*pledge_amounts*). La distribución del primer grupo se ilustra en la Figura 104.

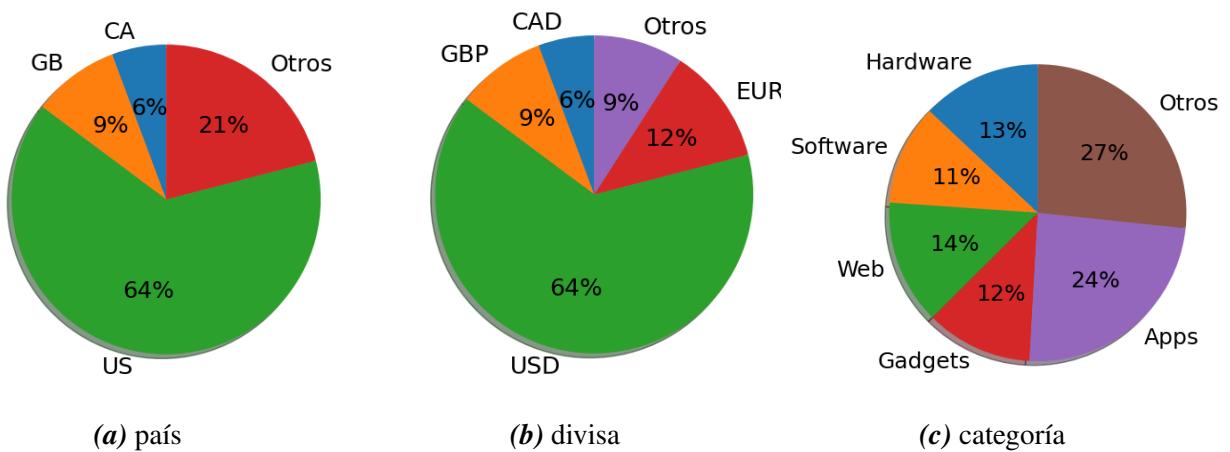


Figura 104. Distribución de las variables categóricas de Metainformación.

Fuente: Elaboración propia.

De las 3 variables categóricas (*country*, *currency* y *category*), se observa que más de la mitad de creadores proyectos provienen de los Estados Unidos (64 %) e invierten en dólares. A ellos los acompañan personas de Gran Bretaña (9 %), que invierten en libras esterlinas, y de Canadá (6 %), que invierten en dólares canadienses. El 21 % restante provienen de otros países, donde el 12 % de ellos invierten en euros. Por el lado de las categorías, las más resaltantes son Apps, Web, Hardware, Software y Gadgets.

Para las potenciales variables numéricas de Metainformación (*backers_count*, *goal*, *pledged*, *usd_pledged* y *duration*), se calcularon sus datos estadísticos de rango de valores, media, mediana, desviación estándar y varianza, con ayuda de diagramas de caja y bigote que se muestran a continuación:

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

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

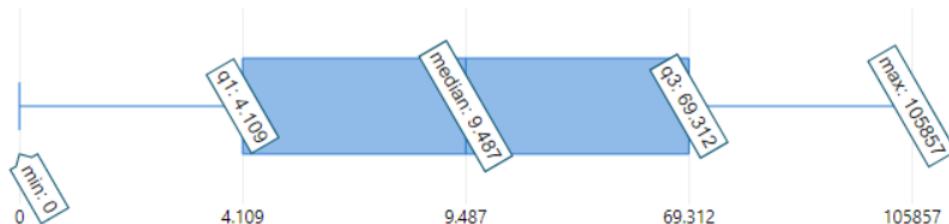


Figura 105. Diagrama de caja y bigote de patrocinadores.

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

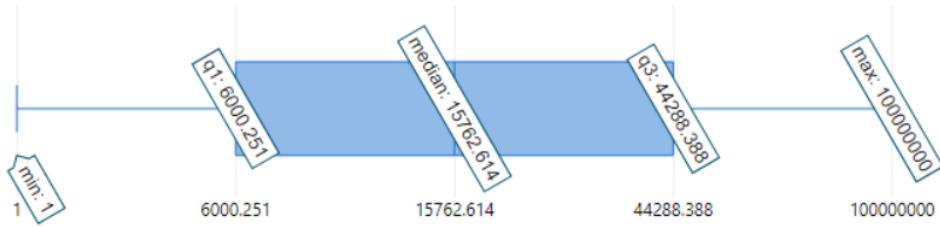


Figura 106. Diagrama de caja y bigote de meta.

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

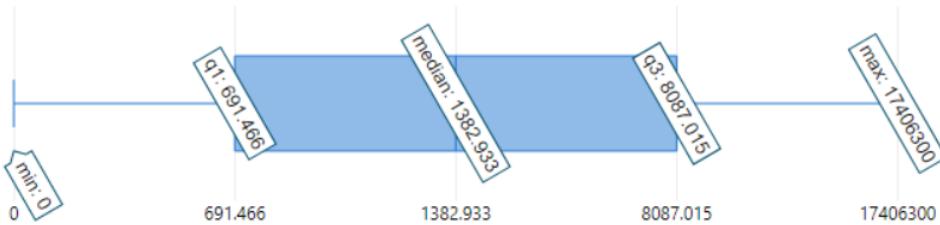


Figura 107. Diagrama de caja y bigote de monto patrocinado.

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

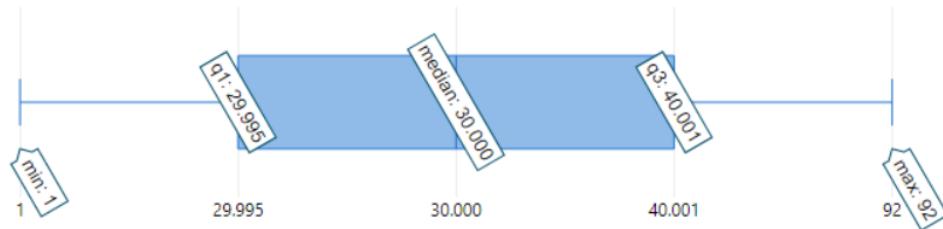


Figura 108. Diagrama de caja y bigote de duración.

Fuente: Elaboración propia.

Posterior a este entendimiento de datos, se elaboró una matriz de correlaciones (Figura 109) para encontrar correlaciones entre ellas y determinar la existencia de alguna variable redundante y descartarla para no afectar el rendimiento del modelo.

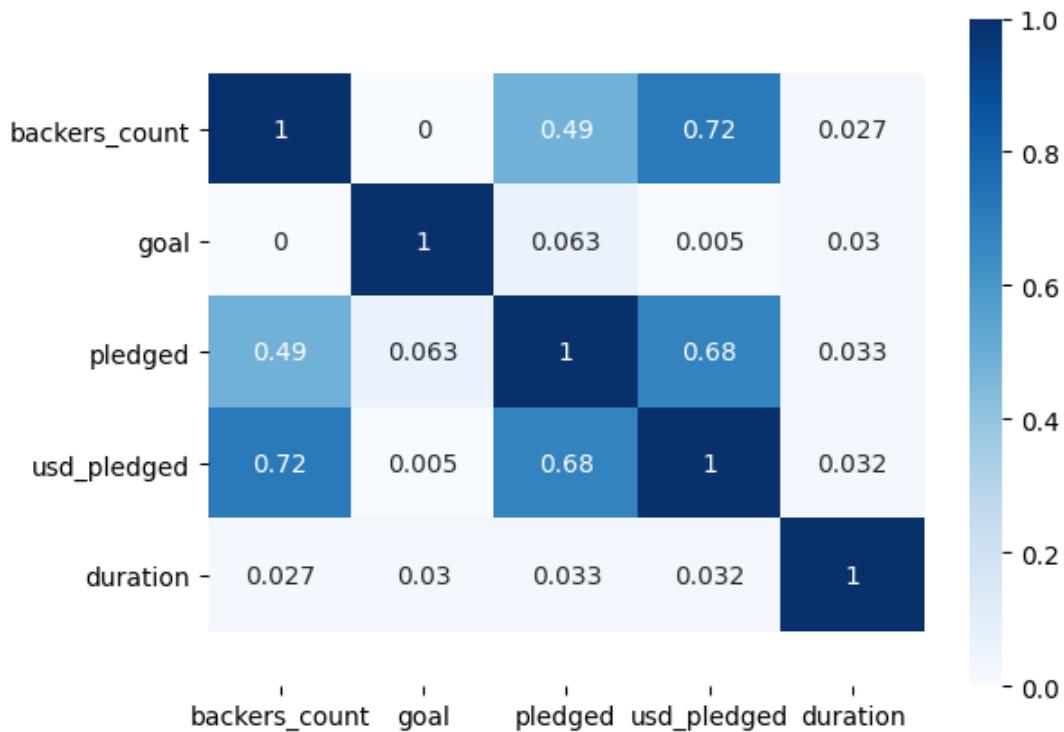


Figura 109. Matriz de correlaciones entre variables independientes.

Fuente: Elaboración propia.

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 110, se confirma y concluye no utilizar las observaciones comentadas.

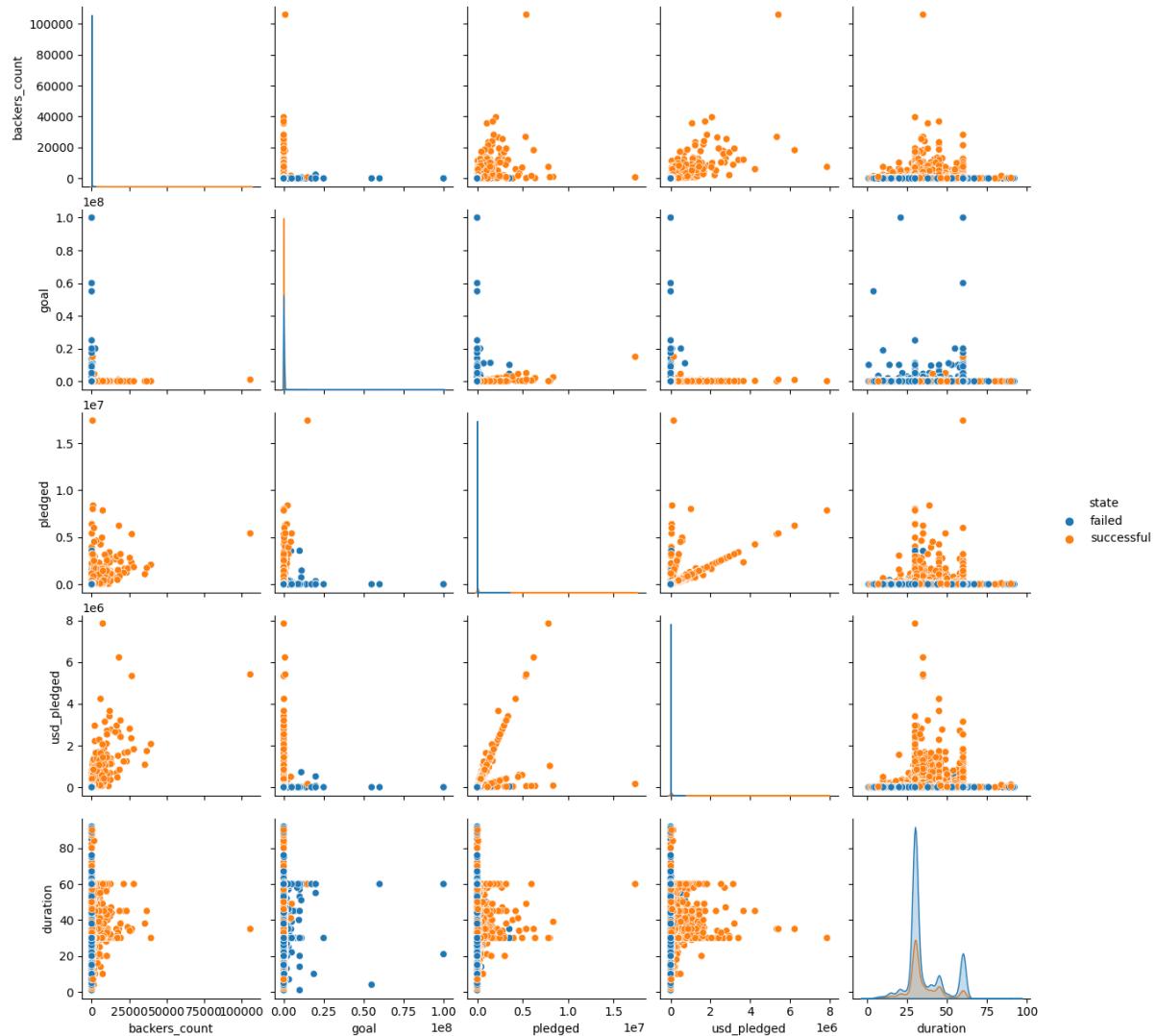


Figura 110. Gráfico de dispersión de correlaciones entre variables independientes.

Fuente: Elaboración propia.

La variable *duration* es la única que sigue una distribución normal debido a la forma de campana de su silueta. Asimismo, los registros de proyectos exitosos y fracasados para las otras 4 variables se encuentran mezcladas en los mismos grupos al cruzarse entre ellas. También se visualizan observaciones en los extremos de cada gráfico, tal y como se detalló en sus valores estadísticos individuales.

Por el lado de Descripción, solo 640 proyectos (2 % del total) no presentaron descripciones por razones externas durante el proceso de extracción de datos. De ellos, 512 (80 % de proyectos sin descripciones) fracasaron en ser financiados. Por el lado de proyectos con descripciones, casi el 30 % fueron exitosos como se grafica en la Figura 111.

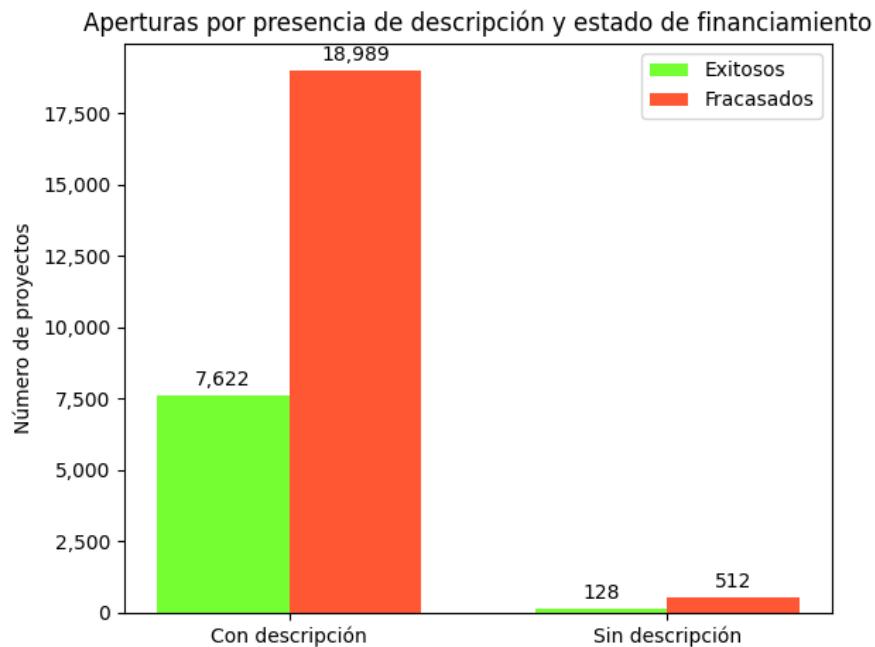


Figura 111. Distribución de proyectos por presencia de descripciones y estado final.

Fuente: Elaboración propia.

Asimismo, el registro con descripción de mayor longitud presentó 5,152 palabras y, a nivel total de proyectos, el vocabulario fue de 165,683 palabras.

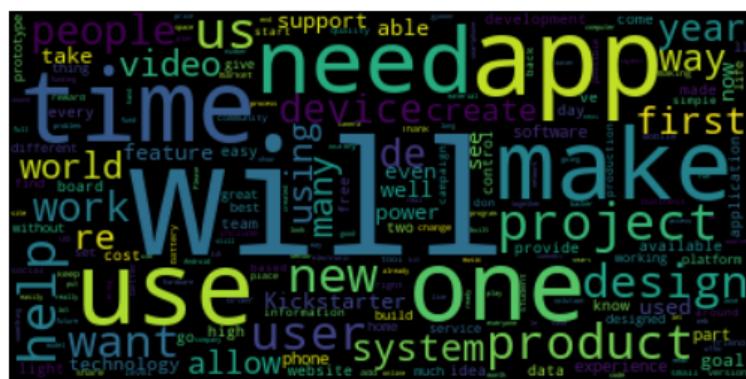


Figura 112. Nube de palabras de descripciones.

Fuente: Elaboración propia.

En la nube de palabras de la anterior Figura 112, los términos más frecuentes en ellas se relacionan con las funcionalidades del producto (*system, app, need, help, product*, etc).

Por el lado de Comentarios, al analizar los proyectos exitosos y fracasados por la presencia de comentarios (Figura 113), se observa que el 60 % de proyectos con comentarios (4,626 registros) fueron exitosos, mientras que el 84 % de los proyectos sin comentarios fracasaron.

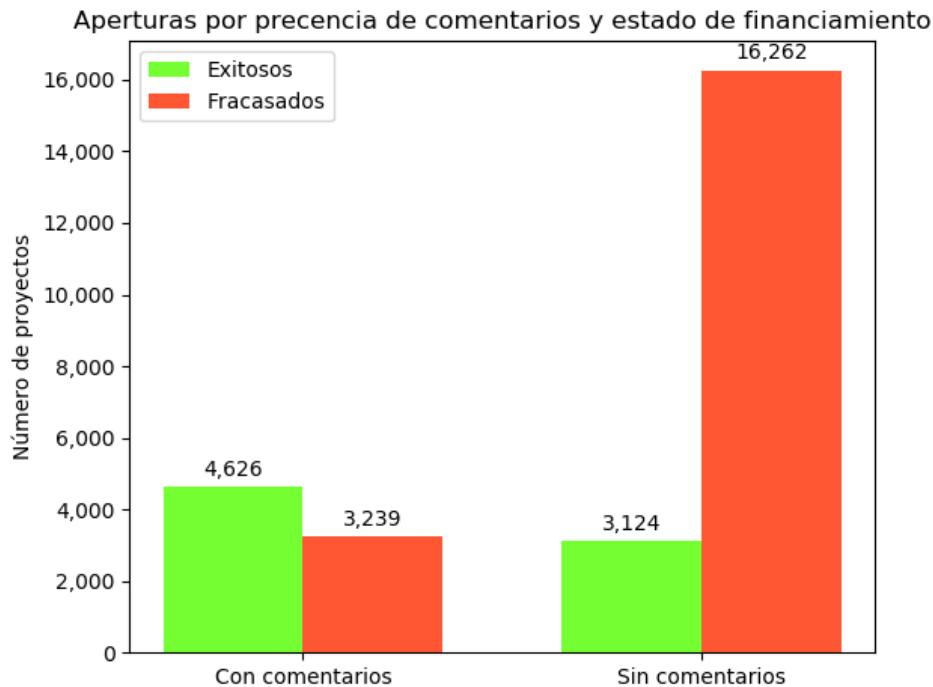


Figura 113. Distribución de proyectos por presencia de comentarios y estado final.

Fuente: Elaboración propia.

Esto señala que es más probable que un proyecto sin recibir comentarios tiende a fracasar por la diferencia notable entre ambas categorías (68 %). Por el contrario, para el caso de aquellos que presentan comentarios, no se puede formular una hipótesis sobre su comportamiento ya que la diferencia de proporciones no es muy alta (20 %) en comparación con el grupo sin comentarios. Por ello, para conocer un poco más a este último grupo, se analizó el impacto de las cantidades de comentarios independientes realizados por patrocinadores exclusivamente en el resultado final de la meta de financiamiento.

