

PREDICCIÓN DEL TIPO Y CANTIDAD DE ACTIVIDADES DE INSTALACIÓN Y
MANTENIMIENTO GESTIONADOS POR EL PERSONAL TÉCNICO DE LA
EMPRESA COLVATEL S.A., USANDO REDES NEURONALES.

FABIO CAMILO LÓPEZ CASTELLANOS

UNIVERSIDAD PILOTO DE COLOMBIA
FACULTAD DE INGENIERÍA
BOGOTÁ D.C.
2018

PREDICCIÓN DEL TIPO Y CANTIDAD DE ACTIVIDADES DE INSTALACIÓN Y
MANTENIMIENTO GESTIONADOS POR EL PERSONAL TÉCNICO DE LA
EMPRESA COLVATEL S.A., USANDO REDES NEURONALES.

FABIO CAMILO LÓPEZ CASTELLANOS

JUAN CARLOS NAVARRO BELTRAN
ASESOR TEMATICO

UNIVERSIDAD PILOTO DE COLOMBIA
FACULTAD DE INGENIERÍA
BOGOTÁ D.C.
2018

CONTENIDO

Pág.

LISTA DE TABLAS.....	5
LISTA DE FIGURAS	6
RESUMEN	7
1. PLANTEAMIENTO DEL PROBLEMA.....	8
2. JUSTIFICACIÓN.....	9
3. OBJETIVOS.....	10
3.1 OBJETIVO GENERAL	10
3.2 OBJETIVOS ESPECÍFICOS	10
4 MARCO REFERENCIAL	11
4.1 ANTECEDENTES	11
4.2 SERIES DE TIEMPO	13
4.3 REDES NEURONALES	14
4.4 REDES NEURONALES RECURRENTE.....	15
4.5 PROCESOS DE APROVISIONAMIENTO Y ASEGURAMIENTO EN OPERADORES DE SERVICIOS DE TELECOMUNICACIONES.....	16
5 ANÁLISIS DE MODELOS DE REDES NEURONALES RECURRENTE.....	17
5.1 DESVANECIMIENTO O EXPLOSIÓN DE GRADIENTE	17
5.2 ELMAN RECURRENT NEURAL NETWORK (ELMAN RNN O ERNN)	18
5.3 ECHO STATE NETWORK (ESN)	19
5.4 LONG SHORT-TERM MEMORY (LSTM)	20
5.5 GATED RECURRENT UNIT (GRU).....	21
6 METODOLOGÍA PARA EL ANÁLISIS, DISEÑO Y CONSTRUCCIÓN DE LA SOLUCIÓN	23
6.1 ENTENDIMIENTO DEL NEGOCIO.....	23
6.2 ADQUISICIÓN Y ENTENDIMIENTO DE LOS DATOS	24
6.3 MODELADO.....	26
6.4 DESPLIEGUE	29
6.5 ACEPTACIÓN DEL CLIENTE.....	29
7 ANÁLISIS DE INFORMACIÓN HISTÓRICA REQUERIDA PARA LA CONSTRUCCIÓN DE LA SOLUCIÓN	30
7.1 ENTENDIMIENTO DEL NEGOCIO.....	30

7.2	ADQUISICIÓN Y ENTENDIMIENTO DE LOS DATOS	31
8	CONSTRUCCIÓN Y ENTRENAMIENTO DE LA RED NEURONAL	41
8.1	INGENIERIA Y SELECCIÓN DE ATRIBUTOS (TRANSFORMACIÓN).....	43
8.2	DIVISIÓN DEL LOTE DE DATOS.....	45
8.3	CONSTRUCCIÓN DEL MODELO	45
8.4	ENTRENAMIENTO DEL MODELO.....	47
9	VALIDACIÓN DEL MODELO PREDICTIVO.....	49
9.1	PUNTUACIÓN DEL MODELO	49
9.2	EVALUACIÓN DEL MODELO.....	49
	CONCLUSIONES.....	54
	BIBLIOGRAFIA	56

LISTA DE TABLAS

Pág.

Tabla 1. Comparación de redes neuronales.	22
Tabla 2. Correcciones tabla ADSLPLUS_APROVISIONAMIENTO	32
Tabla 3. Correcciones tabla ADSLPLUS_LINEABASICA.	33
Tabla 4. Información del set de datos.	35
Tabla 5. Medidas de posición.....	36
Tabla 6. Medidas de dispersión.	36
Tabla 7. Variación y correlación de los datos.....	39
Tabla 8. Asimetría y apuntamiento de los datos.	40
Tabla 9. Herramientas de desarrollo.	41
Tabla 10. Herramientas hardware	41
Tabla 11. Configuración general del modelo predictivo.....	46
Tabla 12. Combinación de escaladores y activadores.	48
Tabla 13. Perdida de datos a través de las iteraciones de entrenamiento.	50
Tabla 14. Errores de predicción por escalador.....	51
Tabla 15. Predicción de series de tiempo por escalador.....	52
Tabla 16. Métricas para la evaluación del modelo predictivo.	53

LISTA DE FIGURAS

Pág.

Figura 1. Ciclo de vida TDSP.	24
Figura 2. Arquitectura de la solución	31
Figura 3. Función para la limpieza de valores atípicos.....	33
Figura 4. Consulta general datos de aseguramiento y aprovisionamiento.	34
Figura 5. Agrupamiento y reducción de los datos.	34
Figura 6. Densidad cantidad de actividades y técnicos.....	37
Figura 7. Histograma cantidad de actividades y técnicos.....	38
Figura 8. Diagrama de cajas para la cantidad de actividades y técnicos.	39
Figura 9. Diagrama de dispersión técnicos vs actividades.....	40
Figura 10. Diseño del modelo predictivo.	42
Figura 11. Ventana deslizable de datos.	44
Figura 12. Segmentación de datos.....	45
Figura 13. Función para construir el modelo predictivo.....	47

RESUMEN

Este proyecto lleva a cabo la aplicación de un modelo ya establecido de red neuronal recurrente para predecir el número y tipo de servicios de instalación y mantenimiento gestionados por el personal técnico de Colvatec S.A. Su ejecución se realiza de acuerdo con el ciclo de vida de la metodología TDSP.

La información empleada para el entrenamiento y predicción corresponde a las actividades atendidas en los segmentos de aprovisionamiento y aseguramiento de los servicios de línea básica y banda ancha cobre desde el 1 de enero de 2013 hasta el 31 de diciembre de 2017.

1. PLANTEAMIENTO DEL PROBLEMA

COLVATEL S.A. E.S.P. Es una sociedad comercial anónima, de servicios públicos mixta, del orden distrital, descentralizada por servicios, creada en diciembre de 1992 como filial de la Empresa de Telecomunicaciones de Bogotá S.A. E.S.P (ETB).¹, que se dedica a la comercialización de bienes y la prestación de servicios de telecomunicaciones y cualquier tecnología informática y de comunicaciones, así como servicios de valor agregado, telemáticos, outsourcing e integración de servicios a terceros en las áreas de gestión técnica, administrativa, financiera, comercial y operativa².

COLVATEL S.A. actualmente presenta una problemática con la distribución del personal técnico especializado el cual no siempre es el adecuado de acuerdo con la cantidad de servicios de instalación y mantenimiento de redes banda ancha, demandados por ETB lo que genera precios menos competitivos frente a la oferta del sector, afectando de manera directa la economía de la empresa.

Esto sumado al hecho de que a COLVATEL S.A. se le pagará menos por cada actividad que no atienda dentro del día en que fue agendada, hace que sea necesario una herramienta que permita conocer cuál podría ser la tendencia a futuro en la gestión de estas actividades y que facilite la toma de decisiones de carácter estratégico de forma efectiva de cara a las necesidades del negocio, tales como la optimización del despliegue del personal especializado requerido para la atención de dichas actividades.

Pregunta: ¿Cómo predecir el número y tipo de servicios de instalación y mantenimiento que debe gestionar el personal técnico de Colvatel S.A., con el fin de facilitar la toma de decisiones estratégicas para el cumplimiento de los niveles de servicio?

¹ Colvatel S.A. E.S.P. Quienes Somos [En línea]. Bogotá D.C.: S.N., enero 2015 [Citado en 21 agosto de 2018]. Disponible en Internet: < <http://www.colvatel.com/somos.php> >.

² Colvatel S.A. E.S.P. Naturaleza jurídica [En línea]. Bogotá D.C.: S.N., enero 2015 [Citado en 21 agosto de 2018]. Disponible en Internet: < <http://www.colvatel.com/juridica.php> >.

2. JUSTIFICACIÓN

El propósito de este proyecto es la aplicación de modelos ya establecidos de redes neuronales recurrentes a la predicción de series de tiempo, con el fin de adaptar su funcionamiento al problema de predicción del número y tipo de servicios de instalación y mantenimiento gestionados por el personal técnico de Colvatec S.A.

Contar con un pronóstico de esta índole facilita la toma de decisiones de tipo estratégico, permitiendo reaccionar de forma oportuna y concreta ante las situaciones o problemas que se presentan en el mercado, sin embargo, requiere de una herramienta eficiente para el procesamiento y análisis de información histórica.

Particularmente para Colvatec S.A. es una necesidad fundamental de la dirección operativa, en cuanto a la posibilidad de optimizar el agendamiento y despliegue del personal técnico especializado requerido para la atención de las actividades de instalación y mantenimiento de acuerdo con la cantidad y tipo de servicios. Adicionalmente, las decisiones tomadas a partir de los pronósticos generados contribuirán de forma indirecta en la percepción del cliente en cuanto al tiempo de espera de atención del servicio solicitado; siendo esto parte de un principio de responsabilidad frente a los clientes.

3. OBJETIVOS

3.1 OBJETIVO GENERAL

Analizar y aplicar un modelo de redes neuronales recurrentes para la predicción de series de tiempo, al problema del pronóstico del número y tipo de servicios de instalación y mantenimiento gestionados por el personal técnico de Colvotel S.A.

3.2 OBJETIVOS ESPECÍFICOS

Analizar diferentes modelos conocidos de redes neuronales y seleccionar el más apropiado para aplicarlo al problema de la predicción del número y tipo de servicios gestionados por el personal técnico de Colvotel S.A.

Identificar la metodología para el análisis, diseño y construcción de una solución usando el modelo de redes neuronales seleccionado.

Analizar las necesidades de información histórica requerida para la construcción del modelo de red neuronal y predicción del comportamiento en cuanto al número tipo de servicios gestionados por el personal técnico de Colvotel S.A.

Construir y entrenar el modelo de red neuronal planteado.

Validar el modelo de red neuronal construido y entrenado, a partir de la información histórica disponible acerca de los servicios gestionados por el personal técnico de Colvotel S.A.

4 MARCO REFERENCIAL

En este capítulo se presentará el marco referencial necesario para conceptualizar una solución al problema planteado en el presente proyecto, partiendo con la exposición de algunos antecedentes sobre investigaciones similares realizadas en este campo, la definición sobre series de tiempo, redes neuronales y redes neuronales recurrentes.

4.1 ANTECEDENTES

Dentro de la investigación de predicción de series de tiempo con redes neuronales, se han encontrado múltiples trabajos, la mayoría enfocados en la predicción de indicadores económicos de la bolsa de valores, predicción del cambio climático, predicción de marcadores de diferentes disciplinas deportivas y unos pocos enfocados a la predicción de la demanda energética de una ciudad y a la carga de la actividad telefónica de un operador móvil; por lo que se tomaran como referencia algunos de estos estudios especialmente aquellos que poseen un enfoque similar al problema planteado en Colvatel S.A. para dilucidar una solución.

El primer caso de estudio analizado se titula “Análisis de Series Temporales Usando Redes Neuronales Recurrentes”³, en donde se propone el uso de redes neuronales recurrentes para el análisis y predicción de series de tiempo. Para demostrar el grado de éxito de este enfoque aplicaron Redes Neuronales Recurrentes (RNN) al análisis del consumo eléctrico de la población de Soller Mallorca en España, donde el objetivo era realizar una estimación futura aproximada del valor total para la demanda en el consumo eléctrico por hora, a partir de la información histórica del consumo eléctrico de esta población en un periodo de tiempo determinado. Dicho estudio logra predicciones con un nivel de confianza del 93% además de que pone en relieve la necesidad de incorporar variables externas que influyan en el comportamiento de la serie en estudio. Siendo en este caso de variable externa, una serie histórica de la temperatura ambiental en esta población durante el mismo intervalo de tiempo.

³ GONZALEZ AVELLA, J. C.; TUDURI, J. M. y RUL LAN, G.; Análisis de Series Temporales Usando Redes Neuronales Recurrentes. Palma de Mallorca: APSL, 2017. 6 p

En el estudio “An overview and comparative analysis of recurrent neural networks for short term load forecasting”⁴, se comparan diferentes arquitecturas de redes neuronales por medio del análisis de un conjunto de datos relativo a la carga de llamadas telefónicas registradas a través de una red móvil por los usuarios de la operadora Orange en Costa de Marfil.

Este trabajo se centró en predecir el volumen en minutos de las llamadas entrantes del día siguiente, observándose que la precisión además de darse por la adición de nuevos conjuntos de datos con variables exógenas que proporcionan información de contexto adicional, requiere un preanálisis de los datos que permitan identificar ciclos estacionales para que, de esta forma las RNN se enfoquen en aprender a predecir cambios en dichos ciclos, lo que permite mejorar considerablemente la precisión de los pronósticos realizados.

Así mismo concluyen que las arquitecturas Long Short Term Memory (LSTM) y Gate Recurrent Unit (GRU) pueden modelar de manera eficiente relaciones de información altamente no lineales, ya que sus mecanismos de activación permiten modificar rápidamente el contenido de la memoria de las redes y la dinámica interna. Sin embargo, la elección más conveniente en los problemas de predicción de series de tiempo en cuanto a rendimiento, simplicidad de implementación y capacitación, recomiendan las arquitecturas Elman Recurrent Neural Network (ERNN) y Echo State Unit (ESN).

El trabajo “Modeling and forecasting the Nord Pool day ahead power market through Deep Learning”⁵, buscaba predecir con el modelo LSTM el precio del megavatio por hora de energía para el siguiente día, a partir del análisis del conjunto de datos históricos sobre demanda, volumen, consumo y reserva de energía hidroeléctrica.

Descubriéndose que la precisión más alta en términos de error absoluto medio se encontraba cuando se realizaba la predicción del precio el cual era de un 0.82 por ciento en contraste con el 3,62 por ciento alcanzado con el modelo oculto de Márkov (HMM), aunque la mayor precisión en términos de error cuadrático medio se alcanzó con el modelo LSTM fusionando el conjunto de datos de precio y consumo del 2014-2017 con una precisión de 127.85 con respecto a error cuadrático medio de 239.99 alcanzado con la fusión del conjunto de datos precio y volumen del 2014-2015; lo que implica de este modo que tanto la cantidad de datos disponibles para el entrenamiento como el tipo de variable exógena a correlacionar influyen en la precisión de las predicciones realizadas por el modelo LSTM.

