

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA



**PONTIFICIA
UNIVERSIDAD
CATÓLICA
DEL PERÚ**

Desarrollo de un software para la gestión de proyectos de desarrollo PM4R que permita automatizar la etapa de planificación para estimar tiempos a través de algoritmos de aprendizaje automático.

Tesis Para optar por el Título de Ingeniero Informático que presenta el bachiller:

Jorge Enrique Espinoza Muñoz

Asesor: Doc. Luis Alberto Flores García

Lima, abril de 2019

Resumen

La gestión de proyectos es un trabajo arduo que demanda de experiencia en el sector para la correcta definición de actividades y estimación de costos y tiempos de los proyectos. Es por ello que PM4R o gestión para resultados propone 7 etapas integradas. En este contexto, esta metodología es potente al permitir tener una ruta en cómo realizar una buena gestión, pero la habilidad de estimación aún queda definida por los gestores de proyectos, los que en base a su experiencia definen los tiempos de los proyectos y en consecuencia los costos. ¿Qué pasaría si este proceso estuviera automatizado?

Ante esta pregunta, en la presente tesis se desarrolla un software para esta metodología. Sin embargo, como componente adicional se integran algoritmos de *Machine Learning* para permitir una estimación de la duración de los proyectos (tiempos) haciendo uso de la data histórica, para lo cual se procede a generar data en base a un conjunto de datos público del Banco Mundial. Además de que el diseño del software se basa en los lineamientos de Nielsen, lo que permitió la publicación de un artículo en el rubro de Interacción Humano-Computador.

En base a lo anterior, los resultados muestran una herramienta potente que ha integrado la metodología en su totalidad (7 pasos) y además los resultados de la integración con *Machine Learning* son prometedores ya que tanto la clasificación como la regresión presentan valores de error acorde a la revisión del estado del arte y quedan pendientes trabajos como el tuneo de los algoritmos para la obtención de mejores resultados. Además de que a partir de la evaluación del incremento de los datos se observan resultados en los que, si bien es cierto se incrementa la precisión, también se incluye ruido o desbalanceo. Es un *trade-off* que el gestor de proyectos puede utilizar a su favor, ya que el software en cuestión es una herramienta de apoyo para los gestores, de tal manera que estos puedan tomar mejores decisiones en la etapa de planificación de nuevos proyectos.



Dedicatoria

A mis padres Jorge Espinoza Fernández y Avelinda Muñoz Mego y
a mi hermano Carlos Espinoza Muñoz por su apoyo constante
y por siempre confiar en mí.

Tabla de Contenido

Tabla de Contenido.....	iii
Índice de Figuras.....	viii
Índice de Tablas.....	x
Capítulo 1. Generalidades.....	1
1.1 Problemática.....	1
1.2 Objetivos.....	4
1.2.1 Objetivo general.....	4
1.2.2 Objetivos específicos.....	4
1.2.3 Resultados esperados.....	5
1.3 Herramientas y Métodos.....	7
1.3.1 Herramientas.....	9
1.3.2 Métodos/ Metodologías.....	11
1.4 Alcance, Limitaciones y Riesgos.....	18
1.4.1 Alcance.....	18
1.4.2 Limitaciones.....	20
1.4.3 Riesgos.....	20
1.5 Viabilidad.....	21
1.5.1 Viabilidad Técnica.....	21
1.5.2 Viabilidad Temporal.....	21
1.5.3 Viabilidad Económica.....	23
1.5.4 Conclusiones de Viabilidad.....	23
Capítulo 2. Marco Conceptual.....	24
2.1 Objetivos del marco conceptual.....	24
2.2 Definiciones.....	24
2.2.1 Proyectos.....	24
2.2.2 Gestión de Proyectos.....	25

2.2.3	Proyectos de desarrollo	26
2.2.4	Gestión de proyectos de desarrollo.....	27
2.2.5	Data y Niveles de uso de la data.....	32
2.2.6	Herramientas de Gestión de Proyectos.....	33
2.2.7	Comportamiento Inteligente	35
Capítulo 3.	Estado del Arte	36
3.1	Revisión y discusión.....	36
3.2	Objetivos de la revisión del estado del arte	36
3.2.1	Herramientas	37
3.2.2	Literatura revisada	40
3.3	Conclusiones del estado del arte	43
Capítulo 4.	Especificación de Requisitos.....	44
4.1	Objetivo.....	44
4.2	Requerimientos funcionales.....	44
4.3	Requisitos no funcionales	44
4.4	Rango de valoración	44
4.5	Verificación del resultado	44
4.5.1	Consistencia	44
4.5.2	Trazabilidad	44
Capítulo 5.	Diseño de interfaces	45
5.1	Objetivo.....	45
5.2	Esquema general.....	45
5.3	Fuentes.....	46
5.4	Paleta de colores	46
5.5	<i>Mockups</i> y flujo base.....	47
5.6	Librerías.....	47

5.7	Verificación de resultado.....	47
Capítulo 6.	Diseño de arquitectura.....	48
6.1	Objetivo.....	48
6.2	Metas y Restricciones.....	48
6.2.1	Metas.....	48
6.2.2	Restricciones.....	48
6.3	Diagramas.....	49
6.3.1	Diagrama de componentes (Alto Nivel).....	49
6.3.2	Diagrama de componentes (Bajo Nivel).....	49
6.3.3	Diagrama de despliegue.....	50
6.3.4	Diagrama de clases.....	51
6.3.5	Diagrama Entidad Relación (DER).....	51
6.4	Justificación de la arquitectura.....	52
6.5	Verificación de resultado.....	52
Capítulo 7.	Implementación de la plataforma.....	53
7.1	Objetivo.....	53
7.2	<i>Back-End</i>	53
7.3	<i>Front-End</i>	53
7.4	Verificación del resultado.....	54
7.4.1	Pruebas.....	54
1.	Pruebas unitarias.....	54
2.	Pruebas de integración.....	54
Capítulo 8.	Generación y almacenamiento de datos.....	55
8.1	Objetivo.....	55
8.2	Módulo generador de datos.....	55
8.3	Verificación del resultado.....	57

Capítulo 9. Definición de atributos y pre-procesamiento de datos	58
9.1 Objetivo.....	58
9.2 Atributos.....	58
9.3 Pre-procesamiento de datos	58
9.3.1 Actividades de pre-procesamiento	58
9.3.2 Diagrama de flujo.....	60
9.4 Verificación del resultado	60
Capítulo 10. Algoritmo de clustering.....	61
10.1 Objetivo	61
10.2 Diagrama de flujo	61
10.3 Verificación del resultado.....	62
Capítulo 11. Módulo predictor.....	63
11.1 Objetivo	63
11.2 Clasificación	63
11.2.1 Diagrama de flujo.....	64
11.3 Regresión.....	64
11.3.1 Diagrama de flujo.....	65
11.4 Verificación del resultado.....	66
11.4.1 Clasificación.....	66
11.4.2 Regresión	66
Capítulo 12. Análisis del error.....	67
12.1 Objetivo	67
12.2 Clasificación	67
12.2.1 Métricas	67
1. F1 score.....	67
2. Accuracy.....	68
12.2.2 Comparación de ambas métricas.....	68

12.2.3	Evolución de las métricas con el incremento de data	69
12.3	Regresión	69
12.3.1	Métricas	69
1.	RMSLE	69
12.4	Evolución de las métricas con el incremento de data.....	70
12.5	Verificación del resultado.....	71
Capítulo 13.	Integración del módulo generador de modelos dinámicos con la plataforma 72	
13.1	Objetivo	72
13.2	Integración <i>Back-End</i>	72
13.3	Integración <i>Front-End</i>	74
13.4	Verificación del resultado.....	75
13.4.1	Pruebas	75
1.	Pruebas unitarias.....	75
2.	Pruebas de integración	75
Capítulo 14.	Conclusiones y trabajos futuros.....	76
14.1	Conclusiones.....	76
14.2	Trabajos futuros.....	78
Referencias.....		79

Índice de Figuras

<i>Figura 1. Representación del árbol de problemas detallado en la problemática. Elaboración propia.</i>	3
<i>Figura 2. Algoritmo de EM-Clustering. Tomado de (Herbold, 2013).</i>	13
<i>Figura 3. Representación de los grupos de procesos de Gestión de Proyectos y la separación en dos fases. Adaptado de (PMI, 2018; Wallace, 2011).</i>	25
<i>Figura 4. Representación de estrategias de desarrollo conjuntas. Adaptado de (Siles & Mondelo, 2015)</i>	26
<i>Figura 5. Elementos de la planificación orientada a resultados. Adaptado de (García López & García Moreno, 2010).</i>	28
<i>Figura 6. Elementos del EDT para el Programa Reconversión Urbana. Adaptado de (Yamanaka, 2015)</i>	29
<i>Figura 7. Cronograma para el Programa Reconversión Urbana. Tomado de (Yamanaka, 2015)</i>	30
<i>Figura 8. Curva S para el Programa Reconversión Urbana. Tomado de (Yamanaka, 2015)</i>	30
<i>Figura 9. Matriz de Adquisiciones para el Programa Reconversión Urbana. Tomado de (Yamanaka, 2015)</i>	31
<i>Figura 10. Matriz de riesgos para el Programa Reconversión Urbana. Tomado de (Yamanaka, 2015)</i>	31
<i>Figura 11. Matriz de comunicaciones para el Programa Reconversión Urbana. Tomado de (Yamanaka, 2015)</i>	32
<i>Figura 12. Matriz de asignación de responsabilidades para el Programa Reconversión Urbana. Tomado de (Yamanaka, 2015).</i>	32
<i>Figura 13. Análisis de riesgo. Adaptado de (Stratejos, 2018).</i>	38
<i>Figura 14. Mensajes & eventos. Adaptado de (Stratejos, 2018).</i>	38
<i>Figura 15. Estimación del tiempo. Adaptado de (Forecast, 2018).</i>	39
<i>Figura 16. Estimación del costo. Adaptado de (Forecast, 2018)</i>	39

<i>Figura 17. RBd. Tomado de (Aptage, 2017).</i>	39
<i>Figura 18. Interfaz de usuario. Tomado de (Saglam, 2017).</i>	42
<i>Figura 19. Paleta de colores. Tomado de (Angular Material, 2018).</i>	46
<i>Figura 20. Diagrama de componentes (Alto Nivel). Elaboración propia.</i>	49
<i>Figura 21. Diagrama de componentes (Bajo Nivel). Elaboración propia.</i>	49
<i>Figura 22. Diagrama de despliegue. Elaboración propia.</i>	50
<i>Figura 23. Diagrama de flujo – pre-procesamiento. Elaboración propia.</i>	60
<i>Figura 24. Diagrama de flujo – clustering parte 1. Elaboración propia.</i>	61
<i>Figura 25. Diagrama de flujo – clustering parte 2. Elaboración propia.</i>	62
<i>Figura 26. Diagrama de flujo - clasificación. Elaboración propia.</i>	64
<i>Figura 27. Diagrama de flujo - regresión. Elaboración propia.</i>	65
<i>Figura 28. Evolución de F1score y Accuracy al incrementar la data. Elaboración propia.</i>	69
<i>Figura 29. Evolución de RMSLE al incrementar la data. Elaboración propia.</i>	71
<i>Figura 30. Clases adicionales. Elaboración propia.</i>	72

Índice de Tablas

<i>Tabla 1. Resultados esperados para el objetivo 1. Elaboración propia.....</i>	<i>5</i>
<i>Tabla 2. Resultados esperados para el objetivo 2. Elaboración propia.....</i>	<i>6</i>
<i>Tabla 3. Resultados esperados para el objetivo 3. Elaboración propia.....</i>	<i>7</i>
<i>Tabla 4. Mapeo de Resultados esperados con Herramientas y Métodos. Elaboración propia.....</i>	<i>7</i>
<i>Tabla 5. Identificación de riesgos. Elaboración propia.....</i>	<i>20</i>
<i>Tabla 6. Actividades del proyecto. Elaboración propia.</i>	<i>21</i>
<i>Tabla 7. Costos de despliegue estimados. Elaboración propia.....</i>	<i>23</i>
<i>Tabla 8. Pasos para la planificación y gestión de proyectos de desarrollo. Adaptado de (Siles & Mondelo, 2015).</i>	<i>29</i>
<i>Tabla 9. Clasificación de herramientas de gestión de proyectos actuales. Elaboración propia.....</i>	<i>34</i>
<i>Tabla 10. Esquema general. Elaboración propia.....</i>	<i>45</i>
<i>Tabla 11. Fuentes. Elaboración propia.....</i>	<i>46</i>
<i>Tabla 12. Score obtenido con la métrica F1 score. Elaboración propia.</i>	<i>67</i>
<i>Tabla 13. Error obtenido con la técnica Cross-validation y kfold igual a 10. Elaboración propia.....</i>	<i>68</i>
<i>Tabla 14. Error obtenido con la métrica RMSLE y la variación de algunos parámetros. Elaboración propia.</i>	<i>70</i>

Capítulo 1. Generalidades

1.1 Problemática

Un proyecto, según el PMI (*Project Management Institute*), es un esfuerzo con un inicio y un fin para lograr un resultado específico (PMI, 2017). La planificación, ejecución y monitoreo del mismo es denominada gestión de proyectos (PM por sus siglas en inglés) y esta ha incrementado su complejidad al mismo tiempo que los proyectos y las técnicas usadas (Wallace, 2011). Ante ello, herramientas tecnológicas (software a través de aplicaciones de escritorio, plataformas web, etc.) fueron creadas y continúan en proceso de desarrollo para facilitar y garantizar su cumplimiento (Wallace, 2011).

Un tipo específico de proyectos, son los proyectos de desarrollo, que se definen como vehículos para que un país o región logre mejorar sus índices de desarrollo y que buscan resultados concretos y sostenibles a largo plazo, para los cuales se presenta a PM4R o gestión para resultados como una metodología para asegurar el logro de sus objetivos específicos (Siles & Mondelo, 2015).

Es cierto que existen herramientas conocidas en el mercado que apoyan en términos generales la gestión de proyectos, por ejemplo la última versión de MS Project, que brinda soporte a procesos como la administración de recursos, seguimiento y reportes (Chatfield & Johnson, 2016). No obstante, estas no ofrecen todos los instrumentos integrados que propone el PM4R, que son: EDT, Cronograma, Curva S, Matriz de adquisiciones, Matriz de riesgos, Matriz de comunicaciones y Matriz de Responsabilidades (Siles & Mondelo, 2015), en este contexto, MS Project no cuenta con una sección para definir la Curva S o una Matriz de adquisiciones.

Debido a la ausencia de software para esta metodología, se evidencia que los instrumentos previamente mencionados se desarrollan a través de software de ofimática como Excel (Nuñez, 2014; Yamanaka, 2015; Herrera, comunicación personal, 14 de mayo de 2018). A partir de ello, se infiere que cada proyecto requiere la creación de un nuevo documento con sus especificaciones, de manera aislada respecto a los proyectos anteriores. Por tanto, el problema es no reutilizar los datos históricos a través de un comportamiento inteligente, sino solo a nivel de la experiencia de los gestores (Chang et al., 2008), lo que conduce a inconvenientes que se mencionarán posteriormente.

Un comportamiento es inteligente cuando cumple con dos requisitos: criterio de efectividad, y el mismo criterio aplicado en contextos variados (como se citó en Bueno Cuadra, 2008). Este se logra a través de sistemas inteligentes basados en algoritmos de aprendizaje automático, que a su vez requieren una recolección de datos representativos (como se citó en Serna, 2017). Los algoritmos de aprendizaje automático son aquellos que construyen modelos para hacer predicciones o tomar decisiones (como se citó en Monleon-Getino, 2016) y los datos mencionados, aunque por sí mismos no tiene valor, con un correcto ordenamiento y procesamiento, aportan conocimiento (Monleón-Getino, 2010) y esto no es un concepto nuevo. Gibson & Nolan (1974), plantean que el nivel de “Madurez” en el uso de los datos está relacionado a sistemas basados en conocimientos integrados con sistemas expertos para permitir a las empresas tener modelos de simulación para obtener beneficios económicos (como se citó en Trujillo, 2012). Esto se ve reforzado con la creación de datos en mayor proporción con el paso del tiempo, y la existencia de dispositivos más eficientes para almacenarla (Monleon-Getino, 2016).

Como ya se mencionó, una de las etapas de la gestión es la planificación y esta requiere de estimaciones de tiempos y costos. El primero, acorde a un ciclo de gobierno, si es un proyecto de una entidad estatal, o a fechas límites, en cambio el segundo, en base a un presupuesto total aprobado (Siles & Mondelo, 2015). En PM4R, esto se define en el Cronograma (estimado de tiempo en días), la Curva S (costos por mes) y la Matriz de adquisiciones (presupuesto estimado). Entonces, para una adecuada planificación se debe aprovechar los datos de los costos y tiempos de proyectos pasados, como factores para la toma de decisiones, caso contrario, uno de los inconvenientes será la propagación del error en cada ciclo de desarrollo de un proyecto, como la sobreestimación de costos y tiempos (Hanchate & Bichkar, 2015).

Para la etapa de ejecución y monitoreo, se deben considerar otros procedimientos como la asignación de recursos humanos o responsables de las tareas (Chang & Christensen, 1999; Chang et al., 2001, 2008; Joslin & Poole, 2005), los que deben optimizar el uso eficiente del personal y sus horas laborables. En PM4R, el instrumento que abarca este punto es la Matriz de Responsabilidades, la cual conecta el organigrama con el instrumento EDT (Siles & Mondelo, 2015). Entonces, para una adecuada ejecución y monitoreo, se debe establecer cursos de acción cuando se presentan anomalías (Baia, 2015), en base a factores como el tiempo laborable, tiempo máximo de sobre-trabajo,

salarios del personal, habilidades, entre otros (Chang & Christensen, 1999; Chang, Christensen, & Zhang, 2001; Chang, Jiang, Di, Zhu, & Ge, 2008).

Todas estas etapas expuestas anteriormente se vuelven más complejas cuando los proyectos son más grandes y por tanto requieren de un procesamiento mayor de los datos para lograr el comportamiento inteligente (efectividad en contextos variados), lo que no se puede realizar solo con la estimación análoga propuesta por PM4R (que se sustenta en revisar proyectos similares) ya que continúa siendo un método no automatizado (Siles & Mondelo, 2015).

Por último, con el paso del tiempo se buscan mejores métodos para generar un comportamiento inteligente y las soluciones están orientadas a la integración de los conceptos de datos más Inteligencia Artificial (Chang & Christensen, 1999; Chang et al., 2001, 2008; Hanchate & Bichkar, 2015; Herbold, 2013; Hosley, 1985; Joslin & Poole, 2005; Saglam, 2017). Con este marco, la pregunta rectora es: ¿De qué manera se puede automatizar la gestión de proyectos de desarrollo (planificación, ejecución y monitoreo) a través de un comportamiento inteligente? (Ver Figura 1).

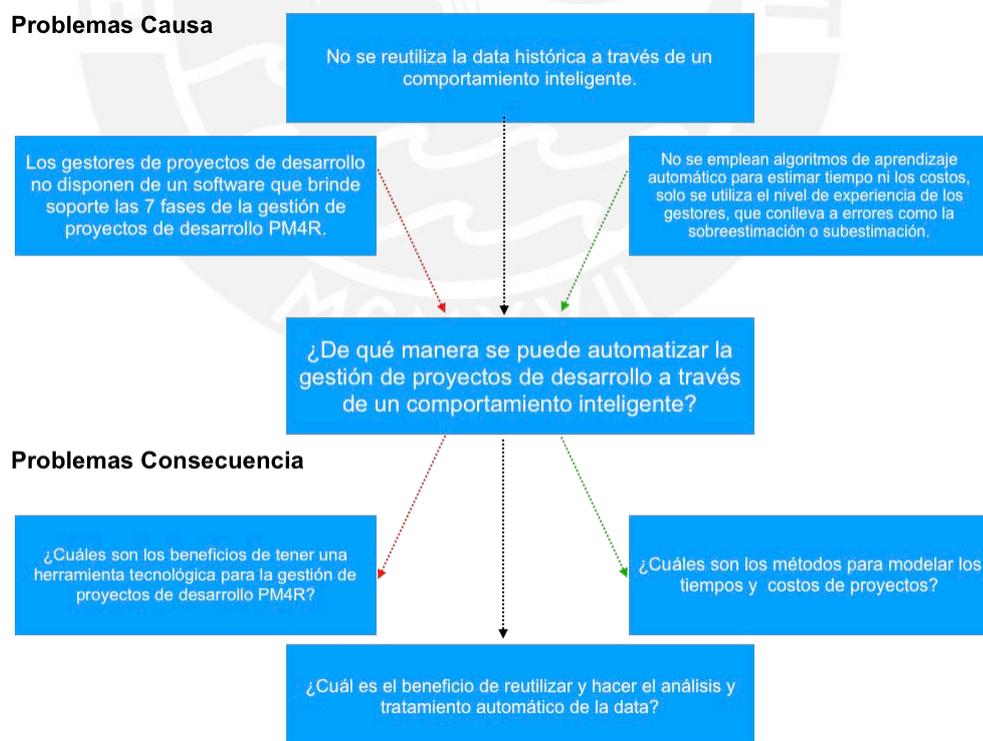


Figura 1. Representación del árbol de problemas detallado en la problemática.

Elaboración propia.

1.2 Objetivos

1.2.1 Objetivo general

Desarrollar un software para la gestión de proyectos de desarrollo PM4R que permita automatizar la etapa de planificación para estimar tiempos a través de algoritmos de aprendizaje automático.

1.2.2 Objetivos específicos

Se definen los objetivos específicos en el orden cronológico de su desarrollo:

- O 1. Desarrollar una plataforma web que cubra las 7 fases de la gestión de proyectos de desarrollo PM4R.**

La plataforma debe automatizar el proceso de la gestión de proyectos de desarrollo de modo que los gestores dispongan de una herramienta específica con los 7 instrumentos de PM4R.

- O 2. Implementar un módulo generador de modelos dinámicos para la estimación computacional de tiempos en proyectos de desarrollo.**

La propuesta inicial se basa en aprendizaje automático a través clasificación y regresión, ambos en base a una clusterización previa a través de *K-means* y *EM*. Para el primero se hará uso de Regresión logística, *Naive Bayes*, *SVM* con *RBF kernel*, árboles de decisión, *RandomForest* y perceptrón multicapa. Por otro lado, para la regresión, los algoritmos de *RandomForest Regressor* y *Catboost*. Se medirá el éxito a través del análisis del *f1-score* y *accuracy* obtenido por *cross-validation* para la clasificación y *RMSLE* para la regresión, ya que es necesario evaluar la precisión de los mismos.

- O 3. Integrar en la plataforma el módulo generador de modelos dinámicos que permita estimar computacionalmente los tiempos en proyectos de desarrollo.**

Se busca integrar el módulo desarrollado en el objetivo específico 2 con el software para la gestión de proyectos PM4R desarrollado en el objetivo específico 1.

1.2.3 Resultados esperados

Se realiza el mapeo entre objetivos, resultados y el medio de verificación (Ver Tablas 1, 2 y 3).

Tabla 1. Resultados esperados para el objetivo 1. Elaboración propia.