De aquellos proyectos con comentarios, el 96% que fueron financiados exitosamente recogieron más de 475 mil comentarios, como se aprecia en la Figura 114.

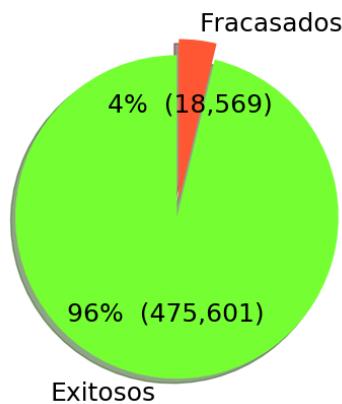


Figura 114. Distribución de comentarios en proyectos exitosos y fracasados.

Fuente: Elaboración propia.

Respecto al contenido, en la Figura 115 se ilustra la nube de palabras más frecuentes, respectivamente, luego de quitar URLs, emoticonos y términos en idioma distinto al inglés.

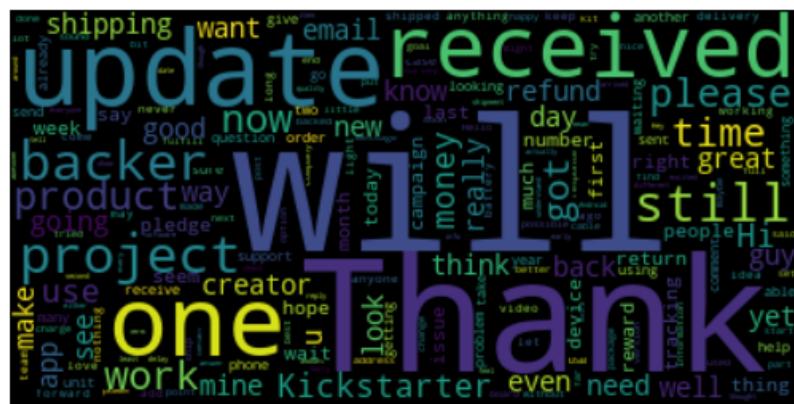


Figura 115. Nube de palabras de comentarios más frecuentes.

Fuente: Elaboración propia.

De esta imagen, se observa que las palabras más frecuentes en los comentarios (tamaño de fuente más grande) se relacionan con términos clave respecto a la campaña (*update, project, backer, product* y *Kickstarter*) y palabras relacionadas a su interacción con el público (*thank, will, received, time, money*). Algunos de estos términos, tanto en su forma raíz como en conjugaciones, suelen aparecer solitarias o acompañados de otros para formar frases recurrentes.

4.3 Preparación de los datos

Actividad 1: Pre-procesar base de datos de Metainformación

De acuerdo a los autores K. Chen et al. (2013), S.-Y. Chen et al. (2015) y Jin et al. (2019), a las 5 potenciales variables numéricas se adicionaron 7 variables basadas en el mecanismo financiero (mediana (*pledges_median*), promedio (*pledges_mean*), valor máximo (*pledges_max*), valor mínimo (*pledges_min*), variación estándar (*pledges_std*) y cantidad de montos disponibles para contribuir (*pledges_num*)) y efecto progresión (porcentaje de financiamiento o completitud (*completeness*) del monto prometido). Esta última se calcula dividiendo el monto alcanzado en el tiempo t, sobre la meta de la campaña, multiplicado por 100%. La nueva matriz de correlación se observa en la Figura 116.

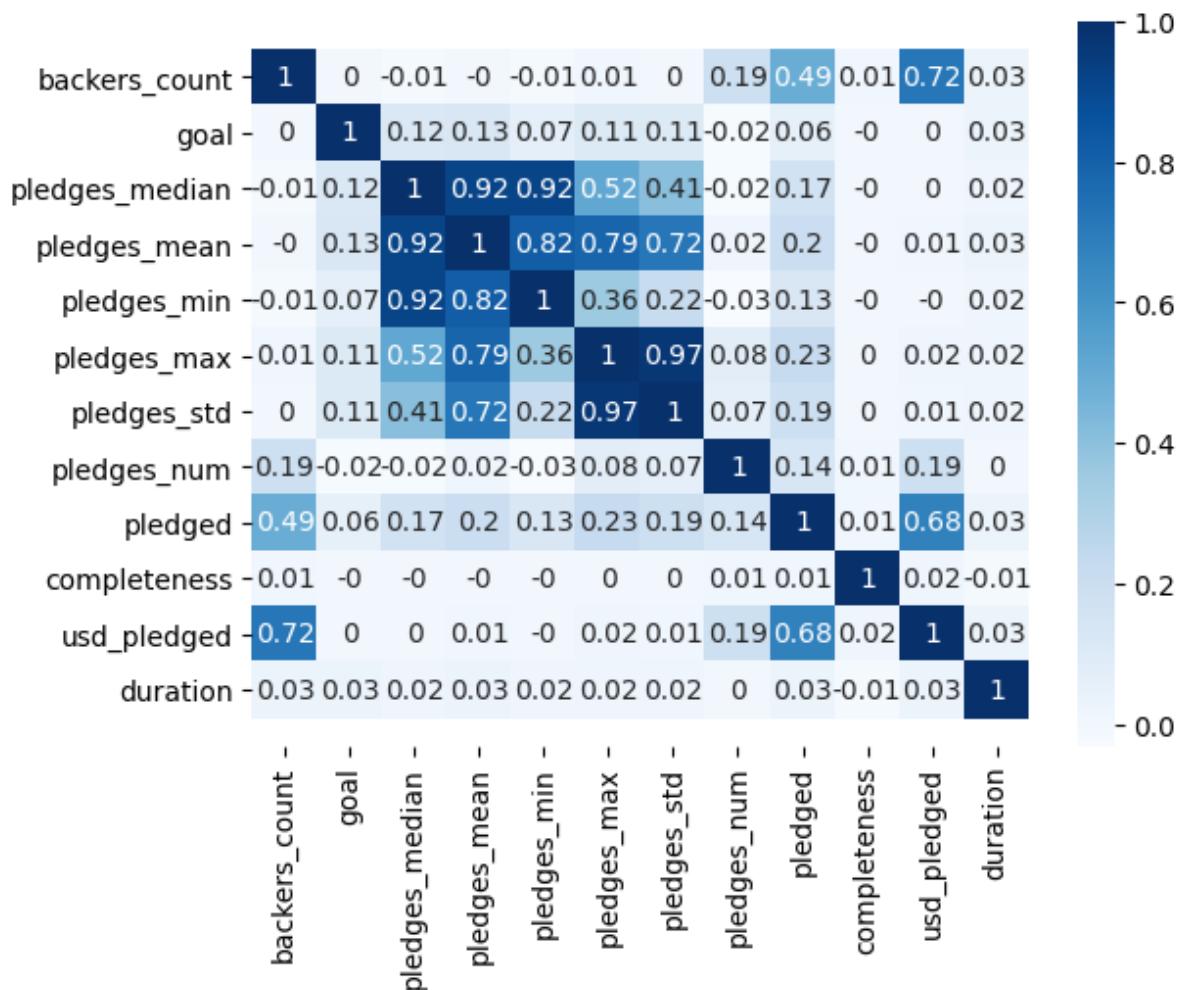


Figura 116. Matriz de correlaciones entre variables independientes considerando adicionales.

Fuente: Elaboración propia.

Para la selección de variables, se consideró a aquellas con correlación clasificada como insignificante, es decir, menor o igual a 0.30 (Mukaka, 2012). Las únicas que cumplen son *goal*, *pledges_num*, *completeness* y *duration*. Sin embargo, algunas de las restantes pueden ser consideradas condicionándose a excluir otras. En la Tabla 14 se listan las 8 combinatorias posibles de variables que se pueden formar.

Tabla 14
Potenciales combinatorias de variables de metainformación.

Combinación 1	Combinación 2	Combinación 3	Combinación 4
goal	goal	goal	goal
completeness	completeness	completeness	completeness
duration	duration	duration	duration
pledges_num	pledges_num	pledges_num	pledges_num
backers_count	backers_count	backers_count	backers_count
pledges_median	pledges_mean	pledges_min	pledges_max
		pledges_std	
Combinación 5	Combinación 6	Combinación 7	Combinación 8
goal	goal	goal	goal
completeness	completeness	completeness	completeness
duration	duration	duration	duration
pledges_num	pledges_num	pledges_num	pledges_num
pledged	pledged	pledged	pledged
pledges_median	pledges_mean	pledges_min	pledges_max
		pledges_std	

Fuente: Elaboración propia.

Una vez generadas las variables independientes (X) y dependiente (Y), el conjunto de datos es separado en subconjuntos de entrenamiento y prueba, con proporciones de 80 % y 20 % respectivamente (L.-S. Chen & Shen, 2019; Mitra & Gilbert, 2014; Sawhney et al., 2016; P.-F. Yu et al., 2018; H. Yuan et al., 2016) y se fija un valor de aleatoriedad. Dentro de los parámetros de separación, se establece el argumento de estratificación según la variable Y, es decir, cada subconjunto mantendrá la distribución 72 % exitosos y 28 % fracasados.

```
train_ratio = 0.80
X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 1 - train_ratio, stratify = Y, random_state=0)
```

Figura 117. Función para dividir base de datos en subconjuntos de entrenamiento y prueba.

Fuente: Elaboración propia.

Luego, utilizando el escalador Mínimo Máximo (*Min-Max scaler*) de la librería Scikit-learn, se normalizaron las variables independientes a un nuevo rango conformando valores entre 0 y 1. La función creada para este proceso se muestra en la Figura 118.

```
# Escalado de variables
from sklearn.preprocessing import StandardScaler # Importan el modulo
from sklearn import preprocessing
sc_X = preprocessing.MinMaxScaler() # Definir la funcion
X_train = sc_X.fit_transform(X_train) # Ajuste
X_test = sc_X.transform(X_test) # Aplicacion
```

Figura 118. Función para normalizar variables.

Fuente: Elaboración propia.

Para calcular los nuevos valores normalizados usando el anterior escalador, se sigue la siguiente fórmula (Ciaburro & Joshi, 2019; Pedregosa et al., 2011; Scikit-learn, s.f.):

$$x_{escalado} = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (28)$$

Donde x representa el valor original de un dato, x_{min} el valor mínimo existente para dicha variable y x_{max} , el valor máximo.

Por ejemplo, tomando como referencia las estadísticas de la variable *duration* en la Figura 108, se desea transformar una duración de 30 días dentro del rango [0; 1]. Para aplicar la fórmula, los valores serían $x = 30$, $x_{min} = 1$ y $x_{max} = 92$. Entonces, el nuevo resultado sería $x_{escalado} = \frac{30-1}{92-1} = 0.32$.

Actividad 2: Pre-procesar base de datos de Descripción

Se realizó la limpieza de texto basándose en los trabajos de los autores Mitra y Gilbert (2014), H. Yuan et al. (2016) y L.-S. Chen y Shen (2019) y además, se agregaron pasos de lematización y supresión de palabras de parada siguiendo el proceso dictado en el curso de Procesamiento de Lenguaje Natural en la Escuela Superior de Economía de la Universidad Nacional de Investigación, Rusia (Zimovnov, 2018). Antes de ejecutarse el proceso, los registros sin descripciones (*NaN*) fueron convertidos en cadena (*string*) para evitar problemas de procesamiento de texto. Se remueven las contracciones, caracteres especiales, enlaces externos y contenidos en otros idiomas. Este resultado fue separado en palabras o tókens para eliminar palabras de parada en inglés, lematizar las restantes y finalmente juntarlas en una lista por su proyecto.

Cada iteración se pudo lograr gracias a elementos de la biblioteca para procesamiento de lenguaje natural Natural Language Toolkit (NLTK), como por ejemplo *word_tokenize*, *stopwords* y *WordNetLemmatizer*. 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 119 representa aquellas palabras que más aparecen en las descripciones.

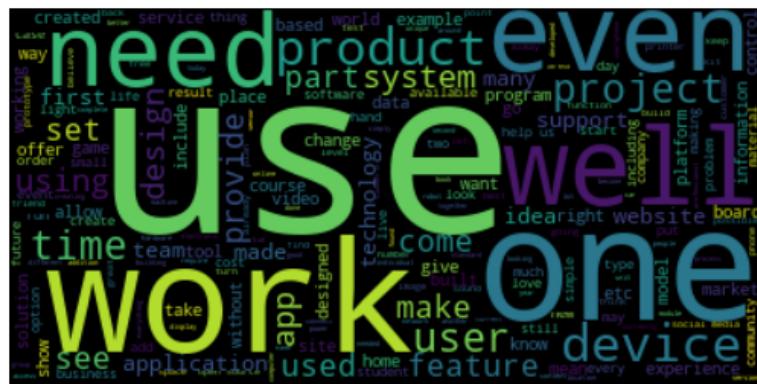


Figura 119. Nube de palabras de descripciones posterior a la limpieza de texto.

Fuente: Elaboración propia.

Luego de la limpieza de textos, la variable independiente *description*, como en el caso de Metainformación, fue dividida en subconjuntos de 80% para entrenamiento y 20% para prueba, estratificados según la distribución de la variable *state*. A continuación, en la Figura 120 se ejecuta el proceso para representar las palabras de las descripciones en vectores.

```
# función para entrenar tokenizador
def create_tokenizer(lines):
    tokenizer = Tokenizer(oov_token=<OOV>")
    tokenizer.fit_on_texts(lines)
    return tokenizer

# tokenizar conjunto de entrenamiento
tokenizer = create_tokenizer(training_sentences)

# función para calcular máxima longitud de un documento (sin palabras de parada)
def max_length(lines):
    return max([len(s.split()) for s in lines])

# calculando máxima longitud de un documento
length_long_sentence = max_length(training_sentences)

# función para codificar lista de líneas
def encode_text(tokenizer, lines, length):
    # codificación
    encoded = tokenizer.texts_to_sequences(lines)
    # llenar secuencias codificadas
    padded = pad_sequences(encoded, maxlen=length, padding='post')
    return padded

# codificar datos
training_sentences = encode_text(tokenizer, training_sentences, length_long_sentence)
testing_sentences = encode_text(tokenizer, testing_sentences, length_long_sentence)
```

Figura 120. Proceso de representación de palabras en vectores codificados.

Fuente: Elaboración propia.

De acuerdo al algoritmo de la figura anterior, se usó la función *Tokenizer* de la librería **tensorflow.keras.preprocessing.text**, para separar las palabras únicas o *tokens* de una línea de texto asignada. En caso se encuentre un término no identificado dentro del diccionario a entrenar, se asignará a dicho token el valor de *<OOV>*. Esta función se aplicó al subconjunto de entrenamiento para elaborar un diccionario a partir de sus tokens. Luego, se creó una función para determinar la mayor longitud de palabras de las descripciones del dataset. Esta cantidad representó 3,671 términos. A continuación, se desarrolló una función para crear una secuencia de las palabras codificadas, homologar hasta la máxima longitud de descripciones y llenar con ceros a la derecha (parámetro *padding='post'*) en caso un vector no alcance esta longitud. Se aplicó este proceso a los subconjuntos originales de entrenamiento y prueba.