⁴ BIANCHI, Filippo María; MAIORINO, Enrico; KAMFFMEYER, Michael C.; RIZZI, Antonello; JENSSEN, Robert. An overview and comparative analysis of Recurrent Neural Networks for Short Term Load Forecasting. Norway: Arxiv, 2017. 41 p.

⁵ WALKER LYSFJORD, MAGNUS J. Modeling and Forecasting the Nord Pool Day-Ahead Power Market through Deep-Learning. Research project submitted for the degree of Masters of Renewable Energy. Norway: University of Agder, 2017. 48 p.

La empresa internacional de vehículos de transporte con conductor: Uber, publicó los artículos “Time series extreme event forecasting with neural networks at Uber”⁶ y “Deep and confident prediction for time series at Uber”⁷, en donde se propone una arquitectura de red basada en LSTM, con un marco codificador-decodificador que captura el patrón inherente en la serie temporal aprendido durante el proceso de preentrenamiento y una red de predicción que toma datos incrustados desde el codificador-decodificador, así como de cualquier característica externa potencial para guiar la predicción.⁶ Con un set de datos compuesto de cinco años de viajes completados en los Estados Unidos entre otras variables exógenas, se buscaba predecir la cantidad de viajes durante fechas especiales (navidad, año nuevo, días feriados, etc.) y la previsión confiable de la incertidumbre o grado de confianza en las predicciones, con el fin de llevar a cabo una asignación óptima de recursos, asignación de incentivos a los conductores, planificación presupuestaria entre otros⁷.

Estas investigaciones lograron alcanzar un margen de confianza del 95% en la predicción realizada para fechas especiales⁶, además de que recomiendan tres elementos que se deben tener en cuenta para aumentar la probabilidad de que la red neuronal pronostique con una mayor precisión, tales como el número de series de tiempo, la longitud (tiempo) de cada una y la correlación entre ellas⁷.

4.2 SERIES DE TIEMPO

Las series de tiempo, también conocidas como series cronológicas es una secuencia de datos, observaciones o valores, medidos en determinados momentos y ordenados cronológicamente. Los datos pueden estar espaciados a intervalos iguales o desiguales⁸. Las series de tiempo corresponden a distribuciones bidimensionales o bivariantes, lo que significa que se analizan conjuntamente dos variables en donde una de ellas corresponde al tiempo, considerada como la variable independiente o explicativa; y la otra corresponde a la variable que se va a

⁶ ZHU, Lingxue y LAPTEV, Nikolay. Deep and confident prediction for time series at Uber. San Francisco, California: Arxiv, 2017. 8 p.

⁷ LAPTEV, Nikolay; YOSINSKI, Jason; ERRAN, Li y SMYL, Slawek. Time-series extreme event forecasting with neural networks at Uber. San Francisco, California: Semantic Scholar - The Allen Institute for Artificial Intelligence, 2017. 4 p.

⁸ Series de tiempo con Python [En línea]. Buenos Aires (Argentina): Raúl E. López Briega, 26 septiembre 2016 [Citado en 22 agosto 2018]. Disponible en Internet:<<https://relopezbriega.github.io/blog/2016/09/26/series-de-tiempo-con-python>>.

estimar, ya sea dentro de la serie (interpolación) o su comportamiento futuro (extrapolación)⁹.

4.3 REDES NEURONALES

Antes de realizar la definición de red neuronal, se debe exponer el significado de Inteligencia Artificial, Aprendizaje Automático, Aprendizaje profundo y como se relacionan entre ellos.

La inteligencia artificial (IA) se conoce como “El esfuerzo para automatizar tareas intelectuales normalmente realizadas por humanos”¹⁰, básicamente se intenta lograr que una computadora sea capaz de “pensar” como lo haría un humano. “Como tal, IA es un campo general que abarca al aprendizaje automático y al aprendizaje profundo, pero que también incluye muchos más enfoques que no implican ningún aprendizaje”¹⁰.

Aprendizaje Automático es un subcampo de investigación dentro de la Inteligencia artificial, el cual consiste en el diseño y estudio de algoritmos que analicen grandes conjuntos de datos, encuentren patrones y realicen predicciones¹¹ sin descuidar la precisión y velocidad con las que lo realiza, lo que implica que será capaz de hacerlo mejor en el futuro a partir de su propia experiencia¹².

Así como el aprendizaje automático es un subcampo de la IA, el aprendizaje profundo es un subcampo del aprendizaje automático. El aprendizaje profundo es una clase específica de algoritmos de aprendizaje automático¹² que se enfoca en aprender representaciones de datos significativos a través de sucesivas capas de aprendizaje, de hecho, la cantidad de capas que contribuyen a un modelo de datos es lo que se conoce como profundidad del modelo¹³.

⁹ BENCARDINO MARTÍNEZ, Ciro. Estadística y muestreo. 13 ed. Bogotá D.C.: ECOE ediciones Ltda., 2012. ISBN:978-958-648-702-3. p. 794-816.

¹⁰ CHOLLET, Francois. Deep learning with python. New York: Manning publications Co., 2017. p. 4-5. ISBN 9781617294433.

¹¹ Differences Between AI, Machine Learning and Deep Learning [En línea]. New York (USA): S.N., 29 noviembre 2017 [Citado en 9 septiembre 2018]. Disponible en Internet:<<https://t2conline.com/differences-between-ai-machine-learning-and-deep-learning>>.

¹² GOLLAPUDI, Sunila. Practical Machine Learning. Birmingham - Mumbai: Packt Publishing Ltd., 2016. p. 34. ISBN 9781784399689.

¹³ CHOLLET. Op. Cit., p 8-9

En ambos casos, los algoritmos que sustentan estos dos tipos de aprendizaje es lo que actualmente se conocen como redes neuronales artificiales que son básicamente un conjunto de unidades neuronales simples¹⁴ o funciones matemáticas compuestas principalmente por muchas y pequeñas multiplicaciones de matrices¹³. Inspiradas en una neurona biológica real, su funcionamiento se resume en el ingreso de una o más entradas (datos de entrenamiento) que luego son sumadas, ponderadas (peso y sesgo) y transferidas a una función no lineal (función de activación o función de transferencia) para producir una salida¹⁵. Además de ser altamente paralelas, sus principales características son la no linealidad, capacidad de adaptación a los datos, escalabilidad y reusabilidad.

Las redes neuronales artificiales (RNA) se han visto fuertemente impulsadas desde el año 2012, cuando los investigadores Alex Krizhevsky, Ilya Sutskever, y Geoff Hinton presentaron el modelo AlexNet para el reconocimiento de imágenes, que condensaba fundamentalmente todo el estado del arte alcanzado en redes neuronales desde su concepción teórica hace más de medio siglo, aportando mejoras como el entrenamiento en paralelo por medio de las unidades de procesamiento gráfico (GPU), lo cual permitía la reducción en el tiempo de entrenamiento y en el margen de error del modelo.

4.4 REDES NEURONALES RECURRENTES

Las Redes neuronales recurrentes nacieron para abordar el problema de persistencia de las RNA tradicionales; la inteligencia biológica es capaz de procesar la información gradualmente mientras mantiene un modelo interno de lo que está procesando, construido a partir de información pasada y constantemente actualizada a medida que ingresa nueva información. Una RNN adopta el mismo principio, aunque en una versión simplificada: procesa una secuencia de datos iterando a través de los elementos de la secuencia y manteniendo un estado que contiene información relativa a lo que ha visto hasta ahora; esto permite que esas iteraciones o conexiones recurrentes construyan una "memoria" de entradas previas en el estado interno de la red y que esta influya en la salida de esta red¹⁶. Esta

¹⁴ Introducción al Deep Learning [En línea]. Buenos Aires (Argentina): Raúl E. López Briega, 13 junio 2017 [Citado en 22 agosto 2018]. Disponible en Internet:<<https://relopezbriega.github.io/blog/2017/06/13/introduccion-al-deep-learning>>.

¹⁵ BONNIN, Rodolfo. Building Machine Learning Projects with TensorFlow. Engaging projects that will teach you how complex data can be exploited to gain the most insight. Birmingham - Mumbai.: Packt Publishing, 2016. p. 112. ISBN 9781786466587.

¹⁶ CHOLLET. Op Cit., p. 8.

naturaleza en forma de cadena revela que las Redes neurales recurrentes están íntimamente relacionadas con las secuencias y listas; por lo que son ideales para trabajar con este tipo de datos¹⁷.

4.5 PROCESOS DE APROVISIONAMIENTO Y ASEGURAMIENTO EN OPERADORES DE SERVICIOS DE TELECOMUNICACIONES

Dentro del marco de procesos empresariales para los proveedores de servicios de Telecomunicaciones (ETOM) propuesto por el TMForum, donde se describen los procesos en la gestión de servicios y tecnologías de la información y las comunicaciones y como se relacionan entre ellos, también aborda la definición de los procesos de cumplimiento, garantía y facturación que conforman el flujo de proceso del área de operaciones (OPS).

El proceso de cumplimiento también conocido como aprovisionamiento, es la agrupación de procesos responsables de proporcionar a los clientes los productos solicitados de manera oportuna y correcta. Traduce la necesidad comercial o personal del cliente en una solución, que puede entregarse utilizando los productos específicos en la cartera de la empresa. Este proceso informa a los clientes sobre el estado de su orden, garantiza que se complete a tiempo y garantiza un cliente satisfecho¹⁸.

El proceso de garantía o aseguramiento es la agrupación de procesos responsables de la ejecución de actividades de mantenimiento proactivas y reactivas para garantizar que los servicios proporcionados a los clientes estén continuamente disponibles y que cumplan con los acuerdos de niveles de servicio (SLA). Este proceso gestiona los SLA e informa el rendimiento del servicio al cliente. Recibe informes de problemas del cliente, informa al cliente sobre el estado del problema y garantiza la restauración y reparación, así como también garantiza un cliente satisfecho¹⁹.

¹⁷ LÓPEZ, Introducción al Deep Learning, Op. Cit.

¹⁸ TELEMAGEMENT FORUM. Business Process Framework (eTOM) - Concepts and principles. GB921CP. Version 14.5.2. Morristown, New Jersey, United States: TM Forum, 2015. p. 22.

¹⁹ Ibid., p. 22.

5 ANÁLISIS DE MODELOS DE REDES NEURONALES RECURRENTE

En esta sección se expone el problema de desvanecimiento o explosión de gradiente el cual se convierte en el punto de partida en la selección del modelo de red neuronal recurrente (RNN) para la predicción de series de tiempo. Además de la exposición de cuatro diferentes arquitecturas empleadas para la predicción de series de tiempo, realizando una breve descripción con sus principales características, ventajas y desventajas de cada modelo.

5.1 DESVANECIMIENTO O EXPLOSIÓN DE GRADIENTE

Los primeros modelos existentes de RNN tienen una importante desventaja debido principalmente a su limitada memoria, esta desventaja mejor conocida como desvanecimiento o explosión de gradiente se hace evidente cuando la información contenida en entradas pasadas debe recuperarse después de un intervalo de tiempo largo y cuando los modelos cuentan con grandes profundidades recurrentes²⁰.

Tanto el desvanecimiento como la explosión del gradiente hace referencia a un crecimiento y propagación exponencial de la evaluación realizada por la función de pérdida a través de la red. Esta función de pérdida es la encargada de alcanzar la convergencia de los datos evaluando y ponderando las diferencias entre los valores predichos y los valores futuros reales²⁰. Mientras que en el desvanecimiento del gradiente esta evaluación converge rápidamente a cero, lo que implica que la red deje de aprender, en la explosión esta evaluación converge en valores mayores a uno haciendo que la red entre en un estado caótico donde su capacidad computacional se ve obstaculizada.

Para manejar este problema dentro de las RNN se han adoptado modelos con compuertas como Long Short-Term Memory (LSTM), Gated Recurrent Unit (GRU) que han sido diseñadas específicamente para mitigar el problema de desvanecimiento y permitir que la red aprenda con mayores profundidades recurrentes²⁰.

Otras soluciones consisten en el uso de Rectified Linear Unit (ReLU) como función de activación cuya derivada no hace que el degradado se desvanezca o explote y la

²⁰ BIANCHI. Op Cit., p. 9.

regularización de los datos que, además de prevenir el sobreajuste en la fase de entrenamiento, resulta útil para mitigar la explosión del gradiente²¹.

5.2 ELMAN RECURRENT NEURAL NETWORK (ELMAN RNN O ERNN)

También conocida como Simple Recurrent Network (SRN), es la versión básica de una RNN, la mayoría de las arquitecturas complejas de RNN generalmente son una extensión de las ERNN²¹. Una ERNN está compuesta de una capa de entrada, una capa recurrente que proporciona información de estado, una capa oculta y una capa de salida. En esta red, las salidas de la capa oculta pueden retroalimentarse a través de una capa de almacenamiento intermedio, llamada capa recurrente mediante el cálculo de una función no lineal de la suma ponderada de las nuevas entradas. Esta retroalimentación permite que una ERNN aprenda, reconozca y genere patrones temporales, así como también patrones espaciales, pero también la hace susceptible al desvanecimiento del gradiente.

Cada neurona oculta está conectada a una sola neurona de capa recurrente a través de un peso constante de valor uno. Por lo tanto, la capa recurrente se convierte en una copia del estado de la capa oculta un instante antes, lo que implica que el número de neuronas recurrentes es el mismo que el número de neuronas ocultas²².

Una red recurrente posee capacidades de predicción no lineal, convergencia más rápida y una capacidad de mapeo más precisa, uso de información contextual, características que son heredadas por las ERNN, aplicándose en diferentes contextos, como el procesamiento del lenguaje natural, donde demostró ser capaz de aprender gramática y predecir palabras sucesivas en oraciones, predicción de series de tiempo demostrando ser útil para abordar la tarea de previsión a través de redes más pequeñas y de fácil acceso²¹.

²¹ Ibid., p. 10.

²² WANG, Jie; WANG, Jun; FANG, Wen y NIU, Hongli. Financial Time Series Prediction Using Elman Recurrent Random Neural Networks. Beijing, China: Hindawi Publishing Corporation, 2016. p. 2.

5.3 ECHO STATE NETWORK (ESN)

ESN se considera como un SRN sin Backpropagation Through Time (BPTT) lo que implica que no se ve afectada por el desvanecimiento del gradiente. Compuesta por una gran capa recurrente y escasamente interconectada no entrenada, contiene una entrada x conectada a una función de transferencia no línea que genera los estados h que son almacenados en depósitos w inicializados con pesos aleatorios, que, una vez que se hayan aplicado todas las entradas de entrenamiento, son usados para una regresión lineal entre los estados capturados y las salidas objetivo²³. Estos pesos de salida pueden luego incorporarse a la red existente y usarse para entradas nuevas.

En lugar de un proceso de aprendizaje difícil, los ESN se basan en la propiedad del inicializado aleatorio no entrenado de las RNN para reflejar la historia de las entradas observadas, a las que se hace referencia como "propiedad del estado del eco"²³. Si se cumple esta condición, solo la adaptación de pesos de salida de red es suficiente para obtener RNN con alto rendimiento. Sin embargo, para grandes dinámicas, se necesitan cientos de unidades ocultas.