Objetivo: O 1. Desarrollar una plataforma web que cubra las 7 fases de la gestión de proyectos de desarrollo PM4R.		
Resultado	Meta física	Medio de verificación
1.1 Especificación de Requisitos	Catálogo de requisitos funcionales y no funcionales	- Verificar la consistencia y trazabilidad de los requisitos.
1.2 Diseño de interfaces	Estándares de diseño y los <i>mockups</i> con un flujo definido.	- Verificación de los lineamientos de Usabilidad de Nielsen.
1.3 Diseño de arquitectura	Diagrama de componentes (alto y bajo nivel), diagrama de despliegue, DER y diagrama de clases.	- Verificar un comportamiento deseado de la Arquitectura Cliente-Servidor a través del reporte técnico de <i>endpoints</i> . - Verificación del cumplimiento de estándares UML.
1.4 Implementación de la plataforma	Software	- Verificación del software, en los requerimientos que se establezcan como prioritarios, a través de pruebas unitarias y pruebas de integración.

Tabla 2. Resultados esperados para el objetivo 2. Elaboración propia.

Objetivo: O 2. Implementar un módulo para generar modelos dinámicos para la estimación computacional de tiempos en proyectos de desarrollo.		
Resultado	Meta física	Medio de verificación
2.1 Generación y almacenamiento de datos.	Bloque de datos ingresados a la plataforma (diferentes proyectos de desarrollo).	<ul style="list-style-type: none"> - Verificar si se logró generar más de 50 proyectos según la guía para escoger el mejor predictor (scikit-learn developers (BSD License), 2016).
2.2 Definición de los atributos y pre-procesamiento de los datos.	Detalle técnico de los atributos a ser usados para los algoritmos de ML y técnicas de pre-procesamiento.	<ul style="list-style-type: none"> - Verificar que se ha realizado una limpieza según (Alatrística Salas, 2016).
2.3 Algoritmo de segmentación o clusterización implementado.	Diagrama de flujo y código fuente	<ul style="list-style-type: none"> - Evaluación de cohesión (SSW) y separación (SSB).
2.4 Módulo generador de modelos dinámicos.	Diagrama de flujo y código fuente	<ul style="list-style-type: none"> - Evaluación de <i>recall</i> y <i>precision</i> a través del <i>f1-score</i>. - Verificación del módulo a través de pruebas unitarias.
2.5 Análisis del error sobre una muestra de proyectos de prueba.	<p>Tablas comparativas del error asumido en las estimaciones.</p> <p>Gráficos de evolución de las métricas con la variación de la cantidad de muestras.</p>	<ul style="list-style-type: none"> - Evaluación del éxito de la clasificación a través del análisis del <i>f1-score</i> y <i>accuracy</i> obtenido por <i>cross-validation</i>. - Evaluación del éxito de la regresión a través del análisis del RMSLE.

Tabla 3. Resultados esperados para el objetivo 3. Elaboración propia.

Objetivo: O 3. Integrar en la plataforma el módulo generador de modelos dinámicos que permita estimar computacionalmente los tiempos en proyectos de desarrollo.		
Resultado	Meta física	Medio de verificación
3.3 Plataforma integrada con el módulo para la generación de modelos dinámicos.	Software.	<ul style="list-style-type: none"> - Verificación del módulo a través de pruebas unitarias. - Verificación del módulo a través de pruebas de integración. - Aprobación de un juicio experto de un profesional en PM con doctorado en Gestión Estratégica.

1.3 Herramientas y Métodos

Para cada resultado se presentan las herramientas y/o métodos utilizados (Ver Tabla 4). Posteriormente se detallan cada una de estas.

Tabla 4. Mapeo de Resultados esperados con Herramientas y Métodos. Elaboración propia.

Resultado	Herramientas	Métodos/Methodologías
1.1 Especificación de Requisitos		<ul style="list-style-type: none"> - Metodología tradicional de desarrollo de software
1.2 Diseño de interfaces	<ul style="list-style-type: none"> - Sketch - Componentes Angular Material 	<ul style="list-style-type: none"> - Lineamientos de usabilidad de Nielsen
1.3 Diseño de arquitectura	<ul style="list-style-type: none"> - StarUML 	

1.4 Implementación de la plataforma	<ul style="list-style-type: none"> - Pycharm con licencia educacional - Django Framework - Angular 6.0 Framework - Repositorio Gitlab 	<ul style="list-style-type: none"> - Metodología tradicional de desarrollo de software
2.1 Generación y almacenamiento de datos de distintas fuentes.	<ul style="list-style-type: none"> - Pg. Admin 4.0 	
2.2 Definición de atributos y pre-procesamiento de los datos.	<ul style="list-style-type: none"> - Pycharm con licencia educacional 	<ul style="list-style-type: none"> - Tareas en el pre-procesamiento (Alatriza Salas, 2016).
2.3 Algoritmo de segmentación o <i>clustering</i> implementado.	<ul style="list-style-type: none"> - Jupyter Notebook 5.4.0 - Librería <i>scikit-learn</i>. 	<ul style="list-style-type: none"> - Clustering. - SSW y SSB.
2.4 Módulo generador de modelos dinámicos.		<ul style="list-style-type: none"> - Estimación o predicción. - <i>Recall, precisión y F1 score</i> - RMSE y RMSLE
2.5 Análisis del error sobre una muestra de proyectos de prueba.	<ul style="list-style-type: none"> - Jupyter Notebook 5.4.0 	<ul style="list-style-type: none"> - <i>F1 score</i> - <i>Cross-validation</i> - RMSE y RMSLE
3.3 Plataforma integrada con el módulo para la generación de modelos dinámicos.	<ul style="list-style-type: none"> - Pycharm con licencia educacional. 	

1.3.1 Herramientas

1. Sketch

Herramienta de prototipado para MacOS que se utilizará para el diseño de interfaces porque permite añadir *pluggins* de estilos adicionales tales como Google Material y mantener pizarras (*artboards*) en base a especificaciones de tamaños. Cuenta con todas las herramientas para insertar, editar y cortar componentes además de proveer reglas, rejillas y formas eficientes de alinear dos componentes (Sketch, 2018).

2. Componentes Material

Framework front-end, el cual es usado en distintas herramientas actuales según la revisión del estado del arte. Dentro de sus características están: Acelerar el proceso de desarrollo por la incorporación de animaciones y estilos de componentes, centrado en la experiencia de usuario (Google, 2018).

Por otra parte, Angular provee una integración con estos componentes a través de Angular Material, el cual no solo cuenta con estilos CSS sino también SASS. Además de que estos son responsivos y se pueden adaptar a los diferentes tamaños de los dispositivos (Angular Material, 2018).

3. StarUML

Software para el modelado de diagramas UML 2.0 (Lenguaje unificado de modelado) , que será usado porque provee de estándares internacionalmente aceptados, algunos de estos son (StarUML, 2018): Diagrama de clases, diagrama de componentes, diagrama de despliegue, diagrama de secuencia, diagrama de comunicación, diagrama de estados, diagrama de actividades y diagrama de entidad relación

Para este proyecto, los diagramas de interés son: el diagrama de clases, diagrama entidad relación, diagrama de componentes y diagrama de despliegue, porque permiten definir lo necesario para una implantación real.

El primero, cuenta con tres secciones: el nombre de la clase, los atributos y los métodos. Además, la cardinalidad entre clases está dada por uno a muchos, cero a muchos y por un número específico. Finalmente, las relaciones entre clases se da como herencia, agregación, asociación y dependencia (Salinas, n.d.). Asimismo, el segundo es una representación de entidades (las tablas generadas) y sus columnas (StarUML, 2018).

Los dos siguientes, se utilizarán para modelar la arquitectura del software, para lo cual se definen componentes y nodos como una representación de software y hardware con diferentes estereotipos (pc, pc-cliente, pc-servidor, almacenamiento, etc.), que generan productos o artefactos (código fuente, documento de diseño, reportes, etc.). Estos se asocian para representar una ruta de comunicación entre los nodos en tiempo de ejecución (Sparx Systems, 2018).

Por último, las características adicionales de StarUML son (StarUML, 2018): es multiplataforma (MacOS, Windows y Linux), tiene métodos de exportación y tiene extensiones como el generador de scripts una vez realizado el modelado.

4. Pycharm

Pycharm es un IDE (Entorno de desarrollo integrado) usado para la programación con Python, razón por la que se usará en este proyecto para el cumplimiento de la integración con los componentes de aprendizaje automático. Algunas de sus características son: proporciona una linterna para errores en la sintaxis del código al igual que un *debugger*, permite la creación de entornos virtuales (*virtual environments*) e integrar Django como *framework*, cuenta con un navegador por nombre de métodos o variables y tiene una integración con *Git* para el manejador de repositorios (JetBrains, 2010). La versión utilizada es la edición profesional con licencia para uso educacional.

5. Django Rest

Django Rest es un *framework* basado en Django, el cual implementa tres capas: modelos, vistas y *templates*, y se usará en este proyecto por las facilidades que brinda para servicios web. Algunas de sus características son: permite la creación de rutas con expresiones regulares y el mapeo objeto relacional (ORM) para la creación de bases de datos a partir de los modelos, proporciona un administrador para la creación de entidades, permite la integración de PostgreSQL y la ejecución de clases middleware. (Django, 2018)

6. Angular 6.0

Es un *framework front-end*, que facilita la interacción con el DOM, funciona como cliente a través de solicitudes Http. Permite el desarrollo multiplataforma y se basa en Typescript y Sass. La versión 6.0 es la más reciente y cuenta con una integración de componentes de *Material Design* (Fluin, 2018).

7. Pg. Admin 4.0

Plataforma de administración y desarrollo para PostgreSQL, a través de un entorno con una herramienta de consultas (queries) , administrador de sesiones, esquemas, vistas, *triggers*, llaves foráneas, procedimientos almacenados, etc. (The PgAdmin Development Team, 2018). Su uso se justifica debido a la creación de una base de datos relacional a través del diagrama de clases.

8. Jupyter Notebook 5.4.0

Denominación para iPython Notebooks, el cual presenta una interfaz web para la interacción con lenguaje Python. Este no es más que un JSON con una lista de celdas y con posibilidad de uso de librerías Python; una de las más significativas es el *ploteo* (Project Jupyter, 2018). Su uso se justifica en sus facilidades para probar las librerías previo a la integración con el software.

1.3.2 Métodos/Metodologías

1. Metodología Tradicional de desarrollo de software

Se basa en la planificación bajo una etapa intensa de análisis y diseño previo al desarrollo que es lo que se busca en este proyecto para evitar la dependencia continua de un *Project owner* como lo sería en una metodología ágil. Es por ello que se utiliza el modelo en cascada, del cual se aplicará las siguientes fases (P. Letelier, 2003):

- Establecer una lista de requerimientos funcionales y no funcionales con su priorización.
- Diseñar el software: centrado en la arquitectura.
- Implementar y realizar pruebas unitarias: desarrollo de los módulos necesarios.

2. Lineamientos de usabilidad de Nielsen

Se consideran los 10 principios de Nielsen porque son un conjunto de reglas que permiten desarrollar sistemas amigables para el usuario que es lo que se busca en este proyecto y son heurísticas porque son generalizables para evaluar los diseños no específicos, estos son (Nielsen, 1995):

- Visibilidad del estado del sistema: El sistema deberá mantener a los usuarios informados acerca de lo que está pasando, a través de una retroalimentación adecuado en un tiempo razonable.

- Relación entre el sistema y el mundo real: El sistema debe hablar el idioma de los usuarios, con palabras, frases y conceptos familiares para este, en vez de términos orientados al sistema. Seguirá las convenciones del mundo real, haciendo que la información aparezca en un orden natural y lógico.
- Control del usuario y libertad: Los usuarios a menudo eligen funciones del sistema por error y debe existir una "salida de emergencia" claramente marcada para salir del estado no deseado sin un dialogo extenso. Soporte para deshacer y rehacer.
- Consistencia y estándares: Los usuarios no deberían tener que preguntarse si diferentes palabras, situaciones o acciones significan lo mismo. Seguirá las convenciones de la plataforma.
- Prevención de errores: Se debe tener un diseño que prevenga la ocurrencia de un problema. Se debe presentar una confirmación al usuario antes de realizar una acción.
- Reconocer en lugar de recordar: Reducir la carga de memoria del usuario. Este no debe tener que recordar la información de una parte a otra. Deben ser visibles instrucciones de uso o fácilmente recuperables.
- Flexibilidad y eficiencia del uso: Aceleradores de interacción para el usuario experto. Permitir a los usuarios adaptar las acciones frecuentes.
- Diseño estético y minimalista: Los diálogos no deben contener información que es irrelevante o raramente necesaria.
- Ayuda para reconocer, diagnosticar y recuperarse de errores: Los mensajes de error deben ser expresados en un lenguaje sencillo (sin códigos), indica con precisión el problema, y constructivamente sugerir una solución.
- Ayuda y documentación: Podría ser necesario proporcionar ayuda y documentación.

3. Tareas en el pre-procesamiento

Dentro de las tareas propuestas, se toma en consideración las de formato para obtener los atributos definidos, limpieza como eliminar los campos nulos, transformación como

agregación, normalización y *discretización*, y muestreo para dividir el conjunto de datos en datos de entrenamiento y de prueba (Alatrística Salas, 2016).

4. Clustering

El *clustering* es un método de aprendizaje no supervisado, el cual a partir de un conjunto de datos iniciales genera grupos o clústeres de acuerdo a características o patrones identificados (León Guzmán, 2016). Tal como en Herbold (2013), a través de este método se busca obtener los proyectos que tienen las mismas características que el proyecto objetivo.

Las alternativas para la generación de clústeres, que competirán por ofrecer la menor distancia inter e *intra-cluster*, son los siguientes algoritmos (Herbold, 2013):

a. K-means

Este algoritmo parte de un número de clústeres conocidos (k), y es un proceso iterativo en el que cada punto se asigna a un grupo considerando la menor distancia entre este y el centroide del grupo, el cual además se actualiza con cada iteración.

Se utiliza la librería *sklearn.cluster* para importar *KMeans*.

b. EM-Clustering

El algoritmo de maximización esperada (Ver Figura 2) es parecido al de *K-means*, pero su importancia radica en la probabilidad de asignación. Se inicia con el desconocimiento de la cantidad de grupos y se generan las distribuciones gaussianas aleatoriamente, luego se computa la probabilidad de que un elemento pertenezca a esa distribución en el denominado Paso E (Mitchell, 2005).

$$P(c_l | x_i) = \text{probabilidad de que el elemento } x_i \text{ pertenezca al Cluster } c_l$$

Luego, se reajustan las distribuciones, en la media μ y covarianza que implica el teorema de Bayes $P(A|B) = (P(B|A) * P(A))/P(B)$ y se itera hasta la convergencia dada por un error σ (Mitchell, 2005).

```

Input: Candidate data sets  $S^{candidate}$ , target data set  $S^{target}$ 
 $S^{train} \leftarrow \emptyset$ ;
 $C^{all} \leftarrow \{C(S) : S \in S^{candidate} \cup \{S^{target}\}\}$ ;
 $EMCluster.createClusters(C^{all})$ ;
 $C^{train} \leftarrow EMCluster.getCluster(C(S^{target}))$ ;
 $S^{train} \leftarrow \{S \in S^{candidate} : C(S) \in C^{train}\}$ ;
return  $S^{train}$ ;

```

Figura 2. Algoritmo de EM-Clustering. Tomado de (Herbold, 2013).

Se utiliza la librería *sklearn* para importar *mixture.GaussianMixture*.

5. SSW y SSB

SSW (*Sum of Squared Within*), es una medida que evalúa la cohesión de los clústeres, es decir la distancia *intra-cluster* (entre los puntos de un clúster). Por otro lado, SSB (*Sum of Squared Between*), evalúa la distancia *inter-cluster* (entre clústeres) (León Guzmán, 2016). Se busca minimizar la distancia *intra-cluster* y maximizar la *inter-cluster* (Berzal, 2012). A partir de estas medidas se evalúan índices tal como el coeficiente de Silhouette, el cual tiene valores entre -1 y 1, desde un mal agrupamiento hasta uno bueno respectivamente (scikit-learn, 2017).

Se utiliza la librería *sklearn* para importar *metrics.silhouette_score*, para evaluar el coeficiente de Silhouette.

6. Estimación o predicción

a. Clasificación

Se utiliza la *discretización* de la duración de los proyectos como parte del pre-procesamiento a través de la agrupación de datos numéricos en nominales (categorías) (Alatrística Salas, 2016). Se discretiza en *n* categorías definidas por el gestor de proyectos, de tal manera que estas describan la distribución de los datos y permitan la generación de un modelo, por ejemplo, estas pueden ser: alta duración (*high*), duración media (*medium*) y baja duración (*low*).

En consecuencia, para estimar estas *n* categorías se presentan como alternativas, según la bibliografía revisada, los siguientes algoritmos: Regresión logística, *Naive Bayes*, *SVM* con *RBF* kernel, árboles de decisión, *RandomForest* y perceptrón multicapa (Herbold, 2013).

i. Regresión Logística

Permite estimar la relación entre una variable categórica o dependiente y variables independientes, con lo cual se obtiene la influencia de estas últimas en la probabilidad de ocurrencia de un suceso específico (Mercedes, n.d.).

Se utiliza la librería *sklearn.linear_model* para importar *LogisticRegression* y generar un modelo de Regresión logística con el parámetro *multiclas="multinomial"*, ya que se cuenta con más de dos valores a predecir.

ii. Naive Bayes

Utiliza el teorema de Bayes, asumiendo que los atributos tienen distribuciones independientes y por tanto es una multiplicación de probabilidades: $P(d|c_j) = P(d_1|c_j) * P(d_2|c_j) * \dots * P(d_n|c_j)$ (Fomby, 2008).

Dentro de sus beneficios está su rápido entrenamiento y clasificación, la no sensibilidad a atributos irrelevantes y que maneja data continua y discreta (Fomby, 2008).

Se utiliza la librería *sklearn.naive_bayes* para importar *GaussianNB*, y generar un modelo de *Naive Bayes*.

iii. SVM

Este modelo de clasificación no se centra en minimizar el error de entrenamiento sino del estructural. Es decir, busca seleccionar un hiperplano (objeto divisor) de separación equidistante a los puntos más cercanos a cada clase para conseguir un margen máximo. Los puntos que caen en la frontera de dichos márgenes se denominan vectores de soporte (Carmona Suárez, 2014).

Se utiliza la librería *sklearn.svm* para importar *SVC* y generar un modelo de *SVM* con `kernel="rbf"` según lo revisado en el estado del arte (Herbold, 2013).

iv. Árboles de decisión

A partir de un nodo inicial se conforma la subdivisión del espacio de muestreo donde cada nodo representa una clase que puede tomar el valor objetivo dependiendo de los valores de los atributos. Se clasifica cada muestra desde la raíz hacia las hojas pasando por algún test definido en los vértices (Rokach & Maimon, 2010).

Se utiliza la librería *sklearn.tree* para importar *DecisionTreeClassifier* y generar un modelo de árbol de decisión.

v. RandomForest Classifier

Considerado como meta clasificador porque utiliza árboles de decisión en varios segmentos de la data. Dentro de sus ventajas está el control sobre el *overfitting* del modelo y el uso del promedio para evaluar una correcta clasificación (scikit-learn developers (BSD License), n.d.).

Se utiliza la librería *sklearn.ensemble* para importar *RandomForestClassifier* y generar un modelo de *RandomForest*.

vi. Perceptrón multicapa

Es una red neuronal conformada por varias capas dentro las cuales se tiene la capa de entrada formada por neuronas que son las que reciben y propagan los atributos a la siguiente capa, por otro lado, las capas ocultas requieren procesamiento no lineal de los datos de las capas anteriores para generar una salida de las siguientes, finalmente la capa de salida donde cada neurona representa una clase (Galván, Valls, Aler, & Huertas, 2017).

Se utiliza la librería *sklearn.neural_network* para importar *MLPClassifier* y generar un modelo de perceptrón multicapa con un capa oculta empezando con el número de 10 neuronas según lo revisado en el “Diseño de un PM para abordar un problema” (Galván et al., 2017).

b. Regresión

Para estimar las duraciones de los proyectos sin un proceso de *discretización*, se presentan como alternativas, según la bibliografía revisada, los siguientes algoritmos: *RandomForest Regressor* y *Catboost*.

i. RandomForest Regressor

Aplica el mismo concepto que el *RandomForest Classifier*, y se utiliza la librería *sklearn.ensemble* para importar *RandomForestRegressor* y generar un modelo de *RandomForest Regressor*.

Adicionalmente se utiliza la estructura propuesta por Currie (2017) para iterar sobre un máximo valor parametrizable y aleatoriamente escoger el número de iteradores (*n_estimators*), profundidad máxima (*max_depth*) y la cantidad mínima de registros para dividir un nodo (*min_samples_split*).

ii. Catboost Regressor

Basado en Gradiend Bootsting, que es un modelo de ensamblado de árboles de decisión que permite la optimización a través de la minimización del error cuadrado medio (MSE) a través de un ratio de aprendizaje.

Adicionalmente se utiliza la estructura propuesta por Currie (2017) para iterar sobre un máximo valor parametrizable y de manera aleatoria escoger las iteraciones (iterations), profundidad (depth), ratio de aprendizaje (learning rate) y el porcentaje de características a usar en cada subgrupo generado (rsm). De igual forma, se define como función de pérdida la distancia media cuadrática mínima o RMSE por sus siglas en inglés (loss_function='RMSE').

Se utiliza la librería *catboost* para importar *CatboostRegresor* y generar un modelo de regresión.

7. Recall y Precision

Recall expresa la proporción de predicciones positivas sobre el total de predicciones positivas más las predicciones falsas negativas. Por otro lado, *precision* es la proporción de las predicciones positivas sobre el total de predicciones positivas más las predicciones falsas positivas (Herbold, 2013). Para la comparación de modelos clasificadores se utiliza F1 score que balancea ambas proporciones.

Se utiliza la librería *sklearn.metrics* para importar *precision_score*, *recall_score* y *f1_score*, y adicionalmente *classification_report* para visualizarlos en un reporte.

8. Cross-validation

El proceso tradicional de ML consiste en dividir la data en un segmento de entrenamiento y otro de prueba, pero a través de cross-validation se realiza de manera iterativa de tal manera que todos puntos pueden ser probados. Se utiliza el parámetro *cv= "kfold"* con el que se divide el conjunto de datos en partes iguales y cada segmento es usado en cada iteración como data de prueba mientras que los restantes como entrenamiento (Refaeilzadeh, Tang, & Liu, 2009).

Se utiliza la librería *sklearn.model_selection* para importar *KFold* y particionar la data en 10 segmentos, valor común en ML (Refaeilzadeh et al., 2009), y *cross_val_score* para calcular el valor de *cross-validation*.

9. RMSE y RMSLE

Error cuadrático medio o MSE muestra el error medio entre los valores reales y predichos (scikit-learn developers (BSD License), 2017), se define por:

$$MSE(real, pred) = \frac{\sum_{i=0}^{n_{samples}-1} (real_i - pred_i)^2}{n_{samples}}$$

Para comparar los modelos de regresión se obtiene el valor de RMSE y se escoge al de menor valor, este se define por $RMSE = \sqrt{MSE}$.

Para validar el modelo seleccionado, se verifica si el valor de RMSLE es inferior o igual a 0.3935 de acuerdo a los resultados de la literatura revisada (Currie, 2017). Este se define por $RMSE = \sqrt{MSLE}$, donde MSLE o error logarítmico cuadrático medio se define por:

$$MSLE(real, pred) = \frac{\sum_{i=0}^{n_{samples}-1} (\log_e(1 + real_i) - \log_e(1 + pred_i))^2}{n_{samples}}$$

Se utiliza la librería *sklearn.metrics* para importar *mean_squared_error* y calcular el valor de MSE y posteriormente calcular la raíz cuadrada de ese valor para obtener RMSE. De igual forma con *mean_squared_log_error* y calcular el valor de RMSLE.