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 como se observa en la Figura 121. Para la presente investigación, se seleccionó la opción Wikipedia 2014 + Gigaword 5, con una matriz de 100 columnas, donde cada una contendrá las incrustaciones de palabras GloVe para las palabras del corpus, cuyos índices se corresponderán con cada número de fila (Malik, 2019).

```
from numpy import array, asarray, zeros
# abrir archivo GloVe y crear diccionario
glove_file = open('Glove/glove.6B.100d.txt', encoding="utf8")
embeddings_dictionary = dict()
# completar diccionario con palabras entrenadas de GloVe
for line in glove_file:
    records = line.split()
    word = records[0]
    vector_dimensions = asarray(records[1:], dtype='float32')
    embeddings_dictionary [word] = vector_dimensions
# cerrar archivo GloVe
glove_file.close()
# crear matriz de incrustación de palabras
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 121. Proceso de creación de matriz de incrustaciones de palabras.

Fuente: Elaboración propia.

En la figura anterior, luego de abrirse el archivo GloVe, se completa un diccionario creado con los registros extraídos del algoritmo GloVe. Luego de terminarse este proceso y cerrarse el archivo, se crea la matriz de incrustación de palabras que será utilizada más adelante en la capa de incrustación del modelo predictivo.

Actividad 3: Pre-procesar base de datos de Comentarios

La base de datos de comentarios está conformada a nivel de 1 proyecto con una lista de comentarios separados en sublistas. De los 7,750 proyectos con comentarios (4,626 exitosos), se removieron aquellos que presentaron términos en idioma distinto al inglés, URLs, emoticonos, emojis, números y caracteres especiales, quedando 7,658 registros (4,574 exitosos).

Debido a la gran cantidad de registros carecientes de comentarios, se propuso rellenarlos con un término aleatorio no relacionado con las temáticas principales: *kuwagatabaizan*. Posteriormente, se repitió el mismo ejercicio para las descripciones. Sin considerar este nuevo término, la nube de palabras se representa en la Figura 122.



Figura 122. Nube de palabras de comentarios posterior a la limpieza de texto.

Fuente: Elaboración propia.

Al igual que el caso de Descripción, se siguieron los procesos de la Figura 120 para obtener la representación de palabras en vectores codificados y la Figura 121 para crear la matriz de incrustación de palabras que se usarán en la capa de incrustaciones, con la diferencia en que el tamaño de la secuencia de palabras codificadas no será la máxima longitud de comentarios ya que estos, inicialmente separados por autor, al ser concatenados en un solo registro por proyecto, ampliaron su longitud exponencialmente. La máxima longitud de todos los registros fue de 30,072 palabras; por ello, se limitó el tamaño de secuencias a 5,000 términos.

Capítulo V: Análisis y Discusión de Resultados

En este capítulo, se continuaron las 3 últimas fases de la metodología seleccionada.

5.1 Modelamiento

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 durante el entrenamiento.

Por último, es importante asignar un peso distinto para cada una de las dos clases de la variable dependiente *state*. 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.

Actividad 1: Desarrollar modelo predictivo de Metainformación

Se diseñó el modelo de descripciones basada en un Perceptrón Multicapa (MLP por sus siglas en inglés) bajo la arquitectura de la Figura 123 y teniendo como referencia a los autores P.-F. Yu et al. Se asignaron 100 épocas y el número de lotes fue 32.

La arquitectura comienza con la capa de entrada alimentadas por las 6 variables consideradas, que representan 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 según Ranjan (2019).

De acuerdo a una de ellas, 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.

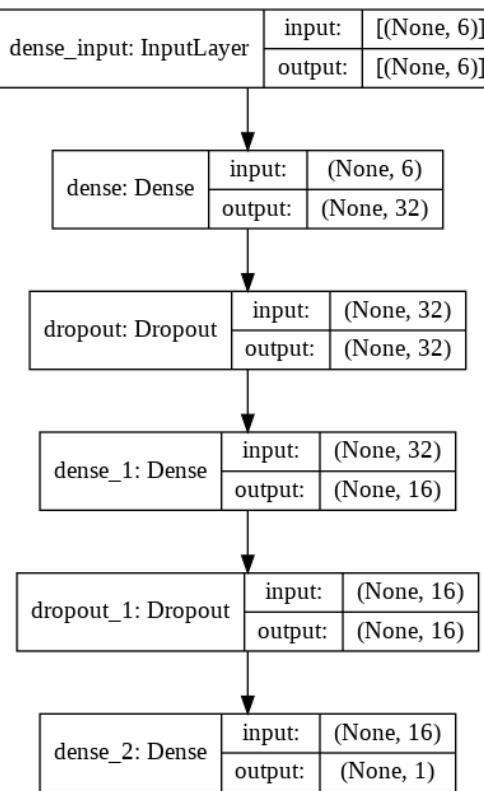


Figura 123. Arquitectura de modelo MLP para la metadata.

Fuente: Elaboración propia.

El autor 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.

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. El criterio para elegir esta función se explica en el modelo de descripción, en el cual también fue aplicado. Por último, además de *adam*, se realizó 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 0.005 debido a que el modelo tenía a

aprender muy rápido durante el transcurso de las épocas. Al ratio de decaimiento se le asignó, entonces, el valor de 0.00005 y para evaluar el modelo se usó la exactitud.

El resumen de la explicación anterior, desde la configuración de parámetros para entrenar hasta los elementos presentes en cada capa, se encuentra en el Anexo F.

Actividad 2: Desarrollar modelo predictivo de Descripción

Se diseñó el modelo de descripciones basada en una Red Neuronal Convolucional unidimensional (Conv1D) bajo la arquitectura de la Figura 124, así como de referencia un trabajo de análisis de sentimientos de películas (Malik, 2019). Se asignaron 100 épocas para entrenar y número de lotes de 128.

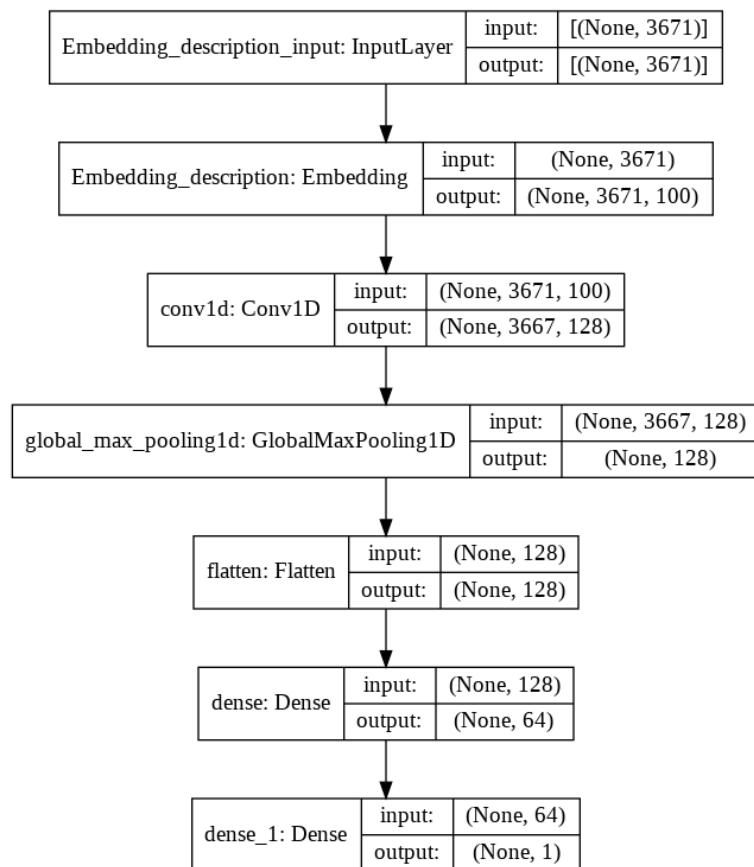


Figura 124. Arquitectura de modelo CNN para las descripciones.

Fuente: Elaboración propia.

Esta red se compone de una capa de incrustación de palabras o *Embedding* alimentada por los datos de entrada en la primera capa *InputLayer* de dimensión de 3,671 vectores de palabras (la mayor longitud de palabras de todas las descripciones), la cual genera como salida una matriz de 3,671 por 100 (número de columnas de incrustaciones de GloVe). Para esta capa se entrenarán 14,827,000 parámetros como resultado del producto de las 100 columnas mencionadas y 148,270 como el tamaño del vocabulario entrenado.

La siguiente capa es la Convolución en 1 dimensión o *Conv1D* (usado frecuentemente para extraer características de datos de textos por ser unidimensionales) que, con 128 características, 5 de tamaño de kernel y función de activación *relu*, generó una salida de 3,667 por 128; así como 64,128 parámetros entrenables.

A continuación, le sigue la capa de reducción *GlobalMaxPooling*. Al igual que en la convolución, esta también fue unidimensional y se caracteriza por realizar agrupamiento global basado en el valor máximo de los bloques seleccionados para reducir el tamaño del vector generado.

Esta nueva salida pasa por la capa de aplanamiento o *Flatten*, en donde se multiplican las filas y columnas y tener un solo vector. Esta sirve para conectar con las 64 neuronas de la nueva capa densa y función de activación *tanh*, agregada con el fin de mejorar la performance del modelo. Según Brownlee (2019), se puede considerar el uso tanto de una función *relu* como una función *tanh* cuando se presente la desaparición de gradientes al propagar hacia atrás a mayor cantidad de capas, hecho presentado en los experimentos. Si bien menciona que el uso de la función tangente hiperbólica en capas ocultas resultó una buena práctica durante las décadas de 1990 y 2000, teniendo mejor rendimiento que la función logística, afirma que ambas son dos opciones válidas para problemas de redes neuronales profundas. Por lo tanto, el criterio para considerar una función *tanh* en la investigación obedece a un mejor desempeño en 2 % más de exactitud en el entrenamiento que utilizando la función *relu*.

Finalmente, la arquitectura culmina con la última capa con función de activación *sigmoid* para regular el valor de salida entre 0 y 1, ya que, al tratarse de un problema de clasificación binaria (predecir si un proyecto será financiado: exitoso, de lo contrario: fracasado), cuenta con solo 1 neurona y su parámetro de pérdida es *binary_crossentropy*. El resumen de todo lo anterior explicado se encuentra en el Anexo G.

Actividad 3: Desarrollar modelo predictivo de Comentarios

Al igual que en el modelo de descripciones de proyectos, se creó un diccionario de palabras con la data luego de tokenizar y codificarlas con las mismas funciones y librerías. Sin embargo, a diferencia del anterior modelo, la longitud de la matriz para el relleno de ceros con el fin de homogenizar el tamaño de cada vector de incrustaciones se limitó a las 5,000 últimas palabras de una oración (parámetro **padding='post'** de la función *pad_sequences*) en lugar de la longitud máxima dado que ésta representa una cantidad considerable (30,072) como para utilizar todos los recursos del entorno de ejecución.

De igual manera, se creó una matriz de incrustaciones de palabras usando GloVe y se diseñó una arquitectura basada en una Red Neuronal Recurrente (RNN por sus siglas en inglés) ilustrada en la Figura 125.

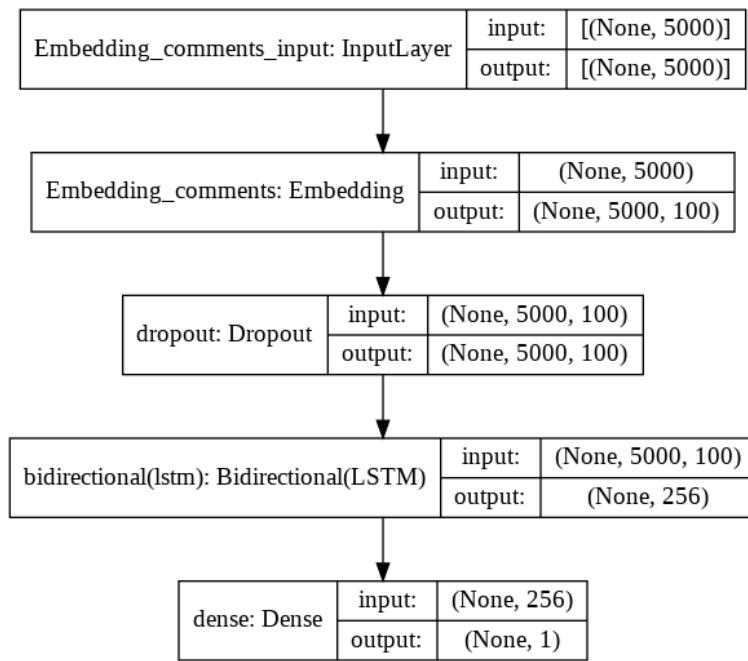


Figura 125. Arquitectura de modelo RNN para los comentarios.

Fuente: Elaboración propia.

Existe una similitud de estructura en las 2 primeras capas con las del modelo de descripción. Luego de la capa de incrustaciones o *Embedding*, para reducir las conexiones entre neuronas se añadió 1 capa de desactivación o *Dropout*.

A continuación, los vectores de palabras ingresan a la capa de la red neuronal recurrente *LSTM*. Para este caso, se consideró envolverla dentro de una Red Bidireccional de 128 neuronas, ya que como se explicó en el Marco Teórico del Capítulo II sobre las RNN Bidireccionales, este tipo es una mejora de la LSTM tradicional al entrenar 2 juntas (la segunda representa una copia invertida de la secuencia de entrada) en donde cada capa ahora puede considerar también información de las capas siguientes junto con la información de las previas que ya tenía en cuenta, es decir, toma información de 2 direcciones (Brownlee, 2017b).

Finalmente, el modelo culmina con una capa densa en donde recibe 256 valores de entrada (2×128 de la capa Bidireccional LSTM) y utiliza la función de activación *sigmoid* para transformar el valor final entre 0 y 1. El resumen de lo anterior se encuentra en el Anexo H.