La propiedad de estado de eco significa que el efecto de una entrada dada sobre el estado del depósito debe desaparecer en un número infinito de instantes de tiempo, generalmente esta dispersión es controlada por un hiper parámetro que determina la cantidad de elementos distintos de cero que deben existir en el depósito. Esto implica que una entrada reciente presentada a la red tiene más influencia en el estado de la red que una entrada anterior, desvaneciéndose gradualmente a mayor historial de señales de entrada²⁴.

La ventaja de este enfoque radica en el procedimiento de entrenamiento increíblemente simple ya que la mayoría de los pesos se asignan solo una vez y al azar²⁵, que los algoritmos de regresión lineal computacionalmente efectivos se pueden usar para ajustar los pesos de salida. Los ESN se han aplicado en una variedad de contextos diferentes, como la clasificación estática, reconocimiento de

²³ ICANN: INTERNATIONAL CONFERENCE ON ARTIFICIAL NEURAL NETWORKS, 17th (9-13 september, 2007, Porto, Portugal.). Comparison of Echo State Networks with Simple Recurrent Networks and Variable-Length Markov Models on Symbolic Sequences. Berlin: Springer Berlin Heidelberg, 2007. p. 618-627.

²⁴ BIANCHI. Op Cit., p. 16.

²⁵ Intuitive explanation of Echo State Networks? [En línea]. Salt Lake City, UT, USA: Adam Conkey, 21 Julio 2015 [Citado en 18 septiembre 2018]. Disponible en internet: <<https://stats.stackexchange.com/questions/140652/what-is-an-intuitive-explanation-of-echo-state-networks>>.

voz, detección de intrusos, control adaptativo, detención de series temporales no estacionarias y, en general, para el modelado de varios tipos de sistemas dinámicos no lineales²⁶.

5.4 LONG SHORT-TERM MEMORY (LSTM)

Una red LSTM es igual a un RNN estándar, excepto que las unidades de sumatoria en la capa oculta se reemplazan por bloques de memoria. Fue desarrollado por Hochreiter y Schmidhuber en 1997, siendo la culminación de su investigación sobre el problema de desvanecimiento del gradiente presente en RNN con muchas capas de profundidad²⁷.

El modelo LSTM consiste en un conjunto de subredes conectadas de manera recurrente, conocidas como bloques de memoria, en donde cada bloque contiene una o más celdas de memoria conectadas automáticamente y tres unidades multiplicativas, las puertas de entrada, salida y olvido, son una análoga de las operaciones de escritura, lectura y reinicio de las celdas²⁸. Las unidades multiplicativas permiten que las celdas de memoria LSTM almacenen y accedan a información durante largos periodos de tiempo, mitigando así el problema de desvanecimiento del gradiente.

La celda de memoria es capaz de recordar la primera entrada siempre que la puerta de olvido esté abierta y la puerta de entrada esté cerrada²⁸, a la vez que la puerta del olvido es capaz de reiniciar celdas de memoria que posee información irrelevante del flujo de datos para que información pasada se reinyecte en un momento posterior²⁷.

Las ventajas del modelo LSTM son más pronunciadas para los problemas que requieren el uso de información contextual de un rango amplio entre ellas, el aprendizaje de lenguajes sin contexto, tareas que requieran tiempos y conteos precisos y en particular, ha resuelto varios problemas que siguen siendo imposibles con cualquier otra arquitectura RNN²⁹.

²⁶ BIANCHI. Op Cit., p. 16.

²⁷ CHOLLET. Op Cit., p. 198.

²⁸ GRAVES, Alex. Supervised Sequence Labelling with Recurrent Neural Networks. Berlin, Springer-Verlag GmbH, 2012. p. 32. ISBN 9783642247972.

²⁹ Ibid., p.32

5.5 GATED RECURRENT UNIT (GRU)

El modelo GRU propuesto por KyungHyun Cho en 2014, es una extensión del modelo LSTM. Donde el modelo GRU de forma similar, tiene unidades de puertas que modulan el flujo de la información para hacer que cada unidad recurrente capture de forma adaptativa dependencias de diferentes escalas de tiempo³⁰.

La gran diferencia entre el modelo GRU y el modelo LSTM radica en que GRU utiliza dos puertas. La primera puerta es de actualización (resultado de combinar la puerta de entrada y la de olvido), que controla cuánto debe actualizarse el contenido actual de la celda con el nuevo estado candidato. El segundo es la puerta de reinicio que, si está cerrada (valor cercano a 0), puede restablecer efectivamente la memoria de la celda y hacer que la unidad actúe como si la siguiente entrada procesada fuera la primera en la secuencia. El estado interno en GRU siempre está completamente expuesto en la salida, debido a la falta de un mecanismo de control, como la puerta de salida en LSTM³¹.

De acuerdo con las evaluaciones empíricas de GRU y LSTM realizadas en los artículos “Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling”³⁰ y “An Empirical Exploration of Recurrent Network Architectures”, no hay un ganador claro. En muchas tareas, ambas arquitecturas producen un rendimiento comparable y los hiper parámetros de ajuste, como el tamaño de capa, son probablemente más importantes que elegir la arquitectura ideal. Los GRU tienen menos parámetros y, por lo tanto, pueden entrenarse un poco más rápido o necesitan menos datos para generalizarse. Por otro lado, si tiene suficientes datos, el mayor poder expresivo de los LSTM puede conducir a mejores resultados³².

Tomando como referencia, la investigación sobre los anteriores cuatro modelos de redes neuronales recurrentes para la predicción de series de tiempo, se realiza un cuadro para la comparación paralela entre las características de cada modelo.

³⁰ CHUNG, Junyoung; GULCEHRE, Caglar; CHO, KyungHyun y BENGIO, Yoshua. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. Montreal, Canadá: Arxiv, 2014. p. 4.

³¹ BIANCHI. Op Cit., p. 13.

³² Recurrent Neural Network Tutorial, Part 4 Implementing a GRU/LSTM RNN with Python and Theano [En línea]. San Francisco (Estados Unidos): Denny Britz, 27 octubre 2015 [Citado en 18 septiembre 2018]. Disponible en Internet: <<http://www.wildml.com/2015/10/recurrent-neural-network-tutorial-part-4-implementing-a-grulstm-rnn-with-python-and-theano>>.

Tabla 1. Comparación de redes neuronales.

Características	ERNN	ESN	LSTM	GRU
Retro propagación	X		X	X
Predicción de series de tiempo	X	X	X	X
Tolerancia al desvanecimiento del gradiente		X	X	X
Tolerancia a la explosión del gradiente		X		
Celdas de memoria			X	X
Documentación	X		X	

Fuente 1. Elaboración propia.

De acuerdo con el anterior cuadro comparativo y debido a que los modelos basados en compuertas actualmente poseen mejores posibilidades de ser escalables a un entorno productivo, este proyecto opta por el uso del modelo LSTM para el desarrollo del proceso predictivo de la cantidad y tipo de actividades de instalación y mantenimiento gestionados por el personal de Colvatel S.A.

6 METODOLOGÍA PARA EL ANÁLISIS, DISEÑO Y CONSTRUCCIÓN DE LA SOLUCIÓN

Para el desarrollo del producto se toma como referencia el ciclo de vida de la metodología ágil e iterativa de ciencias de datos Team Data Science Process (TDSP) propuesto por Microsoft³³. Basado en la metodología Cross Industry Standard Process for Data Mining (CRISP-DM), el TDSP fue creado para ofrecer soluciones analíticas predictivas y aplicaciones inteligentes de manera eficiente.

El TDSP proporciona una definición de infraestructura, recursos, herramientas y ciclo de vida, para estructurar el desarrollo de proyectos de ciencia de datos. Diseñado para proyectos que hacen parte de aplicaciones inteligentes donde se implementan modelos de aprendizaje automático o inteligencia artificial para el análisis predictivo, el ciclo de vida describe los pasos de principio a fin, que los proyectos suelen seguir cuando se ejecutan. En proyectos exploratorios de ciencia de datos o proyectos de análisis también se puede hacer uso de este ciclo. Pero en tales casos, algunos de los pasos descritos pueden no ser necesarios³³.

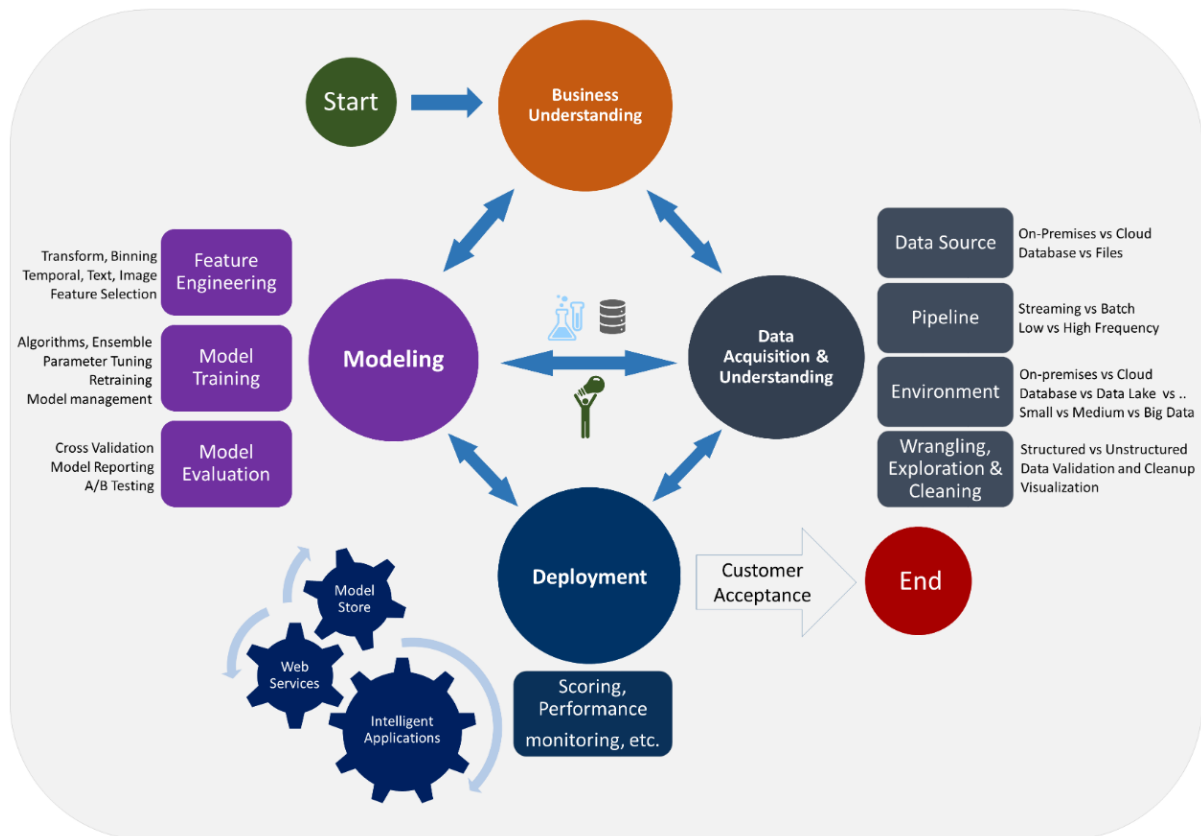
El ciclo de vida TDSP se compone de cinco etapas principales que se ejecutan de forma iterativa. Estas etapas consisten en: entendimiento del negocio, adquisición y entendimiento de los datos, modelado, despliegue y aceptación del cliente.

6.1 ENTENDIMIENTO DEL NEGOCIO

Esta etapa del ciclo de vida consiste en dos tareas principales: Identificar las variables del negocio que son claves para la predicción (Objetivo del modelo), las métricas asociadas con ellos para determinar el éxito del proyecto, e identificar las fuentes de datos relevantes las que la empresa tiene acceso y que son necesarias para elaborar una medida precisa del modelo y de las variables de interés.

³³ Team Data Science Process Documentation [En línea]. Redmond, WA, USA: Microsoft - Debraj GuhaThakurta, 20 octubre 2017 [Citado en 17 octubre 2018]. Disponible en internet: < <https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/overview> >.

Figura 1. Ciclo de vida TDSP.



Fuente 2. Microsoft TDSP.

6.2 ADQUISICIÓN Y ENTENDIMIENTO DE LOS DATOS

Esta etapa es la encargada de producir un conjunto de datos limpio y alta calidad, cuya relación con las variables objetivo se comprenda. Está compuesta de tres tareas principales: Ingesta de datos, Configuración de una tubería o canal de datos y la exploración de los datos.³⁴

6.2.1 Ingesta de datos. Esta tarea se encarga de preparar el proceso para mover los datos de las ubicaciones de origen a las ubicaciones de destino donde se ejecutarán

³⁴ Team Data Science Process Documentation - Data acquisition and understanding [En línea]. Redmond, WA, USA: Microsoft - Debraj GuhaThakurta, 20 octubre 2017 [Citado en 17 octubre 2018]. Disponible en internet: <<https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/lifecycle-data>>.

las operaciones de análisis, entrenamiento y predicciones. Esta tarea se desarrolla en paralelo con la tarea de configuración de una tubería³⁵.

6.2.2 Configuración de una tubería o canal de datos. Esta tarea consiste en la configuración un canal para obtención y actualización de los datos como parte del proceso de aprendizaje continuo. Dependiendo de las necesidades del negocio o de las limitaciones en donde se integra la solución, la canalización puede ser por lote de datos, en tiempo real (streaming) o híbrido³⁵.

6.2.3 Exploración de los datos. Esta tarea busca evaluar y mejorar la calidad de los datos, además de desarrollar una buena comprensión de estos, por lo que a menudo se deben llevar a cabo de forma iterativa las actividades de preprocesamiento y análisis exploratorio de los datos³⁵.

6.2.3.1 Preprocesamiento de datos. Debido a que normalmente el uso de datos de baja calidad implica un proceso de aprendizaje con resultados poco fiables, se hace necesaria la aplicación de actividades de preprocesamiento tales como la limpieza y reducción de los datos, cuyo objetivo principal es convertir los datos importados en un conjunto de datos final que sea de calidad y útil para la fase de entrenamiento del modelo de predicción³⁵.

- Limpieza de datos: Esta tarea consiste en detectar, completar o eliminar datos incompletos o nulos, eliminar datos erróneos y valores atípicos, eliminar caracteres incrustados o ilegibles que pueden causar una desalineación de los datos³⁶.
- Reducción de datos: Consiste en la agregación de los datos en grupos o el muestreo de registros o atributos y la selección del subconjunto representativo de los datos³⁷.

³⁵ Ibid.

³⁶ GARCÍA, Salvador; RAMÍREZ GALLEGO, Sergio; LUENGO, Julián y HERRERA, Francisco. Big Data: Preprocesamiento y calidad de datos. En: Novática. Julio-octubre, 2016, no. 237, p.17-23.