1.4 Alcance, Limitaciones y Riesgos

1.4.1 Alcance

- El proyecto tiene como producto final una herramienta web y estará diseñada para una versión de escritorio.
- El proyecto presenta una herramienta de gestión de proyectos de desarrollo bajo una gestión para resultados PM4R.
- La interfaz del proyecto se basará en los estándares de Google Material por su uso en las herramientas actuales, analizadas en el estado del arte.
- La herramienta contará con un módulo para cada fase del PM4R, estas son: EDT, Cronograma, Curva S, Matriz de adquisiciones, Matriz de Riesgos, Matriz de comunicaciones y Matriz de asignación de responsabilidades.
- Para la Fase de EDT, la herramienta permitirá relacionar proyecto, componentes, subcomponentes, productos, entregables y paquetes de trabajo.
- Para la Fase de Cronograma, la herramienta permitirá detallar el tiempo de duración de los paquetes de trabajo en días y con una representación visual del diagrama de Gantt.

- Para la fase de Curva S, la herramienta permitirá ingresar los costos por mes, el valor planificado y un campo para el valor real y mostrará un gráfico para validar la forma S de los costos.
- Para la fase de Matriz de adquisiciones, la herramienta permitirá ingresar, por cada producto o entregable, un tipo, modalidad, fechas estimadas y presupuesto estimado de adquisiciones requeridas y un campo adicional para el presupuesto real.
- Para la fase de Matriz de riesgos, la herramienta permitirá tener un mapeo entre los riesgos, resultados, síntomas, probabilidad, impacto, prioridad, respuesta y responsable.
- Para la fase de Matriz de comunicaciones, la herramienta permitirá ingresar los elementos, razones, destinatarios, métodos, responsables y tiempos de comunicación.
- Para la fase de Matriz de asignación de responsabilidades, la herramienta permitirá mapear cada producto o entregable con su responsable en base a la matriz RACI.
- El proyecto no abarca el desarrollo de nuevos algoritmos, sino la utilización y adaptación en base a librerías existentes como *scikit-learn*.
- Se utilizará *K-means* y EM-Cluster como algoritmos de segmentación o clusterización.
- Se utilizarán Regresión logística, *Naive Bayes*, SVM con *RBF kernel*, árboles de decisión, *RandomForest* y perceptrón multicapa como algoritmos de clasificación.
- La clasificación será realizada en base a una *discretización* del tiempo.
- Se utilizarán *RandomForest Regressor* y *Catboosts* como algoritmos de regresión.
- El módulo de estimación solo considerará los proyectos que cuenten con duración real y costo real.
- El módulo de estimación no verificará los datos ingresados o no ingresados.

- El proyecto no es un sistema experto, sino que cuenta con componentes reforzados con ML, para apoyar a la planificación de los gestores.

1.4.2 Limitaciones

- La data de proyectos de desarrollo en la web es limitada y solo se obtienen ciertos datos reales como el costo estimado y el real de los proyectos, mas no todas las características implicadas, por lo que la funcionalidad de la estimación de la plataforma estará restringida por la fuente de error del ruido presente en la generación de datos.

1.4.3 Riesgos

A continuación, se presentan los riesgos identificados que pueden generar un impacto negativo en el desarrollo del proyecto y las medidas preventivas (antes de que ocurra algo) o correctivas (como respuesta ante la ocurrencia de algo) para contrarrestarlos (Ver Tabla 5).

Tabla 5. Identificación de riesgos. Elaboración propia.

Riesgo	Impacto en el proyecto	Medidas Preventivas y/o Correctivas
Extensión en el desarrollo del software	<ul style="list-style-type: none"> - Prolongación de las actividades programadas. - Entregables de baja calidad. 	<ul style="list-style-type: none"> - Realizar una revisión constante del cumplimiento de las actividades planificadas y ajustarlas en la medida de que se cumpla con el proyecto.
Obtención de una cantidad no significativa de proyectos de desarrollo	<ul style="list-style-type: none"> - Imposibilidad de probar el módulo de estimación de tiempos. 	<ul style="list-style-type: none"> - Buscar proyectos de desarrollo de otras fuentes y no solo de la web del BID. - Generación de muestras en base a estrategias para parametrizar los variables requeridas.
Error del modelo muy alto	<ul style="list-style-type: none"> - Los resultados de la estimación serán imprecisos. 	<ul style="list-style-type: none"> - Agregar al módulo un nuevo modelo como opción en la búsqueda de la minimización del error.

		- Generar una mayor cantidad de datos de proyectos para evaluar los modelos generados.
--	--	--

1.5 Viabilidad

En esta sección se detallan los factores que determinan la viabilidad del proyecto.

1.5.1 Viabilidad Técnica

Se cuenta con los conocimientos técnicos para el desarrollo del proyecto. Dentro de los cursos llevados en la carrera se han desarrollado múltiples sistemas con un alto nivel de abstracción, además de que se cuenta con los cursos de Aplicaciones de ciencias de la computación, Inteligencia Artificial y Aprendizaje de Máquina para obtener los conocimientos base para la parte de la inmersión de estos en la plataforma a desarrollar. Asimismo, se cuenta con el apoyo de un asesor, profesional altamente capacitado y con experiencia en desarrollo de software y proyectos. De igual forma, las herramientas utilizadas, son todas conocidas por el tesista tanto para la parte de la construcción del software como para las de implementación de aprendizaje de máquina.

Por otro lado, los métodos planteados para el desarrollo brindan una guía en cuanto a las fases y los resultados a obtener para el logro de los objetivos y todas las herramientas están disponibles ya sea por ser software libre o por licencia educacional.

1.5.2 Viabilidad Temporal

En el aspecto temporal se presenta un el cronograma de actividades acorde a los tiempos establecidos para el proyecto (Ver Tabla 6):

Tabla 6. Actividades del proyecto. Elaboración propia.

Actividad	Descripción	Fecha inicio	Fecha fin
1. Obtención de Requerimientos	Se obtienen los requerimientos funcionales y no funcionales pre desarrollo de la plataforma y se les prioriza.	07-05-2018	15-05-2018

2. Diseñar las interfaces	Se obtiene un prototipo y flujo del software a desarrollar.	15-05-2018	01-06-2018
3. Diseñar la arquitectura	Se obtienen los diagramas de clases y despliegue usando STARUML como herramienta de apoyo.	02-06-2018	17-06-2018
4. Implementación de la plataforma	Programación de la interfaz y la lógica del software.	18-06-2018	26-08-2018
5. Generación y almacenamiento de datos de proyectos	Se generarán datos de proyectos en base a la data web y parámetros establecidos.	26-08-2018	10-09-2018
6. Definición de los atributos para los algoritmos de machine learning.	Se definen las variables para los algoritmos de ML.	10-09-2018	11-09-2018
7. Pre-procesamiento de datos.	Pre-procesamiento de los datos.	11-09-2018	16-09-2018
8. Implementación del algoritmo de segmentación o clustering.	Se implementa el algoritmo de segmentación.	16-09-2018	20-09-2018
9. Implementación del módulo para la generación de modelos dinámicos de predicción de tiempos.	Se implementa el módulo de estimación de tiempos para obtener un modelo.	20-08-2018	30-09-2018
10. Análisis del error sobre una muestra de proyectos de prueba.	Se generan tablas comparativas del error asumido en las estimaciones.	30-09-2018	10-10-2018

11. Integrar la plataforma con el módulo.	Se integra el módulo en la plataforma y se realizan las pruebas unitarias.	10-10-2018	30-10-2018
---	--	------------	------------

Para la actividad 5 se planteó obtener datos de la web del World Bank a través de una búsqueda intensiva de proyectos publicados para posteriormente, utilizando datos existentes y otros parámetros, implementar un generador de datos, tal como se realizó en “Diseño de un algoritmo de búsqueda Tabú para resolver problemas de la selección de proyectos” (Rejas Cano, 2014), en el que se generó data de costos, beneficios, tiempo, etc. para la posterior evaluación de su algoritmo heurístico.

1.5.3 Viabilidad Económica

Las herramientas no representan un costo adicional al utilizar las versiones gratuitas o con licencia educacional. En cuanto al despliegue de la herramienta, se utilizó un servidor gratuito. A continuación, se detallan los costos en un entorno real (Ver tabla 7).

Tabla 7. Costos de despliegue estimados. Elaboración propia.

Requerimiento	Empresa	Costo	Costo en soles	Descripción
Dominio	GoDaddy	MXN 18.99/año	PEN 3.0231/año	Dominio .com
Host para <i>Front-End</i>	Google Cloud	\$56.22/mes	PEN 190.42/mes	7.5GB Memoria, 2VCPU's
Host para <i>Back-End</i> (EC2)	AWS	\$9.5/mes	PEN 32.18/mes	t2.micro
BD (RDS)	AWS	\$12.24/mes	PEN 41.46/mes	db.t2.micro

1.5.4 Conclusiones de Viabilidad

En conclusión, el proyecto es viable y requiere de un alto nivel de compromiso; ya que se cuenta con los conocimientos técnicos, el desarrollo del proyecto no demanda un desembolso de dinero y si bien es cierto, existen limitaciones con respecto a la obtención de datos, se establecieron respuestas pertinentes.

Capítulo 2. Marco Conceptual

En esta sección se explican los conceptos mencionados en la problemática y los relacionados.

2.1 Objetivos del marco conceptual

Detallar los conceptos que permitirán un mejor entendimiento del problema de automatizar la gestión de proyectos de desarrollo (planificación, ejecución y monitoreo) a través de un comportamiento inteligente.

2.2 Definiciones

Se presentan a continuación los conceptos relevantes.

2.2.1 Proyectos

“Un proyecto es un esfuerzo temporal que se lleva a cabo para crear un producto, servicio o resultado único” (Project Management Institute, 2017, p. 40).

Según el Project Management Institute PMI (2017) en la Guía del PMBOK, este esfuerzo temporal puede involucrar a una o muchas personas y a diferentes niveles de la organización y el término temporal hace referencia a que un proyecto tiene inicio y fin; sin embargo, pueden existir entregables, de naturaleza social, económica, material o ambiental, que sean posteriores a la finalización.

También se define en la Guía del PMBOK que los proyectos impulsan a la organización hacia un estado futuro y crean valor en el negocio, este último referido al beneficio cuantificable neto en forma de: Tiempo, dinero, bienes e intangibles (PMI, 2017).

Ejemplos de proyectos son:

- Proyecto de estadios para el mundial femenino sub 20 en Chile (PMI, 2010)

Objetivo: Construcción de 4 estadios en 9 meses para el mundial femenino sub 20 en Chile

Cronograma: 9 meses.

Costo: USD 100.000.000

- Central Hidroeléctrica Platanal en Perú (Ministerio del Ambiente, 2011)

Objetivo: generar electricidad utilizando fuentes de energía renovable, para suministrarlos al sistema interconectado nacional

Cronograma: 7 años

Costo: USD 350.000.000

2.2.2 Gestión de Proyectos

También conocida como gerencia, administración o dirección de proyectos, es una disciplina definida por el Instituto de Gestión Proyectos como “la aplicación de conocimientos, habilidades, herramientas y técnicas en las actividades del proyecto para lograr sus requerimientos” (PMI, 2018).

Una concepción similar es la de la Asociación para la Gestión de Proyectos que lo define como “la aplicación de procesos, métodos, conocimientos, habilidades y experiencia para lograr los objetivos del proyecto” (APM, 2018),.

Ambas definiciones coinciden en aplicar ciertas características para cumplir objetivos; sin embargo, Wallace (2011) hace una separación entre establecer y alcanzar. El primero referido a la etapa de planificar y el segundo a la ejecución y control para conseguir o superar objetivos de tiempo, costo y desempeño. Además, incluye conceptos de seguridad y riesgo, ya que las necesidades económicas apuntan a un mayor riesgo y menor seguridad (Wallace, 2011). Los grupos de procesos de gestión con la separación de Wallace son mostrados en la Figura 3.

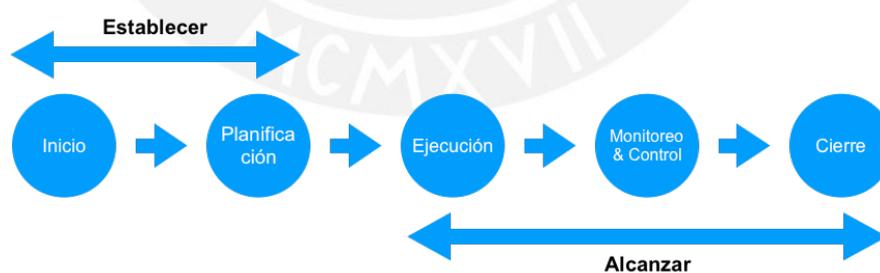


Figura 3. Representación de los grupos de procesos de Gestión de Proyectos y la separación en dos fases. Adaptado de (PMI, 2018; Wallace, 2011).

Según PMI (2018) son 10 áreas en las que se basa la gestión de proyectos: Integración, alcance, tiempo, costo, calidad, adquisiciones, recursos, comunicaciones, gestión del riesgo y gestión de interesados.

2.2.3 Proyectos de desarrollo

Según Siles & Mondelo (2015), los proyectos de desarrollo son vehículos para que un país o región logre mejorar sus índices de desarrollo a través de la obtención de resultados concretos y sostenibles a largo plazo. Se basa en la teoría del cambio (Change Framework), que constituye la consecución gradual de objetivos intermedios para acercarse a un objetivo mayor a largo plazo (Siles & Mondelo, 2015).

Se definen las siguientes características:

- Diversidad de interesados: stakeholders con diversas necesidades, expectativas, demandas, nivel de influencia e interés que requieren un alto nivel de comunicación para definir acuerdos en temas políticos, sociales, ambientales, de género, etc.
- Sostenibilidad: el impacto a largo plazo debe prevalecer por sobre los resultados a corto plazo, por tanto, se monitorean los cambios post a la finalización del proyecto.
- Retorno Social: Se realizan ante una necesidad social, y se basan en una estrategia de desarrollo económico y social. (Siles & Mondelo, 2015)

La estrategia de desarrollo mencionada hace referencia a las prioridades de un país, que incluye objetivos de desarrollo, cooperación internacional en el marco de acuerdos y análisis de la situación socioeconómica (Siles & Mondelo, 2015). Usualmente se complementa el gobierno con una entidad financiadora, relación mostrada en la Figura 4.



Figura 4. Representación de estrategias de desarrollo conjuntas.

Adaptado de (Siles & Mondelo, 2015)

Esta colaboración, según Siles & Modelo (2015) implica la normativa y comunicación para facilitar el trabajo colaborativo y eliminar duplicidad en el mismo, velar por intereses alineados en forma progresiva y combatir la corrupción.

Por otro lado, algunas entidades que también realizan proyectos de desarrollo son las organizaciones no gubernamentales (ONG's), las que interactúan con socios y colaboradores, que pueden ser entidades públicas o privadas, como financiadores (Mans Unides, 2016). La ONG Mans Unides presenta una clasificación por sectores para los proyectos de desarrollo que realizan, estos son: agrícola, promoción social, educativo, promoción de la mujer, sanitario y emergencias (Mans Unides, 2015b, 2016).

A continuación se ejemplifican algunos proyectos, desarrollados en el Perú, para estos sectores (Mans Unides, 2015a):

- Fortalecimiento de la cadena productiva de la fibra de alpaca

Lugar: Juliaca, costo: 71.299,00 Euros, sector: Agropecuario

- Desarrollo inclusivo de mujeres ayacuchanas

Lugar: Lima, costo: 283.688,59 Euros, sector: Promoción de la mujer

- Salud, nutrición y educación para niños menores de 5 años y madres gestantes

Lugar: Apurímac, costo: 92.077,00 Euros, sector: Sanitario

- Ampliación. desarrollo integral en comunidades indígenas

Lugar: UROS CCAPI, costo: 9.000,00 Euros, sector: Social

- Implementación de una biblioteca escolar

Lugar: JULIACA, costo: 9.009,00 Euros, sector: Educativo

2.2.4 Gestión de proyectos de desarrollo

La gestión de proyectos de desarrollo se basa completamente en la gestión para resultados (PM4R) (Siles & Mondelo, 2012, 2015) y su definición está dada como:

“El uso de una combinación de herramientas y técnicas derivadas de buenas prácticas y estándares internacionales para asegurar el logro de los objetivos específicos (resultado, producto o servicio) del proyecto dentro del tiempo (cronograma)” (Siles & Mondelo, 2015, p.12).

Se diferencia de la gestión de proyectos convencionales, por la inmersión de conceptos de prácticas internacionales y por utilizar características de ellas para, según Siles & Mondelo, garantizar que se mantengan las estrategias de desarrollo del país-financiador y la lógica por resultados. Esta última evalúa la priorización de resultados para

posteriormente definir la combinación de recursos con lo que no se enfoca en las actividades, para que la prioridad deje de ser su control (García López & García Moreno, 2010; Siles & Mondelo, 2015).

En este sentido, Siles & Mondelo reafirman la importancia de enfocarse en los resultados a través del uso de indicadores y evaluaciones, tales como cumplir con los tres tipos de planificación mostradas en la Figura 5.

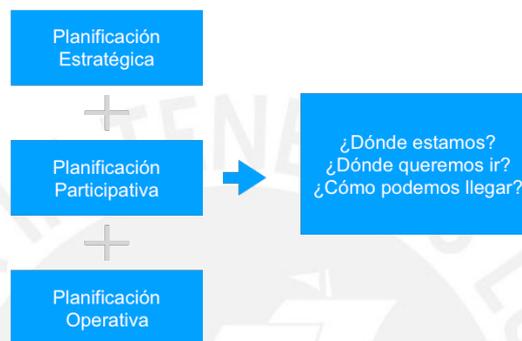


Figura 5. Elementos de la planificación orientada a resultados. Adaptado de (García López & García Moreno, 2010).

La primera está referida a responder ¿a dónde queremos ir?, la última a ¿cómo podemos llegar? y la segunda a la inmersión de los interesados (stakeholders) en el consenso de resultados, para que los beneficiarios finales queden satisfechos, caso contrario no serán exitosos los resultados sin importar si se cumplieron o no los requerimientos en el tiempo planificado (García López & García Moreno, 2010; Siles & Mondelo, 2015).

Las herramientas que se proponen en PM4R son las que se muestran en la Tabla 8 y se detallan posteriormente a través del ejemplo: Programa de Reconversión Urbana del Ministerio de Obras Públicas y Comunicaciones del Gobierno de Paraguay.

Tabla 8. Pasos para la planificación y gestión de proyectos de desarrollo. Adaptado de (Siles & Mondelo, 2015).

Pasos	Descripción
Paso 1	Elaboración del EDT
Paso 2	Elaboración del cronograma
Paso 3	Desarrollo de la curva S
Paso 4	Elaboración de matriz de adquisiciones
Paso 5	Elaboración de matriz de riesgos
Paso 6	Elaboración de matriz de comunicaciones
Paso 7	Creación de MAR: Matriz de asignación de responsabilidades

1. EDT: estructura de desglose de trabajo en la que se crean y editan los componentes, subcomponentes, productos, entregables y paquetes de trabajo (Ver Figura 6) (Siles & Mondelo, 2015).



Figura 6. Elementos del EDT para el Programa Reconversión Urbana. Adaptado de (Yamanaka, 2015).

2. Cronograma: estimación de tiempos análoga (duración, presupuesto y complejidad), paramétrica (estadísticamente horas-hombre, o metros cuadrados) o por tres valores PERT a través de $De = (O + 4M + P)/6$, donde De es la duración esperada, O es la duración optimista, M la duración más probable o realista y P la pesimista. Se puede aplicar fast-tracking o tareas en paralelo (Ver Figura 7) (Siles & Mondelo, 2015).



Figura 7. Cronograma para el Programa Reconversión Urbana. Tomado de (Yamanaka, 2015).

3. Curva S: curva de uso de recursos, y se denomina curva S porque al inicio como al final los costos son menores (Ver Figura 8). Los insumos son el instrumento 1 y 2, las restricciones del proyecto, el documento de aprobación del proyecto en el que se define el total del presupuesto aprobado, las restricciones del uso de recursos y la información financiera histórica (Siles & Mondelo, 2015).

En este instrumento se tiene una visibilidad total de lo planificado y lo ejecutado.

La estimación se puede hacer por estimación análoga, determinación de tarifas (cotizaciones) y en base a índices (costos unitarios) (Siles & Mondelo, 2015).

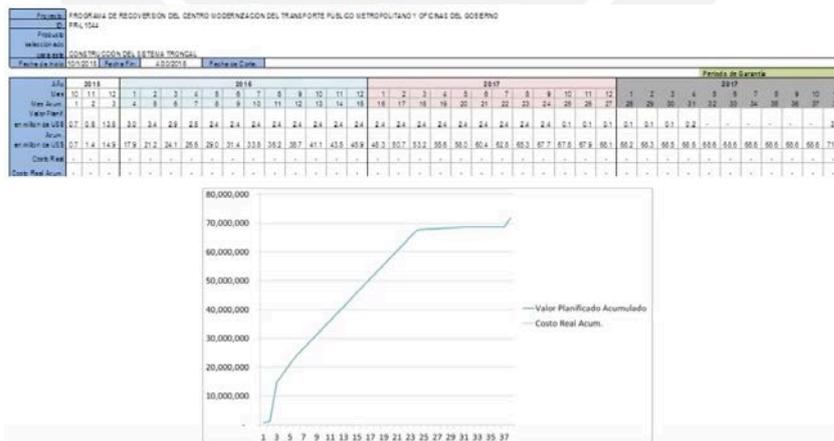


Figura 8. Curva S para el Programa Reconversión Urbana. Tomado de (Yamanaka, 2015).

4. Matriz de adquisiciones: Se detalla la contratación de bienes y servicios, el tipo y modalidad de adquisición de los mismos y las fechas estimadas (Ver Figura 9). Se obtiene un plan de adquisiciones (Siles & Mondelo, 2015).

Proyecto: PROGRAMA DE RECONVERSIÓN DEL CENTRO MODERNIZACIÓN DEL TRANSPORTE PÚBLICO METROPOLITANO Y OFICINAS DEL GOBIERNO							
ID:							
Código EDT	Producto o Entregable	Tipo de Adquisición	Modalidad de Adquisición	Fechas Estimadas		Presupuesto Estimado en Dolares Americanos	
				Inicio	Fin		
1	BTR						
1.1	CONSTRUCCIÓN DEL SISTEMA TRONCAL						
1.1.1	Estudio de Prediales y Catastro	Servicio de consultoría	SBOC	10/1/2015	3/29/2016	\$3,500,000	
1.1.2	Estudio de Impacto Ambiental	Servicio de consultoría	SCC	12/1/2015	3/30/2016	\$200,000	
1.1.3	Diseño del Plan Comunicacional del Programa	Servicio de consultoría	SCC	10/15/2015	4/12/2016	\$200,000	
1.1.4	Plan de Gestión Social	Servicio de consultoría	SD	11/30/2015	5/28/2016	\$100,000	
1.1.5	Fiscalización de las Obras de Construcción del Sistema	Servicio de consultoría	SBOC	2/7/2015	4/30/2018	\$3,500,000	
1.1.6	Construcción del Sistema Troncal	Servicios Fimas	LPI	12/1/2015	4/30/2018	\$64,000,000	
Total						5	71,500,000.00

Figura 9. Matriz de Adquisiciones para el Programa Reconversión Urbana. Tomado de (Yamanaka, 2015).