Antes de continuar con el desarrollo del modelo de Aprendizaje Profundo Multimodal, en la Tabla 15 se presentan las variables de las modalidades que se usaron para entrenarlo. Estas se seleccionaron de acuerdo al Benchmarking aplicado a los antecedentes en el Capítulo II.

Los autores citados por cada variable utilizada se mencionan a continuación:

Tabla 15
Diccionario de datos del conjunto final entrenado.

Variable	Detalle	Tipo de dato
Variables independientes		
goal	Monto de la meta de financiamiento del proyecto.	float64
completeness	Porcentaje de financiamiento o completitud.	float64
duration	Duración de la campaña (en días).	int64
pledges_num	Cantidad de montos disponibles para contribuir.	int64
pledged	Monto contribuido en la campaña.	float64
pledges_median	Mediana de montos disponibles para contribuir.	float64
description	Descripción del proyecto.	object
comments	Comentarios de patrocinadores sobre el proyecto.	object
Variable dependiente		
state	Estado de financiamiento del proyecto.	object

Fuente: Elaboración propia.

- **goal:** K. Chen et al. (2013), Mitra y Gilbert (2014), M. Zhou et al. (2015), S.-Y. Chen et al. (2015), Y. Li et al. (2016), H. Yuan et al. (2016), Sawhney et al. (2016), Kaur y Gera (2017), R. S. Kamath y Kamat (2018), P.-F. Yu et al. (2018), Jin et al. (2019), Cheng et al. (2019).
- **completeness:** S.-Y. Chen et al. (2015).
- **duration:** Mitra y Gilbert (2014), M. Zhou et al. (2015), Y. Li et al. (2016), Sawhney et al. (2016), Kaur y Gera (2017), R. S. Kamath y Kamat (2018), P.-F. Yu et al. (2018), Jin et al. (2019).
- **pledges_num:** K. Chen et al. (2013), Mitra y Gilbert (2014), S.-Y. Chen et al. (2015), H. Yuan et al. (2016), Jin et al. (2019).
- **pledged:** K. Chen et al. (2013), Y. Li et al. (2016), R. S. Kamath y Kamat (2018).
- **pledges_median:** S.-Y. Chen et al. (2015)*, Jin et al. (2019)*.
- **description:** Mitra y Gilbert (2014), M. Zhou et al. (2015), H. Yuan et al. (2016), Sawhney et al. (2016), R. S. Kamath y Kamat (2018), S. Lee et al. (2018), Jin et al. (2019), Cheng et al. (2019), L.-S. Chen y Shen (2019), Chaichi y Anderson (2019).
- **comments:** Y. Li et al. (2016), Kaur y Gera (2017), S. Lee et al. (2018), Jin et al. (2019).

Si bien en los respectivos antecedentes marcados en (*) figuran el promedio de los montos disponibles para patrocinar, se usó la mediana en vez de la media ya que presentó mejor performance en los experimentos.

Actividad 4: Desarrollar modelo ensamblado apilado

Una vez construidos los modelos para cada modalidad (metainformación, descripción y comentarios), se construyó un modelo de Aprendizaje Profundo Multimodal ilustrado en la Figura 126.

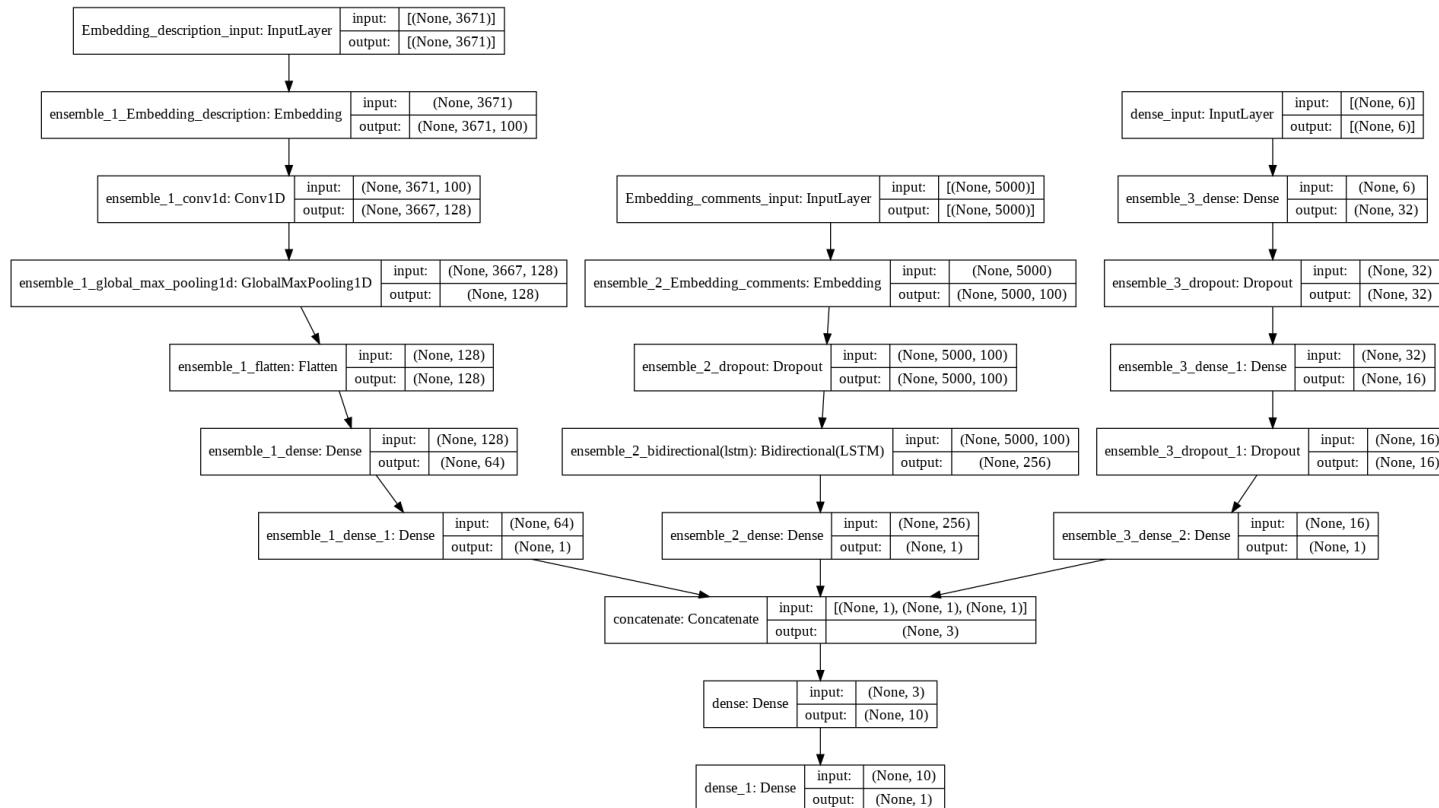


Figura 126. Arquitectura del modelo apilado final The Hydra.

Fuente: Elaboración propia.

La finalidad de este modelo apilado de múltiples cabezas es aprender la mejor manera de combinar las predicciones de cada submodelo para lograr un mayor rendimiento y clasificar mejor la variable dependiente que cada modalidad.

A este modelo se le denominó “*The Hydra*” (La Hidra por su traducción al español) en referencia al monstruo mitológico del lago de Lerna, con 7 cabezas que renacían a medida que se cortaban (Real Academia Española, s.f.).

Al tratarse de un modelo ensamblado apilado, las salidas de cada modelo se concatenaron en una capa debajo de estos, generando 3 valores de entrada para una penúltima capa densa con 10 neuronas de salida y una función de activación *relu*. El modelo apilado culmina con una capa densa de 1 salida y asignándose la función Sigmoide para generar probabilidades entre 0 y 1, los valores de Fracasado o Exitoso respectivamente.

Previo a la compilación del modelo final, se repitió el ejercicio de cada modelo cargado asignar los parámetro de pérdida *binary_crossentropy* para la clasificación binaria, *accuracy* (exactitud) para la métrica del entrenamiento, pesos balanceados para las clases de la variable *state* (0.6987077585764833 para 0 y 1.7581290322580645 para 1), y optimizador *Adam* con la variante de asignarle el ratio de aprendizaje y también de decaimiento de 0.00005. El resumen de todo los parámetros anteriores se encuentra en el Anexo I.

5.2 Evaluación

Como parte de la aplicación de la metodología CRISP-DM, explicada en el sexto subcapítulo del Capítulo III, se mencionaron las métricas usadas en la literatura. La más recurrente fue la exactitud. Dado que la librería Scikit-learn cuenta con un reporte de clasificación con esta métrica y otras 4 más como la precisión, sensibilidad, puntaje F1 y AUC, además que la distribución de proyectos por su estado de financiamiento es desbalanceada y se necesita más de un indicador para poder evaluar y comparar, se decidió usar estas 5 teniendo como referencias a los autores Beckwith (quinto antecedente), H. Yuan et al.* (séptimo antecedente), Kaur y Gera (noveno antecedente), Cheng et al. (decimocuarto antecedente), y L.-S. Chen y Shen** (decimoquinto antecedente).

En el antecedente marcado en (*), los modelos no fueron evaluados por AUC; mientras que en (**), las métricas precisión y AUC no fueron tomadas en cuenta.

Actividad 1: Evaluar desempeño del modelo predictivo de Metainformación

Las 8 combinaciones de variables de la Tabla 14 para el submodelo de Metainformación, luego de ser pre-procesadas utilizando el escalador Min-Max, fueron entrenadas durante 100 épocas cada una. En la Tabla 16 se presentan los resultados obtenidos.

Tabla 16

Exactitud de los conjuntos de datos de validación para las 8 combinaciones.

Combinación 1	Combinación 2	Combinación 3	Combinación 4
0.88405	0.89855	0.89763	0.88369
Combinación 5	Combinación 6	Combinación 7	Combinación 8
0.93102	0.92588	0.92478	0.92478

Fuente: Elaboración propia.

El mejor rendimiento lo produjo la combinación 5 (compuesta por las variables *goal*, *completeness*, *duration*, *pledges_num*, *pledged* y *pledges_median*), la cual alcanzó un valor de exactitud de 0.93102 luego de 19 épocas, con un promedio de 3 segundos de entrenamiento cada una. Al acabar este proceso, el modelo dejó de entrenar dado que durante 7 épocas no registró una reducción en el valor de la pérdida del subconjunto de validación, a pesar de que hace 3 épocas se redujo su tasa de aprendizaje.

Se concluye que la ventaja del grupo de las 4 últimas combinatorias frente a las primeras se da por la inclusión de la variable del monto recaudado al final de la campaña (*pledged*) en lugar del número de patrocinadores alcanzados (*backer_count*). Además, la combinación 5 supera a las siguientes 3 por considerar la mediana de los montos disponibles para contribuir en la campaña, frente a la media, variación estándar, valor máximo y valor mínimo de estos montos.

Así, de acuerdo a la Figura 127, en la época 11 se registran los mejores valores de exactitud y pérdida para el subconjunto de validación, alcanzando 0.9523 y 0.1246 respectivamente.

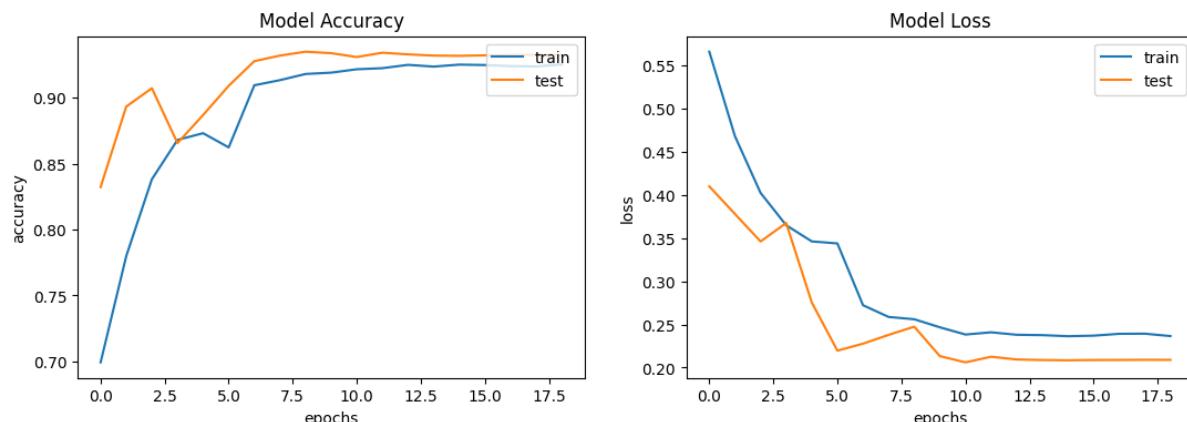


Figura 127. Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo MLP de metadata con 100 épocas.

Fuente: Elaboración propia.

La matriz de confusión resultante se representa en la Figura 128.

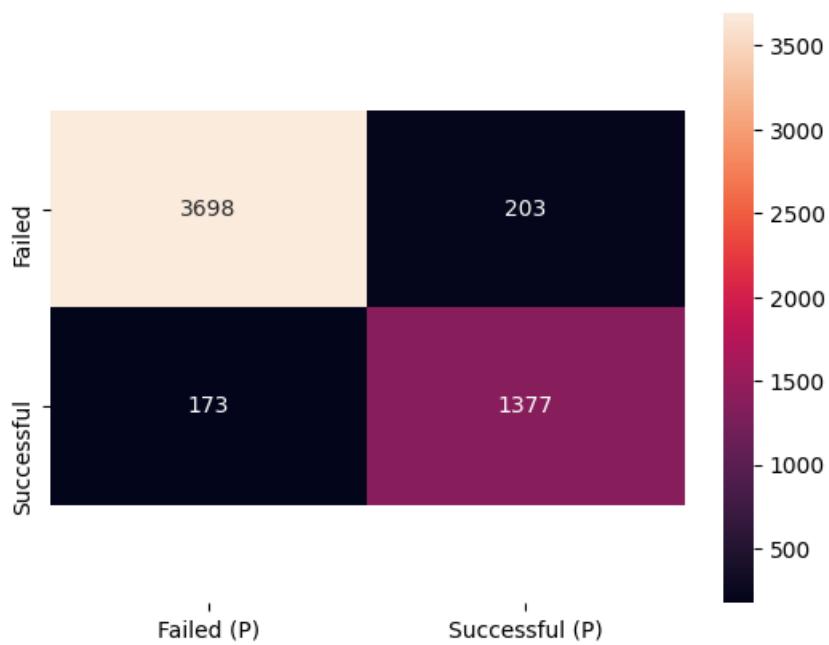


Figura 128. Matriz de confusión para el modelo de metadata.

Fuente: Elaboración propia.

De esta matriz, se derivan los resultados de la Tabla 17 y el AUC en la Figura 129.

Tabla 17
Informe de clasificación para el modelo de metadata.

Valor	Precisión	Sensibilidad	Puntaje F1	Muestras
Fracasado	0.96	0.95	0.95	3,901
Exitoso	0.87	0.89	0.88	1,550
Exactitud			0.93	5,451
Promedio macro	0.91	0.92	0.92	5,451
Promedio ponderado	0.93	0.93	0.93	5,451

Fuente: Elaboración propia.

- El ratio de exactitud se interpreta como: El 93 % de los proyectos de la muestra fueron predichos correctamente.
- El ratio de precisión para los positivos (*Successful*) se interpreta como: El 87 % de los proyectos exitosos predichos de la muestra fueron clasificados correctamente.
- El ratio de sensibilidad para los positivos (*Successful*) se interpreta como: El 89 % de los proyectos exitosos reales de la muestra fueron clasificados correctamente.
- El ratio de Puntaje F1 para los positivos (*Successful*) representa un balance entre las 2 métricas anteriores. En la tabla se observa que su valor es de 88 %, lo cual indica que en general, el modelo mantiene un alto rendimiento.

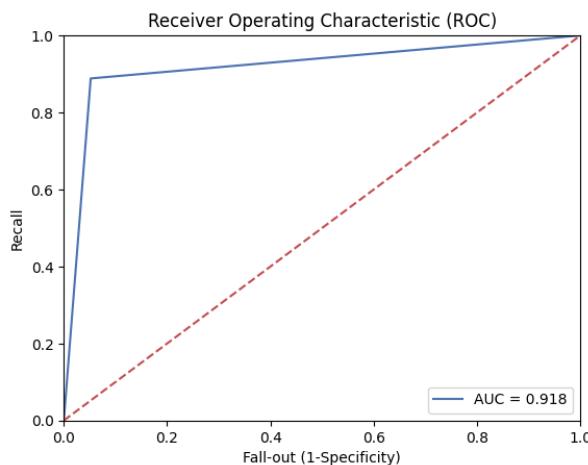


Figura 129. Área bajo la curva ROC de modelo de metadata.

Fuente: Elaboración propia.

El área bajo la Curva ROC presenta un valor de aproximadamente 92 %, del cual se observa en el gráfico que su sensibilidad es muy alta y el ratio de Falsa Alarma es casi nulo. De acuerdo con Britos et al. (2006), el poder discriminante del modelo es excelente.

Actividad 2: Evaluar desempeño del modelo predictivo de Descripción

Luego de 82 épocas, con un promedio de 106 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, a pesar de que hace 5 épocas se redujo su tasa de aprendizaje.

Así, de acuerdo a la Figura 130, en la época 72 se registran los mejores valores de exactitud y pérdida para el subconjunto de validación, alcanzando 0.7683 y 0.4901 respectivamente.

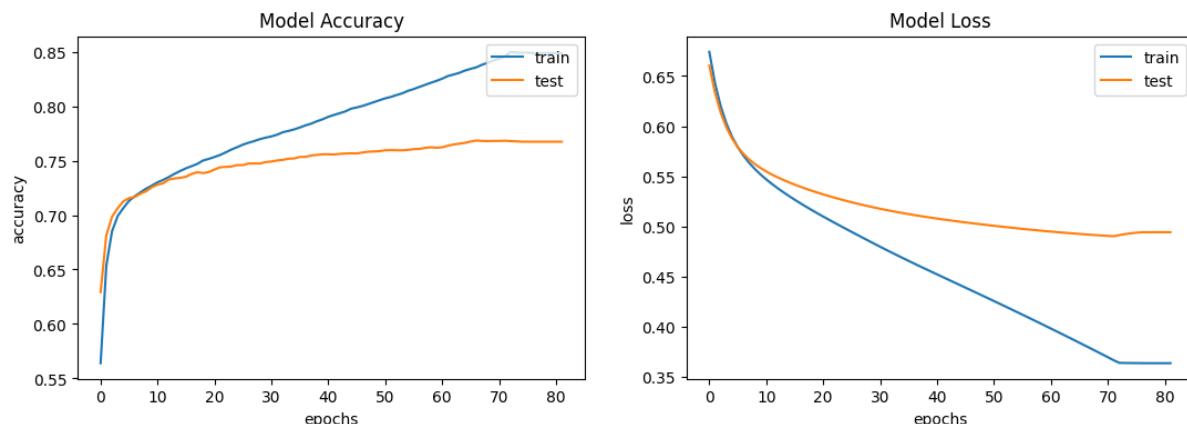


Figura 130. Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo CNN de descripciones con 100 épocas.

Fuente: Elaboración propia.

La matriz de confusión resultante se representa en la Figura 131.

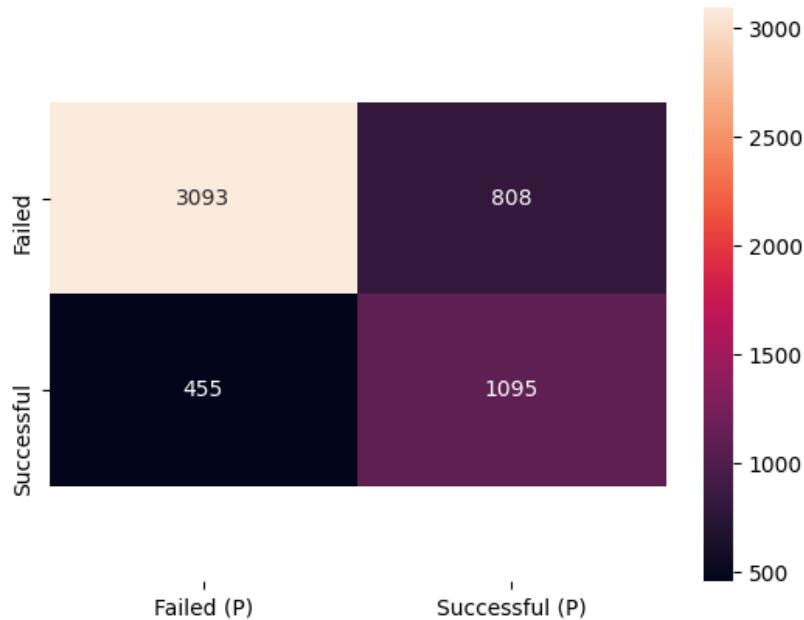


Figura 131. Matriz de confusión para el modelo de descripciones.

Fuente: Elaboración propia.

De esta matriz, se derivan los resultados de la Tabla 18 y el AUC en la Figura 132.

Tabla 18
Informe de clasificación para el modelo de descripciones.

Valor	Precisión	Sensibilidad	Puntaje F1	Muestras
Fracasado	0.87	0.79	0.83	3,901
Exitoso	0.58	0.71	0.63	1,550
Exactitud			0.77	5,451
Promedio macro	0.72	0.75	0.73	5,451
Promedio ponderado	0.79	0.77	0.77	5,451