³⁷ Ibid., p.18.

6.2.3.2 Análisis exploratorio de datos. Esta actividad consiste en realizar el análisis y exploración de variables, cálculos particulares, ejercicios gráficos de donde se puede ir analizando posibles relaciones o visualizando conexiones entre ciertas variables, hasta concluir el comportamiento global de los datos. La exploración se puede clasificar en medidas de posición, dispersión y forma.³⁸

- Medidas de posición o de tendencia central permiten determinar la posición de un valor respecto a un conjunto de datos considerado como representativo o típico para el total de observaciones. Entre estas medidas se consideran el cálculo de la media, del mínimo, del máximo, la moda, los cuantiles³⁹.
- Medidas de dispersión permiten determinar la concentración de datos alrededor del promedio, para saber hasta donde se da la representatividad de ese conjunto de datos. Entre estas se consideran el rango de inter-cuartiles, el rango de los datos, la varianza y la desviación estándar⁴⁰.
- Medidas de asimetría y apuntamiento permiten determinar si una distribución de frecuencias o colección de datos presentan uniformidad. Estas medidas son la asimetría y curtosis⁴¹.

6.3 MODELADO

Esta etapa se encarga de determinar los atributos óptimos para el modelo de aprendizaje automático, de crear el modelo con mayor precisión al predecir el objetivo y que dicho modelo sea adecuado para la producción. Esta etapa se compone de tres tareas principales: Selección de atributos, entrenamiento y evaluación del modelo.

³⁸ Análisis Confirmatorio vs Exploratorio [En línea]. Austin, TX, USA: Daniel Legorreta, 12 marzo 2015 [Citado en 17 octubre 2018]. Disponible en internet: <<https://dlegorreta.wordpress.com/2015/03/12/analisis-confirmatorio-vs-exploratorio/>>

³⁹ BENCARDINO MARTÍNEZ. Op. cit., p 92.

⁴⁰ Ibid., p 144.

⁴¹ Ibid., p 167, 170.

6.3.1 Ingeniería y selección de atributos. Los datos de entrenamiento utilizados en el aprendizaje automático a menudo se pueden mejorar mediante la extracción de características de los datos sin procesar recopilados. La ingeniería de atributos es un acto de equilibrio para encontrar e incluir variables informativas, pero al mismo tiempo tratar de evitar demasiadas variables no relacionadas. Normalmente, la ingeniería de atributos se aplica primero para generar características adicionales, y luego el paso de selección de características se realiza para eliminar características irrelevantes, redundantes o altamente correlacionadas⁴².

6.3.1.1 Ingeniería de atributos. Este proceso intenta crear características relevantes adicionales a partir de las características en bruto existentes en los datos, y aumentar el poder predictivo del algoritmo de aprendizaje. Implica acciones de inclusión y agregación de variables brutas para crear las características utilizadas en el análisis⁴².

6.3.1.2 Selección de atributos. Este proceso selecciona el subconjunto clave de características de datos originales y los normaliza (Transforma) a una misma escala en un intento por reducir el ruido, las dimensiones de los datos y el tiempo invertido durante la etapa de entrenamiento⁴².

6.3.2 Entrenamiento del modelo. El objetivo del entrenamiento del modelo consiste en aprender a generalizar patrones aplicables a datos sin analizar en lugar de memorizar los datos que ha visto durante el entrenamiento. Esta tarea incluye dos procesos: Dividir los datos de entrada al azar en dos conjuntos de datos, uno entrenamiento y otro de evaluación, y construir el modelo utilizando el conjunto de datos de entrenamiento⁴².

6.3.3 Evaluación del modelo. Esta tarea busca determinar la mejor solución para responder la pregunta objetivo, comprobando si el modelo tiene un buen rendimiento en relación con los datos de ejemplo sin analizar que no haya usado

⁴² Team Data Science Process Documentation - Modeling [En línea]. Redmond, WA, USA: Microsoft - Debraj GuhaThakurta, 20 octubre 2017 [Citado en 17 octubre 2018]. Disponible en internet: < <https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/lifecycle-modeling> >.

para el entrenamiento del modelo. Para esto, se llevan a cabo los procesos de puntuación y evaluación⁴³.

6.3.3.1 Puntuación. En este proceso se utiliza el modelo entrenado para predecir la respuesta sobre el conjunto de datos de evaluación (datos omitidos)⁴⁴.

6.3.3.2 Evaluación. Este proceso permite decidir si el modelo cumple con el rendimiento esperado. Se lleva a cabo con la revisión de diferentes métricas que dependen del tipo de modelo que se esté evaluando. Las métricas para los modelos de regresión generalmente están diseñadas para estimar la cantidad de error⁴³.

Se considera que un modelo se ajusta bien a los datos si la diferencia entre los valores observados y predichos es pequeña. Sin embargo, observar el patrón de los residuos (la diferencia entre cualquier punto predicho y su valor real correspondiente) puede decir mucho sobre posibles sesgos en el modelo. Las siguientes métricas son reportadas para evaluar modelos de regresión⁴³.

- El error absoluto medio (MAE) mide qué tan cerca están las predicciones de los resultados reales; Por lo tanto, una puntuación más baja es mejor⁴³.
- El error cuadrático medio (RMSE) crea un valor único que resume el error en el modelo. Al cuadrar la diferencia, la métrica ignora la diferencia entre la predicción excesiva y la predicción insuficiente⁴³.
- El coeficiente de determinación, a menudo denominado R², representa el poder predictivo del modelo como un valor entre 0 y 1. Cero significa que el modelo es aleatorio (no explica nada); 1 significa que hay un ajuste perfecto. Sin embargo, se debe tener cuidado al interpretar los valores de R², ya que los valores bajos pueden ser completamente normales y los valores altos pueden ser sospechosos⁴³.

⁴³ Evaluate Model [En línea]. Redmond, WA, USA: Microsoft - Roope Astala, 10 octubre 2018 [Citado en 17 octubre 2018]. Disponible en internet: < <https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/evaluate-model> >.

⁴⁴ Score Model [En línea]. Redmond, WA, USA: Microsoft - Roope Astala, 24 enero 2018 [Citado en 17 octubre 2018]. Disponible en internet: < <https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/score-model> >.

6.4 DESPLIEGUE

Esta etapa hace referencia al despliegue del canal de datos y del modelo en un entorno de producción o similar, exponiendo los modelos con una interfaz API abierta para que pueda ser consumido por diferentes aplicaciones como sitios web, hojas de cálculo, aplicaciones back-end, etc.⁴⁵.

6.5 ACEPTACIÓN DEL CLIENTE

Esta etapa final busca confirmar que el canal de datos y el modelo implementado genere resultados con una precisión aceptable que satisfaga las necesidades del cliente. El proyecto se transfiere a la entidad responsable de las operaciones tras finalizar y revisar toda la documentación⁴⁶.

⁴⁵ Team Data Science Process Documentation - Deployment [En línea]. Redmond, WA, USA: Microsoft - Debraj GuhaThakurta, 04 noviembre 2017 [Citado en 17 octubre 2018]. Disponible en internet: < <https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/lifecycle-deployment> >.

⁴⁶ Team Data Science Process Documentation - Customer acceptance [En línea]. Redmond, WA, USA: Microsoft - Debraj GuhaThakurta, 04 noviembre 2017 [Citado en 17 octubre 2018]. Disponible en internet: < <https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/lifecycle-acceptance> >.

7 ANÁLISIS DE INFORMACIÓN HISTÓRICA REQUERIDA PARA LA CONSTRUCCIÓN DE LA SOLUCIÓN

En este capítulo se llevará a cabo el desarrollo de las etapas de Entendimiento del negocio y Adquisición de los datos, especificados en el ciclo de vida de la metodología Team Data Science Process (TDSP). Estas dos etapas nos permitirán identificar la información clave para el proceso de entrenamiento del modelo y predicción de valores futuros del número y tipo de servicios de instalación y mantenimiento gestionados por Colvatec S.A.

7.1 ENTENDIMIENTO DEL NEGOCIO

Para el desarrollo de esta etapa del proyecto, se solicitó a la jefatura de Informática de Colvatec S.A. E.S.P. el acceso a los datos almacenados en la aplicación Colvaplus, plataforma para la gestión de los servicios de Aprovisionamiento y Aseguramiento atendidos por Colvatec S.A. E.S.P. y ofertados por ETB S.A. E.S.P. Los datos entregados corresponden a una copia de la Base de datos ADSLplus generado por la herramienta de exportación de MS Azure SQL.

En esta base de datos se encuentran registradas todas las actividades atendidas en los segmentos de aprovisionamiento y aseguramiento de los servicios de línea básica y banda ancha cobre desde el 1 de enero de 2013 hasta el 31 de diciembre de 2017, con información adicional sobre el tipo de actividad atendida, fecha de agendamiento, técnico asignado para la gestión de la actividad, fecha de cierre y código de cierre de la actividad.

Es necesario aclarar que los datos de referencia que se utilizaron desde la ingesta hasta la construcción y evaluación del modelo corresponden a la totalidad de actividades gestionadas con código de cierre exitoso en el segmento de aprovisionamiento, entendiéndose que la mecánica de aquí en adelante es la misma para cualquier tipo de actividad que se desee analizar. Aun así, dentro del canal de datos se adiciona el parámetro Oferta, el cual permitirá obtener solamente la cantidad de actividades gestionadas en dicha oferta.

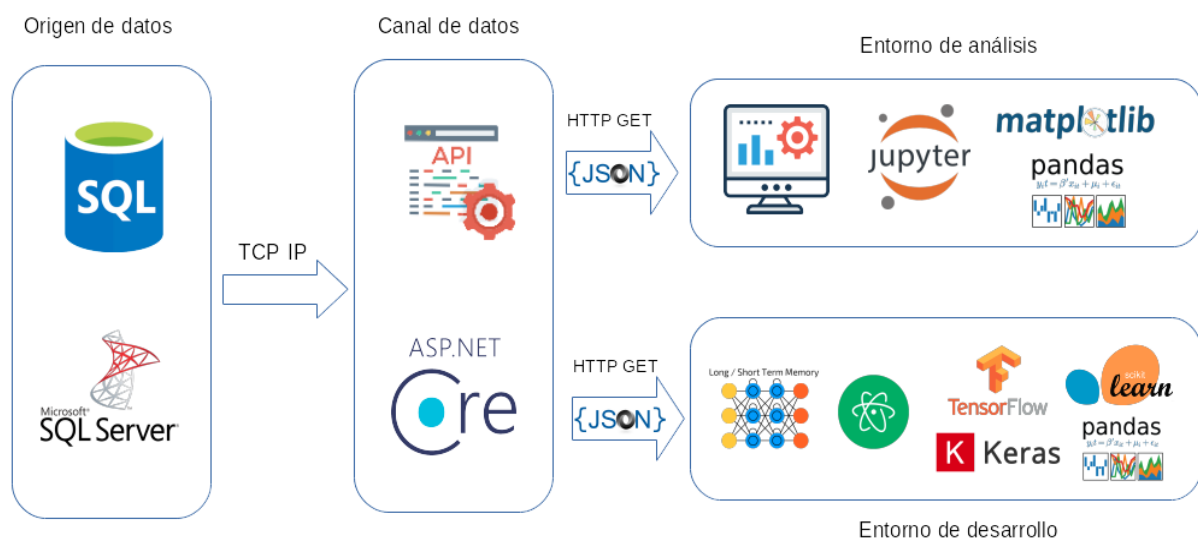
7.2 ADQUISICIÓN Y ENTENDIMIENTO DE LOS DATOS

Con la información disponible, se ejecutarán las etapas de Ingesta de datos, preprocesamiento y exploración. tres etapas encargadas de preparar los datos que serán inyectados en el entrenamiento del modelo predictivo.

7.2.1 Ingesta de datos. En esta etapa, se restaura la copia de la base de datos ADSLplus en una instancia local de SQL Server 2017, el cual es compatible con la versión SQL v14 del servicio MS Azure SQL. Dentro de la aplicación web Jupyter Notebook, entorno para análisis y desarrollo de modelos para el aprendizaje automático, se importan las bibliotecas de código abierto requests, json y pandas por medio de pip (el sistema de gestión de paquetes para Python), con el objetivo importar los datos restaurados a través de peticiones request al canal de datos y se manipulados por medio de un dataframe de pandas.

7.2.2 Construcción del canal de datos. Como la necesidad de los datos corresponden al modelo relacional que actualmente se utiliza en la aplicación Colvapplus, se opta por la construcción de un servicio Web API Rest con ASP.net Core, autogenerando la construcción del modelo relacional con Entity Framework Core. Dentro de este Web API se realiza la reducción de los datos por el método de agregación, el cual se detallará en la etapa de preprocesamiento.

Figura 2. Arquitectura de la solución



Fuente 3. Elaboración propia.

Para el almacenamiento y control de cambios del código fuente en el canal de datos, se creó el siguiente repositorio en GitLab: <https://gitlab.com/TestType/webapi-dl>.

7.2.3 Exploración de los datos. Tomando como referencia las actividades propuestas por la metodología TDSP para esta etapa, a continuación, se describe las actividades que se llevaron a cabo en el preprocesamiento de los datos y las medidas que se utilizaron en el análisis exploratorio.

7.2.3.1 Preprocesamiento. Como primer paso, con las funciones RTRIM y LTRIM de SQL se realizó la eliminación de los espacios en blanco registrados dentro de la columna oferta de las tablas ADSLPLUS_APROVISIONAMIENTO y ADSLPLUS_LINEABASICA almacenadas en la base de datos ADSLplus, actualizando 813401 registros.

En las dos tablas anteriores, también se ejecutó la función LIKE de SQL para buscar las ofertas incorrectas y corregirlas por las ofertas ADSL BASICO, ADSL CORPORATIVO, ADSL MASIVO, ADSL SUPERIOR, CAMBIO DE EQUIPO, CAMBIO DE NUMERO, CAMBIO DE SERVICIO, INSTALACION DUO, INSTALACION SINGLE, INSTALACION TRIO, MIGRACION SINGLE, TRASLADO DE SERVICIO y REDES INTERNAS. Definidas por la dirección de operaciones, registrándose el número de registros afectados en cada caso.

Tabla 2. Correcciones tabla ADSLPLUS_APROVISIONAMIENTO

Oferta incorrecta	Oferta correcta	Registros afectados
ADSL MASIO / ADLS MASIVO	ADSL MASIVO	4
CRUZADA	ADSL MASIVO	944
PRECABLEADO / LEGALIZACION	ADSL MASIVO	138
CAMBIO EQUIPO / CAMBIO_EQUIPO	ADSL MASIVO	49504
CAMBIO PRODUCTO / CAMBIO DE PRODUCTO	CAMBIO DE SERVICIO	26724
NSTALACION SIN	INSTALACION SINGLE	25222
LINEA BASICA / INSTALACION	INSTALACION SINGLE	96
TRIO 11M HOG	INSTALACION TRIO	10
LINEA BASICA + ADSL BASICO	INSTALACION DUO	338
TRASLADO EXTERNO / TRASLADO DEL SERV	TRASLADO DE SERVICIO	987
MIGRACION SIGNLE / GRACION SIN	MIGRACION SINGLE	12529

Fuente 4. Elaboración propia.