5. Matriz de riesgos: presenta los riesgos, la evaluación de la probabilidad de ocurrencia, su impacto y las respuestas asignadas a un responsable (Ver Figura 10) (Siles & Mondelo, 2015).

MATRIZ DE RIESGOS							
PROYECTO: PROGRAMA DE RECONVERSIÓN DEL CENTRO MODERNIZACIÓN DEL TRANSPORTE PÚBLICO METROPOLITANO Y OFICINAS DEL GOBIERNO							
ID PROYECTO: 094.0344 FECHA DE INICIO: 2014 FECHA DE TÉRMINO PROPUESTA: 2018							
#	Riesgo (R)	Posible resultado (R-resultado)	Síntoma	Probabilidad de ocurrencia (R-probabilidad)	Impacto (R-impacto)	Responsable de la acción de respuesta	
1	Seja falta de metas claras para la implementación de programa, y consecuentemente la no producción en la siguiente institución	Faltas en la ejecución de actividades programadas en el cronograma	Atraso en el inicio de ejecución de actividades	Alto	Alto	Coordinación de una "Reunión de Trabajo Estratégica" con representantes de esta gestión municipal; reportes del programa de trabajo para definir definiciones y controlar procesos que aseguren y/o impulsen el correcto desarrollo del Programa, tanto a nivel interno como en tanto al territorio.	Gerencia del Programa
2	Seja incapacidad de gestión para la elaboración de documentos de justificación y consecuentemente falta de pago de proyectos	Documentos de justificación (tramitados) y con errores (errores)	Faltas en la aprobación de documentos de justificación. Faltas de documentos, errores en autorizaciones y protocolos y faltantes	Medio	Alto	Revisión de la elaboración de "Formas de Referencia" y Experiencias Técnicas cuando se requiera o cuando se soliciten. Revisión de documentos de justificación.	Coordinador Técnico de Cooperación Técnica
3	Deficiencia de calidad de la gestión (social) respecto al OdeA Americano, respecto a la que se establece el	Señalamientos (reclamaciones) para cubrir las obras y/o servicios de actividades	Atraso en el inicio de ejecución de obras	Alto	Alto	Gestionar para obtener recursos adicionales de contratación para cubrir la obra de actividades.	Especialista Finanzas
4	Atraso a las obras por parte de las empresas contratadas del trabajo del contratista	Atraso en el inicio y en la ejecución de obras	Deficiencia del proyecto para establecer los requerimientos de los beneficiarios. Atraso en el inicio, ejecución y/o tiempo (de las obras, etc)	Alto	Alto	Caracterización de actividades. Elaboración e implementación de estrategias de comunicación apropiadas (charlas, mesas de comunicación, redes sociales, audiencias públicas, entre otras)	Coordinador Técnico, Especialista de Comunicación, Especialista del Área Social
5	Escasez de materiales de construcción	Atraso en la ejecución de las obras	Deficiencia de la producción nacional de cemento es suministrado únicamente por el Estado	Medio	Alto	Controlar un porcentaje de inventarios para cubrir el costo por importación de cemento	Especialista en Infraestructura
6	Deficiencia de ejecución de obras (servicios sociales) para obras con base optimizada (no metálica)	Atraso en las contrataciones y para obras con base optimizada (no metálica)	Atraso en las acciones que son más a los contratados	Medio	Alto	Revisar y ajustar el cronograma de ejecución considerando los tiempos de procesos administrativos más	Especialista en Adquisiciones, Especialista en Infraestructura
7	Capacidad operativa de la UEP no adecuada a las demandas y necesidades del proyecto	Seja seguimiento y control de los contratados. Resolución de problemas a corto plazo, en donde se requiera recursos y/o bienes	Deficiencia de actividades a corto plazo, en donde se requiera recursos y/o bienes	Medio	Alto	Revisar el cronograma y realizar las contrataciones de personal de necesidad	Coordinador Técnico
8	Por falta de acciones múltiples (reclamo) y/o gestión de 3 de las Municipalidades, se genera un cambio en el comportamiento político de los alcaldes con el proyecto. Las acciones se realizaron en Nov 2014	Cambios que impactan en el avance de proyectos por acciones negativas de gestión por parte de las Municipalidades	Deficiencia de acciones múltiples por parte de las Municipalidades y/o la instancia	Medio	Medio	Ejecución de acciones, importancia y vigencia del proyecto obliga a ser cancelado y seguir un cronograma con metas antes de las elecciones.	Gerencia del Proyecto
9	Atraso en la ejecución debido a que las obras programadas dependen de otras instituciones (trámites de licencias)	Atraso en la ejecución de actividades programadas en el cronograma	Atraso en la recepción de licencias para la ejecución de las obras por falta de recursos económicos para el cumplimiento de trámites y/o licencias	Medio	Alto	Realizar monitoreo permanente a las Solicitudes de Desembolso de acuerdo al Cronograma de ejecución	Especialista Finanzas

Figura 10. Matriz de riesgos para el Programa Reconversión Urbana. Tomado de (Yamanaka, 2015).

6. Matriz de comunicaciones: Especifica la cantidad y calidad de información que se tiene que comunicar en el momento adecuado a los interesados (stakeholders) (Ver Figura 11), por lo que requiere como insumos la matriz de

interesados (datos de los interesados), cronograma y los factores organizacionales de obligatoriedad (Siles & Mondelo, 2015).

Proyecto: RECONVERSIÓN							
ID: PRU104							
¿Qué comunicar?	¿Por qué?	Destinatario	Método de Comunicación	Responsabilidad		Tiempo	
				Preparación	Envío	Inicio	Frecuencia
La implantación y operación del Metrobus en el corredor E. Ayala	Porque serán afectados temporalmente durante las obras en el corredor para la operación del servicio	Comercio Formal del corredor del BRT	Reuniones por subtramos y consultas públicas sobre el proyecto y sus alcances	Coordinación social y equipo social de la Unidad Ejecutora del Proyecto	Unidad Ejecutora de Proyecto UEP	10/15/2015	5 veces por semana para abarcar los 16 subtramos del corredor principal
La implantación y operación del Metrobus en el corredor E. Ayala	Serán afectados en forma permanente si que deberán ser removidos del corredor para la liberación de la franja de dominio	Puestos fijos de venta informal en el corredor del BRT	Talleres participativos y de consulta sobre propuestas para la reubicación de puestos	Empresa tercerizada	Unidad Ejecutora de Proyecto UEP y la empresa tercerizada	11/30/2015	1 vez por semana por 2 meses
Cambio en el sistema de transporte público en el corredor E. Ayala y Ruta Mariscal Estigarribia	Dos líneas de transporte actuales ya no podrán operar cómodas en el corredor	Transportistas	Reuniones, talleres, consultas	Empresa tercerizada	Unidad Ejecutora de Proyecto UEP y la empresa tercerizada	5/30/2015	1 vez por empresa durante 2 meses
Modernización del transporte público para pasajeros a través del primer BRT	Será la beneficiaria del servicio y se garantizará el derecho a la movilidad urbana	Ciudadanía en general	Medios de prensa, entrega de volantes informativos, spot de radio, video de presentación	Empresa tercerizada	Unidad Ejecutora de Proyecto UEP y la empresa tercerizada	5/30/2015	Campaña de difusión masiva por medios y callejera durante el tiempo de obras
Avances del proyecto	Son las instituciones que recibirán los beneficios del proyecto en las áreas intervenidas	Instituciones involucradas (Municipalidad en, entidades de servicios públicos)	Conformación de mesa de interinstitucional de trabajo, notas impresas, envío de mail	Gerencia del proyecto y área de comunicación	Gerencia del proyecto	11/7/2015	Semanal antes del inminente inicio de obras y quincenal durante las obras
Avances del proyecto según según alcances y ejecución financiera	Son las entidades que financian gran parte del proyecto y deben monitorear el avance y la ejecución correcta de los recursos	Entidades financieras	Reunión de coordinación, notas impresas, envíos de mail	Gerencia del proyecto y la UEP	Gerencia del proyecto	10/7/2014	Reuniones semanales y/o quincenales según la necesidad o requiera

Figura 11. Matriz de comunicaciones para el Programa Reconversión Urbana. Tomado de (Yamanaka, 2015).

7. Matriz de asignación de responsabilidades: conecta el organigrama con el EDT y se basa en RACI (Ver Figura 12).

EDT	Producto o entregable	Responsable				
		R	A	C	I	F
2	Primer Corredor de Transporte Público Masivo - BTR					
2.1	Mejora de la infraestructura y del Ambiente Urbano					
2.1.1	Diseño y Ejecución de las Obras de Construcción del Corredor Central - Tramos 2 y 3					
2.1.1.1	licitaciones	Peggy (EA)	José Tomás (CT)	Ulían (UOC) Martín (CE) Jicós (CE)	Angel (OP)	
2.1.1.2	Diseño y Ejecución de Obras	Sally (E)	José Tomás (CT)	Karen (EA), Claudia (EA) Papalardo (EDV) Daniel (DSSA) Cesar (CE)	Angel (OP)	
2.1.2	Diseño y Ejecución de las Obras de Construcción de las Pistas - Tramos 1 y 4					
2.1.2.1	licitacion	Peggy (EA)	José Tomás (CT)	Ulían (UOC) Martín (CE) Jicós (CE)	Angel (OP)	
2.1.2.2	Diseño y Ejecución de Obras	Sally (E)	José Tomás (CT)	Karen (EA), Claudia (EA), Papalardo (EDV) Daniel (DSSA), Cesar (CE)	Angel (OP)	
2.2	Centro de Control y Operación					
2.2.1	Plan Operacional del Sistema BTR	Bianca (EC)	José Tomás (CT)	Serman (CE)	Angel (OP)	
2.2.2	Modelo tarifario y del Plan de Negocios	Bianca (EC)	José Tomás (CT)	Serman (CE)	Angel (OP)	
2.2.3	Centro de Control en Operación	Bianca (EC)	José Tomás (CT)	Serman (CE)	Angel (OP)	
2.2.4	Centro de Monitoreo Ambiental en Operación	Bianca (EC)	José Tomás (CT)	Serman (CE)	Angel (OP)	
2.2.5	Sistema de transporte concesionados	Bianca (EC)	José Tomás (CT)	Serman (CE)	Angel (OP)	
2.3	Visitación Socio Ambiental					
2.3.1	Capacitación Empresarial y Asistencia Técnica	Gloria (ES)	José Tomás (CT)	Claudia (EA), Rowda (CE)	Angel (OP)	
2.3.2	Conductores y trabajadores técnicos capacitados	Gloria (ES)	José Tomás (CT)	---	Angel (OP)	
2.3.3	Campañas de Comunicación	Roxa (EC)	José Tomás (CT)	Oscar (D) HCDM	Angel (OP)	
2.3.4	Reubicación de vendedores informales	Gloria (ES)	José Tomás (CT)	Claudia, Elena (CE), Daniel (DSSA)	Angel (OP)	
2.3.5	Reubicación de vendedores del Mercado de San Lorenzo	Gloria (ES)	José Tomás (CT)	Claudia, Elena (CE), Daniel (DSSA)	Angel (OP)	

Referencias
R = Responsable de Ejecutar
A = Autorizada
C = Consultado
I = Informado

Equipo Interno
OP = Gerente de Programa
CT = Coordinador Técnico BTR
E = Especialista en Infraestructura
EA = Especialista en Arg., Urbanismo y Catastro
ES = Especialista Social
EE = Especialista en Educación
EA = Especialista en Adquisiciones
EE = Especialista Económico
EA = Especialista Ambiental

Equipo Externo
UOC = Unidad Operativa de Contrataciones - MOPC
Dir COM = Dirección de Comunicación - MOPC
Dir = Dirección de Visitas
CE = Consultor externo
DSSA = Dirección General Social y Ambiental - MOPC

Figura 12. Matriz de asignación de responsabilidades para el Programa Reconversión Urbana. Tomado de (Yamanaka, 2015).

2.2.5 Data y Niveles de uso de la data

Data o data cruda es una colección de textos, números o símbolos sin ningún significado. Información por otra parte, es la agrupación o procesamiento de la data de manera organizada para ser usada en un contexto y tenga un significado, y

conocimiento es la aplicación o uso de la información (Cambridge International Examinations, 2017).

Por otro lado, los niveles de uso de la data de acuerdo a Gibson & Nolan (1974) son seis:

- Iniciación: Primera integración de una computadora en la organización.
- Contagio o Expansión: Independización del área de sistemas de información (SI) y se contrata especialistas. La capacidad computacional es utilizada para aplicaciones más avanzadas.
- Control o Formalización: Se formalizan planeamiento y reportes. Los sistemas de información (SI) alcanzan un nivel gerencial.
- Integración: De los sistemas a la organización. Autonomía generada por los desarrollos tecnológicos.
- Administración de datos: Reconocer la información como bien valioso. Integridad y compartir del mismo cobran importancia.
- Madurez: La informática está al más alto nivel de dirección. Integración de sistemas basados en conocimientos y sistemas expertos (simulaciones) (como se citó en Trujillo, 2012).

2.2.6 Herramientas de Gestión de Proyectos

Son las herramientas que ayudan en los procesos de la gestión de proyectos con el objetivo de garantizar una correcta utilización de recursos y que los resultados logren el impacto para la empresa o para un tercero al que se le brinda el servicio (como se citó en Santos D., A. Parra, Gallardo V., & Ingram R., 1996).

Existe una gran variedad de herramientas en el mercado actual, con prevalencia de algunos como Microsoft Project (MS Project). Según una encuesta a finales del 2016, a más de 200 gestores de proyectos de EEUU con un 45% perteneciente al sector TI, el 67% utilizaba MS Project como herramienta de gestión de proyectos; no obstante, el 66% estaba dispuesto a cambiar a una nueva tecnología y las principales razones para ello, según los mismos entrevistados, eran la administración de tareas y el trabajo colaborativo como características deseadas (Maffeo, 2016). Es por ello que las nuevas

herramientas, en respuesta a la demanda (requerimientos de los gestores), desarrollan en su gran mayoría estos tipos de soluciones mostrados en la Tabla 9:

Tabla 9. Clasificación de herramientas de gestión de proyectos actuales. Elaboración propia.

Conceptos Usados	Herramientas
Basados en tareas (Task Based)	JIRA, Asana, Wrike, Trello, TimeCamp, Jive, SmartSheet
Servicios Web (Web Services)	Slack, iMeetCentral, Microsoft Project Online Professional, BaseCamp, Celoxis, Team Gant, Confluence
Instalables (Desktop Software)	Hansoft, OmniPlan, Merlin Project
Basado en almacenamiento en la nube (Cloud based)	ProjectLibre

Los conceptos detallados basados en las Categorías de software de *Capterra*, son los siguientes (Capterra, 2018a):

- Software basados en tareas (*Task based software*): Basados en el seguimiento y cumplimiento de una lista de tareas (Capterra, 2018b). Según los ejemplos mostrados en la Tabla 9, están relacionados con metodologías ágiles.
- Servicios Web (*Web Services*): cuentan con plataformas web y algunos presentan su versión instalable. Considerados también como soluciones Cloud por su almacenamiento en la nube (no locales) (Douglas, 2018b, 2018a).
- Instalables (*Desktop Software*): requieren de un instalador y el almacenamiento es local (Elmblad, 2018). Según los ejemplos mostrados en la Tabla 9, suelen tener versiones para móviles.
- Basado en almacenamiento en la nube (*Cloud based*): Según el ejemplo mostrado en la Tabla 9, es la combinación de un software instable con almacenamiento en la nube (no locales).

En contraste, herramientas como *stratejos.ai*, *forecast.it* y *aptage* están reforzadas con IA (Inteligencia Artificial). Esto significa que cuentan con componentes de inteligencia artificial que les permiten reforzar la gestión de proyectos (Aptage, 2018; Forecast, 2018; Stratejos, 2018).

2.2.7 Comportamiento Inteligente

El comportamiento es inteligente cuando cumple con dos requisitos: criterio de efectividad, y el mismo criterio aplicado en contextos variados (como se citó en Bueno Cuadra, 2008). Este se logra a través de sistemas inteligentes basados en algoritmos de aprendizaje automático, que a su vez requieren una recolección de data representativa. Este aprendizaje automático también conocido como *Machine Learning* es una rama de la Inteligencia Artificial (Vegega, Pytel, & Pollo, 2017).

Esta implica el entendimiento de nuestra inteligencia y la construcción de agentes inteligentes. La inteligencia está referida a responder el ¿cómo pensamos?, esto incluye el entender la forma como percibimos, entendemos, predecimos y manipulamos nuestro entorno (Russell & Norvig, 2010).

Existen diferentes categorías de inteligencia artificial; sin embargo, según Russell & Norvin (2010) la aproximación actual está más relacionada a las dos últimas categorías: “Actuando como humano” y “Actuando racionalmente”.

“Actuando como humano” implica otorgar capacidades a las máquinas, y según Kurzweil (1990) estas están referidas a funciones que requieren inteligencia (Como se cita en Russell & Norvig, 2010). Una de sus capacidades es el Aprendizaje de máquina (en adelante ML por sus siglas en inglés *Machine Learning*) para establecer modelos y extrapolar patrones (Russell & Norvig, 2010).

Kovahi & Provost (1998) definen *Machine Learning* como la disciplina que permite la construcción de algoritmos para aprender de la data (como se citó en Monleon-Getino, 2016), también denominados algoritmos de aprendizaje automático. Estos algoritmos según Bishop (2006) construyen modelos para hacer predicciones o tomar decisiones (como se citó en Monleon-Getino, 2016).

Capítulo 3. Estado del Arte

Esta sección, por un lado, detalla los conceptos, componentes, estándares y formato de las herramientas de gestión de proyectos que presentan soluciones de inteligencia artificial. Por otro lado, se presentan los algoritmos, estrategias, factores, cantidades de muestras, métodos de medición, comparación de algoritmos y conclusiones de algunas investigaciones sobre la integración de un software de gestión de proyectos con conceptos de aprendizaje automático.

3.1 Revisión y discusión

Se realizó una revisión de las herramientas actuales de gestión de proyectos, y la literatura existente en los repositorios de ACM, tesis PUCP y del Politécnico de Milán, la revista científica “Applied Discrete Mathematics and Heuristic Algorithms” y una solución presentada para la competencia de Kaggle cuya finalidad fue predecir la duración de los viajes en taxi en New York. Para ello, se definieron objetivos previos a la búsqueda de los términos como criterios de inclusión o exclusión.

3.2 Objetivos de la revisión del estado del arte

En las herramientas tecnológicas:

- Conocer los conceptos de inteligencia artificial que han usado las herramientas de gestión de proyectos para estimar ciertas variables, el formato de su presentación y conocer los componentes de estas herramientas.
- Conocer los estándares actuales y la distribución de sus componentes

En la literatura revisada:

- Conocer el procedimiento utilizado para predecir características de proyectos y los algoritmos de inteligencia artificial implementados, que pueden ser replicados para resolver la estimación de tiempos.
- Conocer la comparación entre los algoritmos usados para la predicción sobre características de proyectos y los métodos de medición del éxito del predictor.
- Conocer la cantidad de datos de proyectos evaluados y las estrategias de selección de la data para los algoritmos de segmentación o clusterización.

- Conocer las fases de desarrollo replicables para la implementación de un algoritmo de Inteligencia Artificial en un sistema.

3.2.1 Herramientas

Las herramientas de gestión de proyectos que presentan soluciones de inteligencia artificial identificadas en la revisión son las siguientes:

- Stratejos.ai
- Forecast.ai
- Aptage

1. **Conocer los conceptos de inteligencia artificial que han usado las herramientas de gestión de proyectos para estimar ciertas variables, el formato de su presentación y conocer los componentes de estas herramientas.**

Una característica reforzada con inteligencia artificial es el análisis de riesgo que realiza Stratejos.ai y notifica cuando hay imprecisiones en las estimaciones del presupuesto, es decir, presentan una solución para la estimación de costos a través de un sistema de alertas, como se observa en la Figura 14 (Stratejos, 2018).

Este sistema de alertas, es un bot auto-configurable por el usuario, por lo que cuenta con un componente para este fin (Ver Figura 14). De igual forma, otros componentes presentados son el Dashboard (contiene las prácticas y progresos del proyecto, las personas y el trabajo realizado), Data (obtención de información por filtros), Bot (sistema de alertas) y Admin (configuraciones del proyecto) (Stratejos, 2018).

Para el caso de Forecast.ai, se realiza la automatización de los procesos manuales de programación de calendarios con predicción basada en data histórica. De igual forma, presentan una estimación de costos del proyecto basada en *machine learning* (Forecast, 2018), es decir, responden al problema de estimación de tiempos y costos a través de campos adicionales (Ver Figuras 15 y 16).

Por último, Aptage usa conceptos de *machine learning* para proveer recomendaciones y revela información de los proyectos antes que se desalineen de lo planificado, además de simulaciones en base a cambios en los datos (Aptage, 2018).

2. Conocer los estándares actuales y la distribución de sus componentes

Dentro de los estándares de Stratejos.ai, se presenta un menú lateral izquierdo para un proyecto y una barra de navegación superior, su interfaz presenta colores claros con precedencia de blancos y grises (Ver Figuras 13 y 14).

En el caso de Forecast.ai (Ver Figuras 15 y 16), los estándares de diseño están basados en *Material Design*, que es el estilo de Google, por el uso del botón clásico “añadir” y la sombra en los componentes de la interfaz (Google, 2018). Los colores, al igual que Stratejos.ai, son claros: blanco y gris, y solo cuenta con una barra de navegación superior.

En cuanto a los estándares de Aptage (Ver Figura 17), en esta herramienta se presentan también los colores claros, pero con precedencia del color verde. El componente de navegación es una barra en la parte superior y submenús.

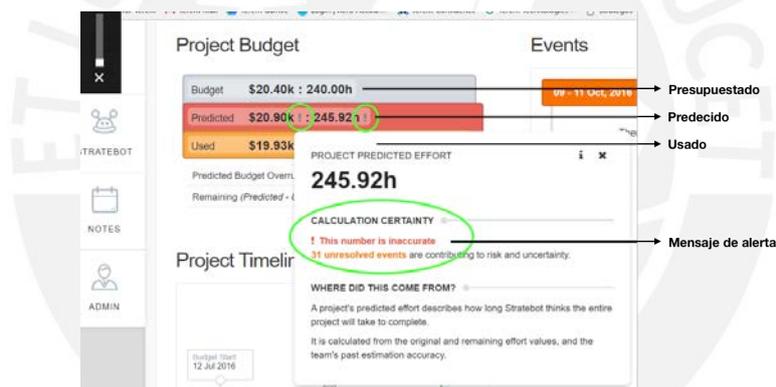


Figura 13. Análisis de riesgo. Adaptado de (Stratejos, 2018).