Fuente: Elaboración propia.

- El ratio de exactitud se interpreta como: El 77 % de los proyectos de la muestra fueron predichos correctamente.
- El ratio de precisión para los positivos (*Successful*) se interpreta como: El 58 % de los proyectos exitosos predichos de la muestra fueron clasificados correctamente.
- El ratio de sensibilidad para los positivos (*Successful*) se interpreta como: El 71 % de los proyectos exitosos reales de la muestra fueron clasificados correctamente.
- El ratio de Puntaje F1 para los positivos (*Successful*) representa un balance entre las 2 métricas anteriores. En la tabla se observa que su valor es de 63 %, lo cual indica que en general, el modelo presenta un rendimiento regular.

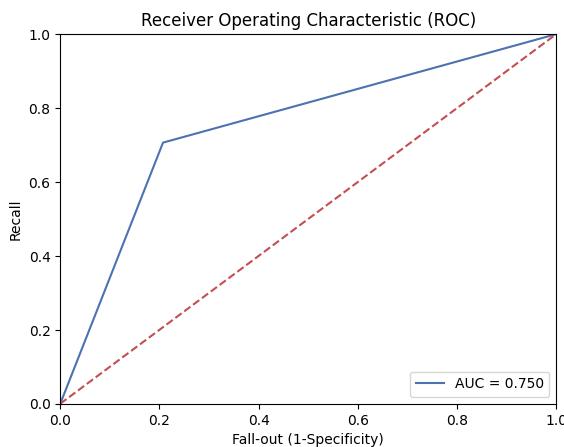


Figura 132. Área bajo la curva ROC de modelo de descripciones.

Fuente: Elaboración propia.

El área bajo la Curva ROC presenta un valor de aproximadamente 75 %, del cual se observa en el gráfico que su sensibilidad es medianamente alta y el ratio de Falsa Alarma es medianamente baja. De acuerdo con Britos et al. (2006), el poder discriminante del modelo es aceptable .

Actividad 3: Evaluar desempeño del modelo predictivo de Comentarios

Luego de 43 épocas, con 77 segundos en promedio 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 1 época antes se había reducido su tasa de aprendizaje.

Así, de acuerdo a la Figura 133, en la época 33 se registran los mejores valores de exactitud y pérdida para el subconjunto de validación, alcanzando 0.8510 y 0.4472 respectivamente.

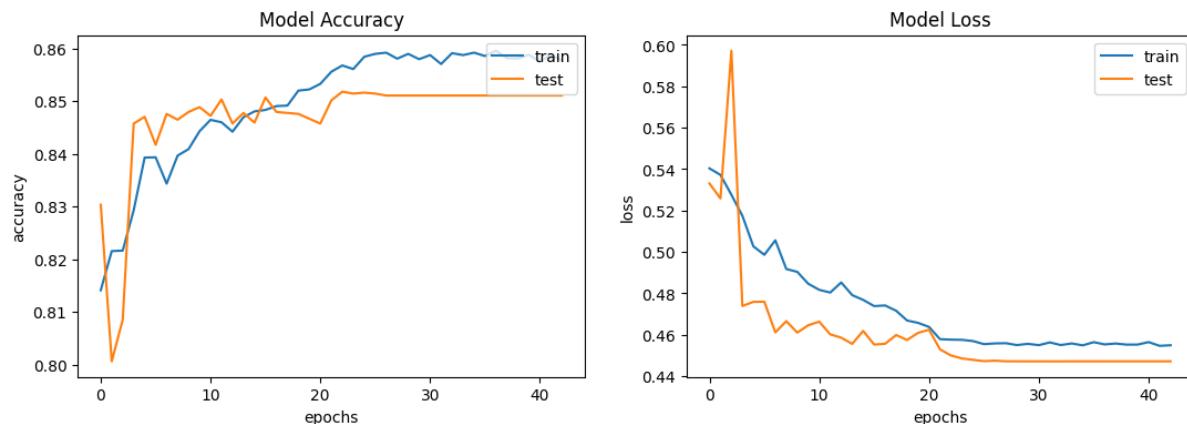


Figura 133. Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo RNN de comentarios con 50 épocas.

Fuente: Elaboración propia.

La matriz de confusión resultante se representa en la Figura 134.

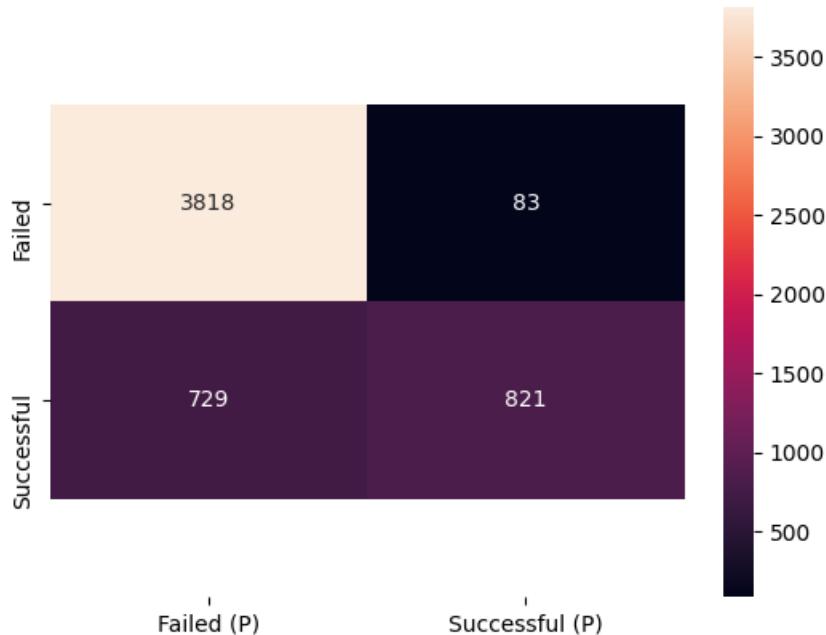


Figura 134. Matriz de confusión para el modelo de comentarios.

Fuente: Elaboración propia.

De esta matriz, se derivan los resultados de la Tabla 19 y el AUC en la Figura 135.

Tabla 19
Informe de clasificación para el modelo de comentarios.

Valor	Precisión	Sensibilidad	Puntaje F1	Muestras
Fracasado	0.84	0.98	0.90	3,901
Exitoso	0.91	0.53	0.67	1,550
Exactitud			0.85	5,451
Promedio macro	0.87	0.75	0.79	5,451
Promedio ponderado	0.86	0.85	0.84	5,451

Fuente: Elaboración propia.

- El ratio de exactitud se interpreta como: El 85% de los proyectos de la muestra fueron predichos correctamente.
- El ratio de precisión para los positivos (*Successful*) se interpreta como: El 91% de los proyectos exitosos predichos de la muestra fueron clasificados correctamente.
- El ratio de sensibilidad para los positivos (*Successful*) se interpreta como: El 53% de los proyectos exitosos reales de la muestra fueron clasificados correctamente.
- El ratio de Puntaje F1 para los positivos (*Successful*) representa un balance entre las 2 métricas anteriores. En la tabla se observa que su valor es de 67%, lo cual indica que en general, el modelo presenta un rendimiento regular.

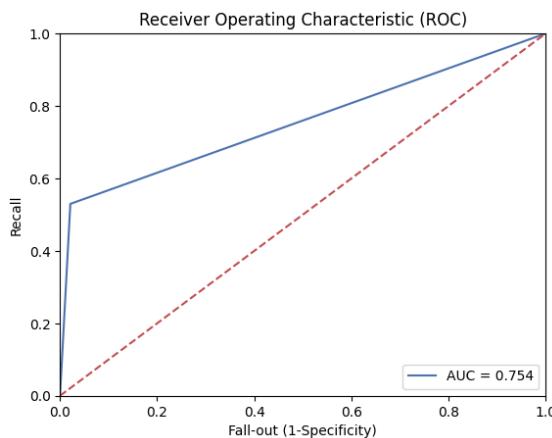


Figura 135. Área bajo la curva de modelo de comentarios.

Fuente: Elaboración propia.

El área bajo la Curva ROC presenta un valor de aproximadamente 75 %, del cual se observa en el gráfico que su sensibilidad es baja pero su ratio de Falsa Alarma es casi nulo. De acuerdo con Britos et al. (2006), el poder discriminante del modelo es aceptable.

Actividad 4: Evaluar desempeño del modelo ensamblado apilado

Luego de 13 épocas, con 152 segundos de entrenamientos 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, a pesar de que hace 2 épocas se redujo su tasa de aprendizaje.

Así, de acuerdo a la Figura 136, en la época 3 se registran los mejores valores de exactitud y pérdida para el subconjunto de validación, alcanzando 0.9336 y 0.1810 respectivamente.

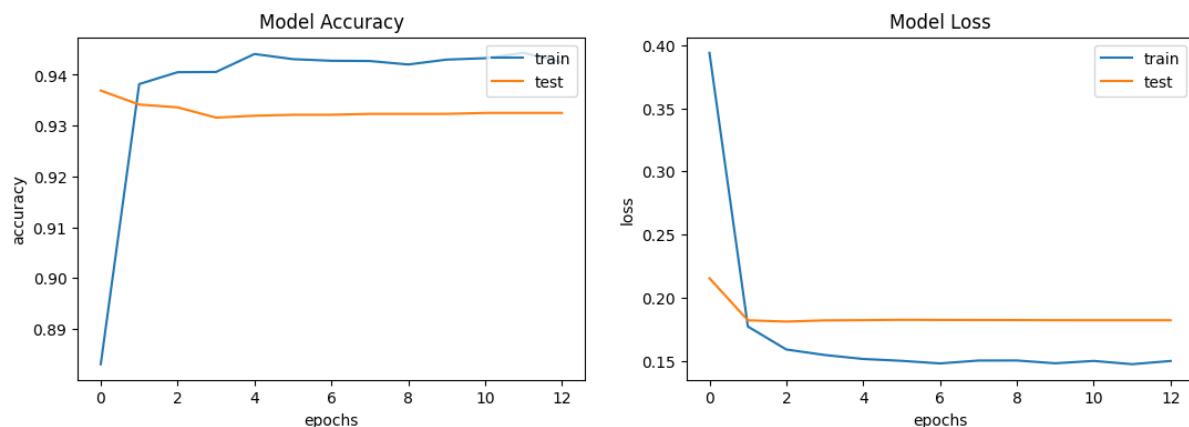


Figura 136. Exactitud y pérdida respectivamente de los subconjuntos de entrenamiento y validación para el modelo apilado con 200 épocas.

Fuente: Elaboración propia.

La matriz de confusión resultante se representa en la Figura 137.

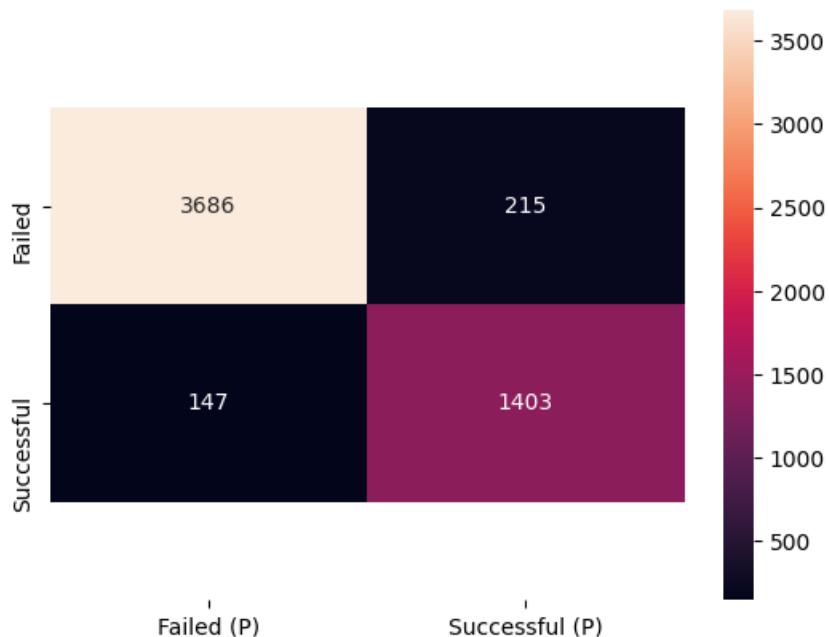


Figura 137. Matriz de confusión para el modelo apilado.

Fuente: Elaboración propia.

De esta matriz, se derivan los resultados de la Tabla 20 y el AUC en la Figura 138.

Tabla 20

Informe de clasificación para el modelo apilado.

Valor	Precisión	Sensibilidad	Puntaje F1	Muestras
Fracasado	0.96	0.94	0.95	3,901
Exitoso	0.87	0.91	0.89	1,550
Exactitud			0.93	5,451
Promedio macro	0.91	0.93	0.92	5,451
Promedio ponderado	0.93	0.93	0.93	5,451

Fuente: Elaboración propia.

- El ratio de exactitud se interpreta como: El 93 % de los proyectos de la muestra fueron predichos correctamente.
- El ratio de precisión para los positivos (*Successful*) se interpreta como: El 87 % de los proyectos exitosos predichos de la muestra fueron clasificados correctamente.
- El ratio de sensibilidad para los positivos (*Successful*) se interpreta como: El 91 % de los proyectos exitosos reales de la muestra fueron clasificados correctamente.

- El ratio de Puntaje F1 para los positivos (*Successful*) representa un balance entre las 2 métricas anteriores. En la tabla se observa que su valor es de 89 %, lo cual indica que en general, el modelo presenta un rendimiento regular.

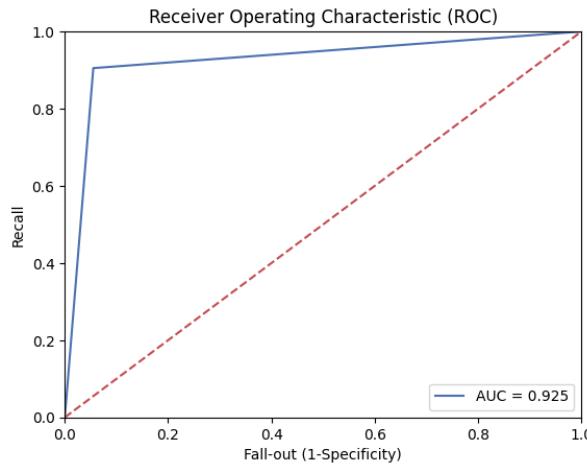


Figura 138. Área bajo la curva de modelo apilado.

Fuente: Elaboración propia.

El área bajo la Curva ROC presenta un valor aproximado de 93 %, del cual se observa en el gráfico que su sensibilidad es muy alta y su ratio de Falsa Alarma es casi nulo. De acuerdo con Britos et al. (2006), el poder discriminante del modelo es excepcionalmente bueno.

Finalmente, considerando los modelos independientes para cada modalidad, así como un trabajo previo del autor de la presente investigación cuyo trabajo sirvió de base (Puente, 2019), se armó el cuadro comparativo de la Tabla 21.

Tabla 21

Comparación de resultados de modelos propuestos con antecedentes.

Modelos		Exactitud	Precisión	Sensibilidad	Puntaje F1	AUC
Tesis de pregrado	Metainformación	0.89	0.86	0.72	0.75	0.84
	Descripción	0.75	0.59	0.53	0.35	0.68
Propuesta	Metainformación	0.93	0.91	0.92	0.92	0.92
	Descripción	0.77	0.72	0.75	0.73	0.75
	Comentarios	0.85	0.87	0.75	0.79	0.75
	The Hydra	0.93	0.91	0.93	0.92	0.93

Fuente: Elaboración propia.

Los modelos citados de la Tesis de pregrado para Metainformación y Descripción utilizaron una Máquina de Vectores de Soporte (SVM) y una SVM entrenada con el algoritmo TF-IDF, respectivamente.

Para comparar ambas investigaciones, se usaron las mismas bases de datos para las 2 modalidades, con proyectos tecnológicos de Kickstarter finalizados entre 2009 y 2019, así como también las mismas métricas para evaluar cada modelo. El tiempo de entrenamiento en el antecedente mencionado fue mayor (aproximadamente 16 segundos para la metainformación y 4 horas para la descripción), en contraste con los elaborados en este trabajo (aproximadamente 38 segundos para la metainformación y 2 horas y media para la descripción).

Como se observa en la Tabla 21, a nivel general, la performance de The Hydra fue mejor tanto contra los modelos individuales de cada modalidad (superando en más de 0.03 y más de 0.05 al modelo de comentarios y descripción respectivamente en las 5 métricas, y en 0.01 al de metainformación en AUC) como contra los modelos referenciados en los antecedentes (más de 0.05 en todas las métricas para el modelo de metainformación y más de 0.18 en todas las métricas para el modelo de descripción). El concepto (con otros modelos y variantes en el desarrollo) utilizado en la Tesis de pregrado se basó en el trabajo de los autores Cheng et al. (2019), el cual utilizó un marco de trabajo de Aprendizaje Profundo Multimodal (*Multimodal Deep Learning* en inglés), donde se combinan las características de metainformación, descripción e imagen principal del proyecto en la capa totalmente conectada. Sin embargo, en dicha ocasión no se alcanzó lograr los objetivos dado que el modelo de contenido visual presentó problemas para clasificar adecuadamente un proyecto según su estado. Esto se dio en parte a la variedad de imágenes dentro de la misma categoría Tecnología, que contiene asimismo 16 subcategorías, lo cual dificultó en su momento a la red a encontrar patrones a partir de su características. Se decidió, entonces, cambiar el criterio de reemplazar el contenido visual por otra modalidad respaldada por varios antecedentes (enunciados en la descripción del prototipo de investigación del Capítulo III) y que no fue tomada en cuenta en su momento, los comentarios realizados durante la campaña por los patrocinadores del proyecto.

5.3 Despliegue

Actividad 1: Diseñar prototipo de sistema con modelo propuesto

La última fase de la metodología CRISP-DM comienza con el diseño del prototipo que contemplará el sistema conformado por la captura de datos y la predicción del estado de financiamiento de un proyecto consultado. Para ello, previamente se deberán cargarse todos los complementos necesarios para que el modelo de Aprendizaje Profundo Multimodal funcione correctamente. Desde librerías de Python que incluyen elementos utilizados en la recolección de datos como Selenium y las librerías de Keras para la carga de las capas del modelo, hasta algoritmos usados para la limpieza de texto como NLTK y las métricas de clasificación por parte de Scikit-learn.

Luego, se diseñó el código para ejecutar el proceso de la Figura 87, el cual puede ser descargado desde el link <https://www.kaggle.com/alonsopuente/tesis-titulacion-demo>

Actividad 2: Ejecutar prototipo con proyectos tecnológicos vigentes

Una vez diseñado el proceso del prototipo, el software es ejecutado localmente desde Jupyter Notebook y recibe como dato de entrada el enlace web de un proyecto vigente en Kickstarter, representado en la Figura 139.