Tabla 3. Correcciones tabla ADSLPLUS_LINEABASICA.

Oferta incorrecta	Oferta correcta	Registros afectados
INSTALACION DE REDES INTERNAS	REDES INTERNAS	90037
INSTALACION DUO LB+ INTERNET / LINEA BASICA + ADSL	INSTALACION DUO	1251414
INSTALACION TRIO LB+INTERNET+TV / LINEA BASICA + ADSL SUPERIOR + TELEFONIA IP	INSTALACION TRIO	30985
INSTALACION SINGLE LB / LINEA BASICA	INSTALACION SINGLE	472968
TRASLADO INTERNO / TRASLADO EXTERNO / TRASLADO DEL SERVICIO	TRASLADO DE SERVICIO	203680

Fuente 5. Elaboración propia.

Para la limpieza de valores atípicos, se crea la función limpiar_outliers dentro del entorno de análisis y desarrollo. Esta función recibe como parámetro un dataframe de pandas y a partir de este calcula el rango intercuartil con la diferencia entre el tercer cuartil y el primer cuartil. Este rango intercuartil, multiplicado por la diferencia entre el cuartil uno y el margen de aceptación y multiplicado por la suma del cuartil tres y el margen de aceptación, define los elementos que en definitiva no encajan con la muestra de datos procesada, retornando un dataframe sin estos valores.

Figura 3. Función para la limpieza de valores atípicos.

```

24 # Funcion para limpiar los valores atipicos del set de datos
25 def limpiar_outliers(df):
26     filtered = None
27     rango = list(range(df.index.min().year, df.index.max().year+1))
28     for x in rango:
29         Q1, Q3 = df[str(x)].cantidad_servicios.quantile([0.25,0.75])
30         IQR = Q3 - Q1
31         datatemp = df[str(x)].query('(@Q1 - 1.5 * @IQR) <= cantidad_servicios <= (@Q3 + 1.5 * @IQR)')
32         filtered = pd.concat([filtered,datatemp])
33     return filtered

```

Fuente 6. Elaboración propia.

La reducción de los datos por el método de agregación se realizó dentro del servicio Web API, por medio de consultas con expresiones lambda, en donde a partir de la unión de los datos en las tablas ADSLPLUS_APROVISIONAMIENTO y ADSLPLUS_LINEABASICA, se realiza una agrupación por fecha de agendamiento, retornando los datos de Fecha, cantidad de actividades gestionadas como exitosas y la cantidad de técnicos disponibles en esa fecha.

Figura 4. Consulta general datos de aseguramiento y aprovisionamiento.

```

59     public List<DLActividades> ConsultarActividadesHistorico(DateTime FechaIni, DateTime FechaFin, string oferta = "") {
60         List<DLActividades> listaactividades = new List<DLActividades>();
61
62         listaactividades.AddRange(_context.AdsIplusAprovisionamiento
63             .Where(x => x.CodEstado.Contains("RDY")
64                 && x.FechaAgenda >= FechaIni && x.FechaAgenda <= FechaFin
65                 && (oferta != null ? x.Oferta.Contains(oferta) : true))
66             .Select(x => new DLActividades {
67                 Linea = x.Linea,
68                 Solicitud = x.Solicitud,
69                 FechaAgenda = x.FechaAgenda,
70                 FechaCierre = x.FechaCierre,
71                 Oferta = x.Oferta,
72                 TipoProducto = x.TipoProducto,
73                 Estado = x.CodEstado,
74                 CodigoCierre = x.CodigoCierre,
75                 IdTecnico = _context.AdsIplusAprovAgendas
76                     .Where(i => i.LineaFk == x.Linea && i.SolicitudFk == x.Solicitud && i.FechaAgendaFk == x.FechaAgenda)
77                     .OrderBy(i => i.CtivoAgenda)
78                     .Select(i => i.IdTecnicoFk)
79                     .LastOrDefault()
80             }).ToList());
81
82         listaactividades.AddRange(_context.AdsIplusLineabasica
83             .Where(x => x.CodEstado.Contains("RDY")
84                 && x.Fecha >= FechaIni && x.Fecha <= FechaFin
85                 && (oferta != null ? x.Tipo.Contains(oferta) : true))
86             .Select(x => new DLActividades
87             {
88                 Linea = x.Codigoa,
89                 Solicitud = x.Solicitud,
90                 FechaAgenda = x.Fecha,
91                 FechaCierre = x.FechaCierre,
92                 Oferta = x.Tipo,
93                 TipoProducto = x.TipoProducto,
94                 Estado = x.CodEstado,
95                 CodigoCierre = x.CodCierre,
96                 IdTecnico = x.IdTecnico.ToString() ?? ""
97             }));
98
99         return listaactividades;
100     }

```

Fuente 7. Elaboración propia.

Figura 5. Agrupamiento y reducción de los datos.

```

114     public dynamic ConsultarCantidadActividadesPorDia(DateTime FechaIni, DateTime FechaFin, string oferta)
115     {
116         List<DLActividades> listaactividades = ConsultarActividadesHistorico(FechaIni, FechaFin, oferta);
117
118         var lstagrupamiento = listaactividades
119             .GroupBy(x => new { Fecha = x.FechaAgenda.Date })
120             .Select(x => new {
121                 x.Key.Fecha,
122                 cantidad_servicios = x.Count(),
123                 cantidad_tecnicos = x.Select(c => c.IdTecnico).Distinct().Count() })
124             .OrderBy(x => x.Fecha)
125             .ToList();
126
127         return lstagrupamiento;
128     }

```

Fuente 8. Elaboración propia.

El set de datos original contenía dos atributos categóricos no numéricos (tipo de actividad atendida y código de cierre de la actividad) los cuales son utilizados como parámetros para filtrar la información que será utilizada en el análisis exploratorio y entrenamiento de la red neuronal, por lo que no fue necesario aplicar ningún tipo de técnica de transformación de datos ya que la información resultante corresponde a datos ordinales y nominales (fechas y cantidades).

7.2.3.2 Análisis exploratorio. Para el desarrollo del análisis de los datos, se toma el dataframe generado por la biblioteca pandas y a partir de este, se ejecutan las instrucciones adjuntas en las siguientes tablas, con el fin de realizar las medidas de posición, dispersión, asimetría y apuntamiento expuestos en el TDSP. El set de datos procesado estaba compuesto de 1542 registros y 3 columnas sin ningún valor vacío o nulo.

Tabla 4. Información del set de datos.

Características	Instrucción	Información
Dimensiones (filas, columnas)	df.shape	(1542, 3)
Columnas	df.columns	cantidad_servicios, cantidad_tecnicos, fecha
Tipo de datos	df.dtypes	cantidad_servicios int64 cantidad_tecnicos int64 fecha datetime64[ns]
Registros por columna y valores nulos	df.info ()	cantidad_servicios. 1542 non-null cantidad_tecnicos 1542 non-null fecha 1542 non-null

Fuente 9. Elaboración propia.

- Medidas de posición. El rango de datos que se toma para este análisis corresponde a las fechas del 1 de enero del 2013 hasta el 31 de diciembre del 2017. Con la información registrada la siguiente tabla se puede inferir la existencia de valores atípicos a partir de los valores máximos y mínimos disponibles en el set de datos, siendo crítico el hecho de que el dato que se repite con más frecuencia en la cantidad de actividades sea 1.

Tabla 5. Medidas de posición.

Características	Instrucción	Información
Media	df.mean()	cantidad_servicios: 1242.311284 cantidad_tecnicos: 213.584955
Mediana	df.median()	cantidad_servicios: 1278.0 cantidad_tecnicos: 202.0
Moda	df.mode()	cantidad_servicios: 1 cantidad_tecnicos: 196
Min	df.describe()	Min cantidad_servicios: 1 Min cantidad_tecnicos: 1
Max	df.describe()	Max cantidad_servicios: 2263 Max cantidad_tecnicos: 333
Q1(25%)	df.describe()	Q1 cantidad_servicios: 947 Q1 cantidad_tecnicos: 177
Q2(50%)	df.describe()	Q2 cantidad_servicios: 1278 Q2 cantidad_tecnicos: 202
Q3(75%)	df.describe()	Q3 cantidad_servicios: 1541 Q3 cantidad_tecnicos: 269.75
Top (fecha)	df.describe()	2016-09-08 00:00:00
Primero (fecha)	df.describe()	2013-01-01 00:00:00
Ultimo (fecha)	df.describe()	2017-12-30 00:00:00

Fuente 10. Elaboración propia.

- Medidas de dispersión. Con la información registrada en el siguiente cuadro, se puede observar que los datos cantidad_servicios poseen un promedio de variación muy alto con respecto a la media, lo cual sirve como sospecha de que el valor de la moda en cantidad_servicios es en realidad un valor atípico que influye notablemente en el promedio de dispersión de los datos.

Tabla 6. Medidas de dispersión.

Características	Instrucción	Información
Varianza	df.var(ddof=0)	cantidad_servicios: 185737.926449 cantidad_tecnicos: 3982.799203
Desviación estándar	df.std(ddof=0)	cantidad_servicios: 430.973232 cantidad_tecnicos: 63.109422

Tabla 6. (Continuación.)

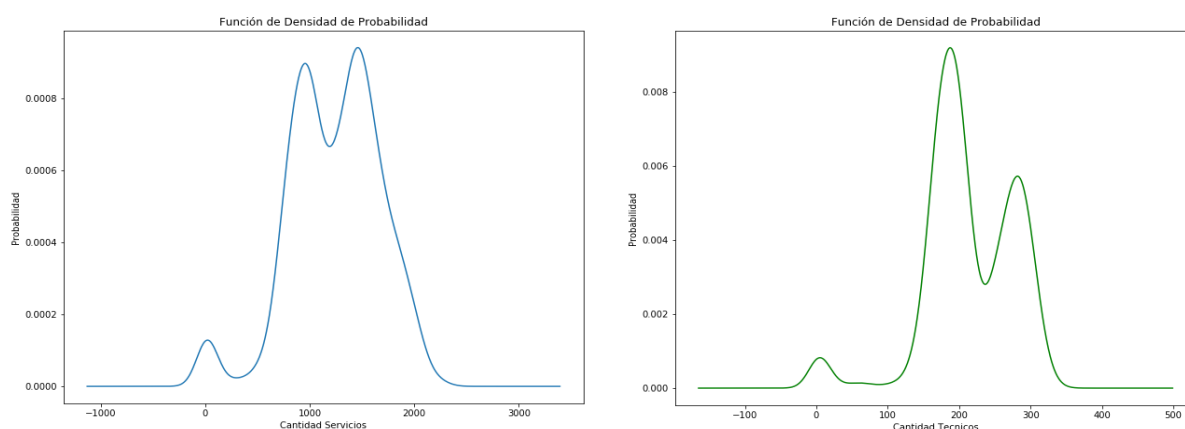
Características	Instrucción	Información
Rango máximo y mínimo	df.max()- df.min()	cantidad_servicios: 2262 cantidad_tecnicos: 332
Rango Inter cuartil	df.quantile(0.75) - df.quantile(0.25)	cantidad_servicios: 594.00 cantidad_tecnicos: 92.75

Fuente 11. Elaboración propia.

En el gráfico de densidad de la cantidad de actividades se observa que la probabilidad de que se registren diariamente, cantidades similares a las que actualmente se encuentran almacenadas es realmente muy baja. Aun así, se observa cierta regularidad entre valores de 700 y 1500 actividades atendidas diariamente. También se observa una cota cercana a cero que estaría influyendo en la desviación estándar de los datos.

El gráfico de densidad para la cantidad de técnicos disponibles presenta un comportamiento similar al gráfico de densidad para la cantidad de servicios. En ambos casos se observa que, si bien ambos datos poseen una cota superior, esta no es la única, por lo que la distribución de los datos actualmente no es una distribución normal, lo que implica la necesidad de normalizar ambos datos.

Figura 6. Densidad cantidad de actividades y técnicos.



Fuente 12. Elaboración propia.

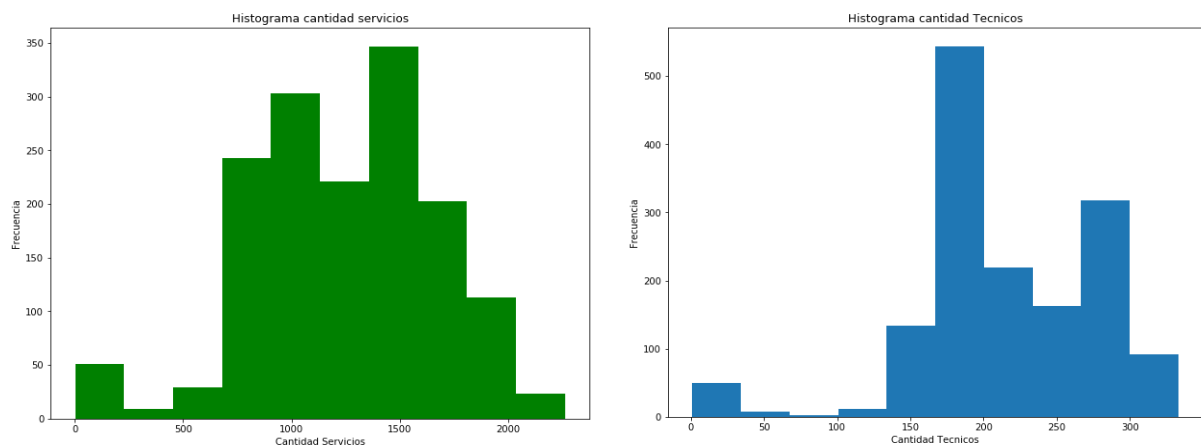
El grafico conocido como histograma, permite observar con más detalle los grupos de valores que poseen mayor frecuencia de aparecer en los datos. En el histograma

de cantidad de actividades, se puede confirmar que grupos de valores con mayor frecuencia están entre las 1000 y 1500 actividades atendidas, con una caída entre las 1200 y 1300 actividades atendidas, lo que implica que Colvatel por lo general, diariamente está gestionando entre 1000 y 1500 actividades diariamente.

También se observa que valores de 1 a 100 en la cantidad de actividades atendidas tienen una frecuencia moderada lo que haría que la influencia de este grupo de valores atípicos en la media sea algo inminente.

El histograma para la cantidad de técnicos requeridos diariamente en la gestión de actividades refleja que se ha recurrido con mayor frecuencia a grupos entre 170 y 190 técnicos y en ocasiones particulares a grupos con el doble de técnicos. También se observa que grupos de 1 y 30 técnicos pueden estar relacionados con valores atípicos en la cantidad de actividades gestionadas.

Figura 7. Histograma cantidad de actividades y técnicos.