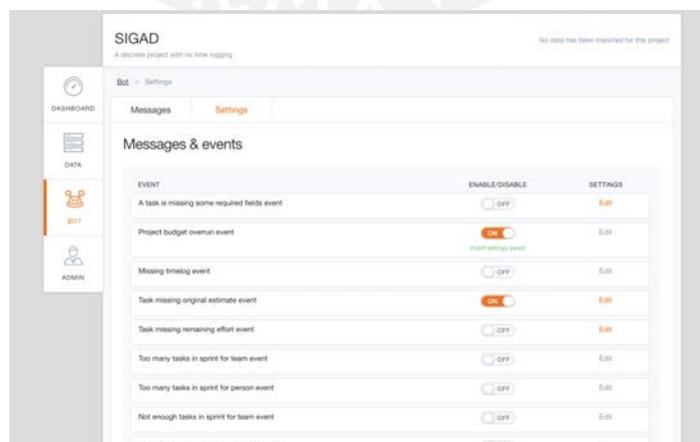
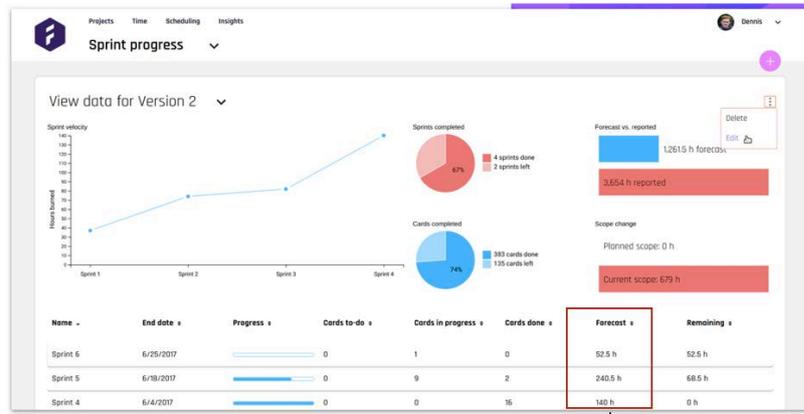
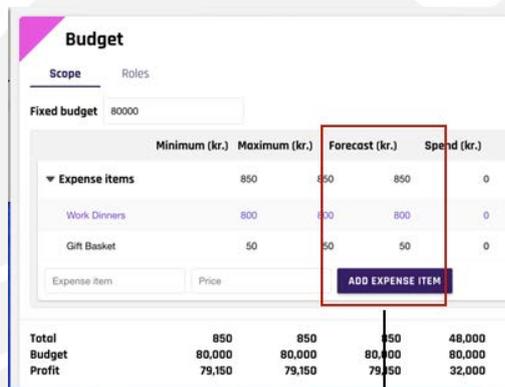


Figura 14. Mensajes & eventos. Adaptado de (Stratejos, 2018).



Valor estimado

Figura 15. Estimación del tiempo. Adaptado de (Forecast, 2018).



Valor estimado

Figura 16. Estimación del costo. Adaptado de (Forecast, 2018).

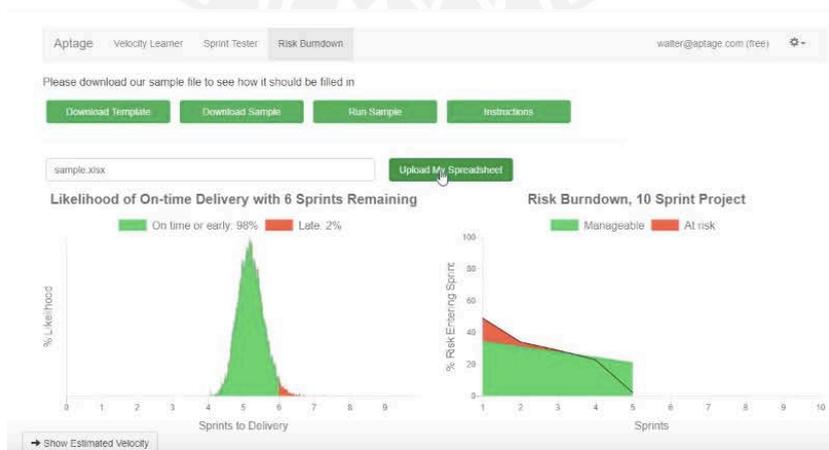


Figura 17. RBd. Tomado de (Aptage, 2017).

3.2.2 Literatura revisada

Las investigaciones revisadas son las siguientes:

- Training data selection for cross-project defect prediction (Herbold, 2013).
- Time-line based model for software project scheduling with genetic algorithms (Chang et al., 2008).
- Applied Machine Learning: Project Management Performance Prediction At Information Technology Company Project Management Office (Saglam, 2017).
- New York City Taxi Trip Duration (Currie, 2017).

1. **Conocer el procedimiento utilizado para predecir características de proyectos y los algoritmos de inteligencia artificial implementados, que pueden ser replicados para resolver la estimación de tiempos.**

Herbold (2013) busca establecer la mejor manera para seleccionar un grupo de proyectos parecidos a un proyecto objetivo y con estos establecer un predictor para estimar el éxito del mismo. Entonces, propone una solución de clasificación binaria (éxito/fracaso del proyecto) basada en distancias en la distribución de las características de la data disponible de los proyectos a través de algoritmos de segmentación o clusterización para finalmente implementar un algoritmo de predicción.

Las estrategias de segmentación que fueron usadas y que pueden ser replicadas para obtener un resultado intermedio en la solución del modelamiento de tiempos son:

- *EM-Clustering*: Se plantea un algoritmo de clusterización en base a todos los proyectos en conjunto con el proyecto a modelar, para luego seleccionar el clúster de proyectos que lo incluya. Finalmente, los proyectos que pertenezcan a ese clúster serán el grupo de entrenamiento para generar el predictor.
- *Nearest Neighbor Selection*: Se usa la menor distancia euclidiana entre todos los proyectos y el proyecto a modelar, para obtener el grupo de entrenamiento con un máximo de elementos k .

Posterior a esta clusterización, los algoritmos de predicción usados fueron los siguientes: Regresión logística, *Naive Bayes*, Redes Bayesianas, *SVM* con *RBF kernel*, árboles de decisión, *RandomForest* y perceptrón multicapa.

Por otro lado, en la búsqueda de algoritmos para la estimación de la duración, Currie (2017) propone una solución de regresión para estimar la duración de los viajes en taxi en la ciudad de New York. En esta, aunque no se soluciona un problema dirigido a proyectos, los algoritmos usados pueden ser replicables: red neuronal *feed-forward*, *RandomForest Regressor*, *Lightgbm*, y *Catboost*.

2. Conocer la comparación entre los algoritmos usados para la predicción sobre características de proyectos y los métodos de medición del éxito del predictor.

Herbor (2013) realiza una comparación sobre los algoritmos previamente mencionados. En los resultados donde no se aplicaron pesos, las Redes Bayesianas superaron por mucho al resto de predictores en términos de *success rate* y *mean recall*, y el peor caso fue *SVM*. En el caso de igualdad de pesos, *SVM* fue el mejor seguido por redes bayesianas.

Es también importante mencionar a Chang et. al (2008), ya que en este artículo se realiza un experimento para medir el éxito de su predictor de esfuerzo en proyectos. Se experimentó la comparación del algoritmo con las estimaciones de dos expertos (gestores de proyectos senior), los que demoraron en promedio 3 horas en dar una solución para un problema, en cambio el algoritmo planteado en 33 minutos ya tenía una solución, por tanto se mejoró la eficiencia del tiempo en la obtención de la estimación en 82%, además de que ambos obtuvieron costos de 546353 y 559920 respectivamente; sin embargo, el algoritmo obtuvo el mínimo costo de 385818, y en promedio las soluciones no óptimas pero aceptables fueron 527104, por tanto la mejora en la estimación del costo comparado al promedio de soluciones no óptimas es del 3.52% y 5.86% para cada experto (4.69% en promedio).

Por otro lado, Currie (2017) utiliza como método de comparación el Error medio cuadrático mínimo o RMSE por sus siglas en inglés. Un valor menor indica un mejor regresor. Asimismo, en la competencia en la que se presentó esta solución, califica los modelos en base a el Error logarítmico cuadrático medio o RMSLE. En esta, el mejor resultado y ganador del concurso obtuvo un valor de 0.28976 de RMSLE; sin embargo, se va a considerar un valor de 0.3719 ya que es el máximo valor dentro de los 100 primeros puestos de la competencia (Kaggle, 2017).

3. Conocer la cantidad de datos de proyectos evaluados y las estrategias de selección de la data para los algoritmos de segmentación o clusterización.

En Herbold (2013) se evaluaron 44 data sets obtenidos de 14 proyectos de software de código abierto. La aproximación planteada es determinar la similitud entre proyectos, en base a características de distribución (estadísticas), ya que según He et al., existe una correlación entre estas métricas y la factibilidad de escoger ese proyecto como parte del grupo de entrenamiento (Como se citó en Herbold, 2013). Estos son: el promedio, la mediana y la desviación estándar.

4. Conocer las fases de desarrollo replicables para la implementación de un algoritmo de Inteligencia Artificial en un sistema.

Saglam (2017) establece un procedimiento, que puede ser replicado, para el desarrollo de un sistema de inteligencia de negocios (*Business intelligence*) reforzado con algoritmos de inteligencia artificial, en este caso ML (*Machine Learning*).

Primero, se desarrolló la siguiente interfaz de usuario en la que se ingresan valores del proyecto y el sistema calcula valores de la gestión del valor ganado (EVM). En este punto, lo replicable es el formato de ingreso de valores y las variables que se consideraron (Ver Figura 18) como el presupuesto, el tamaño del equipo, la duración del proyecto, los días consecutivos, el costo actual para generar un valor planificado, los canales de comunicación, el estimado para terminarlo y al terminarlo.

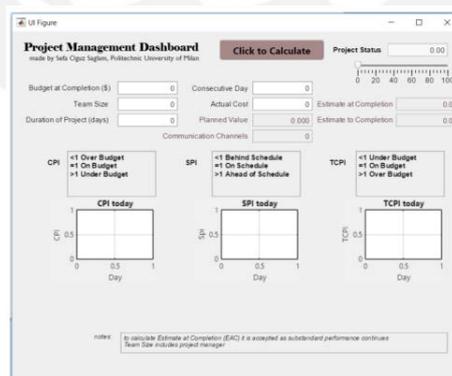


Figura 18. Interfaz de usuario. Tomado de (Saglam, 2017).

Seguidamente, se procedió a recolectar datos por 50 días, para luego implementar el algoritmo de aprendizaje de máquina en base a redes neuronales. Finalmente, el algoritmo fue aplicado para predecir valores del EVM.

5. Conocer las conclusiones a las que llegaron las investigaciones.

Herbold (2013) concluye que los resultados de los algoritmos de inteligencia artificial para la predicción del fracaso de proyectos cruzados, muestran una mejora en el ratio de éxito; sin embargo, menciona también que aún no presenta calidad suficiente para competir con la predicción dentro de un proyecto.

Por otra parte, Chang et. al (2008) en las conclusiones del artículo, indican claramente que el algoritmo de inteligencia artificial superó en gran medida los resultados de las predicciones de los gestores sobre el esfuerzo dentro de un proyecto; sin embargo, los mismos autores reafirman que su modelo debe ser una herramienta de soporte.

Por último, Saglam concluye afirmando la importancia de la inteligencia artificial en la gestión de proyectos y su validez en la predicción, además de que se da hincapié a interrelacionar varios departamentos y considerar otros efectos en la realización de proyectos.

3.3 Conclusiones del estado del arte

En primer lugar, los softwares comerciales presentados orientan sus productos a metodologías ágiles o basados en tareas y sus soluciones no aplican la gestión para resultados. Estos serán considerados en cuanto a su forma de presentar las estimaciones y sus estándares de diseño. Por otro lado, existen conceptos que son transversales en las investigaciones como el aprendizaje de máquina que, de ser implementados en proyectos de desarrollo, podrían mejorar significativamente el impulso que se le puede dar a un país o institución a través de un software con ML.

La prevalencia de gestión de proyectos de software (SPM) se da por la disponibilidad de la data ya que los *datasets* públicos son en su mayoría de este tipo. La ausencia de investigaciones para el modelamiento de tiempos de proyectos en empresas privadas y mucho menos en instituciones públicas conlleva a un nivel de desconocimiento inicial sobre los factores de importancia en los proyectos de desarrollo; sin embargo, gracias a que el punto de partida es la gestión de proyectos, esta es replicable.

Por último, las investigaciones avizoran la persistencia de la inteligencia artificial en la gestión de proyectos para conseguir mejores resultados, lo que brinda motivación para seguir con el proyecto en la búsqueda de la integración de proyectos de desarrollo más inteligencia artificial.

Capítulo 4. Especificación de Requisitos

En esta sección se define el catálogo de requisitos.

4.1 Objetivo

El objetivo de esta sección es especificar los requisitos funcionales y no funcionales del software de gestión de proyectos de desarrollo PM4R (en adelante software), así como la priorización de los mismos. Finalmente se realiza una verificación de consistencia y trazabilidad de los mismos.

4.2 Requerimientos funcionales

(Ver Apéndice 1)

4.3 Requisitos no funcionales

(Ver Apéndice 1)

4.4 Rango de valoración

- Prioridad: determina la importancia del requisito para la implementación del software. (Ver Apéndice 1)
- Exigible: determina si el requisito será desarrollado en el presente software o posteriormente. (Ver Apéndice 1)

4.5 Verificación del resultado

4.5.1 Consistencia

No deben existir contradicciones entre requisitos para lo cual una propuesta es realizar un prototipado de la interfaz de usuario para entender más fácilmente los requisitos (Junta de Andalucía, 2018c). Se presenta un *wireframe* inicial considerando los requisitos exigibles (Ver Apéndice 2).

Con el *wireframe* realizado se ha verificado la consistencia de los requisitos ya que estos no se contradicen y desarrollan un flujo coherente del software.

4.5.2 Trazabilidad

La trazabilidad de los requisitos está definida por la prioridad y las fechas de las actividades del proyecto (Ver Tabla 6).

Capítulo 5. Diseño de interfaces

En esta sección se definen los estándares de diseño y los *mockups* con un flujo base.

5.1 Objetivo

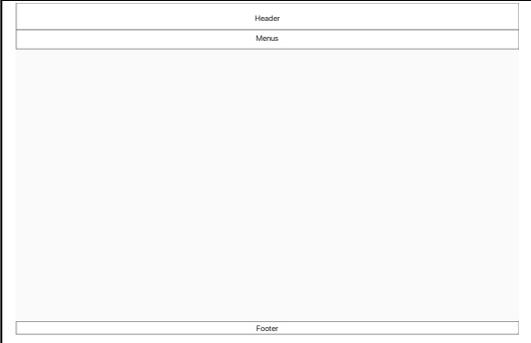
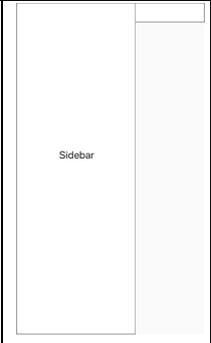
El objetivo de esta sección es definir los estándares de interfaz gráfica a utilizar durante el desarrollo del software, lo que incluye el esquema general, fuentes, paleta de colores, *mockups* y librerías específicas. Finalmente, la verificación se hace a través de los lineamientos de usabilidad de Nielsen.

5.2 Esquema general

Se definen las características generales de la aplicación (Ver Tabla 10), considerando el *wireframe* realizado en el capítulo anterior:

- Se utilizará el estándar “*Google Material Design*” para todos los elementos de la interfaz.
- Se tendrá una cabecera, un pie de página y una barra de menús.
- La barra de navegación lateral solo aparecerá en un dispositivo móvil, esta incluirá todas las opciones disponibles en la cabecera de la aplicación de escritorio.
- Los elementos incluidos en la sección principal como botones, inputs, dropdowns, serán elementos definidos en el estándar *Material Design*.

Tabla 10. Esquema general. Elaboración propia.

Versión escritorio	Versión móvil
 <p>Header</p> <p>Menu</p> <p>Footer</p>	 <p>Sidebar</p>

5.3 Fuentes

Se utilizará la fuente Roboto con estilos Regular, Thin y Bold, los tamaños a utilizar dependiendo del elemento en que la fuente sea utilizada variarán entre los tamaños de 14, 16 y 18 puntos (Ver Tabla 11).

Tabla 11. Fuentes. Elaboración propia.

Detalle	Fuente	Tamaño	Estilo
Títulos	Roboto	18	Bold
Subtítulos		16	Regular
Texto		14	Regular
Botones		14	Regular

5.4 Paleta de colores

Se utilizarán las paletas de colores definidas en la documentación de *Material Design* (Ver Figura 19).

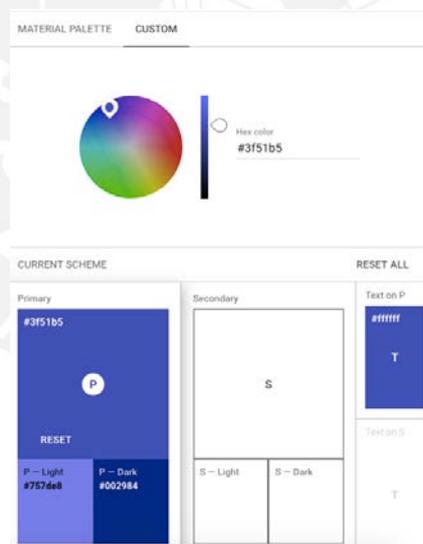


Figura 19. Paleta de colores. Tomado de (Angular Material, 2018).

5.5 Mockups y flujo base

El *wireframe* realizado en el resultado anterior fue perfeccionado y adecuado a los estilos actualmente definidos (Ver Apéndice 3) con la herramienta Sketch. El flujo base entre pantallas está disponible en:

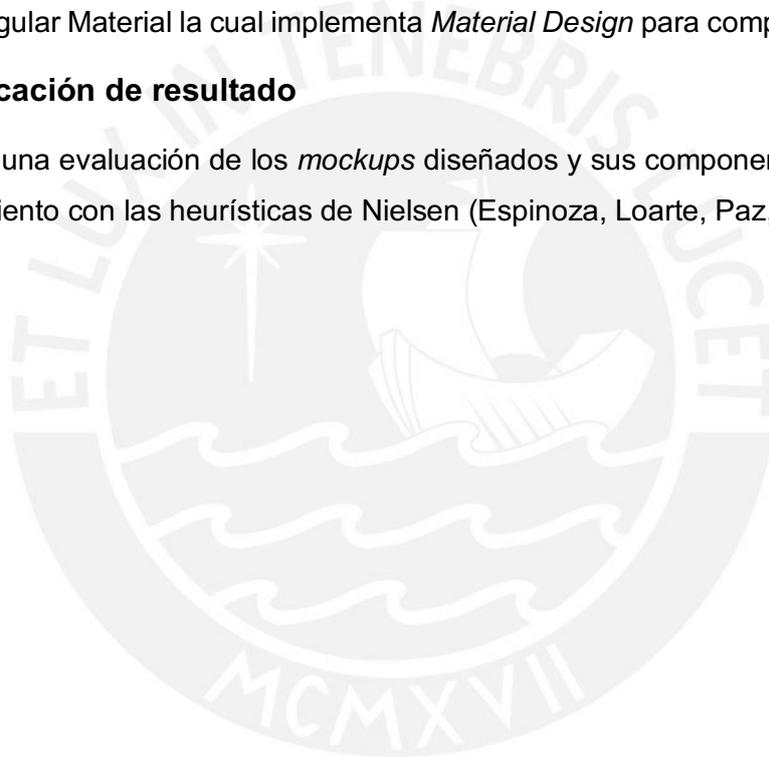
<https://marvelapp.com/7e164hb/screen/49848726>.

5.6 Librerías

Se utiliza el estándar “Google material”, para lograr esto en la plataforma se utilizará la librería Angular Material la cual implementa *Material Design* para componentes Angular.

5.7 Verificación de resultado

Se realizó una evaluación de los *mockups* diseñados y sus componentes para verificar el alineamiento con las heurísticas de Nielsen (Espinoza, Loarte, Paz, & Flores, 2019).



Capítulo 6. Diseño de arquitectura

En esta sección se detalla la arquitectura del software mediante distintos diagramas bajo el estándar UML.

6.1 Objetivo

El objetivo de esta sección es mostrar la especificación de la arquitectura a través de los siguientes diagramas necesarios para la solución ofrecida: diagrama de clases, entidad relación, componentes y despliegue. Asimismo, se justifica el uso de una arquitectura cliente-servidor. Finalmente se hace una verificación del comportamiento deseado a través de un reporte técnico de los *endpoints* y adicionalmente el cumplimiento de los estándares UML.

6.2 Metas y Restricciones

6.2.1 Metas

- La arquitectura permitirá el cumplimiento de todos los puntos especificados en el catálogo de requisitos.
- La arquitectura soportará el uso de las tecnologías elegidas.

6.2.2 Restricciones

- El sistema emplea una base de datos PostgreSQL.
- El sistema contará con una plataforma desarrollada en Angular 6 y Django Rest.

6.3 Diagramas

6.3.1 Diagrama de componentes (Alto Nivel)

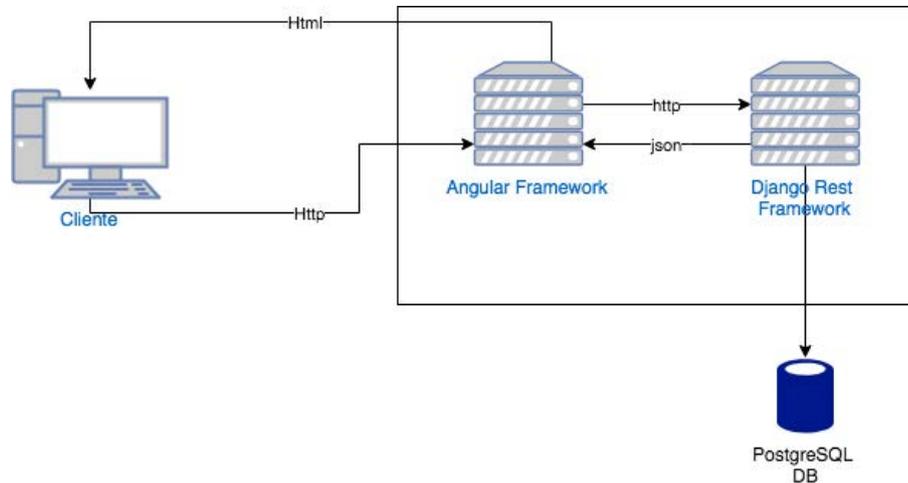


Figura 20. Diagrama de componentes (Alto Nivel). Elaboración propia.

Componentes:

- Cliente: Navegador web o móvil que ingresa a la plataforma web.
- Angular: *Framework Front-end* que permite el desarrollo multiplataforma. Se basa en Typescript y SASS. La versión 6.0 cuenta con una integración de componentes de *Material Design*.
- Django Rest: *Framework back-end* que permite el desarrollo de un API REST para exponer diferentes funcionalidades, que serializa las clases en formato JSON.