```
print("Por favor, ingrese el URL del proyecto que desea analizar:\n")
url_input = input()
url_input = url_input + '?'

Por favor, ingrese el URL del proyecto que desea analizar:

https://www.kickstarter.com/projects/2125914059/revopoint-pop-high-precision-3d-scanner-for-3d-printing?ref=discovery_category_ending_soon
```

Figura 139. Proyecto consultado para la demostración. Captura de pantalla: 15/02/21.

Fuente: Elaboración propia.

Uno de los experimentos hechos el 23 de enero del 2021 se realizó con la campaña vigente de ejemplo de la Figura 140.

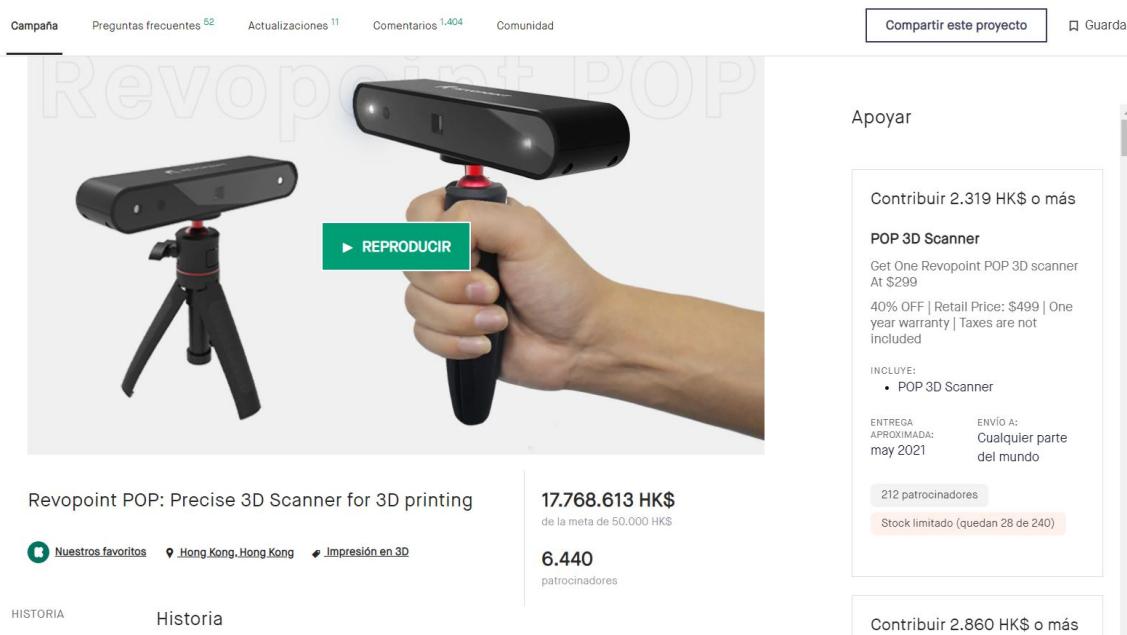


Figura 140. Campaña del proyecto consultado. Captura de pantalla: 15/02/21.

Fuente: Elaboración propia.

La primera acción hecha por el sistema fue extraer la metainformación, descripción y comentarios del proyecto desde el ingreso al URL por un navegador. Debido al cambio de políticas de acceso a la plataforma en 2020, Kickstarter detecta la presencia de bots y restringe la navegación usando CAPTCHAs para evitar su accionar. Algunas veces fue detectado el bot del sistema. Ante ello, la única acción manual por parte del usuario en el sistema se da en este paso presionando por 5 segundos el botón de “I'm a human” (Soy humano por su traducción

al español) que se muestra en la ventana. Desde este punto, luego de ingresar al enlace de la campaña del proyecto, el sistema primero se redirige a la sección de la metainformación y extrae las variables usadas para el entrenamiento del modelo.

Esta secuencia es repetida para las otras modalidades. En el caso de los comentarios, se encuentran en una sección distinta, para lo cual el sistema primero se redirige a ella a partir del dominio de la sección principal, en donde se encontraron la metainformación y la descripción.

Los datos extraídos de cada modalidad se muestran en la Figura 141 respectivamente.

Revopoint POP: Precise 3D Scanner for 3D printing
Cantidad de montos de contribución disponibles: 14
Mediana de montos de contribución disponibles: 2318800.0
Meta de la campaña: 50000 HKD
Monto contribuído en la campaña: 13741035 HKD
Porcentaje contribuído: 27482 %
Duración de la campaña: 43

(a) Metainformación

"Is there more detail about the led light anywhere? I have some lighting already and need to know where about the light to set it if is worth getting over my existing lighting options. Does it integrate or connect to the scanner in any way or is it just a stand alone light?"
"Hi, I am new to UK, I have two questions. 1) Will you be shipping directly to the UK as it is done through the USA? 2) UK customers will have to incur additional VAT and duty on delivery. This can be 20% + fees etc. Please ship these direct to UK and avoid the extra fees. 3) I messaged you last week on Facebook messenger to claim my free markers but never got a reply. Thanks!"
"I am painting what need to scan a different color (90% of the time I just can't do), how can the user solve the issue with the scanner being unable to see Black and having a very difficult time with dark colors like grays, blues, purples, browns, etc.? This can be a real deal breaker for me as a lot of what I'd be scanning will be gray or black. Can the IR be used off-camera?"
"Hello! I'm Backer #318. I added 15400 to my 500 marker. Please, Thanks! How to get updates for the software later on?"
"You have considered a way to switch between the two IR sensors their current distance apart and closer together for scanning items smaller than 90mm x 50mm x 50mm? It seems that there are a number of bakers that want accurate scans of small items..."
"What would be possible to accurately scan small objects with hollow inside like motorcycle intake pipe?"
"I just added \$35 for the LED light, but I can't find my baker number "anywhere"..."
"#903 Thanks for the email!"

(b) Descripción

(c) Comentarios

Figura 141. Variables extraídas por modalidad del proyecto consultado.

Fuente: Elaboración propia.

La información extraída es pre-procesada de la misma manera que la data usada para entrenar cada modelo. A continuación, el modelo cargado The Hydra recibe los datos procesados y concatenados para realizar la predicción. Si el umbral es por lo menos 0.50, el resultado será **EXITOSO** (*Successful* en inglés) como en la Figura 142.

Predicción de probabilidad de financiamiento: 99.86 %
Estado final de la campaña será: EXITOSO

Figura 142. Resultado de predicción de The Hydra para el proyecto consultado.

Fuente: Elaboración propia.

Capítulo VI: Conclusiones y Recomendaciones

6.1 Conclusiones

Luego de identificar y formular el problema general, plantear los objetivos y las hipótesis, se logró cumplir el objetivo de predecir el estado de financiamiento de un proyecto, ya sea detectando aquellos que podrían ser exitosos como los que podrían fracasar, bajo un nuevo criterio, solo considerar proyectos de tecnología, y un nuevo enfoque, implementar un modelo de Aprendizaje Profundo Multimodal que fue denominado «The Hydra».

Respecto a los objetivos específicos, se determinó que el análisis de las alternativas propuestas en los trabajos previos sí influyó en la selección de características y del desarrollo del marco de trabajo de la investigación, ya que se validaron algunas hipótesis de la literatura respecto al rendimiento de modelos de acuerdo a las variables, técnicas y parámetros establecidos en el desarrollo, como por ejemplo, la efectividad de modelos de redes neuronales frente a modelos convencionales de Aprendizaje Automático. Esto se puede corroborar en los resultados mostrados en la Tabla 21.

De la mencionada tabla, también se concluye que el modelo de Aprendizaje Profundo Multimodal se vio afectado por las características consideradas en su desarrollo, ya que presentó mejor rendimiento tanto contra sus submodelos como contra los modelos de tesis de pregrado evaluado por las 5 métricas, siendo 0.01 la diferencia contra el segundo mejor modelo bajo el valor AUC y 0.57 contra el peor modelo bajo la sensibilidad. Sin embargo, la desventaja de la propuesta de la investigación radica en que la ausencia o poca presencia de datos en alguna de las modalidades basadas en texto disminuye el ratio de éxito predicho para un proyecto. Esto se debe al comportamiento mencionado, con tendencia al fracaso, que se observó al probar el prototipo con proyectos de tecnología con dichas características, por ejemplo, descripción muy breve o pocos comentarios recibidos por patrocinadores.

Otra desventaja relacionada con el punto anterior fue el comportamiento de la modalidad de comentarios, ya que si bien hasta en 4 antecedentes se resalta su buena performance y el submodelo entrenado presentó niveles entre 0.75 y 0.87 en las 5 métricas evaluadas, solo el 29 % de proyectos de tecnología de este trabajo presentó comentarios, y de este universo, el 41 % fracasaron en ser financiados. Se encontró, además, una notable brecha entre el promedio de palabras de proyectos exitosos (aproximadamente 103 palabras en todos los comentarios) frente al de aquellos que fracasaron (en promedio 6 palabras en todos los comentarios). Como dato adicional, de las más de 74 mil palabras únicas en el diccionario desarrollado, se observaron términos que no fueron del todo lematizados dado a la complejidad de interpretación por parte del algoritmo ante los errores ortográficos encontrados en cada palabra (por ejemplo,

repeticiones de vocales o sustitución de consonantes), comúnmente expuestas en el lenguaje informal de las comunicaciones online. Esta observación no afectó significativamente en la etapa de entrenamiento del submodelo. Sin embargo, encontrar una manera de lidiar con ella, es decir, aumentar la valorización del procesamiento de textos sobre todo en los comentarios, sí hubiese permitido mejorar su actual rendimiento al utilizar su lema correcta que aparece con mayor frecuencia.

A nivel individual, The Hydra bajo cada métrica (desde las más usadas como la exactitud hasta aquellas más recomendadas para problemas con data desbalanceada como el puntaje F1) mantuvo niveles parejos y conllevó sin problemas su entrenamiento, pese a que tanto el modelo de descripción como de comentarios se obstaculizaron con el sobreajuste luego de muchas épocas. Entre una de las razones por las cuales ambos modelos no progresaban luego de una avanzada cantidad de épocas se encontró en el contenido textual, en especial, el de comentarios ya que la interacción social muchas veces no está sujeta a estrictas normativas de la gramática hacia los usuarios que expresan libremente su opinión. Por lo tanto, algunas palabras incorrectamente redactadas no pudieron ser lematizadas al 100% por la librería NLTK. El modelo de metainformación, en cambio, ayudó a mejorar el rendimiento del modelo apilado, ya que al combinar sus predicciones con los otros dos modelos, la nueva performance del conjunto se incrementó al evaluarse con las 5 métricas (un poco menos de 0.01 en la exactitud, precisión, sensibilidad y puntaje F1, y un poco más de 0.01 en el AUC).

A pesar de presentarse una data desbalanceada (72% proyectos fracasados y 28% exitosos), fraccionar la base total en subconjuntos de entrenamiento y pruebas de forma estratificada, es decir, mantener la distribución de 72% fracasados y 28% exitosos para cada subconjunto, y luego previo a la creación de cada modelo balancear los pesos de las clases (0.6987077585764833 para proyectos fracasados y 1.7581290322580645 para exitosos) fueron también determinantes para que los modelos eviten caer en sobreajuste tempranamente y presenten comportamientos de exactitud y pérdida en la validación cercanas al entrenamiento como se presentó en la Figura 136.

Asimismo, la evaluación de la factibilidad técnica del ambiente de desarrollo para las características del modelo de Aprendizaje Profundo Multimodal determinó la aplicabilidad de las condiciones propuestas en la literatura. Algunos de los trabajos del Capítulo II que inicialmente se tenía en mente implementar eran modelos Seq2seq o LDA. Por ejemplo, Shafqat y Byun (2019) planteó una arquitectura de este último tipo para resolver el problema de clasificación que encajaba con el marco de trabajo de la actual tesis de investigación, aplicando segmentaciones de comentarios según el tema de su contenido para luego alimentar a su sistema de recomendación de proyectos. Sin embargo, como se explica en las especificaciones de sus requerimientos para llevar a cabo estos experimentos, se necesitó tener al menos una

memoria RAM de 32 GB y una GPU potente como Nvidia GForce 1080 para llevar a cabo los experimentos con más de 504 mil comentarios filtrados provenientes de 600 proyectos de Kickstarter. Esto resultó inviable para las condiciones presentes en el entorno ya que, si bien la suscripción a Google Colab Pro permite utilizar GPU con hasta aproximadamente 26 GB de memoria, el conjunto recolectado de comentarios representó más de 10 veces (7,865 proyectos con comentarios) la cantidad mencionada con un total de más de 494 mil comentarios. De igual manera, el modelo Seq2seq implicaba el uso de una extensa memoria RAM y su otra desventaja se encontró en no poder alterar internamente una arquitectura ya modificada, es decir, modificar las capas y sus conexiones. Ante este escenario, se optó por la opción de un modelo LSTM Bidireccional, acortando el número de palabras del total de comentarios por proyecto a un valor estándar para poder diseñar la capa de incrustación correspondiente.

Finalmente, el último objetivo específico cumplido fue la implementación de una herramienta analítica en tiempo real para ayudar a los emprendedores y creadores de proyectos de tecnología en la toma de decisiones y estrategias de sus campañas. El prototipo del sistema funciona localmente en la computadora del investigador y fue puesto a prueba con al menos 1 proyecto vigente de tecnología.

6.2 Recomendaciones

Para futuros trabajos de investigación, se recomienda crear una plataforma web que contenga el prototipo del sistema descrito en la sección de despliegue del Capítulo V, que integre tanto la parte de extracción y pre-procesamiento del input como el modelo The Hydra, con una interfaz que permita al usuario aprender a utilizarla de manera autodidacta, siguiendo las buenas prácticas de experiencia del usuario y motivada por la aplicación de la extensión en Google Chrome que realizaron los autores K. Chen et al. (2013).

Para afinar el desarrollo y los resultados de este trabajo, se sugiere comenzar con continuar ajustando los hiperparámetros de los modelos de contenido textual para progresar en la etapa de entrenamiento, mediante técnicas como validación cruzada (*k-fold Cross Validation*), *Grid Search*, *Random Search*, entre otros. Además, se sugiere también buscar otras alternativas de optimizadores (por ejemplo, *RMSProp*, *SGD*) o alterar más parámetros de la opción usada *Adam*, usando otros inicializadores de kernel, entre otros, con el fin de optimizar los resultados de la predicción de los submodelos.

En caso se cuente con herramientas tecnológicas más potentes de hardware y software para el desarrollo de modelos predictivos más profundos como modelos Seq2seq, multimodales o híbridos del tipo DC-LDA, se recomienda limitar la extensión de palabras a un valor no mayor a la cantidad de comentarios presentada en el trabajo de Shafqat y Byun (2019).

Para lidiar con las oportunidades de mejora del tercer y cuarto párrafo de las conclusiones, se sugiere considerar otras variables y modalidades, como por ejemplo, la interacción social externa entre el creador y la comunidad, ya sea en la sección de comentarios como se consideró en esta investigación, así como también en las menciones del proyecto en las redes sociales para efectuar un análisis de sentimientos más profundo. Estas deberían considerarse en un segundo modelo de Aprendizaje Profundo Multimodal (como lo trabajaron los autores Shafqat y Byun (2019) en su investigación de predicción de temas en varios documentos) que luego sería concatenado con el primero, en las que el creador de la campaña interviene directamente (metainformación y descripción del proyecto), ya que la dependencia de una modalidad en la cual interviene una tercera parte (patrocinadores) podría afectar negativamente la poca o nula presencia de esta información, para lo cual se le podría asignar a este último grupo un menor peso en la fase de entrenamiento. Bajo este escenario, se podría tomar en cuenta opciones de limpieza de texto más avanzadas u otras disponibles (por ejemplo, realizar experimentos con *stemming*) para reducir lo máximo posible la aparición de palabras únicas de un mismo significado u origen pero reconocidas como distintas en el diccionario entrenado, y con esto implementar un clasificador de estados de financiamiento más sólido.

Se prefiere evitar usar la modalidad de imagen y/o video principal de la campaña, ya que como se comentó en la fase de Evaluación, en un trabajo previo desarrollado por el autor, no se encontraron características similares entre ellos para el caso particular de la categoría Tecnología. Otras alternativas potenciales también figuran la de predicción de las mejores opciones de valores que deberían contener las variables (tanto cuantitativas como cualitativas) para un proyecto antes del lanzamiento de su campaña. Este enfoque permitiría evaluar indefinidamente la información que un creador asigne a su campaña hasta encontrar la mejor combinación gracias a un modelo de recomendación.

Como penúltima sugerencia, debería seguirse alguna metodología para definir el valor del punto de corte o *threshold*, ya sea explorando más a detalle los puntos del Área bajo la curva ROC (AUC) u otra técnica, y poder implementar una clasificación más precisa.

Finalmente, como en varias referencias y libros sobre Aprendizaje Automático y Aprendizaje Profundo que se pueden encontrar, no existe una regla definida para asegurar el rendimiento excelente de cualquier modelo. El factor del logro de objetivos principalmente se debe a la continua experimentación y uso de distintas técnicas para alcanzar la performance esperada.

Referencias

- A Not So Random Walk. (2019). Backpropagation Example With Numbers Step by Step. Recuperado de <https://www.anotsorandomwalk.com/backpropagation-example-with-numbers-step-by-step/>.
- Alpaydin, E. (2014). *Introduction to Machine Learning* (3^a ed.). MIT Press.
- Ardi, M. (2020, 28 de septiembre). *1-Dimensional Convolution Layer for NLP Task*. Recuperado de <https://becominghuman.ai/1-dimensional-convolution-layer-for-nlp-task-5d2e86e0229c>.
- Baheti, P. (2020). *Introduction to Multimodal Deep Learning*. Recuperado de <https://heartbeat.fritz.ai/introduction-to-multimodal-deep-learning-630b259f9291>.
- BBVA OpenMind. (2019). ¿Qué es el aprendizaje Profundo? (A. Banafa, Ed.). Recuperado de <https://www.bbvaopenmind.com/tecnologia/mundo-digital/que-es-el-aprendizaje-profundo/>.
- Beckwith, J. (2016). *Predicting Success in Equity Crowdfunding* [Tesis de grado]. Universidad de Pensilvania. Recuperado de 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. Recuperado de <http://laboratorios.fi.uba.ar/lsi/bertona-tesisingenieriainformatica.pdf>.
- Betancourt, G. A. (2005). Las Máquinas de Soporte Vectorial (SVMs). *Scientia et Technica*. Recuperado de <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* (Reporte técnico). Universitat Oberta de Catalunya. Barcelona, España. Recuperado de <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>.
- Brinton, C., & Inouye, D. (2020). *Python for Data Science: n-grams and basic natural language processing* (Reporte técnico). Universidad Purdue. Recuperado de <https://www.cbrinton.net/ECE20875-2020-Spring/W10/ngrams.pdf>.
- Britos, P. V., García Martínez, R., Hossian, A., & Sierra, E. (2006). *Minería de Datos*. Nueva Librería.
- Brownlee, J. (2017a). *Deep Learning for Natural Language Processing: Develop Deep Learning Models for your Natural Language Problems*. Machine Learning Mastery. Recuperado de http://ling.snu.ac.kr/class/AI_Agent/deep_learning_for_nlp.pdf.

- Brownlee, J. (2017b). *How to Develop a Bidirectional LSTM For Sequence Classification in Python with Keras*. Machine Learning Mastery. Recuperado de <https://machinelearningmastery.com/develop-bidirectional-lstm-sequence-classification-python-keras/>.
- Brownlee, J. (2018). *Stacking Ensemble for Deep Learning Neural Networks in Python*. Machine Learning Mastery. Recuperado de <https://machinelearningmastery.com/stacking-ensemble-for-deep-learning-neural-networks/>.
- Brownlee, J. (2019). *How to Fix the Vanishing Gradients Problem Using the ReLU*. Machine Learning Mastery. Recuperado de <https://machinelearningmastery.com/how-to-fix-vanishing-gradients-using-the-rectified-linear-activation-function/>.
- Bujokas, E. (2020). *Creating Word Embeddings: Coding the Word2Vec Algorithm in Python using Deep Learning*. Towards Data Science. Recuperado de <https://towardsdatascience.com/creating-word-embeddings-coding-the-word2vec-algorithm-in-python-using-deep-learning-b337d0ba17a8>.
- Cai, Y., Moore, K., Pellegrini, A., Elhaddad, A., Lessel, J., Townsend, C., Solak, H., & Semret, N. (2017, abril). *Crop yield predictions - high resolution statistical model for intra-season forecasts applied to corn in the US* (Reporte técnico). Recuperado de https://www.researchgate.net/publication/316278341_Crop_yield_predictions_-_high_resolution_statistical_model_for_intra-season_forecasts_applied_to_corn_in_the_US.
- Calvo, D. (2017). Clasificación de redes neuronales artificiales. Recuperado de <http://www.diegocalvo.es/clasificacion-de-redes-neuronales-artificiales/>.
- Calvo, D. (2018a). Definición de Red Neuronal Recurrente. Recuperado de <http://www.diegocalvo.es/red-neuronal-recurrente/>.
- Calvo, D. (2018b). Función de activación - Redes neuronales. Recuperado de <http://www.diegocalvo.es/funcion-de-activacion-redes-neuronales/>.