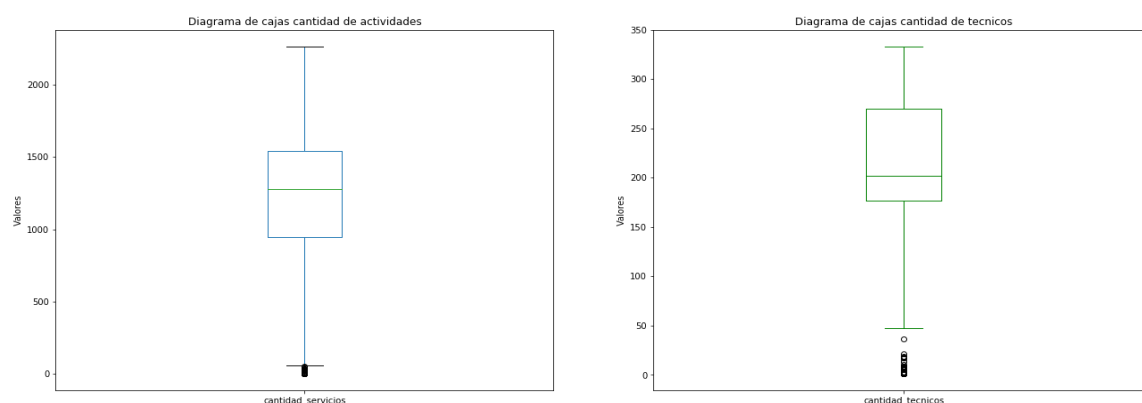


Fuente 13. Elaboración propia.

El grafico conocido como diagrama de cajas permite visualizar con facilidad posibles valores atípicos presentes en los datos. En el diagrama de cajas para la cantidad de actividades gestionadas por día se confirma que el valor de la moda para la cantidad de actividades, en realidad corresponde a un grupo de valores atípicos con valores entre 1 y 100.

En el diagrama de cajas para la cantidad de técnicos disponibles en la gestión de actividades, también existen valores atípicos que podrían estar asociados con los valores atípicos detectados en la cantidad de actividades gestionadas.

Figura 8. Diagrama de cajas para la cantidad de actividades y técnicos.



Fuente 14. Elaboración propia.

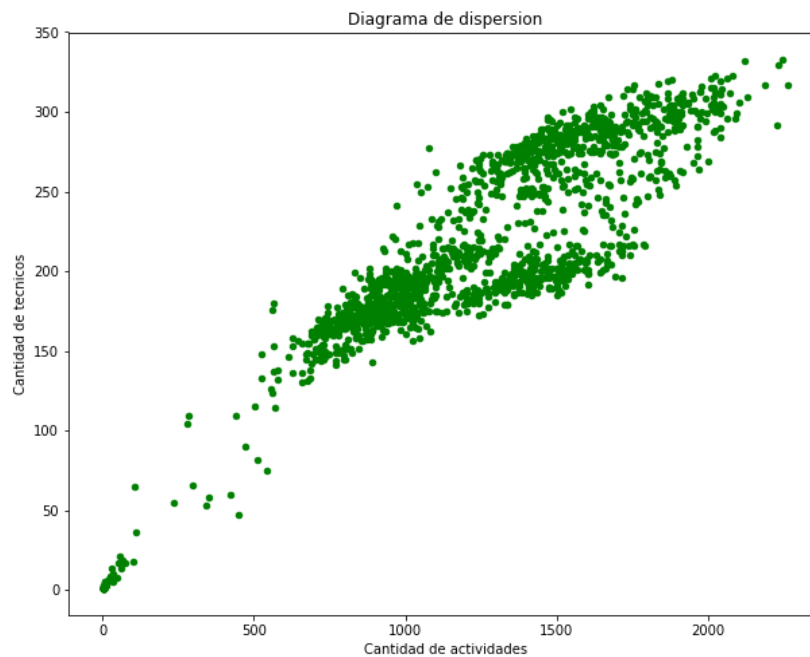
Para finalizar las medidas de dispersión, se elabora la tabla de correlación de variables y el grafico dispersión, en donde se puede notar una alta correlación entre la cantidad de actividades gestionadas y la cantidad de técnicos necesarios para la atención de estas actividades. Esto significa que, a mayor cantidad de actividades a gestionar, mayor cantidad de técnicos se necesitaran. Así mismo se observa una brecha entre 700 actividades y 130 técnicos, lo que implica que valores menores entre estas dos variables, es poco frecuente.

Tabla 7. Variación y correlación de los datos.

Características	Instrucción	Información
Coeficiente de variación Actividades	ss.variation()	0.3469124343005351
Coeficiente de variación técnicos	ss.variation()	0.29547691022816214
Covarianza	df.cov()	24265.745115
Correlación	df.corr(method='pearson')	0.891595

Fuente 15. Elaboración propia.

Figura 9. Diagrama de dispersión técnicos vs actividades.



Fuente 16. Elaboración propia.

- Medidas de asimetría. Con esta información podemos inferir que la cola de la distribución de los datos se alarga para valores inferiores a la media (Asimetría), pero posee cierta concentración de datos en la media (Curtosis). Lo cual indica que los datos no poseen una distribución normal y por ende son difícilmente estacionarios. Este escenario hace que sea necesario aplicar un método de normalización o escalado de los datos.

Tabla 8. Asimetría y apuntamiento de los datos.

Características	Instrucción	Información
Asimetría	ss.skew(df)	Array ([-0.48496439, -0.85994488])
Curtosis	ss.kurtosis(df)	Array ([0.41510192, 1.82231771])

Fuente 17. Elaboración propia.

8 CONSTRUCCIÓN Y ENTRENAMIENTO DE LA RED NEURONAL

En esta etapa del proyecto, se lleva a cabo los procesos de transformación de atributos, división del lote de datos, construcción y entrenamiento del modelo predictivo. Para su desarrollo, se establece el diseño del modelo a partir del grafo del flujo de datos que debe realizar el modelo predictivo, especificación de bibliotecas y recursos tecnológicos utilizados.

Tabla 9. Herramientas de desarrollo.

Herramienta/Biblioteca	Versión	Descripción
Pandas	0.23.4	Biblioteca para el procesamiento de estructuras de datos en análisis de datos, series de tiempo y estadística.
Numpy	1.13.3	Biblioteca para el procesamiento de arreglos de números, cadenas, registros y objetos.
Scikit-learn	0.20.0	Set de módulos para el aprendizaje automático y minería de datos.
Tensorflow	1.9.0	Biblioteca para el aprendizaje automático y la realización de cálculos numéricos mediante diagramas de flujo de datos.
Tensorboard	1.8.0	Herramienta para la depuración, visualización y optimización de flujo de datos.
Keras	2.2.4	API de redes neuronales de alto nivel
Matplotlib	3.0.0	Biblioteca para graficado 2D
Jupyter notebook	1.0.0	Bloc de notas para la computación interactiva sobre un entorno web.

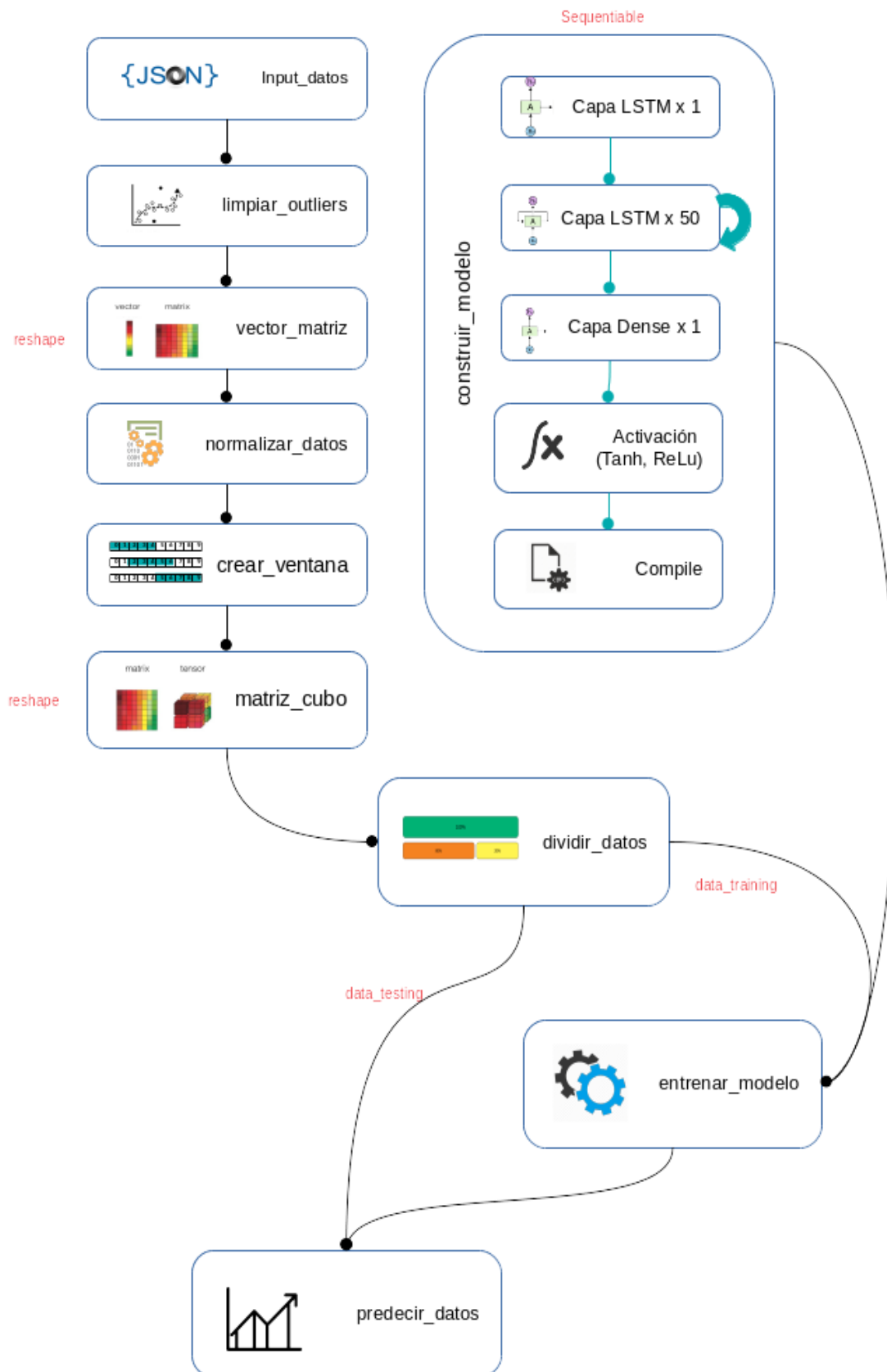
Fuente 18. Elaboración propia.

Tabla 10. Herramientas hardware

Hardware	Especificaciones
Procesador Intel core i7 HQ	N°. núcleos (físicos, hilos): 4,8. Cache: 6 MB Frecuencia (Base, Max): 2.60 GHz, 3.50 GHz
GPU Nvidia GTX 1050Ti	Núcleos CUDA: 768 Frecuencia (Gráficos, Procesamiento): 1290 Mhz, 1392 Mhz Frecuencia memoria: 7 Gbps Memoria estándar: 4gb GDDR5 128-bit

Fuente 19. Elaboración propia.

Figura 10. Diseño del modelo predictivo.



Fuente 20. Elaboración propia.

Para el almacenamiento, seguimiento y control de cambios del código fuente en el modelo predictivo, se creó el siguiente repositorio en la plataforma GitLab: <https://gitlab.com/TestType/ModeloPredictivoDL>.

8.1 INGENIERIA Y SELECCIÓN DE ATRIBUTOS (TRANSFORMACIÓN)

Debido a que, en la etapa de análisis exploratorio de los datos, se detectó la influencia de valores atípicos en la cantidad de actividades gestionadas y en la cantidad de técnicos disponibles, se optó por ejecutar los métodos RobustScaler, PowerTransformer, QuantileTransformer y MinMaxScaler para la normalización o escalado de los datos, los cuales cuentan con implementación disponible en la biblioteca Scikit-learn.

El método RobustScaler realiza el escalado de los datos sin eliminación de valores atípicos ya que las estadísticas de centrado y escalado de este método se basan en percentiles y, por lo tanto, no están influenciadas por un número reducido de valores atípicos marginales muy grandes⁴⁷.

El método PowerTransformer realiza el escalado de los datos con valores atípicos eliminados, aplicando una transformación de potenciación a cada atributo para hacer que los datos sean más similares a una Distribución Gaussiana (Distribución normal)⁴⁷.

El método QuantileTransformer aplica una transformación no lineal de manera que la función de densidad de probabilidad de cada característica se asignará a una distribución uniforme. En este caso, todos los datos se asignarán en el rango [0, 1], incluso los valores atípicos que ya no se pueden distinguir de los valores típicos⁴⁷.

El método MinMaxScaler escala el conjunto de datos de tal manera que todos los valores de las características quedan en el rango [0, 1]. Sin embargo, esta escala comprime todos los valores internos en el rango estrecho [0, 0.005]. MinMaxScaler es muy sensible a la presencia de valores atípicos⁴⁸.

⁴⁷ Compare the effect of different scalers on data with outliers [En línea]. Domaine de Voluceau, Rocquencourt, France: INRIA - Fabian Pedregosa, 1 febrero 2010 [Citado en 19 octubre 2018]. Disponible en internet: < http://scikit-learn.org/stable/auto_examples/preprocessing/plot_all_scaling.html >

⁴⁸ Ibid.

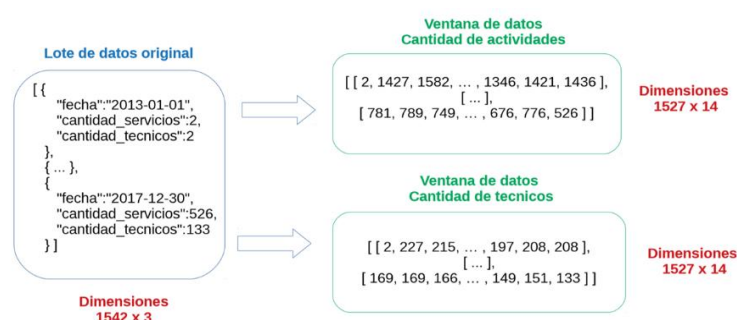
La selección de estos cuatro métodos de escalado de datos se realizó con el fin de abarcar dos posibles escenarios frente a los valores atípicos. El primer escenario posible consiste en entender estos valores atípicos como resultados de la mecánica operativa de Colvatel, por lo cual no deben ser eliminados. Este sería el escenario para el uso del método RobustScaler. El segundo escenario posible, consiste en entender estos valores atípicos como casos muy particulares y que no son el comportamiento habitual de la gestión operativa del negocio, por lo cual deben ser eliminados. Este sería el escenario para el uso de los métodos PowerTransformer, QuantileTransformer y MinMaxScaler

Teniendo en cuenta los posibles escenarios descritos frente a los valores atípicos, se usa la función limpiar_outliers descrita en la etapa de exploración de datos, calcula el rango intercuartil de los datos y a partir de este, realiza la eliminación de los valores atípicos. Esta función es utilizada únicamente para el segundo escenario descrito en el anterior párrafo.