6.3.2 Diagrama de componentes (Bajo Nivel)

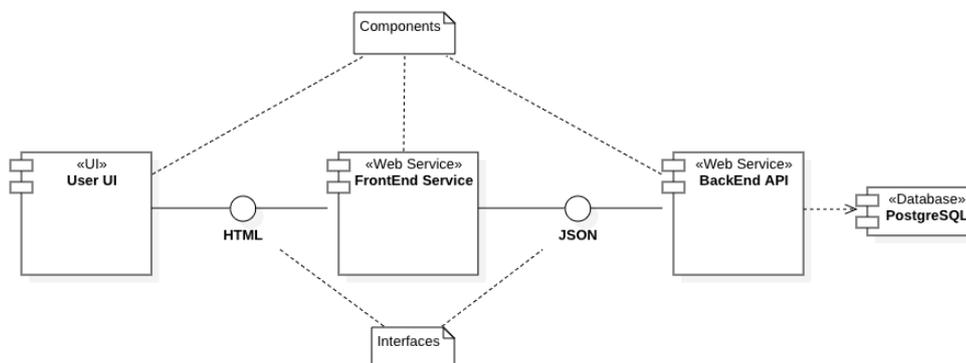


Figura 21. Diagrama de componentes (Bajo Nivel). Elaboración propia.

Componentes:

- User UI: Componente de software del usuario mediante el cual interactúa con el software, con estereotipo UI (Interfaz de usuario)
- *Front-End* Service: Componente de software desarrollada en Angular 6 y desplegado en el servidor para el manejo de urls e interacción con las páginas y elementos DOM. Interactúa con la interfaz del usuario a través de *http request* y retorna elementos HTML.
- *Back-End* API: Componente de software desarrollado en Django Rest y desplegado en el servidor. Interactúa con el servicio de *Front-End* a través de *http request* y JSON. Tiene una relación de dependencia con la base de datos.
- PostgreSQL: Componente de base de datos.

6.3.3 Diagrama de despliegue

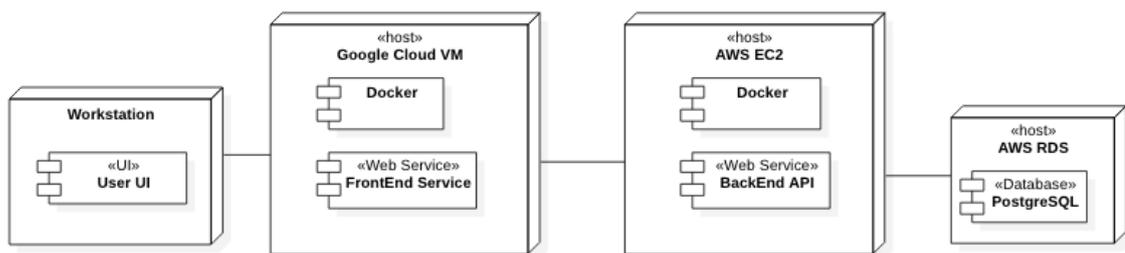


Figura 22. Diagrama de despliegue. Elaboración propia.

Los nodos representan el hardware en el cual están los componentes (software):

- Workstation: Computadora o móvil del usuario con el que accede a una URL específica.
- Host: Servidor donde están desplegados el servicio de *front* y *back-end*, al igual que la Base de datos.
 - o Docker: Se dockerizó ambos servicios (*front* y *back-end*) para una facilidad mayor en cuanto a la actualización de los mismos, ya que, gracias a este, solo es necesario volver a generar un contenedor en el cual ya se encuentran todas las librerías necesarias y con *docker-compose* se automatiza el levantamiento de los servicios en un puerto con comandos específicos dependiendo de las tecnologías usadas.

6.3.4 Diagrama de clases

A través del modelado de estas clases, el *framework* Django Rest genera las tablas en la base de datos (Ver Apéndice 4).

6.3.5 Diagrama Entidad Relación (DER)

Se cuenta con 38 tablas (Ver Apéndice 5), un grupo generadas por Django y otro por los modelos implementados. Se detallan las más importantes:

- mainapp_person: Tabla de usuarios.
- mainapp_acquisition: Tabla de adquisiciones del proyecto.
- mainapp_acquisition_x_project: Tabla de adquisiciones por proyecto.
- mainapp_activity: Tabla de actividades.
- mainapp_asignation: Tabla de asignaciones.
- mainapp_communication: Tabla de comunicaciones.
- mainapp_communication_x_project: Tabla de comunicaciones por proyecto.
- mainapp_component: Tabla de componentes.
- mainapp_country: Tabla de países.
- mainapp_deliverable: Tabla de entregables.
- mainapp_goal: Tabla de objetivos.
- mainapp_monthly_cost: Tabla de costos mensuales.
- mainapp_product: Tabla de productos.
- mainapp_project: Tabla de proyectos.
- mainapp_project_goals: Tabla de proyectos por objetivos.
- mainapp_project_sectors: Tabla de proyectos por sector.
- mainapp_project_users: Tabla de proyectos por usuario.
- mainapp_region: Tabla de regiones.
- mainapp_responsibility: Tabla de responsabilidades.
- mainapp_risk: Tabla de riesgos.

- mainapp_risk_x_project: Tabla de riesgos por proyecto.
- mainapp_sector: Tabla de sectores del proyecto.
- mainapp_subcomponent: Tabla de subcomponentes.
- mainapp_workpackage: Tabla de paquetes de trabajo.

6.4 Justificación de la arquitectura

Debido a que se busca integrar módulos de aprendizaje de máquina en una plataforma web, además de hacer uso de librerías como *scikit-learn* (implementada en lenguaje Python) para realizar un proceso de estructuración para la generación dinámica de modelos y un tuneo inicial de estas funciones en base a procesos iterativos y, además, estar al margen de los estándares de Google Material, se utiliza un diseño cliente-servidor, para separar la lógica de negocio y la interfaz gráfica. De tal manera que, el primero (*back-end*) cuente con independencia del segundo (*front-end*) referido a la tecnología usada. En este sentido, el *back-end* se desarrolla en Django Rest por el soporte al lenguaje Python y la extensa documentación para exponer diferentes funcionalidades en un API, lo que permite la escalabilidad de esta solución a diferentes estructuras o *frameworks front-end*, que en este caso se selecciona Angular 6 por su integración con los estándares Google Material y su continuo soporte.

6.5 Verificación de resultado

Se realiza un reporte técnico para verificar el comportamiento deseado de la arquitectura cliente-servidor. Se hace uso de una plantilla propia para probar las funcionalidades expuestas acorde a los requisitos previamente definidos (Ver Apéndice 6).

Con este reporte, se ha visualizado la trazabilidad de los requisitos, ya que se exponen las funcionalidades necesarias para estos e incluso adicionales que permiten la escalabilidad de la arquitectura para otras estructuras o *frameworks front-end*.

Adicionalmente, los diagramas, exceptuando el de alto nivel, cumplen con estándares definidos por UML.

Capítulo 7. Implementación de la plataforma

En esta sección se detalla el proceso de desarrollo del software a través de las distintas herramientas.

7.1 Objetivo

El objetivo de esta sección es presentar el software generado en base a las actividades realizadas y las herramientas utilizadas, para finalmente mostrar el modo de verificación a través de las pruebas.

7.2 Back-End

Existen varios aspectos a considerar una arquitectura cliente-servidor para el desarrollo del *back-end*. En primera instancia está la forma en la que se hará la transmisión de datos para luego definir las rutas. En este caso, se definió el uso de JSON (*JavaScript Object Notation*) y el uso de la librería *routers* en Python.

El siguiente paso es definir los modelos/clases que son los que generan la base de datos mediante migraciones con el *framework* de Django, las vistas (interactúa con las clases y serializadores) y los serializadores (transforma una clase/objeto en JSON o a la inversa) (Ver Apéndice 7). Además, se utilizaron los modelos de Django-auth mediante herencia para la definición de usuarios.

Finalmente se definieron algunas configuraciones, como el tamaño de página en 50 (cantidad de JSON a enviar como máximo por página/url) y la conexión con la base de datos PostgreSQL.

7.3 Front-End

Se manejaron los conceptos de componentes y rutas (Ver Apéndice 8: Tabla 22), módulos (Ver Apéndice 8: Tabla 23), servicios (Ver Apéndice 8: Tabla 24) y clases idénticas a los modelos de la Tabla 17.

Los componentes representan una página con la que interactúa un usuario y contienen la lógica (Typescript) y el diseño (HTML, SASS), los módulos son librerías externas o propias, los servicios permiten establecer las llamadas http y las clases son las definiciones de los objetos a ser usados en Typescript. Además, se implementó un módulo para el *routing* o ruteo, mediante el cual el usuario interactúa con los componentes.

Además de ello, se adaptó un diseño para las tablas usadas en base a un código abierto en SASS de una tabla con estilos de Google Material (Goresht, 2015) para ajustarse a los *mockups* definidos en capítulo 5. De igual forma, se adaptó la librería Frappe Gantt, licenciada por el MIT (Frappe Technologies Pvt., n.d.), para el componente Project-step-2.

7.4 Verificación del resultado

7.4.1 Pruebas

En esta sección se describen las pruebas unitarias y de integración realizadas sobre el software y las funcionalidades establecidas en el catálogo de requisitos.

1. Pruebas unitarias

Son aquellas pruebas que aseguran el correcto funcionamiento de los componentes del código de forma aislada (Junta de Andalucía, 2018b). En este sentido, se realizaron pruebas unitarias relacionadas a los requisitos planteados en el Capítulo 4, con la definición de prerrequisitos y la descripción de los casos de prueba. (Ver Apéndice 9).

2. Pruebas de integración

Son aquellas pruebas que aseguran el correcto funcionamiento de todo el sistema, es decir los componentes integrados (Junta de Andalucía, 2018a). Para esto, se relacionaron múltiples requisitos con una descripción del caso de prueba, los prerrequisitos y el resultado obtenido (Ver Apéndice 10).

Capítulo 8. Generación y almacenamiento de datos

En esta sección se detalla el proceso de generación y almacenamiento de datos de prueba de proyectos de desarrollo para el software implementado en el capítulo anterior.

8.1 Objetivo

El objetivo de esta sección es presentar el módulo generador de datos, con el fin de permitir el desarrollo subsiguiente de los modelos de clusterización y estimación (clasificación y regresión) en base a datos iniciales. Finalmente mostrar los medios de verificación.

8.2 Módulo generador de datos

Para la generación de datos de proyectos de desarrollo se realizaron los siguientes pasos:

- Obtención de un dataset del World Bank: datos públicos de las operaciones y proyectos del Banco Mundial, disponible en Kaggle.
- Sobre este dataset se realizaron técnicas de tratamiento de datos como tokenización y la eliminación de caracteres especiales.
- Paralelamente, se obtuvo datos de ejemplo de proyectos de desarrollo públicos como los del Gobierno de Paraguay (Nuñez, 2014) para tener como referencia la estructura interna de los proyectos (componentes, subcomponentes, productos, entregables, paquetes de trabajo, actividades, etc).
- Se buscaron datos de ejemplo de usuarios y actividades.
- En base a los elementos anteriores y en conjunto con el docente experto en PM, se definieron constantes de los máximos por proyecto.
 - o MAX_COUNT_USERS = 20, máxima cantidad de usuarios por proyecto.
 - o MAX_COUNT_COMPONENTS = 4, máxima cantidad de componentes por proyecto.
 - o MAX_COUNT_SUBCOMPONENTS = 5, máxima cantidad de subcomponentes por componente.
 - o MAX_COUNT_PRODUCTS = 7, máxima cantidad de productos por subcomponente.

- MAX_COUNT_DELIVERABLES = 6, máxima cantidad de entregables por producto.
- MAX_COUNT_WORKPACKAGES = 6, máxima cantidad de paquetes de trabajo por entregable.
- MAX_COUNT_ACTIVITIES = 6, máxima cantidad de actividades por paquete de trabajo.
- MAX_COUNT_ACQUISITIONS = 10, máxima cantidad de adquisiciones por proyecto.
- MAX_COUNT_RISKS = 10, máxima cantidad de riesgos por proyecto.
- MAX_COUNT_COMMUNICATIONS = 6, máxima cantidad de comunicaciones por proyecto.
- MAX_COUNT_RESPONSIBILITIES = 1, una responsabilidad por cada usuario dentro del proyecto.
- PROP_TASKS = 0.28, Proporción de duración de la tarea entre fecha inicial de la tarea y fecha final del proyecto, aleatorio entre 0 y 0.28.
- PROP_ACQUISITION = 0.6, Proporción de duración de la adquisición entre fecha inicial de la adquisición y fecha final del proyecto, aleatorio entre 0 y 0.60.
- PROP_COST_IN_ACQUISITION = 0.4, proporción del costo del proyecto en adquisiciones, aleatorio entre 0 y 0.4.
- PROP_COST_IN_WORKPACKAGES = 0.5, proporción del costo total del Proyecto en los paquetes de trabajo, aleatorio entre 0 y 0.5.

Así, por ejemplo, un proyecto puede tener los siguientes valores:

Cantidad de usuarios: 8

Cantidad de componentes: 3

Cantidad de subcomponentes: 9

Cantidad de productos: 26

Cantidad de entregables: 96

Cantidad de paquetes de trabajo: 504

Cantidad de actividades: 1743

Cantidad de adquisiciones: 8

Cantidad de riesgos: 8

Cantidad de comunicaciones: 2

Cantidad de responsabilidades: 8

Costo Real total: 4400000000.0

- Adicionalmente se generó data en los campos suplementarios como descripciones en base a un diccionario de palabras recurrentes.
- Finalmente se utilizó la librería *random* en Python para la selección de elementos sobre listas.

Con estos pasos, se desarrolló un módulo generador de datos, en el que se implementa la lectura de los *datasets* y archivos csv, la definición de las constantes como los máximos y las funciones respectivas por cada elemento de un proyecto los que son almacenados en los modelos de base de datos.

8.3 Verificación del resultado

A través de este módulo, se generaron 132 usuarios, 14 sectores, 7 regiones, 206 países, 3 objetivos, 272 proyectos, 701 componentes, 1774 subcomponentes, 7307 productos, 25448 entregables, 139978 paquetes de trabajo, 490829 actividades, 1547 adquisiciones, 1468 riesgos, 963 comunicaciones y 2747 responsabilidades. Con esto se cumple con una cantidad mayor a 50 proyectos según la guía para escoger el mejor predictor.

Capítulo 9. Definición de atributos y pre-procesamiento de datos

En esta sección se definen los atributos iniciales para los algoritmos de *clustering* y predicción. Además, se detalla el proceso realizado para el pre-procesamiento de la data y la selección de los más importantes previa la generación de las muestras.

9.1 Objetivo

El objetivo de esta sección es establecer el flujo automatizado para definir los atributos de los datos para los algoritmos de ML. Finalmente mostrar los medios de verificación.

9.2 Atributos

A partir de las clases definidas y los datos generados en el capítulo anterior, el siguiente paso es obtener los atributos, los cuales pueden ser estáticos o dinámicos. Los primeros no se verán afectados si se generan nuevos valores en la base de datos, por ejemplo, las cantidades (Ver Apéndice 11: Tabla 44). Caso contrario, los dinámicos dependerán de los valores disponibles, por ejemplo, al crear un nuevo Sector, se agrega un atributo nuevo que determina si un proyecto pertenece o no a este nuevo sector (Ver Apéndice 11: Tabla 45).

9.3 Pre-procesamiento de datos

9.3.1 Actividades de pre-procesamiento

Para la etapa de pre-procesamiento, se consideran las siguientes actividades en el orden presentado:

- Formato: Se obtienen los atributos previamente definidos a partir de los modelos de la base de datos.
- Limpieza: Eliminar las filas con campos nulos.
- Transformación de los datos
 - o Agregación: los atributos como la cantidad de componentes, subcomponentes, productos, entregables, paquetes de trabajo, actividades, adquisiciones, riesgos, comunicaciones y responsabilidades, se suman en una sola variable denominada “*elements*”.

- Normalización: evita que los grandes intervalos entre valores genere una tendencia a favor de estos (Alatrística Salas, 2016). Por tanto, se implementó una función para realizar una normalización en base a dos alternativas:

- *Min-Max*

$$z = \frac{x - \min(x)}{\max(x) - \min(x)}$$

- *Z-score*

$$z = \frac{x - \text{mean}(x)}{\sigma(x)}$$

- *Discretización*: Esto se aplica como actividad adicional para el módulo predictor de clasificación, para convertir los datos numéricos en categorías.
 - Transformación de características: selección de atributos en base a PCA (*Principal Component Analysis*), el cual busca obtener aquellos que ofrezcan una mayor varianza acumulada. El valor mínimo es parametrizable en la plataforma e inicia con 0.85.
- Muestreo: Esta actividad se aplica en el módulo predictor, y se obtiene una partición de la data total en dos segmentos: 80% de datos para el entrenamiento y un 20% para prueba.

9.3.2 Diagrama de flujo

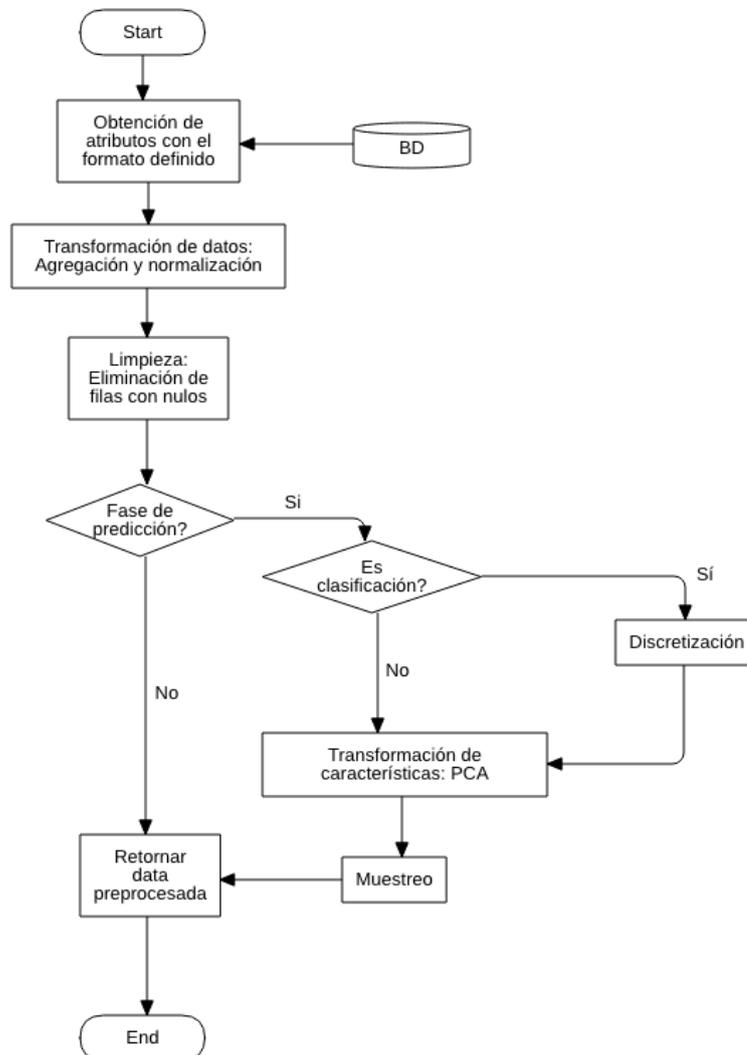


Figura 23. Diagrama de flujo – pre-procesamiento. Elaboración propia.

9.4 Verificación del resultado

Se han cumplido con las actividades para el pre-procesamiento de los datos (Alatriza Salas, 2016). Por otro lado, el código integra pruebas unitarias para verificar que se cumpla el flujo propuesto.

Capítulo 10. Algoritmo de clustering

En esta sección se implementa una función para obtener el mejor modelo de *clustering* aplicados a la data pre-procesada del capítulo anterior.

10.1 Objetivo

El objetivo de esta sección es verificar si el *clustering* es factible para obtener el mejor modelo en base a la comparación del SSW y SSB a través del coeficiente de Silhouette. Caso contrario, este procedimiento no será tomado en cuenta para la fase de predicción. Finalmente se muestran los medios de verificación.

10.2 Diagrama de flujo

El proceso realizado es iterativo e inicia con la selección del tipo de normalización (Min-Max y Zscore), luego el pre-procesamiento de los datos a partir de la data normalizada, la obtención de los modelos válidos de *clustering* con coeficiente de Silhouette mayor a 0.6 (scikit-learn, 2017) para los algoritmos de *K-means* y *EM* y la selección del mejor modelo válido (Ver Figuras 24 y 25).

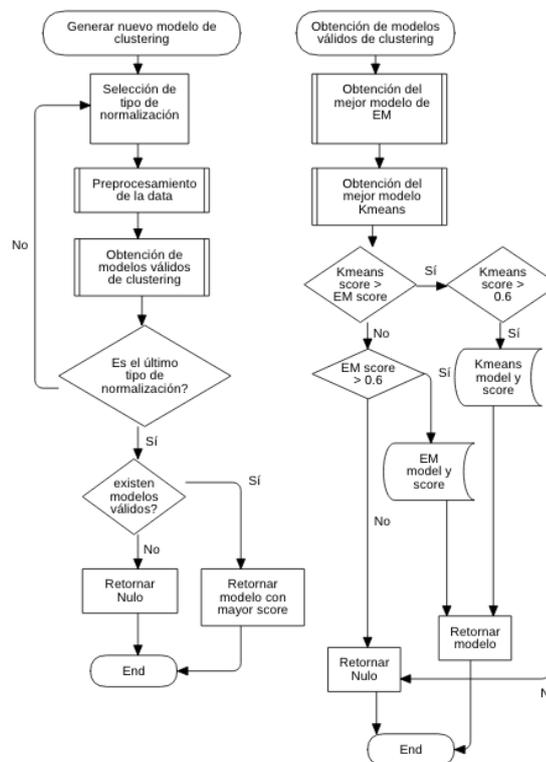


Figura 24. Diagrama de flujo – clustering parte 1. Elaboración propia.

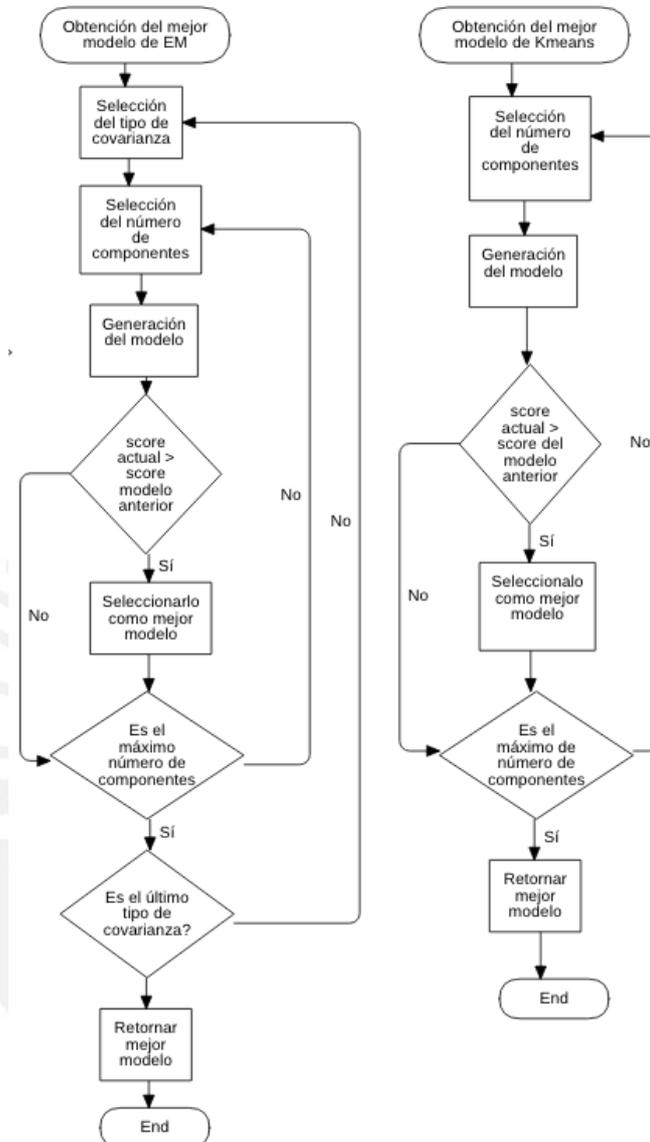


Figura 25. Diagrama de flujo – clustering parte 2. Elaboración propia.