- Chaichi, N., & Anderson, T. (2019). Deploying Natural Language Processing to Extract Key Product Features of Crowdfunding Campaigns: The Case of 3D Printing Technologies on Kickstarter. *2019 Portland International Conference on Management of Engineering and Technology (PICMET)*, 1-9. doi: 10.23919/picmet.2019.8893839.
- Chen, K., Jones, B., Kim, I., & Schlamp, B. (2013). *KickPredict: Predicting Kickstarter Success* (Reporte técnico). Instituto de Tecnología de California. California, Estados Unidos de América. Recuperado de <http://courses.cms.caltech.edu/cs145/2013/blue.pdf>.
- Chen, L.-S., & Shen, E.-L. (2019). Finding the Keywords Affecting the Success of Crowdfunding Projects. *2019 IEEE 6th International Conference on Industrial Engineering and Applications (ICIEA)*, 567-571. doi: 10.1109/iea.2019.8714815.
- Chen, P. (2012). *The New Project Page*. Kickstarter. Recuperado de <https://www.kickstarter.com/blog/the-new-project-page>.

- Chen, S.-Y., Chen, C.-N., Chen, Y.-R., Yang, C.-W., Lin, W.-C., & Wei, C.-P. (2015). Will Your Project Get the Green Light? Predicting the Success of Crowdfunding Campaigns. *Pacific Asia Conference on Information Systems (PACIS) 2015*, 79. Recuperado de <http://aisel.aisnet.org/pacis2015/79>.
- Cheng, C., Tan, F., Hou, X., & Wei, Z. (2019). Success Prediction on Crowdfunding with Multimodal Deep Learning. *Twenty-Eighth International Joint Conference on Artificial Intelligence*, 2158-2164. doi: 10.24963/ijcai.2019/299.
- Chengwei. (2018). *Simple Stock Sentiment Analysis with news data in Keras*. DLology. Recuperado de <https://www.dlogy.com/blog/simple-stock-sentiment-analysis-with-news-data-in-keras/>.
- Ciaburro, G., & Joshi, P. (2019). *Python Machine Learning Cookbook* (2^a ed.). Recuperado de https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781789808452/1/ch01lvl1sec15/data-scaling.
- 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? Recuperado de <https://www.cyberclick.es/publicidad/campana-publicitaria>.
- Das, S. (2020). *Reinforced Pac-man*. Towards Data Science. Recuperado de <https://towardsdatascience.com/reinforced-pac-man-8e51409f4fc>.
- Deng, L., & Liu, Y. (2018). *Deep Learning in Natural Language Processing*. Springer. doi: 10.1007/978-981-10-5209-5.
- Divina, F., Gilson, A., Gómez-Vela, F., Garcia Torres, M., & Torres, J. (2018). Stacking Ensemble Learning for Short-Term Electricity Consumption Forecasting. *Energies*, 11, 949. doi: 10.3390/en11040949.
- 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.), *2012 International Conference on Environment, Energy and Biotechnology* (p. 40, Vol. 33). IACSIT Press. Recuperado de <http://www.ipcbee.com/vol33/008-ICEEB2012-B021.pdf>.
- Fernandez-Blanco, A., Villanueva-Balsera, J., Rodriguez-Montequin, V., & Moran-Palacios, H. (2020). Key Factors for Project Crowdfunding Success: An Empirical Study. *Sustainability*, 12(2), 599. doi: 10.3390/su12020599.
- Figueroa M., G. (s.f.). Convolución y transformadas. *Revista digital Matemática*. Recuperado de <https://tecdigital.tec.ac.cr/revistamatematica/cursos-linea/EcuacionesDiferenciales/EDO-Geo/edo-cap5-geo/laplace/node7.html>.

- Gandhi, R. (2018, junio). *Support Vector Machine — Introduction to Machine Learning Algorithms*. Towards Data Science. Recuperado de <https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47>.
- Gartner. (2019a). Gartner IT Glossary - Machine Learning. Recuperado de <https://www.gartner.com/it-glossary/machine-learning/>.
- Gartner. (2019b). Gartner IT Glossary - Predictive Modeling. Recuperado de <https://www.gartner.com/it-glossary/predictive-modeling/>.
- Goldberg, Y. (2017). *Neural Network Methods for Natural Language Processing* (G. Hirst, Ed.). Morgan & Claypool Publishers. doi: 10.2200/s00762ed1v01y201703hlt037.
- González, L. (2019). Curvas ROC y Área bajo la curva (AUC). Recuperado de <http://ligdigonzalez.com/curvas-roc-y-area-bajo-la-curva-auc-machine-learning/>.
- Google Cloud. (s.f.). Programa gratuito de Google Cloud. Recuperado de <https://cloud.google.com/free/docs/gcp-free-tier>.
- Google Developers. (2018). Glosario sobre aprendizaje automático. Recuperado de <https://developers.google.com/machine-learning/glossary/?hl=es-419>.
- Gupta, P. (2017, mayo). *Decision Trees in Machine Learning*. Towards Data Science. Recuperado de <https://towardsdatascience.com/decision-trees-in-machine-learning-641b9c4e8052>.
- Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucio, P. (2014). *Metodología de la Investigación* (G. Hirst, Ed.; 6^a ed.). McGraw Hill Education. Recuperado de <https://academia.utp.edu.co/grupobasicoclinicayaplicadas/files/2013/06/Metodolog%C3%A3de-la-Investigaci%C3%B3n.pdf>.
- 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.). Recuperado de <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. Recuperado de <https://iartificial.net/regresion-logistica-para-clasificacion/>.
- IArtificial.net. (2019b). Método del Gradiente Descendiente. Recuperado de <https://iartificial.net/gradiente-descendiente-para-aprendizaje-automatico/>.
- IBM. (2019). Regresión Logística. Recuperado de 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. Recuperado de <https://towardsdatascience.com/understanding-neural-networks-what-how-and-why-18ec703ebd31>.
- Izco, F. (2018). Base de Datos Corporativa. Recuperado de https://bookdown.org/f_izco/BDC-POC/metricas.html.

- Jiang, W., Chen, Z., Xiang, Y., Shao, D., Ma, L., & Zhang, J. (2019). SSEM: A Novel Self-Adaptive Stacking Ensemble Model for Classification. *IEEE Access, PP*, 1-1. doi: 10.1109/access.2019.2933262.
- Jin, B., Zhao, H., Chen, E., Liu, Q., & Ge, Y. (2019). Estimating the Days to Success of Campaigns in Crowdfunding: A Deep Survival Perspective. *33rd AAAI Conference on Artificial Intelligence (AAAI'2019)*, 33, 4023-4030. doi: 10.1609/aaai.v33i01.33014023.
- 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). doi: 10.5281/zenodo.1228716.
- Kamath, U., Liu, J., & Whitaker, J. (2019). *Deep Learning for NLP and Speech Recognition*. Springer.
- Kaur, H., & Gera, J. (2017). Effect of Social Media Connectivity on Success of Crowdfunding Campaigns. *Procedia Computer Science*, 122, 767-774. doi: 10.1016/j.procs.2017.11.435.
- Keras. (s.f.). *Model training APIs*. Recuperado de https://keras.io/api/models/model_training_apis/.
- Kickstarter. (s.f.-a). *Acerca de nosotros: Kickstarter*. Recuperado de <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*. Recuperado de <https://www.kickstarter.com/learn?lang=es>.
- Kickstarter. (s.f.-d). *Financiamiento: Kickstarter*. Recuperado de <https://www.kickstarter.com/help/handbook/funding?lang=es>.
- Kickstarter. (s.f.-e). *Nuestras normas*. Recuperado de https://www.kickstarter.com/rules?ref=learn_faq.
- Kickstarter. (s.f.-f). *Prensa: Kickstarter*. Recuperado de <https://www.kickstarter.com/press?ref=hello>.
- Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. *2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1746-1751. doi: 10.3115/v1/d14-1181.
- Kohavi, R., & Provost, F. (1998). Glossary of Terms Journal of Machine Learning. Recuperado de <http://ai.stanford.edu/~ronnyk/glossary.html>.
- Kostadinov, S. (2019). *Understanding Encoder-Decoder Sequence to Sequence Model*. Towards Data Science. Recuperado de <https://towardsdatascience.com/understanding-encoder-decoder-sequence-to-sequence-model-679e04af4346>.

- Kraus, S., Richter, C., Brem, A., Cheng, C.-F., & Chang, M.-L. (2016). Strategies for reward-based crowdfunding campaigns. *Journal of Innovation & Knowledge*, 1, 13-23. doi: 10.1016/j.jik.2016.01.010.
- Lee, H.-D. (2019). Factors affecting successful crowdfunding. *ACM International Conference Proceeding Series*, 379-382. doi: 10.1145/3306500.3306524.
- Lee, I., & Shin, Y. (2018). Fintech: Ecosystem, business models, investment decisions, and challenges. *Business Horizons*, 61(1), 35-46.
- Lee, S., Lee, K., & Kim, H.-c. (2018). Content-based Success Prediction of Crowdfunding Campaigns: A Deep Learning Approach. *Companion of the 2018 ACM Conference on Computer Supported Cooperative Work and Social Computing*, 193-196. doi: 10.1145/3272973.3274053.
- Li, F.-F., Johnson, J., & Yeung, S. (2019). *Convolutional Neural Networks* (Reporte técnico). Universidad Standford. Recuperado de http://cs231n.stanford.edu/slides/2019/cs231n_2019_lecture05.pdf.
- Li, Y., Rakesh, V., & Reddy, C. K. (2016). Project Success Prediction in Crowdfunding Environments. *Ninth ACM International Conference on web search and data mining*, 247-256. doi: 10.1145/2835776.2835791.
- Lichtig, B. (2015). Crowdfunding Success: The Short Story - Analyzing the Mix of Crowdfunded Ventures. *Wharton Research Scholars*, 121. Recuperado de https://repository.upenn.edu/wharton_research_scholars/121/.
- Liu, K., Li, Y., Xu, N., & Natarajan, P. (2018). Learn to Combine Modalities in Multimodal Deep Learning. *arXiv e-prints*, Artículo arXiv:1805.11730. Recuperado de <https://arxiv.org/pdf/1805.11730.pdf>.
- López Briega, R. E. (2016). Redes neuronales convolucionales con TensorFlow. Recuperado de <https://relopezbriega.github.io/blog/2016/08/02/redes-neuronales-convolucionales-con-tensorflow/>.
- López-Golán, M., Vaca Tapia, A. C., Benavides García, N., & Coronado Otavalos, 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. doi: 10.23919/cisti.2017.7976016.
- Machine Learning for Artists. (2019). Redes Neuronales. Recuperado de https://ml4a.github.io/ml4a/es/neural_networks/.
- Maimon, O., & Rokach, L. (2010). *Data Mining and Knowledge Discovery Handbook* (1^a ed.). Springer.

- Malik, U. (2019). *Python for NLP: Movie Sentiment Analysis using Deep Learning in Keras*. Recuperado de <https://stackabuse.com/python-for-nlp-movie-sentiment-analysis-using-deep-learning-in-keras/>.
- MathWorks. (s.f.). Máquina de vectores de soporte (SVM). Recuperado de <https://la.mathworks.com/discovery/support-vector-machine.html>.
- Merino, M. (2019). Conceptos de inteligencia artificial: qué es el aprendizaje por refuerzo. Recuperado de <https://www.xataka.com/inteligencia-artificial/conceptos-inteligencia-artificial-que-aprendizaje-refuerzo>.
- Microsoft. (2018). Algoritmos de minería de datos (Analysis Services: Minería de datos). Recuperado de <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. Recuperado de <https://docs.microsoft.com/es-es/sql/analysis-services/data-mining/data-mining-concepts?view=sql-server-2017>.
- MissingLink.ai. (s.f.). *Keras Conv1D: Working with 1D Convolutional Neural Networks in Keras*. Recuperado de <https://missinglink.ai/guides/keras/keras-conv1d-working-1d-convolutional-neural-networks-keras/>.
- Mitra, T., & Gilbert, E. (2014). The Language that Gets People to Give: Phrases that Predict Success on Kickstarter. *17th ACM conference on Computer supported cooperative work & social computing*, 49-61. doi: 10.1145/2531602.2531656.
- Molina Arias, M., & Ochoa Sangrador, C. (2017). Pruebas diagnósticas con resultados continuos o politómicos. Curvas ROC. *Evidencias en Pediatría*, 13, 12. Recuperado de http://archivos.evidenciasenpediatria.es/files/41-13133-RUTA/Fundamentos_MBE_12.pdf.
- Mukaka, M. M. (2012). Statistics Corner: A guide to appropriate use of Correlation coefficient in medical research. *Malawi Medical Journal*, 24(3), 69-71. Recuperado de https://www.researchgate.net/publication/236604665_Statistics_Corner_A_guide_to_appropriate_use_of_Correlation_coefficient_in_medical_research.
- Ng, A. (2018). *RNN WILL : Bidirectional RNN* [Archivo de video]. Recuperado de <https://youtu.be/bTXGpATdKRY>.
- Nishida, N., & Nakayama, H. (2015). *Multimodal gesture recognition using multi-stream recurrent neural network* (Reporte técnico). Universidad de Tokio. Recuperado de <http://www.nlab.ci.i.u-tokyo.ac.jp/projects/deeplearn-e.html>.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830.

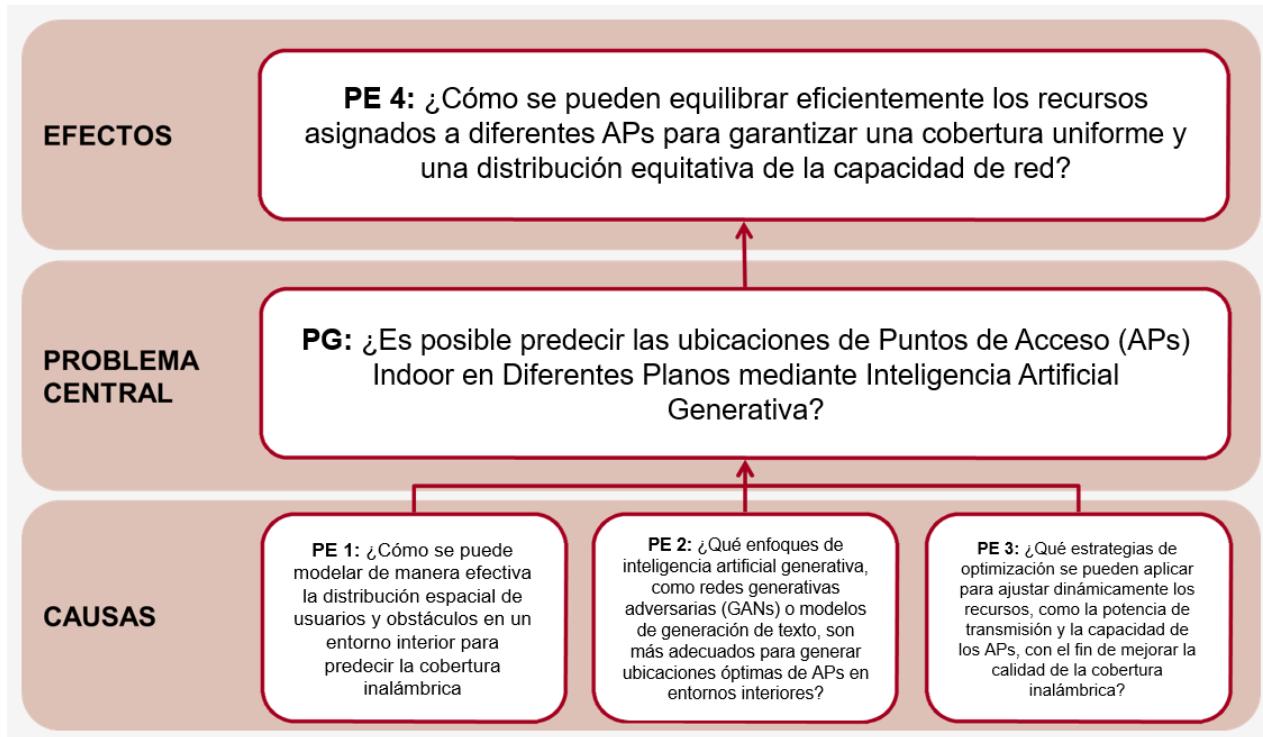
- Pennington, J., Socher, R., & Manning, C. D. (2014a). *GloVe: Global Vectors for Word Representation* (Reporte técnico). Universidad Standford. California, Estados Unidos de América. Recuperado de <https://nlp.stanford.edu/pubs/glove.pdf>.
- Pennington, J., Socher, R., & Manning, C. D. (2014b). *GloVe: Global Vectors for Word Representation*. Recuperado de <https://nlp.stanford.edu/projects/glove/>.
- Phi, M. (2018). *Illustrated Guide to LSTM's and GRU's: A step by step explanation*. Towards Data Science. Recuperado de <https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21>.
- Poole, D., Mackworth, A., & Goebel, R. (1998). Computational Intelligence: A Logical Approach. *Oxford University Press*.
- Prabhu, R. (2018, marzo). *Understanding of Convolutional Neural Network (CNN) — Deep Learning*. Medium. Recuperado de <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)* (6^a ed.). PMBOK.
- Puente, A. (2019). *Predicción del estado de financiamiento de proyectos de tecnología en web de crowdfunding Kickstarter mediante modelo(s) de Aprendizaje Automático* [Tesis de pregrado]. Universidad ESAN.
- Ranjan, C. (2019, julio). *Rules-of-thumb for building a Neural Network*. Towards Data Science. Recuperado de <https://towardsdatascience.com/17-rules-of-thumb-for-building-a-neural-network-93356f9930af>.
- Rao, D., & McMahan, B. (2019). *Natural Language Processing with PyTorch: Build Intelligent Language Applications Using Deep Learning*. O'Reilly.
- Real Academia Española. (s.f.). Hidra. Recuperado de <https://dle.rae.es/hidra>.
- Real Academia Española. (2020). Diccionario de la lengua española (23^a ed.). Recuperado de <https://dle.rae.es/>.
- 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.; 2^a ed.). Pearson Educación, S.A. Recuperado de <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* (3^a ed.). Prentice Hall.
- Sancho Caparrini, F. (2017). *Entrenamiento de Redes Neuronales: mejorando el Gradiente Descendiente* (Reporte técnico). Universidad de Sevilla. Sevilla, España. Recuperado de <http://www.cs.us.es/~fsancho/?e=165>.

- Sancho Caparrini, F. (2018). *Clasificación Supervisada y No Supervisada* (Reporte técnico). Universidad de Sevilla. Sevilla, España. Recuperado de <http://www.cs.us.es/~fsancho/?e=77>.
- SAS Institute. (s.f.). ¿Qué es Deep Learning? Recuperado de https://www.sas.com/es_pe/insights/analytics/deep-learning.html.
- Sawhney, K., Tran, C., & Tuason, R. (2016). *Using Language to Predict Kickstarter Success* (Reporte técnico). Universidad Standford. California, Estados Unidos de América. Recuperado de <https://stanford.edu/~kartiks2/kickstarter.pdf>.
- Scikit-learn. (s.f.). *sklearn.preprocessing.MinMaxScaler*. Recuperado de <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.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. Recuperado de <http://www.ijisr.issr-journals.org/abstract.php?article=IJISR-14-281-04>.
- Shafqat, W., & Byun, Y.-C. (2019). Topic Predictions and Optimized Recommendation Mechanism Based on Integrated Topic Modeling and Deep Neural Networks in Crowdfunding Platforms. *Applied Sciences*, 9(24), 5496. doi: 10.3390/app9245496.
- SitioBigData.com. (2019a). Machine Learning: Selección Métricas de clasificación. Recuperado de <http://sitiobigdata.com/2019/01/19/machine-learning-metrica-clasificacion-parte-3/#>.
- SitioBigData.com. (2019b). ReLU: Funciones de activación. Recuperado de <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. Recuperado de <https://link.springer.com/article/10.1023/B:STCO.0000035301.49549.88>.
- Sutton, R. S., & Barto, A. G. (2018). Finite Markov Decision Processes. En *Reinforcement Learning: An Introduction* (2^a ed., p. 48). MIT Press. Recuperado de <http://incompleteideas.net/book/RLbook2018.pdf>.
- The Hustle. (2019, 9 de febrero). What are your chances of successfully raising money on Kickstarter? Recuperado de <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. *6th International Symposium on Computational and Business Intelligence, ISCBI 2018*, 84-88. doi: 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. doi: 10.3390/jrfm13040080.

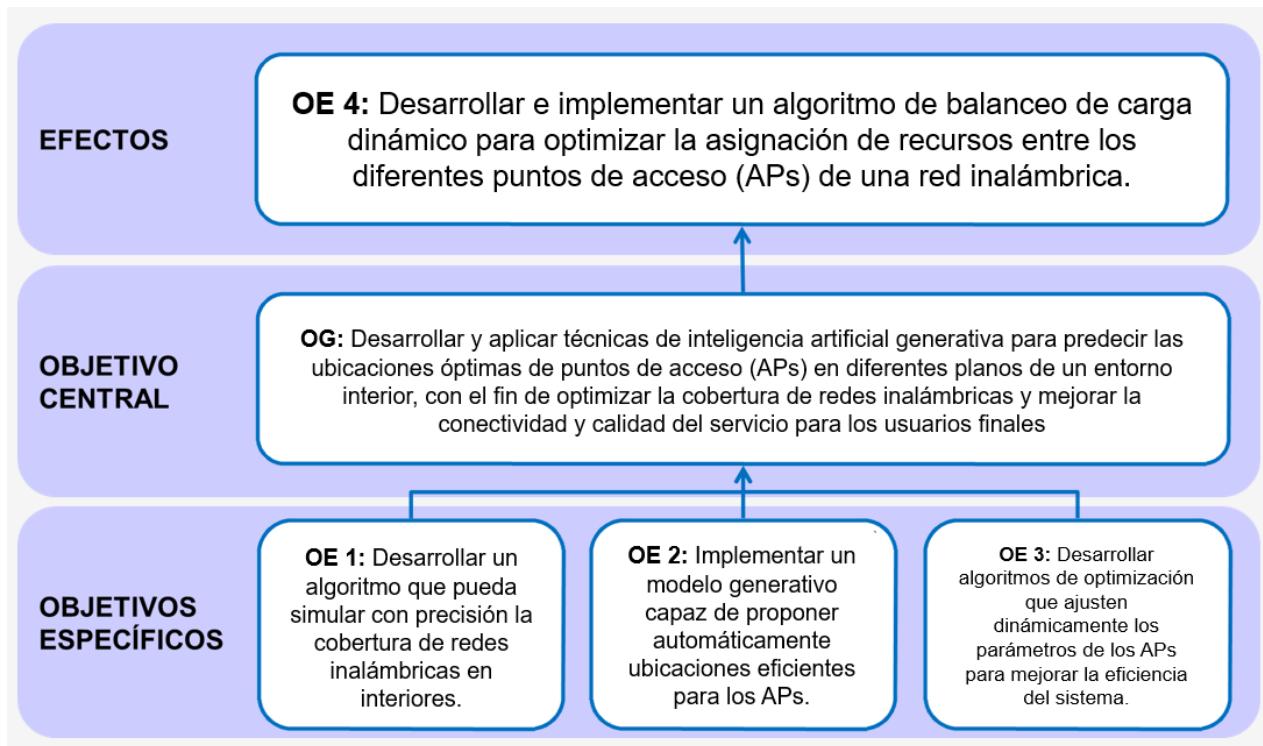
- Viera Balanta, V. (2013). *Backpropagation explicación* [Archivo de video]. Recuperado de <https://youtu.be/0odQ286nsIY>.
- Web Robots. (2019). Kickstarter Datasets. Recuperado de <https://webrbots.io/kickstarter-datasets/>.
- Xuefeng, L., & Zhao, W. (2018). Using Crowdfunding in an Innovative Way: A Case Study from a Chinese Crowdfunding Platform. *2018 Portland International Conference on Management of Engineering and Technology (PICMET)*, 1-9. doi: 10.23919/picmet.2018.8481838.
- Yu, A. (2017). *The Complete Crowdfunding Course for Kickstarter & Indiegogo* (Diapositivas de PowerPoint). Udemy.
- 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. *2018 IEEE 15th International Conference on e-Business Engineering (ICEBE)*, 1-8. doi: 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. doi: 10.1016/j.dss.2016.08.001.
- Yuan, X., Li, L., & Wang, Y. (2019). Nonlinear Dynamic Soft Sensor Modeling With Supervised Long Short-Term Memory Network. *IEEE Transactions on Industrial Informatics*, 1-1. doi: 10.1109/tii.2019.2902129.
- Zambrano, J. (2018, marzo). *¿Aprendizaje supervisado o no supervisado? Conoce sus diferencias dentro del machine learning y la automatización inteligente*. Medium. Recuperado de <https://medium.com/@juanzambrano/aprendizaje-supervisado-o-no-supervisado-39ccf1fd6e7b>.
- Zhang, Y., & Wallace, B. C. (2017). A Sensitivity Analysis of (and Practitioners' Guide to) Convolutional Neural Networks for Sentence Classification. *8th International Joint Conference on Natural Language Processing*, 253-263. Recuperado de <https://www.aclweb.org/anthology/I17-1026.pdf>.
- Zhou, M., Zhang, X., Wang, A. G., Du, Q., Qiao, Z., & Fan, W. (2015). Money Talks: A Predictive Model on Crowdfunding Success Using Project Description. *Twenty-first Americas Conference on Information Systems, Puerto Rico, 2015*. Recuperado de <http://aisel.aisnet.org/amcis2015/BizAnalytics/GeneralPresentations/37>.
- Zhou, V. (2019). *A Simple Explanation of the Bag-of-Words Model*. Towards Data Science. Recuperado de <https://towardsdatascience.com/a-simple-explanation-of-the-bag-of-words-model-b88fc4f4971>.
- Zimovnov, A. (2018). *NLP - Text Preprocessing and Text Classification (using Python)* [Archivo de video]. Recuperado de <https://youtu.be/nxhCyeRR75Q>.

Anexos

A Árbol de Problemas



B Árbol de Objetivos



C Matriz de Consistencia

Título de la tesis	Optimización de Cobertura de Redes Inalámbricas mediante Inteligencia Artificial GENERATIVA para la Predicción de Ubicaciones de Puntos de Acceso (APs) Indoor en Diferentes Planos					
Problema General	Objetivo General	Hipótesis General	Variables	Dimensiones	Indicadores	Metodología
¿Es posible predecir las ubicaciones de Puntos de Acceso (APs) Indoor en Diferentes Planos mediante Inteligencia Artificial Generativa?	Desarrollar y aplicar técnicas de inteligencia artificial generativa para predecir las ubicaciones óptimas de puntos de acceso (APs) en diferentes planos de un entorno interior, con el fin de optimizar la cobertura de redes inalámbricas y mejorar la conectividad y calidad del servicio para los usuarios finales.	La aplicación de técnicas de inteligencia artificial generativa para la predicción de la ubicación de puntos de acceso (Aps) en planes distribuidos en zonas interiores supone una mejora significativa en la cobertura y calidad del servicio de los recursos sin datos.	Independiente: Inteligencia Artificial Generativa	Modelo de Inteligencia Artificial Generativa	Precisión y predicción del modelo de IA Generativa	<ul style="list-style-type: none"> – Tipo de investigación: Diseño Experimental. – Alcance de la investigación: Descriptivo, porque busca describir las características de un fenómeno. – Enfoque de investigación: Cuantitativa.
Problemas Específicos	Objetivos Específicos	Hipótesis Específicas			Efectividad del modelo de IA Generativa	

¿Cómo se puede modelar de manera efectiva la distribución espacial de usuarios y obstáculos en un entorno interior para predecir la cobertura inalámbrica?

Desarrollar un algoritmo que pueda simular con precisión la cobertura de redes inalámbricas en interiores