Los datos originales, antes de ser normalizados, tuvieron que ser transformados de lista a matriz (arreglo de dos dimensiones) por medio de la función reshape disponible en la biblioteca numpy, puesto que todos los métodos de escalado anteriormente mencionados solo reciben matrices como parámetro de entrada. Posteriormente a esto, se crea la función normalizar_datos, en donde se encapsulan los métodos de transformación anteriormente expuestos.

Con la información retornada desde la función normalizar_datos, se procedió con la transformación de los datos por medio de la función crear_ventana. Esta función cuenta con la implementación del algoritmo ventana deslizante que busca controlar el flujo de datos que debe procesar el modelo predictivo y evitar su saturación, además que servirá para que el modelo sea capaz de notar y aprender de la variación de los datos. El rango de la ventana se estableció para saltos de 14 días, esto implicó que la matriz de datos de entrada se convirtiera en una matriz de sub-arreglos, en donde cada sub-arreglo se compone de 14 registros consecutivos, hasta que el ultimo arreglo tenga el ultimo registro de la matriz original.

Figura 11. Ventana deslizante de datos.



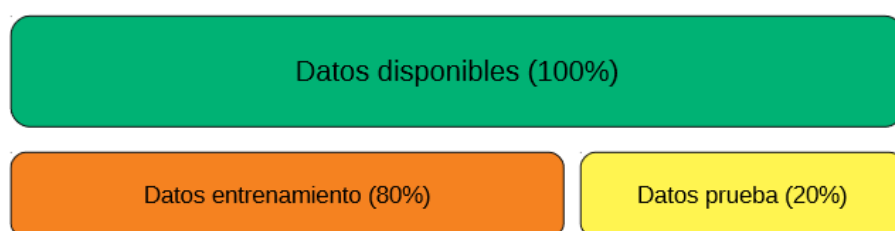
Fuente 21. Elaboración propia.

8.2 DIVISIÓN DEL LOTE DE DATOS

Con el objetivo de entrenar y validar el modelo predictivo a partir de los datos disponibles, se procede con la división del lote de datos de entrada en dos lotes de datos, uno de entrenamiento y otro de validación. Para esto, se establece un punto de corte en donde, del 100% de los datos disponibles, 80% serán utilizados para la etapa de entrenamiento del modelo predictivo y el 20% excedente serán utilizados para la verificación de la predicción realizada por el modelo previamente entrenado. Este punto de corte se envía junto al lote de datos como parámetro a la función `dividir_datos` el cual retorna las matrices de datos respectivas.

Puesto que el modelo predictivo requiere de lotes de datos en tres dimensiones (cubo de datos) para su correcto funcionamiento, las matrices de datos resultantes del proceso de división son transformadas en arreglos de tres dimensiones, una vez más por medio de la función `reshape` disponible en la biblioteca `numpy`. Esta acción puede ser realizada antes o después de la división de datos, indistintamente.

Figura 12. Segmentación de datos.



Fuente 22.Elaboración propia.

8.3 CONSTRUCCIÓN DEL MODELO

La arquitectura de red neuronal Long Short Term Memory (LSTM) seleccionada durante el proceso de análisis y comparación realizado al inicio de este proyecto, hace parte de las arquitecturas enfocadas en el aprendizaje profundo. Por lo que, para llevar a cabo una actividad que impliquen la revisión de datos históricos y secuenciales como lo es la predicción, un modelo debe estar compuesto de sucesivas capas de redes neuronales, a fin de que cada operación realizada por una capa sea transferida a la siguiente y esta a su vez, compare y evalúe esta

operación con una nueva entrada de datos y así de este modo reducir la diferencia entre la información prevista y la información real.

Partiendo del anterior principio, se opta por el uso de un modelo secuencial como base para la construcción del modelo predictivo, por lo que se crea la función `construir_modelo`, en donde se realiza la importación y uso de las capas `Sequential`, `Dense` y `LSTM` disponibles en la biblioteca `tensorflow-keras`, construcción y configuración del modelo predictivo. `Sequential`, es un modelo de pila para diferentes capas de redes neuronales, lo que permitirá encapsular diferentes capas una tras otra emulando una revisión a profundidad de los datos procesados. Para su correcto funcionamiento es necesario indicar en la primera capa que se adiciona, las dimensiones del set de datos que irá a procesar, con el propósito de que, a partir de esta, las dimensiones de las capas subsiguientes sean ajustadas a partir de dichas dimensiones.

La configuración inicial de un modelo para la predicción de series temporales, por lo general dependen de las dimensiones del lote de datos que será procesado en la etapa de entrenamiento. Además de que algunos parámetros principales deben ser seleccionados de forma arbitraria, tales como el número de capas ocultas, la tasa de abandono, el número de iteraciones del modelo, etc. Para la configuración del modelo se establecieron los siguientes parámetros iniciales de forma general.

Tabla 11. Configuración general del modelo predictivo.

Parámetro	Valor
N.º Capas ocultas	50
N.º Capas densas	1
Tasa de abandono	20%
Dimensiones de la ventana de tiempo	14x1
Iteraciones de entrenamiento	50 iteraciones

Fuente 23. Elaboración propia.

El número de capas ocultas hace referencia a la cantidad de neuronas conectadas de forma secuencial que se encargaran de procesar y analizar la información para aprender a partir de ella, el comportamiento de los datos y la estacionalidad de estos. El número de capas densas hace referencia a la cantidad de neuronas disponibles tanto en la entrada como en la salida de los datos. La tasa de abandono es el porcentaje de aceptación o rechazo de los nodos generados por la red neuronal, con el propósito de evitar el sobreajuste de la red (evitar que la red se aprenda los datos y no el comportamiento de estos). Ventana de tiempo y

dimensiones por ventana son el tamaño de cada secuencia de datos que la red debe revisar y entender antes de intentar realizar una predicción.

La anterior configuración expuesta, es enviada como parámetros de entrada a la función `construir_modelo`, encargada de la construcción, configuración y compilación del modelo predictivo. Dicha función cuenta también con la escala de las métricas del error absoluto medio (MAE) y del error cuadrático medio (MSE), necesarias para realizar la evaluación del modelo.

Figura 13. Función para construir el modelo predictivo.

```
7  # Configura el modelo de prediccion
8  def construir_modelo(input_time_step, input_dim, hidden_unit, dropout):
9      model = Sequential()
10     model.add(LSTM(1, input_shape=(input_time_step, input_dim), return_sequences=True))
11     model.add(Dropout(dropout))
12     #model.add(LSTM(hidden_unit, batch_input_shape=(1, input_time_step, input_dim), return_sequences=True, stateful=True))
13     model.add(LSTM(hidden_unit, return_sequences=False))
14     model.add(Dropout(dropout))
15     model.add(Dense(1))
16     model.add(Activation("tanh"))
17     #model.add(Activation("relu"))
18     model.compile(loss="mse", optimizer="rmsprop", metrics=['mae', 'mape'])
19     return model
```

Fuente 24. Elaboración propia.

8.4 ENTRENAMIENTO DEL MODELO

Con el propósito de evitar posibles casos de pérdida de datos o generación de ruido dentro de los pronósticos realizados por el modelo predictivo, se tuvo que considerar el tipo de escala en la cual se estaban normalizando los datos y el tipo de función de activación que se iría a utilizar para ajustar la salida generada por el modelo predictivo. Ya que una función de activación que no logre contener la escala de los datos normalizados incurrirá en un caso de pérdida de datos, al clasificar resultados correctos como irrelevantes, y del mismo modo en caso de que la función de activación supere la escala de los datos normalizado, existirá una alta posibilidad de clasificar resultados errados como correctos.

Teniendo en cuenta el escenario anteriormente expuesto y con el fin de seleccionar la combinación de parámetros que mejor se adapte a los datos disponibles y que de este modo retorne un modelo con métricas de error aceptables, se planearon y se ejecutaron cuatro escenarios de entrenamiento distintos, combinando los cuatro escaladores descritos en la etapa de transformación de atributos, con las funciones de activación tangencial hiperbólica (Tanh) y la unidad lineal rectificada (ReLU).

Tabla 12. Combinación de escaladores y activadores.

Escalador – Normalizador	Intervalo	Valores atípicos	Función de Activación
RobustScaler	-2,3	Si	Tanh
PowerTransformer	-1,2	No	Tanh
QuantileTransformer	0,1	No	ReLu
MinMaxScaler	0,1	No	ReLu

Fuente 25. Elaboración propia.

Debido a la naturaleza iterativa de este proceso, fue necesaria la modificación de la función de activación registrada en el método `construir_modelo` conforme el rango de escala de los datos cambiaba, de acuerdo con la relación expuesta en la tabla anterior. Para llevar a cabo el proceso de entrenamiento, se creó la función `entrenar_modelo`, donde se encapsulo el llamado de la herramienta TensorBoard, con el propósito de capturar el log de precisión del modelo durante el entrenamiento. Posteriormente se pasa como parámetros de entrada de esta función, los datos de entrenamiento y el número de iteraciones de entrenamiento, que para este caso consistió en 50 repeticiones de entrenamiento, a fin de que el modelo tuviese más oportunidades de aprender a partir de los datos sin recurrir al sobre-entrenamiento y por ende, el no aprendizaje.

9 VALIDACIÓN DEL MODELO PREDICTIVO

Esta etapa del proyecto consistió en la puntuación del modelo predictivo con el set de datos de prueba y a partir de esta puntuación, se realiza la evaluación de las métricas establecidas por el TDSP para un modelo de regresión.

9.1 PUNTUACIÓN DEL MODELO

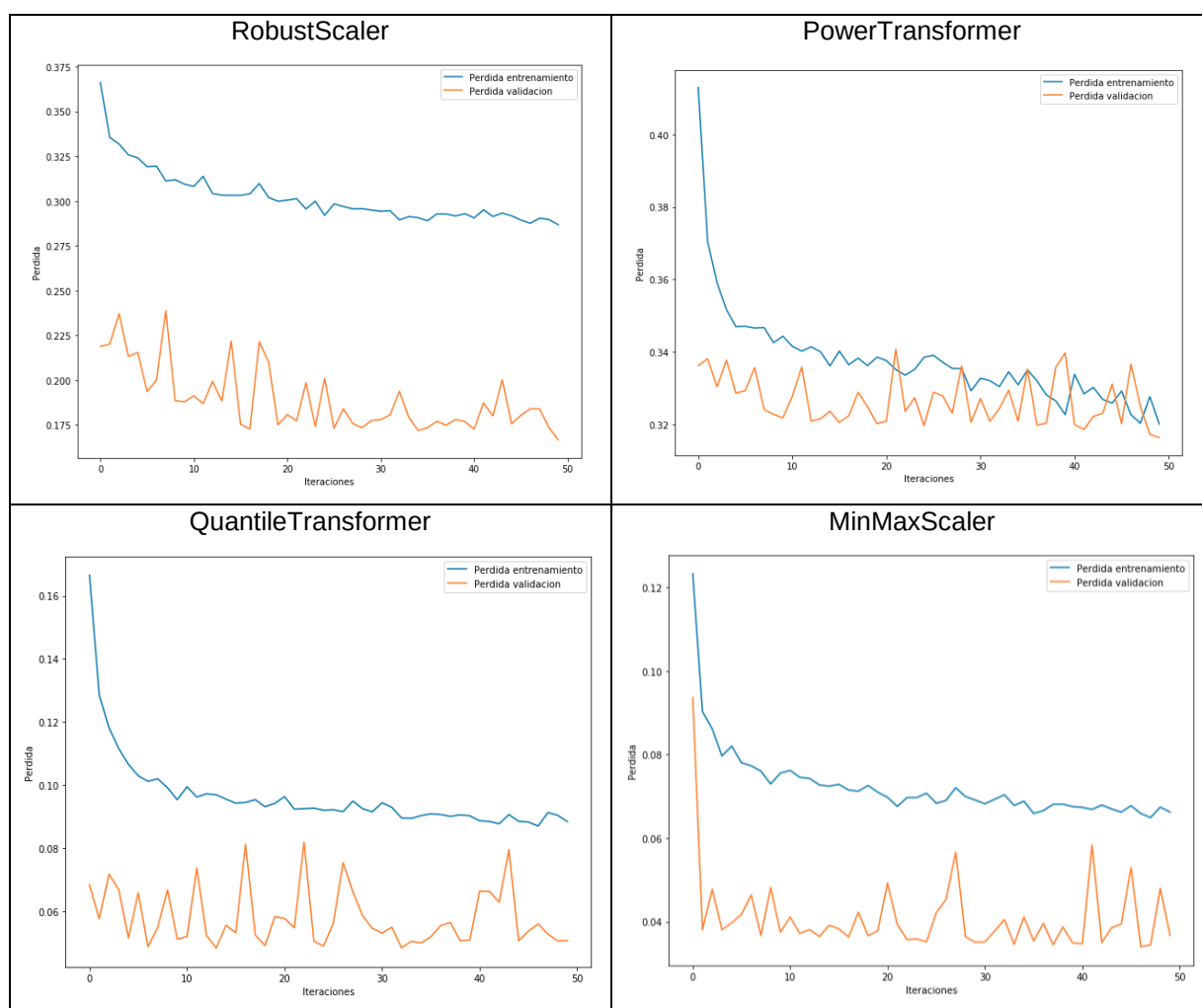
Para esta fase se utiliza el set de datos de prueba generado por la función `dividir_datos` y se procede con la ejecución de una sesión de predicción. La fase de puntuación ha sido ejecutada de forma iterativa con la función `predecir_datos`, utilizando las diferentes combinaciones entre escaladores de datos y funciones de activación que fueron propuestas en la fase de entrenamiento del modelo, con el objetivo de generar la información que será base para ejecutar el proceso de evaluación y elección del modelo predictivo que mejor represente el comportamiento de los datos ingresados.

9.2 EVALUACIÓN DEL MODELO

Para el desarrollo de esta etapa del proyecto, se comienza con la revisión y análisis de la diferencia entre la pérdida de datos realizada durante la etapa de entrenamiento y la etapa de validación, comparación de los histogramas sobre los errores de predicción y las series de tiempo pronosticadas por cada combinación de escalador y activador.

Adicionalmente, se construye la función `generar_metricas`, en donde se importa la biblioteca `sklearn.metrics` para el cálculo del coeficiente de determinación conocido como R^2 que determina el poder predictivo del modelo, el error absoluto medio (MAE), error cuadrático medio (MSE), raíz del error cuadrático medio (RMSE), son importados para evaluar la diferencia entre el valor predicho y el valor real. Su medida está dada en la misma escala en que los datos originales fueron medidos.

Tabla 13. Perdida de datos a través de las iteraciones de entrenamiento.