10.3 Verificación del resultado

La verificación del resultado se realiza como parte del proceso, al evaluar el coeficiente de Silhouette como indicador de una buena o mala clusterización considerando la cohesión y separación de clústeres. Por otro lado, el código integra pruebas unitarias para verificar que se cumpla el flujo propuesto

Capítulo 11. Módulo predictor

En esta sección se implementa el módulo generador de modelos dinámicos de estimación de la duración de proyectos para el software desarrollado.

11.1 Objetivo

El objetivo de esta sección es detallar el flujo realizado para la generación del mejor modelo, tanto de clasificación como de regresión, para la estimación de la duración de proyectos.

11.2 Clasificación

La clasificación se realiza en base a una lista de posibles algoritmos a ser usados, los que son: Regresión logística, *Naive Bayes*, *SVM* con *RBF kernel*, árboles de decisión, *RandomForest* y perceptrón multicapa. Previo a este proceso iterativo de selección del mejor modelo de clasificación, se realiza el pre-procesamiento revisado en capítulo 9 (normalización, *discretización*, PCA y muestreo de los proyectos). Luego, los algoritmos son comparados en base a f1-score (ponderado de *precision* y *recall*) (scikit-learn developers (BSD License), 2014) y finalmente se valida si el modelo seleccionado supera los valores de 0.7 y 0.5 en *precision* y *recall* (Ver Figura 26), para ser considerado como un modelo aceptable (Herbold, 2013).

11.2.1 Diagrama de flujo

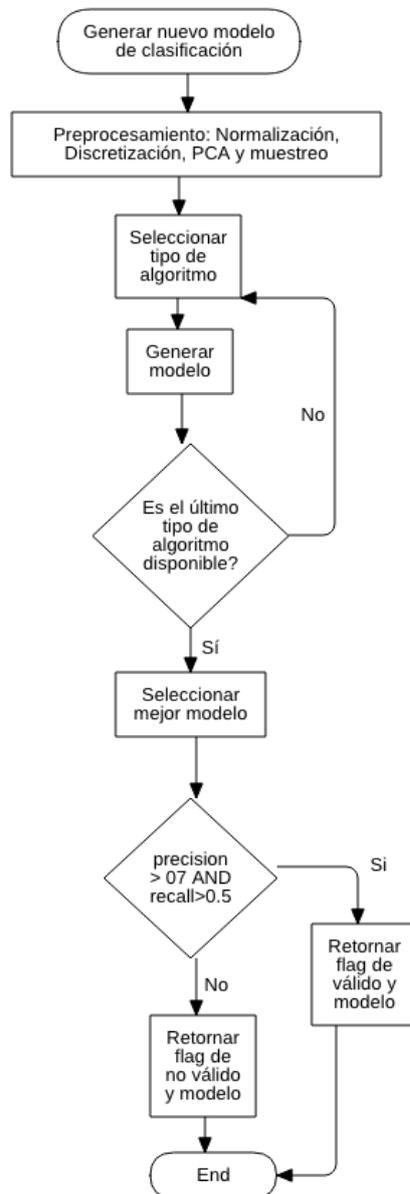


Figura 26. Diagrama de flujo - clasificación. Elaboración propia.

11.3 Regresión

La regresión se realiza sin un proceso de *discretización* y se busca la generación dinámica de modelos predictivos en base los siguientes algoritmos presentes en la bibliografía revisada: *RandomForest Regressor* y *Catboost* (Currie, 2017). Previo a este proceso iterativo de selección del mejor modelo de regresión, se realiza el pre-procesamiento revisado en capítulo 9 (normalización, PCA y muestreo de los proyectos).

Luego, los algoritmos son comparados en base al RMSE, siendo el menor valor el seleccionado y finalmente se valida si este modelo supera el valor de 0.3719 en RMSLE (Ver Figura 27) para ser considerado como un modelo válido (Kaggle, 2017).

11.3.1 Diagrama de flujo

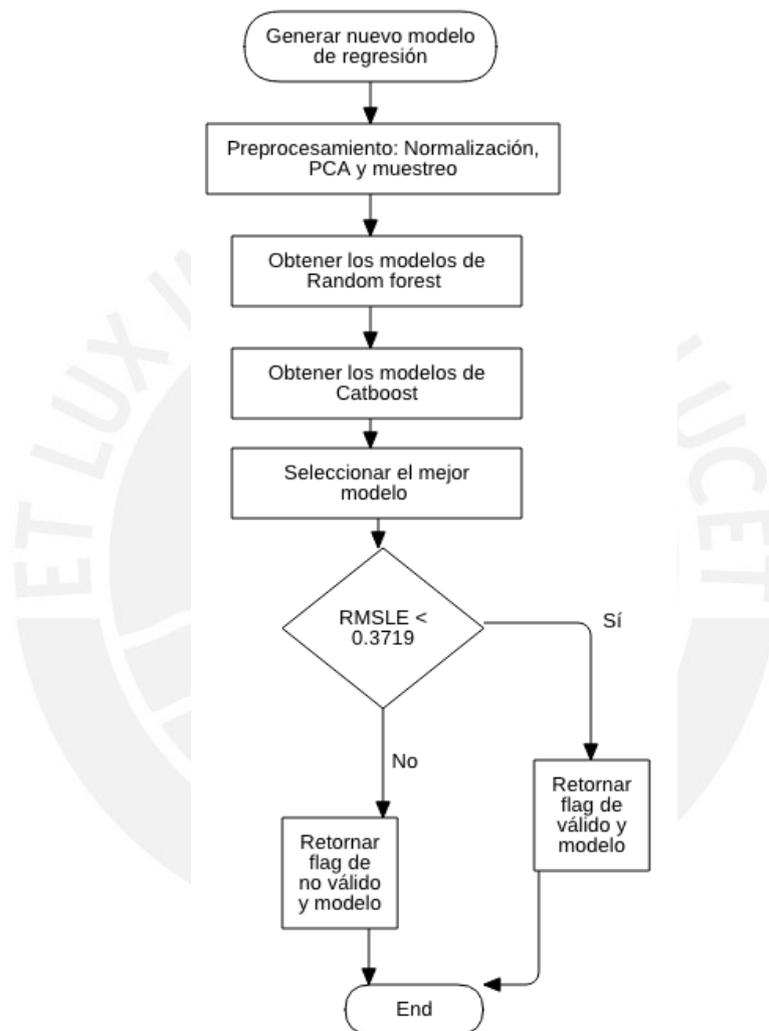


Figura 27. Diagrama de flujo - regresión. Elaboración propia.

11.4 Verificación del resultado

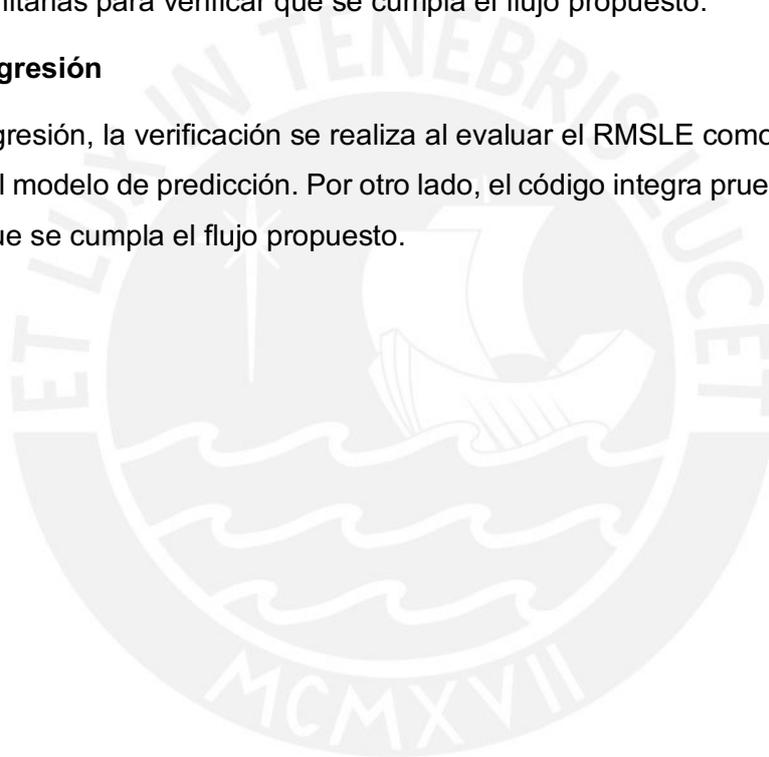
Para ambos tipos de modelos predictivos, parte de la verificación del resultado se realiza durante el proceso iterativo de selección considerando la factibilidad o no de la clusterización, y de las métricas de selección. Se detalla a continuación.

11.4.1 Clasificación

Para la clasificación, la verificación se realiza al evaluar *precision* y *recall* como indicadores de un buen o mal modelo de predicción. Por otro lado, el código integra pruebas unitarias para verificar que se cumpla el flujo propuesto.

11.4.2 Regresión

Para la regresión, la verificación se realiza al evaluar el RMSLE como indicador de un buen o mal modelo de predicción. Por otro lado, el código integra pruebas unitarias para verificar que se cumpla el flujo propuesto.



Capítulo 12. Análisis del error

En esta sección se presenta el análisis del error asumido en la predicción de la duración de proyectos, considerando la muestra de 272 proyectos generados en el capítulo 8.

12.1 Objetivo

El objetivo de esta sección es comparar el error asumido en el módulo predictor del capítulo anterior y evaluar la mejora obtenida a medida que se incrementa la data en la plataforma.

12.2 Clasificación

Para la clasificación, se considera el caso de ejemplo en el que se ha discretizado los datos de la duración de los proyectos en 3 conjuntos: Alta duración (*high*), duración media (*medium*) y baja duración (*low*). Sin embargo, estos rangos serán parametrizables en la plataforma (Ver Apéndice 3).

12.2.1 Métricas

Se utilizan dos métricas para validar el éxito de los clasificadores: la primera es F1 score que considera *precision* y *recall*. Por otro lado, se muestra el *accuracy*, el error y la desviación estándar obtenidos con el método de *cross-validation*.

1. F1 score

Se muestran a continuación los valores de F1 score para los distintos algoritmos planteados (Ver Tabla 12).

Tabla 12. Score obtenido con la métrica F1 score. Elaboración propia.

Valor/Algoritmo	Regresión Logística	Naive Bayes	SVM	Árbol de decisión	Random Forest	Perceptrón Multicapa
Precision	0.67	0.67	0.67	0.65	0.67	0.75
Recall	0.82	0.82	0.82	0.69	0.80	0.76
F1 score	0.74	0.74	0.74	0.67	0.73	0.76

Como se puede observar el mejor F1 score está dado por el perceptrón multicapa con un valor de 0.76, seguido por SVM, Naive Bayes y RL con un valor de 0.74. Un valor mayor de F1-score indica una mejor clasificación debido a que considera el promedio

armónico de precisión y recall. Caso contrario, el peor valor es para el árbol de decisión con un valor de 0.67, inferior en comparación al resto de algoritmos.

2. Accuracy

A través de esta técnica el conjunto de datos total ha sido entrenado y probado, definido el parámetro de *kfold* con un valor igual a 10 se obtuvieron los siguientes resultados para los distintos algoritmos planteados (Ver Tabla 13).

Tabla 13. Error obtenido con la técnica Cross-validation y *kfold* igual a 10.

Elaboración propia.

Valor/Algoritmo	Regresión Logística	Naive Bayes	SVM	Árbol de decisión	Random Forest	Perceptrón Multicapa
Accuracy	0.80	0.83	0.83	0.69	0.82	0.71
Desviación estándar (+/-)	0.28	0.29	0.29	0.23	0.29	0.19
Error (1 – accuracy)	0.20	0.17	0.17	0.31	0.18	0.29

Como se puede observar el menor error es compartido por SVM y Naive Bayes con 0.17. Caso contrario, el peor error es para el perceptrón multicapa con un valor de 0.29. Adicionalmente se observa que los valores de varianza no superan el 30%, pero aun es un valor alto que requiere de atención.

12.2.2 Comparación de ambas métricas

Como se aprecia en las tablas 12 y 13, el perceptrón multicapa según el F1 score es el mejor clasificador; sin embargo, cuando se evalúa el *accuracy*, este resulta ser el que cuenta con mayor error con un valor de 0.29. Esto indica que tiene un alto error *bias* y puede requerir un tuneo adicional en cuanto al número de capas ocultas, las neuronas por capa, el ratio de aprendizaje, entre otros. Sin embargo, este algoritmo tiene también el menor valor de desviación estándar, esto indica que tiene el menor error de varianza.

Por otro lado, los siguientes mejores son SVM y Naive Bayes en ambos casos con F1 score de 0.74 y con el menor error con un valor de 0.17. Este resultado va acorde a lo revisado en el estado del arte donde SVM y redes bayesianas fueron los mejores resultados (Herbold, 2013).

12.2.3 Evolución de las métricas con el incremento de data

Para evaluar el efecto progresivo del incremento de data en la plataforma, se divide el conjunto total de proyectos generados (272) en cinco segmentos de 50, 100, 150, 200 y 272 valores respectivamente y se evalúan los valores de F1-score y *accuracy* (Ver Figura 28).

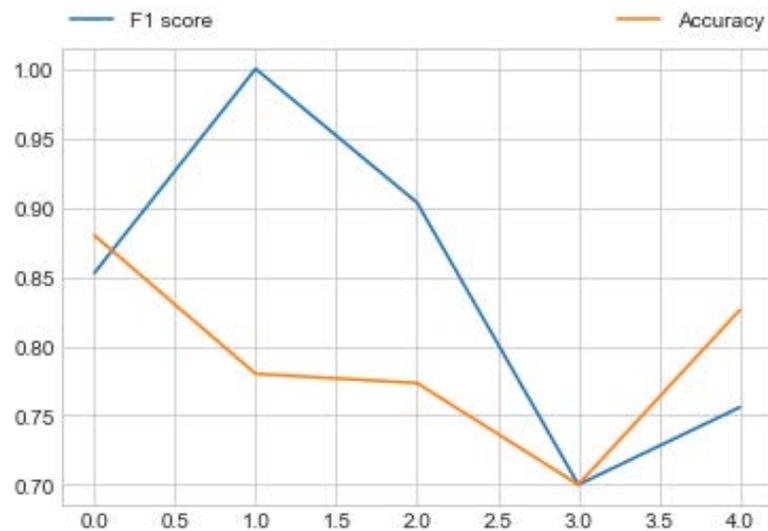


Figura 28. Evolución de F1score y Accuracy al incrementar la data. Elaboración propia.

Como se puede observar en la Figura 28, tanto el F1 score como *accuracy* muestran un decremento en sus valores para el conjunto de datos de [100, 150], y posteriormente incrementan su valor. Esto se debe a que una cantidad mayor de datos puede representar el desbalanceo de las clases, incluso en el total de los 272 proyectos las clases no cuentan con la misma cantidad de elementos. Es por ello, que para los grupos de [50, 100[las clases están mejor distribuidas y el F1 score logra llegar a ser 100%. Por otra parte, se observa que el menor valor de F1 score y Accuracy es 0.7 aproximadamente para el conjunto de 150 proyectos.

12.3 Regresión

12.3.1 Métricas

1. RMSLE

Para la regresión la métrica para validar el éxito es el RMSLE, a través del cálculo $RMSE = \sqrt{MSLE}$, donde MSLE es el error logarítmico cuadrático medio. Se considera

como ejemplo el valor máximo de RMSLE en 0.3719 de acuerdo a los resultados de la literatura revisada (Kaggle, 2017);sin embargo, este será parametrizable en la plataforma (Ver Apéndice 3).

Se muestran a continuación los valores del RMSLE para los distintos algoritmos planteados (Ver Tabla 14). Es importante recordar que los valores a continuación son los obtenidos al aplicarse el modelo sobre todo el conjunto de datos.

Tabla 14. Error obtenido con la métrica RMSLE y la variación de algunos parámetros. Elaboración propia.

Parámetros	RMSLE		Tiempo (segundos)
	Random Forest	Catboost	
Número de iteraciones Catboost = 1 Número de iteraciones RandomForest = 10 PCA = 0.85	0.3745	0.4649	5.84
Número de iteraciones Catboost = 5 Número de iteraciones RandomForest = 15 PCA = 0.85	0.3296	0.4279	39.34
Número de iteraciones Catboost = 5 Número de iteraciones RandomForest = 15 PCA = 1	0.3236	0.3268	36.34

Como se puede observar, al incrementar el número de iteraciones para el algoritmo *RandomForest* y *Catboost* se obtuvieron mejores valores de RMSLE con 0.3296 y 0.4279 comparado al 0.3745 y 0.4649 anterior respectivamente, pero con el costo de un tiempo mayor (39.34 s). Por otro lado, al variar el valor de PCA a 1, el cual define la varianza acumulada mínima, disminuye el valor de RMSLE para ambos algoritmos y se obtienen los mejores valores respecto a la variación de parámetros de ejemplo.

12.4 Evolución de las métricas con el incremento de data

Para evaluar el efecto progresivo del incremento de data en la plataforma, se divide el conjunto total de proyectos generados (272) en cinco segmentos de 50, 100, 150, 200 y 272 valores respectivamente y se evalúan los valores del RMSLE (Ver Figura 29).

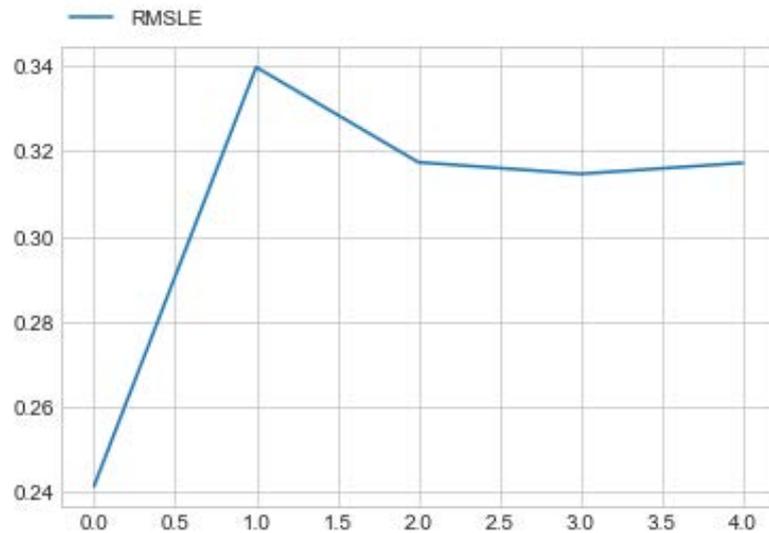


Figura 29. Evolución de RMSLE al incrementar la data. Elaboración propia.

Como se puede observar en la Figura 29, al inicio con el conjunto de datos de 50 elementos se obtuvo el menor valor de RMSLE, para luego incrementar significativamente y luego mantenerse sin cambios distanciados. Esto se debe a que al inicio posiblemente existió un *overfitting* al ser tan poca cantidad de elementos. Es por ello, que para los grupos de $[100, 272[$ el RMSLE es superior pero ya no hay cambios drásticos. Por otra parte, se observa que el mayor valor de RMSLE es de 0.34 para el conjunto de 100 proyectos.

12.5 Verificación del resultado

Se utiliza la métrica *F1-score* y el *accuracy* a través del método de *cross-validation* para los modelos de clasificación y RMSLE para los modelos de regresión.

Capítulo 13. Integración del módulo generador de modelos dinámicos con la plataforma

En esta sección se presentan los pasos seguidos para la integración de los componentes de aprendizaje automático previamente desarrollados en el software de PM4R.

13.1 Objetivo

El objetivo de esta sección detallar el proceso de integración.

13.2 Integración *Back-End*

Para esta fase se añadieron tres clases adicionales (Ver Figura 30):

- ML_class: clase para las categorías generadas por la *discretización* de la duración de los proyectos en rangos definidos por el gestor de proyectos (Ver imagen 30).
- ML_model: clase para los modelos de aprendizaje de máquina generados, tanto de clasificación como de regresión.
- ML_configurations: configuraciones para los modelos tales como la obtención de modelos validados, número de iteraciones, valores mínimos de los scores para la validación de modelos, etc.

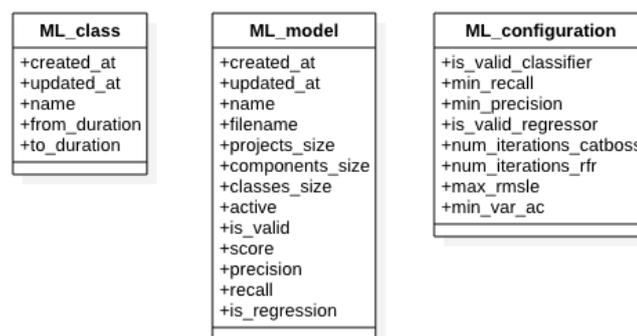


Figura 30. Clases adicionales. Elaboración propia.

Para la primera y tercera se implementaron los serializadores y las vistas correspondientes para permitir la creación, edición y eliminación de instancias, pero para ML_configurations se modificó la vista para no permitir la creación de más de una sola instancia y actualizar la existente.

Para ML_model se generaron dos vistas: tanto para clasificación como para regresión, las cuales interactúan con un script de integración.

En este se realiza el siguiente flujo:

La primera tarea a realizar es extraer todos los atributos posibles de los modelos y guardarlos en un archivo csv, si es que este existe. Luego, se verifica que:

- Si los proyectos han sido actualizados, entonces se genera nuevamente este archivo.
- Si los proyectos no han sido actualizados, pero hay nuevos registros en la base de datos y son válidos (si tienen un costo real y tiene las fechas de inicio y fin reales), se insertan al final del archivo csv.

Se utilizan archivos csv con el fin de optimizar la generación de los modelos, ya que el extraer todos los atributos de la base de datos en cada *request* demanda de gran cantidad de tiempo, en cambio al tenerlos en un csv se cargan los registros en un tipo de dato llamado *Dataframe*, de la librería de pandas, para su manipulación con un costo de tiempo mínimo.

Luego,

- Si no existe ningún proyecto, no se genera ningún modelo y se retorna un JSON vacío.
- Caso contrario, se verifica la posibilidad de la aplicación del algoritmo de clustering, en base al resultado del capítulo 10. Si este retorna un modelo válido y además se verifica que cada clúster cuenta con 50 elementos según la guía para escoger el mejor predictor (scikit-learn developers (BSD License), 2016), entonces se aplica el flujo de generación de modelos de clasificación y regresión sobre cada clúster. Caso contrario se considera como no factible y se omite la clusterización.

Seguidamente, se pre-procesa la data y si es una clasificación se discretiza en base a las categorías definidas por el gestor de proyectos y registradas en ML_class, en caso sea una regresión se omite este proceso.