Los algoritmos de inteligencia artificial generativa, como las GAN, se pueden utilizar para predecir con precisión la ubicación óptima de los puntos de acceso (AP) en un entorno interior.

¿Qué enfoques de inteligencia artificial generativa, como redes generativas adversarias (GANs) o modelos de generación de texto, son más adecuados para generar ubicaciones óptimas de APs en entornos interiores?

Implementar un modelo generativo capaz de proponer ubicaciones eficientes para los APs

La combinación de datos de sensores de señales inalámbricos con datos como planes arquitectónicos mejora la precisión y predicción de la ubicación de AP mediante inteligencia artificial generativa.

Estructura del modelo de IA Generativa

Complejidad de la estructura del modelo de IA Generativa

<p>¿Qué estrategias de optimización se pueden aplicar para ajustar dinámicamente los recursos, como la potencia de transmisión y la capacidad de los APs, con el fin de mejorar la calidad de la cobertura inalámbrica?</p>	<p>Desarrollar algoritmos de optimización que ajusten dinámicamente los parámetros de los APs para mejorar la eficiencia del sistema.</p> <p>La optimización de la cobertura de la red inalámbrica mediante técnicas generativas reduce los costos operativos relacionados con la instalación y el mantenimiento de la infraestructura de red en entornos interiores.</p> <p>Dependiente: Predicción de Ubicaciones de Puntos de Acceso (APs) Indoor en Diferentes Planos</p>	<p>Cobertura de Red</p>	<p>Estado de financiamiento de un proyecto</p>
<p>¿Cómo se pueden equilibrar eficientemente los recursos asignados a diferentes APs para garantizar una cobertura uniforme y una distribución equitativa de la capacidad de red?</p>	<p>Desarrollar e implementar un algoritmo de balanceo de carga dinámico para optimizar la asignación de recursos entre los diferentes puntos de acceso (APs) de una red inalámbrica.</p> <p>La implementación del sistema basado en funciones y el posicionamiento AP de IA generativa mejora la experiencia del usuario: conexiones estables y de alta calidad en diferentes planos complejos.</p>	<p>Experiencia del usuario</p>	<ul style="list-style-type: none"> – Metainformación. – Descripción. – Comentarios.

D Comparación de metodologías de antecedentes

Autor	Título de la Investigación	Metodología	Grupo
Chen, Jones, Kim & Schlamp	KickPredict: Predicting Kickstarter Success	<ul style="list-style-type: none"> – Recolección de datos. – Modelamiento. – Evaluación. – Despliegue. 	GG
Mitra & Gilbert	The Language that Gets People to Give: Phrases that Predict Success on Kickstarter	<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. – Despliegue. 	GE
Zhou, Zhang, Wang, Du, Qiao & Fan	Money Talks: A Predictive Model on Crowdfunding Success Using Project Description	<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. 	GD
Chen, Chen, Chen, Yang, Lin & Wei	Will Your Project Get the Green Light? Predicting the Success of Crowdfunding Campaigns	<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. 	GF
Beckwith	Predicting Success in Equity Crowdfunding	<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. 	GF
Li, Rakesh & Reddy	Project Success Prediction in Crowdfunding Environments	<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. 	GD

		– Recolección de datos.
		– Pre-procesamiento de datos.
		– Modelamiento.
		– Evaluación.
		– Despliegue.
		GE
Yuan, Lau & Xu	The Determinants of Crowdfunding Success: A Semantic Text Analytics Approach	
		– Recolección de datos.
		– Modelamiento.
		– Evaluación.
		GH
Sawhney, Tran & Tuason	Using Language to Predict Kickstarter Success	
		– Recolección de datos.
		– Pre-procesamiento de datos.
		– Modelamiento.
		– Evaluación.
		GF
Kaur & Gera	Effect of Social Media Connectivity on Success of Crowdfunding Campaigns	
		– Recolección de datos.
		– Pre-procesamiento de datos.
		– Modelamiento.
		– Evaluación.
		GD
Kamath & Kamat	Supervised Learning Model For Kickstarter Campaigns With R Mining	
		– Formulación del problema.
		– Recolección y pre- procesamiento de datos.
		– Exploración y extracción de datos.
		– Modelamiento.
		– Evaluación.
		GE
Yu, Huang, Yang, Liu, Li & Tsai	Prediction of Crowdfunding Project Success with Deep Learning	
		– Recolección de datos.
		– Exploración de datos.
		– Pre-procesamiento de datos.
		– Modelamiento.
		– Evaluación.
		GH
Lee, Lee & Kim	Content-based Success Prediction of Crowdfunding Campaigns: A Deep Learning Approach	
		– Recolección de datos.
		– Modelamiento.
		– Evaluación.

Jin, Zhao, Chen, Liu & Ge	Estimating the Days to Success of Campaigns in Crowdfunding: A Deep Survival Perspective	<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. – Despliegue. 	GA
Cheng, Tan, Hou & Wei	Success Prediction on Crowdfunding with Multimodal Deep Learning	<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Pre-procesamiento de datos. – Modelamiento. – Evaluación. – Despliegue. 	GA
Chen & Shen	Finding the Keywords Affecting the Success of Crowdfunding Projects	<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Selección de características. – Modelamiento. – Evaluación. 	GC
Chaichi & Anderson	Deploying Natural Language Processing to Extract Key Product Features of Crowdfunding Campaigns: The Case of 3D Printing Technologies on Kickstarter	<ul style="list-style-type: none"> – Recolección de datos. – Pre-procesamiento de datos. – Selección de características. – Modelamiento. – Evaluación. 	GC
Shafqat & Byun	Topic Predictions and Optimized Recommendation Mechanism Based on Integrated Topic Modeling and Deep Neural Networks in Crowdfunding Platforms	<ul style="list-style-type: none"> – Formulación del problema. – Recolección de datos. – Selección de características. – Modelamiento. – Evaluación. – Despliegue. 	GB

Fernández-		
Blanco,		
Villanueva-	Key Factors for Project	
Balsera,	Crowdfunding Success: An	CRISP-
Rodriguez-	Empirical Study	DM
Montequin &		
Moran-Palacios		

E Comparación de objetivos específicos de antecedentes

Año	Autor	Publicación	Título de la Investigación	Objetivo General	Objetivos Específicos	Item
2013	Chen, Jones, Kim, Schlamp	Reporte técnico	KickPredict: Predicting Kickstarter Success	Desarrollar un sistema para predecir si un proyecto de Kickstarter será financiado exitosamente o no antes de su culminación.	Desarrollar aplicaciones en Android y Google Chrome para predecir en tiempo real el estado de financiamiento y el porcentaje.	OE4
					Analizar las características más importantes que influyen en el éxito de financiamiento a partir de las propiedades de los proyectos.	OE3
					Estudiar el impacto de contribuciones durante el transcurso de la campaña en el estado de financiamiento.	
2014	Mitra, Gilbert	Acta de conferencia	The Language that Gets People to Give: Phrases that Predict Success on Kickstarter	Determinar los factores que conducen a financiar con éxito un proyecto de crowdfunding.	Desarrollar frases predictivas junto con variables de control para ayudar a posteriores estudios y creadores de proyectos.	OE4
					Evaluar el impacto del uso de ciertas frases de la descripción en el éxito de financiamiento.	OE3

	Zhou, Zhang, Wang, Du, Qiao, Fan	Acta de conferencia	Money Talks: A Predictive Model on Crowdfunding Success Using Project Description	Estudiar la influencia de las descripciones de proyectos en el éxito de financiamiento de proyectos crowdfunding.	Examinar el impacto de la calidad argumentativa y fuente de credibilidad de la descripción de un proyecto en su performance durante la campaña. Evaluar e implementar características previamente consideradas en trabajos previos.	OE3 OE1
2015	Chen, Chen, Chen, Yang, Lin, Wei	Acta de conferencia	Will Your Project Get the Green Light? Predicting the Success of Crowdfunding Campaigns	Predecir si una campaña de crowdfunding tendrá éxito a través de extracción y posterior uso de características estáticas y dinámicas.	Analizar relación entre exactitud del modelo y el uso de características dinámicas y/o estáticas. Evaluar efectividad de predicción en distintas etapas de campaña.	OE3 OE1
2016	Beckwith	Tesis de grado	Predicting Success in Equity Crowdfunding	Determinar la relación entre las características de una empresa y su capacidad para recaudar fondos en una plataforma de crowdfunding de capital.	Demostrar la relación entre la probabilidad de éxito de crowdfunding de una compañía y su historial previo de financiamiento. Predecir el éxito de financiamiento de una compañía con precisión, sensibilidad y puntaje F1 mayor a la literatura.	OE1

					Predicción del estado de financiamiento de proyectos de tecnología en sitio web de crowdfunding Kickstarter mediante modelo de Aprendizaje Profundo Multimodal
2016	Li, Rakesh, Reddy	Acta de conferencia	Project Success Prediction in Crowdfunding Environments	Formular la predicción del éxito del proyecto como un problema de análisis de supervivencia y aplicar el enfoque de regresión censurada.	Estudiar la distribución del tiempo de éxito del proyecto de los datos de crowdfunding. Demostrar que el desempeño de los modelos con proyectos exitosos y fracasados es mejor que aquellos que solo comprenden exitosos.
2016	Yuan, Lau, Xu	Artículo	The Determinants of Crowdfunding Success: A Semantic Text Analytics Approach	Implementar un marco de análisis textual para analizar y predecir el éxito de recaudación de proyectos de crowdfunding.	Identificar las características de temas a partir de las descripciones. Identificar las características discriminatorias que influyen en el éxito de financiamiento de proyectos.
2016	Sawhney, Tran, Tuason	Reporte técnico	Using Language to Predict Kickstarter Success	Predecir el éxito de una campaña a partir de su contenido, características lingüísticas y metainformación.	Ayudar a emprendedores a identificar las características textuales más influyentes que afectan los resultados de la recaudación de fondos. OE4 Estudiar relación entre información inicial de campaña (sin incluir variables del tiempo o externas) y estado de financiamiento.

2017	Kaur, Gera	Artículo	Effect of Social Media Connectivity on Success of Crowdfunding Campaigns	Analizar la relación entre la conectividad de las redes sociales y el desempeño de una campaña de crowdfunding.	Evaluar la correlación entre variables de conectividad y variables principales de la campaña. Examinar el impacto de variables principales de la en el desempeño del modelo predictivo. OE3
2018	Kamath, Kamat	Artículo	Supervised Learning Model For Kickstarter Campaigns With R Mining	Implementar un sistema con técnicas de Aprendizaje Automático aplicadas al conjunto de datos de campañas de Kickstarter para clasificar proyectos.	Evaluando el entorno técnico para el desarrollo de fases de metodología. OE2 Analizar propuestas de la literatura para el diseño del marco de trabajo de la investigación. OE1 Determinar el impacto de las propiedades del proyecto para predecir su éxito de financiamiento. OE3
2018	Yu, Huang, Yang, Liu, Li, Tsai	Acta de conferencia	Prediction of Crowdfunding Project Success with Deep Learning	Desarrollar un modelo de Aprendizaje Profundo para predecir el éxito de un proyecto de crowdfunding.	Analizar alternativas de algoritmos de Aprendizaje Automático con conjunto de datos utilizado. Evaluar el rendimiento del modelo desarrollado utilizando solamente la metainformación.

				Implementar modelo más rápido, preciso y eficiente de recursos que línea base.
2018	Lee, Lee, Kim	Acta de conferencia	Content-based Success Prediction of Crowdfunding Campaigns: A Deep Learning Approach	<p>Predecir el estado de financiamiento de proyectos de tecnología con DNN utilizando solo el contenido textual de los proyectos.</p> <p>Alcanzar nivel de rendimiento del estado del arte de por lo menos 89-91 % de exactitud.</p> <p>Estimar valor de variable dependiente en cualquier momento de la campaña con modelo adaptado a nuevos datos.</p>
2019	Jin, Zhao, Chen, Liu, Ge	Acta de conferencia	Estimating the Days to Success of Campaigns in Crowdfunding: A Deep Survival Perspective	<p>Predecir la distribución de promesas y la duración para lograr el éxito de financiamiento implementando un modelo Seq2seq con arquitectura SMP.</p> <p>Identificar la distribución de las contribuciones de acuerdo a la evolución del tiempo (días) de la campaña.</p> <p>Identificar el tiempo correcto de duración que debe tener la campaña a partir del análisis de supervivencia.</p> <p>Ayudar a los creadores de proyectos a identificar características relevantes para lograr alcanzar el financiamiento exitoso de sus proyectos.</p> <p>OE4</p>

2019	Cheng, Tan, Hou, Wei	Acta de conferencia	Success Prediction on Crowdfunding with Multimodal Deep Learning	Estudiar la influencia de interacciones sofisticadas entre modalidades textuales, visuales y metainformación en la predicción de éxito de proyectos.	Investigar esquemas de fusión con diferentes modalidades y evaluar arquitectura multimodal en el con- junto de datos de crowdfunding. OE1
2019	Chen, Shen	Acta de conferencia	Finding the Keywords Affecting the Success of Crowdfunding Projects	Analizar el impacto del contenido textual en un proyecto de Kickstarter a partir del análisis de sus palabras clave que determinen el éxito de financiamiento.	Investigar la contribución de imáge- nes al éxito del proyecto. OE3
2019	Chaichi, Anderson	Acta de conferencia	Deploying Natural Language Processing to Extract Key Product Features of Crowdfunding Campaigns: The Case of 3D Printing Technologies on Kickstarter	Analizar impacto de la predicción temprana en los resultados. OE4	Analizar alternativas de clasifica- ción a partir de distinta selección de características en trabajos previos. OE1

2019	Shafqat, Byun	Artículo	Topic Predictions and Optimized Recommendation Mechanism Based on Integrated Topic Modeling and Deep Neural Networks in Crowdfunding Platforms	Desarrollar un sistema integrado de recomendación de proyectos de crowdfunding a partir del análisis textual de sus comentarios.	Ayudar a encontrar proyectos seguros a inversores a partir de sistema de recomendación. OE4
	Fernández-Blanco, Villanueva-				Diseñar proceso de arquitectura integrada de modelos de recomendación y de predicción de tendencia de temas de comentarios.
	Balsera, Rodriguez-Montequin, Moran-Palacios	Artículo	Key Factors for Project Crowdfunding Success: An Empirical Study	Identificar atributos de proyectos financiados exitosamente y definir estereotipos de comportamiento que puedan estar asociados a nuevos proyectos.	Evaluando ambiente de implementación y experimentación de modelos propuestos. OE2
					Determinar la relación entre la distribución de variables entre clústeres.
					Estimar qué grupo potencial desarrollado sería el resultado de un nuevo proyecto.
					Ayudar al creador a definir una estrategia o reorientar un proyecto para llevarlo al éxito, en función de su posición en el sistema. OE4

F Parámetros para modelo predictivo de Metainformación

Hiperparámetro	Valor
Pesos de clases	<ul style="list-style-type: none"> – Exitoso (1): 1.7581290322580645 – Fracasado (0): 0.6987077585764833
Optimizador	<ul style="list-style-type: none"> – Nombre: ADAM – Ratio de aprendizaje: $5 * 10^{-3}$ – Ratio de decaimiento: $1 * 10^{-5}$
Parada temprana	<ul style="list-style-type: none"> – Monitor: <i>val_accuracy</i> – Modo: <i>auto</i> – Paciencia: 10
Reducción de ratio de aprendizaje	<ul style="list-style-type: none"> – Monitor: <i>val_accuracy</i> – Modo: <i>max</i> – Paciencia: 3 – Factor: 0.1
Punto de guardado del modelo	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i>
Función de pérdida	<i>binary_crossentropy</i>
Número de épocas	100
Tamaño de lote	128
Datos de entrada	<i>X_train</i>
Datos de destino	<i>y_train</i>
Datos de validación	<i>X_test</i> , <i>y_test</i>
Longitud de datos de entrada - capa de entrada	6
Número de neuronas de salida - capa densa 1	32
Función de activación - capa densa 1	<i>relu</i>
Ratio de capa de desactivación 1	0.25
Número de neuronas de salida - capa densa 2	16
Función de activación - capa densa 2	<i>tanh</i>
Ratio de capa de desactivación 2	0.30
Función de activación - capa final	<i>sigmoid</i>

G Parámetros para modelo predictivo de Descripción

Hiperparámetro	Valor
Pesos de clases	<ul style="list-style-type: none"> – Exitoso (1): 1.7581290322580645 – Fracasado (0): 0.6987077585764833
Optimizador	<ul style="list-style-type: none"> – Nombre: ADAM – Ratio de aprendizaje: $1 * 10^{-5}$ – Ratio de decaimiento: $1 * 10^{-5}$
Parada temprana	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i> – Paciencia: 10
Reducción de ratio de aprendizaje	<ul style="list-style-type: none"> – Monitor: <i>val_accuracy</i> – Modo: <i>max</i> – Paciencia: 5 – Factor: 0.01
Punto de guardado del modelo	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i>
Función de pérdida	<i>binary_crossentropy</i>
Número de épocas	100
Tamaño de lote	128
Datos de entrada	<i>training_sentences</i>
Datos de destino	<i>training_labels</i>
Datos de validación	<i>testing_sentences</i> , <i>testing_labels</i>
Longitud de datos de entrada - capa de entrada	3,671
Tamaño de vocabulario - capa Embedding	148,270
Dimensión de incrustación - capa Embedding	100
Número de filtros - capa Conv1D	128
Tamaño de kernel - capa Conv1D	5
Número de neuronas de salida - capa densa 1	64
Función de activación - capa densa 1	<i>tanh</i>
Función de activación - capa final	<i>sigmoid</i>

H Parámetros para modelo predictivo de Comentarios

Hiperparámetro	Valor
Pesos de clases	<ul style="list-style-type: none"> – Exitoso (1): 1.7581290322580645 – Fracasado (0): 0.6987077585764833
Optimizador	<ul style="list-style-type: none"> – Nombre: ADAM – Ratio de aprendizaje: $1 * 10^{-3}$ – Ratio de decaimiento: $1 * 10^{-6}$
Parada temprana	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i> – Paciencia: 10
Reducción de ratio de aprendizaje	<ul style="list-style-type: none"> – Monitor: <i>val_accuracy</i> – Modo: <i>max</i> – Paciencia: 5 – Factor: 0.01
Punto de guardado del modelo	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i>
Función de pérdida	<i>binary_crossentropy</i>
Número de épocas	50
Tamaño de lote	64
Datos de entrada	<i>training_sentences</i>
Datos de destino	<i>training_labels</i>
Datos de validación	<i>testing_sentences</i> , <i>testing_labels</i>
Longitud de datos de entrada - capa de entrada	5,000
Tamaño de vocabulario - capa Embedding	74,096
Dimensión de incrustación - capa Embedding	100
Ratio de capa de desactivación	0.50
Número de neuronas de salida - capa Bidireccional LSTM	128
Función de activación - capa final	<i>sigmoid</i>

I Parámetros para modelo de Aprendizaje Profundo Multimodal

Hiperparámetro	Valor
Pesos de clases	<ul style="list-style-type: none"> – Exitoso (1): 1.7581290322580645 – Fracasado (0): 0.6987077585764833
Optimizador	<ul style="list-style-type: none"> – Nombre: ADAM – Ratio de aprendizaje: $1 * 10^{-5}$ – Ratio de decaimiento: $1 * 10^{-5}$
Parada temprana	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i> – Paciencia: 10
Reducción de ratio de aprendizaje	<ul style="list-style-type: none"> – Monitor: <i>val_accuracy</i> – Modo: <i>max</i> – Paciencia: 5 – Factor: 0.01
Punto de guardado del modelo	<ul style="list-style-type: none"> – Monitor: <i>val_loss</i> – Modo: <i>min</i>
Función de pérdida	<i>binary_crossentropy</i>
Número de épocas	200
Tamaño de lote	32 ^a
Datos de entrada	<i>X_train_stacked</i> ^b
Datos de destino	<i>y_train</i>
Datos de validación	<i>X_test_stacked</i> ^c , <i>y_test</i>
Longitud de datos de entrada ^d	3,671, 5,000, 6
Número de neuronas de entrada - capa concatenada ^e	1, 1, 1
Número de neuronas de salida - capa concatenada	3
Número de neuronas de salida - capa densa 1	10
Función de activación - capa densa 1	<i>relu</i>
Función de activación - capa final	<i>sigmoid</i>

^a Al no especificarse, por defecto se asignó el valor de 32 (Keras, s.f.); ^b comprende la concatenación de *X_train_description*, *X_train_comments* y *X_train_metadata*; ^c comprende la concatenación de *X_test_description*, *X_test_comments* y *X_test_metadata*; ^d representan las longitudes de datos de entrada de cada modalidad; ^e representan las salidas de cada modalidad