A continuación, se obtienen los modelos de clasificación o regresión de la base de datos registrados en ML_model en base a las siguientes consideraciones:

- Tiene que tener el *flag* de activo.
- La cantidad de proyectos nuevos no supera los 50 elementos.
- El número de componentes post un análisis PCA es el mismo que el registrado en el modelo de base de datos.
- La cantidad de categorías discretizadas en caso de clasificación es la misma que la registrada en el modelo.

Si es que no se puede utilizar un modelo de la base de datos, se cambia el *flag* de todos a falso para no usarlos y se procede a generar uno nuevo en base al capítulo 11.

Finalmente se exponen estas funcionalidades mediante los *endpoints* mostrados en la Tabla 26 (Ver Apéndice 12). El detalle técnico se encuentra en el Apéndice 6.

13.3 Integración *Front-End*

Para el *front-end* se añadieron seis componentes (pantallas HTML, estilos y lógica) adicionales:

- *ml-classification-conf*: componente para las configuraciones del módulo estimador con modelos de clasificación
- *ml-regression-conf*: componente para las configuraciones del módulo estimador con modelos de regresión
- *project-estimator-classification*: componente de reporte que incluye la pantalla distribuida en tres secciones. En la primera sección se muestran los proyectos, la duración de los mismos, las clases asignadas en la configuración y las generadas por el mejor modelo de clasificación. En la segunda sección se muestra la cantidad de proyectos clasificados agrupados por las clases asignadas. Finalmente, en la última sección se muestra la cantidad de proyectos correcta e incorrectamente clasificados.
- *project-estimator-regression*: componente de reporte que incluye la pantalla distribuida en tres secciones. En la primera sección se muestran los proyectos, la duración y la duración estimada por el mejor algoritmo de regresión obtenido. En la segunda sección se muestra un gráfico con la comparación de ambas duraciones. Finalmente, en la última sección se muestra la media de la duración agrupada por fase en la que se encuentra el proyecto.

- `project-tab-regression`: componente *front-end* para instanciar el componente de configuración o reporte para los resultados de regresión.
- `project-tab-classification`: componente *front-end* para instanciar el componente de configuración o reporte para los resultados de clasificación.

Adicionalmente, se agregaron las siguientes clases:

- `ML_class`: clase que instancia una categoría de duración de proyectos.
- `ML_configuration`: clase que instancia las configuraciones.

De igual forma, el medio de comunicación fue un servicio en el que se implementaron las funciones para realizar las llamadas a las que los componentes se subscriben para obtener un resultado, estas llamadas al API usan los *endpoints* definidos en la Tabla 26 (Ver Apéndice 12). El detalle técnico se encuentra en el Apéndice 6.

13.4 Verificación del resultado

13.4.1 Pruebas

En esta sección se describen las pruebas unitarias y de integración realizadas sobre el software integrado con el módulo predictor del capítulo 11.

1. Pruebas unitarias

Se realizaron pruebas unitarias relacionadas a los requisitos planteados en el Capítulo 4, con la definición de prerequisites y la descripción de los casos de prueba (Ver Apéndice 13).

2. Pruebas de integración

Para esto, se relacionaron múltiples requisitos con una descripción del caso de prueba, los prerequisites y el resultado obtenido (Ver Apéndice 14). En estos se puede observar que la generación de modelos dependerá de la exigencia de las configuraciones como la mínima varianza acumulada para la selección de atributos, mínimo valor de *recall* y *precision* para clasificación y máximo RMSLE para la regresión. Además de que, al incrementar algunos parámetros como el número de iteraciones, se incrementa también el tiempo para la generación del modelo. Es un *trade-off* que el gestor de proyectos debe asumir.

Capítulo 14. Conclusiones y trabajos futuros

En esta sección se detallan las conclusiones obtenidas luego del desarrollo de la presente tesis y los trabajos futuros derivados.

14.1 Conclusiones

En la presente tesis se buscó desarrollar un software para la gestión de proyectos de desarrollo PM4R que permita automatizar la etapa de planificación para estimar tiempos a través de algoritmos de aprendizaje automático. A partir de esto se puede concluir lo siguiente:

- Se desarrolló un software que brinda soporte a la gestión de proyectos de desarrollo PM4R. Este se basa en los estándares de Google Material acorde a las herramientas actuales de gestión de proyectos y abarca todas las fases del PM4R que son: EDT, Cronograma, Curva S, Matriz de adquisiciones, Matriz de Riesgos, Matriz de comunicaciones y Matriz de asignación de responsabilidades.
 - Para la Fase de EDT la herramienta permite relacionar un proyecto con componentes, subcomponentes, productos, entregables y paquetes de trabajo.
 - Para la Fase de Cronograma, la herramienta permite detallar el tiempo de duración de los paquetes de trabajo en días y con una representación visual del diagrama de Gantt.
 - Para la fase de Curva S, la herramienta permite ingresar los costos por mes, el valor planificado y un campo para el valor real y muestra un gráfico para validar la forma S de los costos.
 - Para la fase de Matriz de adquisiciones, la herramienta permite ingresar, por cada producto o entregable, un tipo, modalidad, fechas estimadas y presupuesto estimado de adquisiciones requeridas y un campo adicional para el presupuesto real.
 - Para la fase de Matriz de riesgos, la herramienta permite tener un mapeo entre los riesgos, resultados, síntomas, probabilidad, impacto, prioridad, respuesta y responsable.

- Para la fase de Matriz de comunicaciones, la herramienta permite ingresar los elementos, razones, destinatarios, métodos, responsables y tiempos de comunicación.
- Para la fase de Matriz de asignación de responsabilidades, la herramienta permite mapear cada producto o entregable con su responsable en base a la matriz RACI.
- Se definió una arquitectura cliente-servidor para enmarcar la independencia de tecnologías usadas, que en este caso son Django Rest para el *back-end* y Angular 6 para el *front-end*. De tal manera que el *back-end* expone una lista de funcionalidades adicionales a las definidas por la lista de requerimientos en la presente tesis, de tal manera que favorece la escalabilidad de la misma.
- Se logró realizar un módulo generador de modelos dinámicos (módulo predictor) en base al tuneo mediante procesos iterativos tanto de algoritmos de *clustering* (*K-means* y *EM-Cluster*), clasificación (Regresión logística, *Naive Bayes*, SVM con RBF kernel, árboles de decisión, *RandomForest* y perceptrón multicapa) y regresión (*RandomForest Regressor* y *Catboosts*).
- Se logró integrar el software con el módulo predictor, por tanto, ofrece soporte en la etapa de planificación para la toma de decisiones a través de la estimación de la duración de los proyectos.
 - Se logró parametrizar ciertos valores necesarios para la generación de modelos de ML: clases para la *discretización*, mínimos o máximos valores de validación y el número máximo de iteraciones referido al tiempo en el que los modelos se pueden optimizar. Por tanto, la decisión de definir estos parámetros le pertenece al gestor de proyectos. Esto se integra en la plataforma como componentes de configuración tanto para clasificación como regresión.
 - Se realizó un análisis del error obtenido por los modelos generados a través del módulo predictor y su evolución con la cantidad de datos. A partir de esto, se concluye que la cantidad de datos va a determinar el nivel éxito de los modelos; sin embargo, la plataforma permite la adaptabilidad de los mismos mediante un tuneo inicial.

- Se implementó un componente *front-end* para el reporte de la estimación realizada por el módulo predictor tanto para clasificación como regresión.

14.2 Trabajos futuros

Si bien se logró realizar la integración de Gestión de Proyectos de Desarrollo PM4R con componentes de ML, quedan pendientes trabajos futuros que serán detallados a continuación:

- Incrementar la gama de algoritmos usados como redes neuronales densas y convolucionales (las que requieren de una mayor cantidad de datos) para la clasificación, y *SVR* y *GradientBoost* para la regresión. Esto requiere de mayor investigación, implementación, tuneo automatizado y pruebas del tiempo requerido.
- Realizar un mejor proceso de tuneo a los algoritmos actuales ya que la solución propuesta brinda un tuneo inicial en base a lo revisado en la literatura.
- Incrementar las validaciones para aceptar un proyecto como muestra, ya que la incorrecta selección puede generar ruido (error) en la generación de los modelos.
- Evaluación del rendimiento del software para su optimización.
- Generar reportes dinámicos en base a los datos obtenidos.
- Continuar con el proceso de integración de gestión de proyectos de desarrollo con conceptos de inteligencia artificial:
 - Sugerencias en la asignación de usuarios a un proyecto (fase 7: Matriz de Responsabilidades).
 - Sugerencias de fechas disponibles para nuevas tareas (fase 2: Cronograma).
 - Utilizar NLP (*Natural Language Processing*) para que a partir de los datos como las descripciones de los proyectos se pueda aportar valor a las estimaciones.
- Componente de autenticación y seguridad para permitir el trabajo colaborativo en la plataforma.

Referencias

- Alatrística Salas, H. (2016). Orientación del curso Aplicaciones de Ciencias de la Computación: Pre-procesamiento de datos. Lima.
- Angular Material. (2018). Angular Material Design Library. Retrieved August 26, 2018, from <https://material.angular.io/>
- APM. (2018). What is Project Management? Retrieved April 15, 2018, from <https://www.apm.org.uk/resources/what-is-project-management/>
- Aptage. (2017). How to use the Aptage RBd tool. Retrieved from <https://www.youtube.com/watch?v=o992mUVcwiA>
- Aptage. (2018). Announcing aptage. Retrieved from <https://get.aptage.com>
- Berzal, F. (2012). Introducción Medidas de similitud. *Decsai*, 1(5), 35–50. Retrieved from [elvex.ugr.es/decsai/intelligent/slides/dm/D3 Clustering.pdf](http://elvex.ugr.es/decsai/intelligent/slides/dm/D3%20Clustering.pdf)
- Bueno Cuadra, R. (2008). *El comportamiento inteligente. La visión interconductual*. Lima.
- Cambridge International Examinations. (2017). Cambridge International AS & A Level Information:Technology Data, information and knowledge.
- Capterra. (2018a). Browse Our Software Categories. Retrieved May 25, 2018, from <https://www.capterra.com/categories>
- Capterra. (2018b). Task Management Software FAQs. Retrieved May 25, 2018, from <https://www.capterra.com/task-management-software/faq/>
- Carmona Suárez, E. J. (2014). Máquinas de Vectores Soporte (SVM). *Dpto. de Inteligencia Artificial, ETS de Ingeniería Inforática, Universidad Nacional de Educación a Distancia (UNED)*, 1–25. Retrieved from [http://www.ia.uned.es/~ejcarmona/publicaciones/\[2013-Carmona\] SVM.pdf](http://www.ia.uned.es/~ejcarmona/publicaciones/[2013-Carmona] SVM.pdf)
- Chang, C. K., & Christensen, M. (1999). A Net Practice for Software Project Management. *IEEE Software*, 16(6), 80. Retrieved from <http://search.ebscohost.com/login.aspx?direct=true&db=bth&AN=2511916&site=ehost-live>
- Chang, C. K., Christensen, M. J., & Zhang, T. (2001). Genetic Algorithms for Project Management. *Ann. Softw. Eng.*, 11(1), 107–139. <http://doi.org/10.1023/A:1012543203763>
- Chang, C. K., Jiang, H. yi, Di, Y., Zhu, D., & Ge, Y. (2008). Time-line based model for software project scheduling with genetic algorithms. *Information and Software*

- Technology*, 50(11), 1142–1154. <http://doi.org/10.1016/j.infsof.2008.03.002>
- Chatfield, C., & Johnson, T. (2016). *Microsoft Project 2016 Step by Step*.
- Currie, D. (2017). NYC Taxi Trip Duration. Retrieved from <https://github.com/Currie32/NYC-Taxi-Trip-Duration>
- Django. (2018). Django documentation. Retrieved from <https://docs.djangoproject.com/en/2.0/>
- Douglas, B. (2018a). Web Services and Cloud Computing. Retrieved May 25, 2018, from https://www.service-architecture.com/articles/cloud-computing/web_services_and_cloud_computing.html
- Douglas, B. (2018b). Web Services Definition. Retrieved May 25, 2018, from https://www.service-architecture.com/articles/web-services/web_services_definition.html
- Elmblad, S. (2018). How Is Desktop Software Different From an App? Retrieved May 26, 2018, from <https://www.thebalance.com/what-is-desktop-software-1293673>
- Espinoza, J., Loarte, P., Paz, F., & Flores, L. (2019). Usability in the development of a project management software reinforced with machine learning, 1–19.
- Fluin, S. (2018). Version 6 of Angular Now Available. Retrieved from <https://blog.angular.io/version-6-of-angular-now-available-cc56b0efa7a4>
- Fomby, T. (2008). Naïve Bayes Classifier, (1), 2–4. <http://doi.org/10.1111/j.1365-2753.2011.01762.x>
- Forecast. (2018). Get Started. Retrieved from <https://www.forecast.it>
- Frappe Technologies Pvt. (n.d.). Frappe Gantt. Retrieved from <https://frappe.io/gantt>
- Galván, I. M., Valls, J. M., Aler, R., & Huertas, J. (2017). Redes de neuronas artificiales: Perceptron Multicapa. Retrieved from <http://ocw.uc3m.es/ingenieria-informatica/redes-de-neuronas>
- García López, R., & García Moreno, M. (2010). *La gestión para resultados en el desarrollo. Avances y desafíos en América Latina y el Caribe. Naciones Unidas*. Retrieved from http://repositorio.cepal.org/bitstream/handle/11362/35244/lcl3252_es.pdf?sequence=1
- Gibson, C. F., & Nolan, R. L. (1974). Managing the four stages of EDP growth. *Harvard Business Review*, 52(February), 76–88. <http://doi.org/10.1145/2579442.2579447>
- Google. (2018). Material Design. Retrieved from <https://material.io>

- Goresht, S. (2015). material-table.
- Hanchate, D. B., & Bichkar, R. S. (2015). The machine learning in software project management : A journey . Part II. *Applied Discrete Mathematics and Heuristic Algorithms*, 3(January), 21–47.
- Herbold, S. (2013). Training data selection for cross-project defect prediction. *Proceedings of the 9th International Conference on Predictive Models in Software Engineering*, 1–10. <http://doi.org/10.1145/2499393.2499395>
- Hosley, W. N. (1985). Innovative Approaches to Quality Assurance in Project Management. *Project Management Journal*, 16(2), 90–101. Retrieved from <https://www.pmi.org/learning/library/innovative-approaches-quality-assurance-1812>
- JetBrains. (2010). JetBrains Strikes Python Developers with PyCharm 1.0 IDE. Retrieved from <http://www.eweek.com/development/jetbrains-strikes-python-developers-with-pycharm-1.0-ide>
- Joslin, D., & Poole, W. (2005). Agent-based simulation for software project planning. In *Proceedings of the Winter Simulation Conference, 2005*. (pp. 1059–1066). <http://doi.org/10.1109/WSC.2005.1574359>
- Junta de Andalucía. (2018a). Plantilla Plan de Pruebas de Integración. Retrieved from <http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/461>
- Junta de Andalucía. (2018b). Plantilla Plan de Pruebas Unitarias.
- Junta de Andalucía. (2018c). Técnicas de validación de requisitos. Retrieved September 8, 2018, from <http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/419>
- Kaggle. (2017). New York City Taxi Trip Duration. Retrieved from <https://www.kaggle.com/c/nyc-taxi-trip-duration/leaderboard>
- León Guzmán, E. (2016). Métricas para la validación de Clustering Contenido. Retrieved from http://www.disi.unal.edu.co/profesores/eleonguz/cursos/md/presentaciones/Sesion13_validacion_Clustering.pdf
- Mans Unides. (2015a). *Libro De Proyectos*. Retrieved from https://docs.google.com/viewerng/viewer?url=https://mansunides.org/sites/default/files/publicaciones-pdf/libro_proyectos_2015.pdf
- Mans Unides. (2015b). Proyectos en Perú de cooperación al desarrollo. Retrieved April

- 15, 2018, from <https://mansunides.org/es/proyectos-peru>
- Mans Unides. (2016). Proyectos de desarrollo. Retrieved from <https://mansunides.org/es/proyectos-de-desarrollo>
- Mercedes, C. (n.d.). Capítulo 2 modelo de regresión logística 2.1.
- Ministerio del Ambiente. (2011). Central Hidroeléctrica el Platanal. Retrieved from <http://cambioclimatico.minam.gob.pe/central-hidroelectrica-el-platanal/>
- Mitchell, T. M. (2005). Expectation Maximization, and Learning from Partly Unobserved Data. *Carnegie Mellon University*.
- Monleon-Getino, A. (2016). El impacto del Big-data en la Sociedad de la Información. Significado y utilidad". *Historia y Comunicación Social*, 20(2), 427–445. http://doi.org/10.5209/rev_HICS.2015.v20.n2.51392
- Monleón-Getino, T. (2010). El tratamiento numérico de la realidad. Reflexiones sobre la importancia actual de la estadística en la sociedad de la información. *Arbor*, 186(743), 489–497. <http://doi.org/10.3989/arbor.2010.743n1213>
- Nielsen, J. (1995). 10 Usability Heuristics for User Interface Design.
- Núñez, H. (2014). Video de Presentación Final del Curso PM4R Paraguay 6 - 2014. Retrieved from <https://www.youtube.com/watch?v=X3qrEhmtr-Y>
- P. Letelier. (2003). Proceso de Desarrollo de Software. *Universidad Politécnica de Valencia*, 1–14. Retrieved from <http://ldc.usb.ve/~abianc/materias/ci4712/ProcesoSW-Letelier.pdf>
- PMI. (2010). Proyectos exitosos en América Latina. Retrieved from <http://americalatina.pmi.org/~media/Files/latam/Argentina-Capitulo-Nuevo-Cuyo/2011-AR-NC-Buchtik-ProyectosExitosos.aspx>
- PMI. (2017). *Guía de los fundamentos para la dirección de proyectos (Guía del PMBOK®)*. (Project Management Institute, Ed.). Project Management Institute.
- PMI. (2018). What is Project Management? Retrieved April 15, 2018, from <https://www.pmi.org/about/learn-about-pmi/what-is-project-management>
- Project Jupyter. (2018). About Project Jupyter. Retrieved from <https://jupyter.org/about>
- Refaeilzadeh, P., Tang, L., & Liu, H. (2009). Cross-Validation BT - Encyclopedia of Database Systems. In L. LIU & M. T. ÖZSU (Eds.), (pp. 532–538). Boston, MA: Springer US. http://doi.org/10.1007/978-0-387-39940-9_565
- Rejas Cano, E. A. (2014). *Diseño de un algoritmo de búsqueda Tabú para resolver el problema de la selección de proyectos*. Pontificia Universidad Católica del Perú.

- Retrieved from
http://tesis.pucp.edu.pe/repositorio/bitstream/handle/123456789/10175/ARCE_CI_GUENAS_SISTEMA_AUTONOMO_CONTROL_TRAFICO.pdf?sequence=1&isAllowed=y
- Rokach, L., & Maimon, O. (2010). Decision Trees. *Data Mining and Knowledge Discovery Handbook*, 165–192. http://doi.org/10.1007/0-387-25465-X_9
- Russell, S. J., & Norvig, P. (2010). *Artificial Intelligence A Modern Approach* (3rd ed.).
- Saglam, S. O. (2017). *Applied Machine Learning: Project Management Performance Prediction At Information Technology Company Project Management Office*. POLITECNICO DI MILANO.
- Salinas, P. (n.d.). Modelo de Clases. Retrieved from
<https://users.dcc.uchile.cl/~psalinas/uml/modelo.html#clase>
- Santos D., M., A. Parra, I., Gallardo V., S., & Ingram R., J. W. (1996). Herramientas corporativas de gestión de proyectos: Una visión integral, (February), 146–152.
- scikit-learn. (2017). Selecting the number of clusters with silhouette analysis on KMeans clustering. Retrieved from http://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_silhouette_analysis.html
- scikit-learn developers (BSD License). (n.d.). RandomForestClassifier. Retrieved from <http://scikit-learn.org/stable/modules/ensemble.html#random-forests>
- scikit-learn developers (BSD License). (2014). Model evaluation : quantifying the quality of predictions. Retrieved from http://scikit-learn.org/stable/modules/model_evaluation.html
- scikit-learn developers (BSD License). (2016). Choosing the right estimator — scikit-learn 0.18.1 documentation. Retrieved from http://scikit-learn.org/stable/tutorial/machine_learning_map/
- scikit-learn developers (BSD License). (2017). Mean squared error. Retrieved from http://scikit-learn.org/stable/modules/model_evaluation.html#mean-squared-error
- Serna, E. (2017). *Desarrollo e innovación en ingeniería. Desarrollo e innovación en ingeniería*. Retrieved from
https://www.researchgate.net/profile/Jhon_Fredy_Narvaez/publication/320170890_Desarrollos_de_la_Ingenieria_ambiental_en_la_evaluacion_de_la_calidad_de_los_recursos_naturales_y_la_salud_ambiental/links/59d26bfca6fdcc181ad611ce/Desarrollos-de-la-Ingenieria-

- Siles, R., & Mondelo, E. (2012). *pm4R: Gestión de Proyectos para Resultados*. Retrieved from <http://link.springer.com/10.1007/s10346-009-0146-7>
<http://www.tandfonline.com/doi/abs/10.1080/17499510801958711>
<http://www.ncbi.nlm.nih.gov/pubmed/15003161>
<http://cid.oxfordjournals.org/lookup/doi/10.1093/cid/cir991>
<http://www.scielo.cl/pdf/ud>
- Siles, R., & Mondelo, E. (2015). *Gestión de proyectos de desarrollo. Certificación Project Management Associate (PMA). GESTIÓN DE PROYECTOS DE DESARROLLO* (4th ed.).
- Sketch. (2018). Sketch. The digital design toolkit. Retrieved June 3, 2018, from <https://sketchapp.com>
- Sparx Systems. (2018). Diagrama de Despliegue UML 2. Retrieved from http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html
- StarUML. (2018). StarUML 3. Retrieved from <http://staruml.io>
- Stratejos. (2018). Project Assistant. Retrieved from <https://stratejos.ai>
- The PgAdmin Development Team. (2018). pgAdmin 4. Retrieved from <https://www.pgadmin.org/docs/pgadmin4/dev/>
- Trujillo, L. (2012). Metodología para determinar la evolución de los sistemas informáticos de organizaciones regionales * 1 [Methodology to determine the evolution of the computer systems of regional organizations] Resumen Introducción. *Ventana Informática*, (28), 151–166. Retrieved from <http://revistasum.umanizales.edu.co/ojs/index.php/ventanainformatica/article/view/188/236>
- Veerasamy, R., Rajak, H., Jain, A., Sivadasan, S., Varghese, C. P., & Agrawal, R. K. (2015). Validation of QSAR Models - Strategies and Importance. *Prescrire International*, 24(166), 294. <http://doi.org/10.1016/j.febslet.2005.06.031>
- Vegega, C., Pytel, P., & Pollo, M. (2017). Método basado en el empujamiento para evaluar los datos aplicables para entrenar algoritmos de aprendizaje automático. In *Desarrollo e innovación en ingeniería*.
- Wallace, W. (2011). Gestión de Proyectos. *Cdi*, 2014(1106), 30. <http://doi.org/http://www.cdi.gob.mx/jovenes/documentos.html>
- Yamanaka, M. (2015). Programa Reconversión Urbana, Modernización del Transporte Público Metropolitano y Oficinas del Gobierno (PR-L1044). Yamanaka, Megumi. Retrieved from https://www.youtube.com/watch?v=Pk_WiiG0FKU