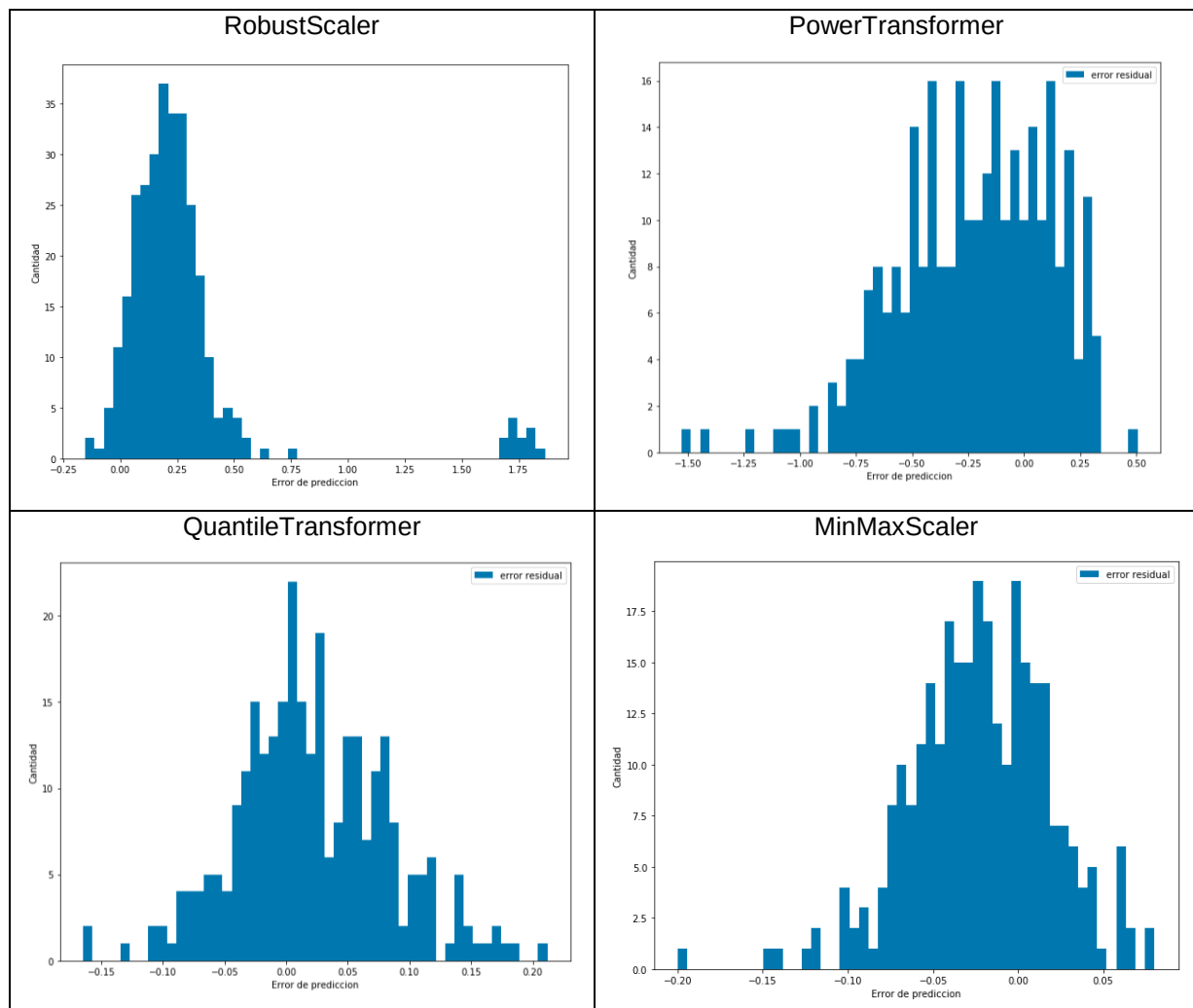
Fuente 26. Elaboración propia.

Con los datos obtenidos sobre la variación histórica del Error Absoluto Medio (MAE: la diferencia entre el valor real y valor pronosticado) durante la etapa de entrenamiento (Naranja) y en la etapa de puntuación (Azul), se elaboran las graficas para la visualización y comparación de la perdida de datos para estas dos etapas. Observándose que, si bien en todos los escaladores la perdida fue mayor en la etapa de entrenamiento, solo en el escalador PowerTransformer se intenta alcanzar una convergencia entre ambas etapas, con una variación absoluta entre 0.32 y 0.34 unidades. Aun así, el comportamiento obtenido con los demás escaladores sigue siendo algo deseable, considerando que la situación no se deba a un sobreajuste en el entrenamiento del modelo.

En los escaladores QuantileTransform y MinMaxScaler se observa que poseen un comportamiento similar en la perdida de datos, esto debido probablemente a que ambos escaladores comparten el mismo intervalo para la normalización de los datos originales. Esto podría indicar por extensión, que existe una alta posibilidad que la

precisión del proceso predictivo para los datos normalizados con estos escaladores sea muy similar entre ellos. Es necesario aclarar que las unidades de MAE están dadas en función de la escala utilizada para la normalización de los datos, por lo que las unidades de variación en la pérdida de datos obtenida con los diferentes escaladores no son comparables entre sí.

Tabla 14. Errores de predicción por escalador



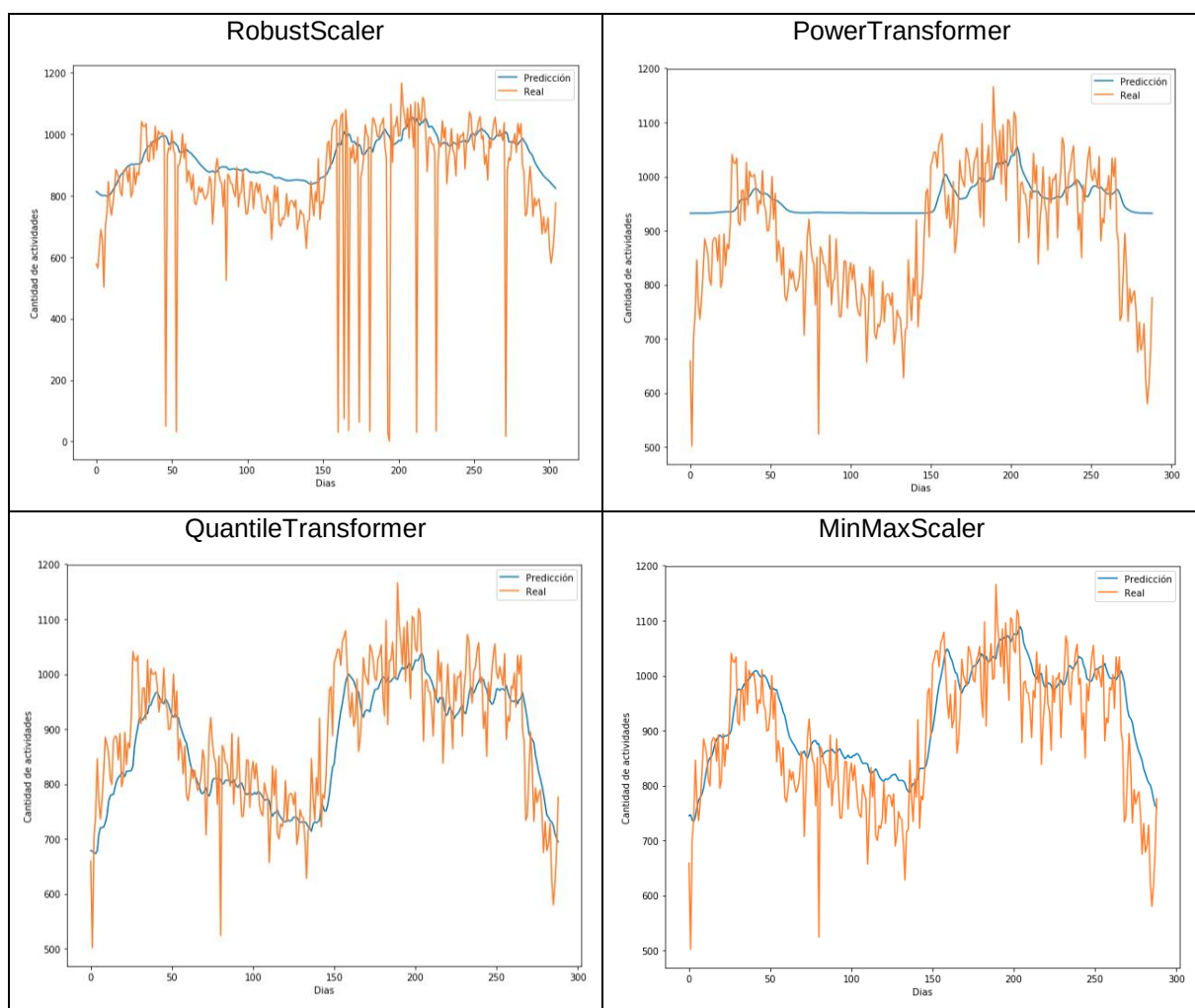
Fuente 27. Elaboración propia.

La tabla comparativa sobre los histogramas de errores en la predicción se generó a partir de la frecuencia en las variaciones entre el valor real y el valor pronosticado (también conocido como error residual). En estos se observa que el rango de variación del error en los escaladores RobustScaler y PowerTransformer, corresponde del $[-0.20, 1.90]$ y $[-1.55, 0.50]$ respectivamente. En ambos casos la cota máxima de error no es 0.00, generando desconfianza en la capacidad de predecir con mayor frecuencia los valores reales.

Adicionalmente se observa una alta irregularidad en las cotas superiores para la cantidad de errores de predicción con la escala PowerTransformer, lo que permitiría inferir que dicho escalador puede no ser útil para que la red sea capaz de aprender el comportamiento de los datos utilizados. El escalador QuantileTransformer en cambio, posee una concentración mayor entre los rangos $[-0.05$ y $0.05]$ logrando una cota superior en el error 0.00 , lo cual genera confianza en que, con mayor frecuencia, los datos pronosticados estarán más cerca con respecto a los datos reales.

Un comportamiento similar se presenta con el escalador MinMaxScaler, el cual conserva sus cotas superiores en el rango $[-0.05, 0.03]$, aunque en este caso existen dos cotas máximas, una en el error 0.00 y otra en el error -0.028 , revelando cierta dificultad en predecir con mayor frecuencia los valores reales y no valores aproximados a estos.

Tabla 15. Predicción de series de tiempo por escalador.



Fuente 28. Elaboración propia.

Con los resultados obtenidos a partir de la etapa de puntuación del modelo predictivo, se crean la función generar_métricas disponible en el documento modulo_evaluacion. Esta función que recibe el arreglo de datos reales y datos pronosticados como parámetros de entrada, calcula la variación general del modelo predictivo, precisión y poder predictivo del modelo. Obteniéndose los resultados expuestos en la siguiente tabla:

Tabla 16. Métricas para la evaluación del modelo predictivo.

Escalador	RMSE	MAE	MAPE	R2
RobustScaler	0.33058302	0.17857893	25,35%	0.08841786
PowerTransformer	0.40650088	0.31587399	25,62%	-0.06234443
QuantileTransformer	0.06513099	0.05002511	26,80%	0.61044871
MinMaxScaler	0.04605736	0.03596575	26,25%	0.55215896

Fuente 29. Elaboración propia.

A partir de la información anteriormente expuesta, se observa que a pesar de que la métrica del error porcentual absoluto medio (MAPE), encargada de evaluar la variación de las predicciones realizadas por cada escalador, genera resultados con poca diferencia entre los cuatro escaladores (entre el 25 y 26%), en la métrica MAE el escalador con peor desempeño fue el PowerTransformer llegando a variar una media de 0.31 unidades y los escaladores con mejor desempeño fueron el QuantileTransformer y MinMaxScaler con una variación media de 0.050 y 0.035 unidades respectivamente. Lo anteriormente expuesto corrobora el comportamiento observado en la predicción de la serie de tiempo para la cantidad de actividades gestionadas por Colvotel S.A.

Tomando como referencia el coeficiente de determinación (R2), se puede definir que la combinación entre escalador y función de activación que mejor se adapta a los datos, corresponde al escalador QuantileTransformer con la función ReLu. Debido puntualmente a que, a pesar de no tener el mejor resultado en las métricas de la variación media MAE y en la variación media porcentual MAPE, si posee el mejor poder predictivo para el modelo entrenado, al alcanzar una puntuación de 0.610. Esto implica que dicha combinación fue la mejor en ser capaz de aprender a generalizar el comportamiento de los datos, haciéndose apta para la elaboración de predicciones.

CONCLUSIONES

Del presente trabajo investigativo, se puede concluir que, para lograr una aplicación viable de un modelo de predicción se debe contar con datos que posean un comportamiento recurrente (estacionalidad) a fin de que el modelo aprenda a generalizar dicho comportamiento y mejore las posibilidades de una predicción con bajos márgenes de error. Así mismo, la construcción y configuración de un modelo predictivo necesita del análisis exploratorio de los datos que permitan confirmar la existencia de valores atípicos, evaluar la necesidad de eliminarlos en caso de que puedan generar ruido en el proceso de predicción o de conservarlos en caso de que estos correspondan a la mecánica operativa del negocio.

El modelo TDSP, al ser una metodología ágil para el desarrollo de soluciones inteligentes de reciente creación, posee un esquema general y ciclo de vida de fácil entendimiento, que se adaptó con naturalidad al desarrollo del presente proyecto, permitiendo que durante todas las etapas del proyecto fuese posible observar los resultados esperados y como estos se interrelacionaba con las actividades a desarrollar en siguientes etapas.

Poseer una medida aceptable en la variación media porcentual MAPE, no es garantía de que el modelo predictivo realmente haya aprendido el comportamiento de los datos, para esto es necesario revisar gráficamente el comportamiento de la serie de tiempo real vs pronosticada. Además de observar y analizar qué tan grande es la variación media absoluta MAE y la puntuación alcanzada en la métrica R2, que en ultimas es quien define el poder predictivo alcanzando por el modelo entrenado.

Como se evidencia en los resultados obtenidos durante la etapa de evaluación del modelo predictivo, a pesar de que las cuatro combinaciones de escaladores tenían resultados cercanos en la variación media porcentual MAPE, la variación media absoluta MAE y el coeficiente de determinación (R2) demostraron resultados totalmente contundentes, principalmente con los escaladores RobustScaler y PowerTransformer. Estos escaladores obtuvieron las dos mejores puntuaciones en la métrica MAPE. Sin embargo, tanto las gráficas de las series de tiempo como la puntuación obtenida en la métrica R2, revelaron que realmente el modelo predictivo entrenado con datos normalizados por estos dos escaladores, realmente no cumplen con el propósito de aprender a generalizar el comportamiento de los datos y realizar predicciones a partir de esta.

Por lo anteriormente expuesto, la selección quedo reducida a la combinación entre los escaladores QuantileTransformer, MinMaxScaler y la función de activación ReLu. En donde se observó que el escalador MinMaxScaler a pesar de contar con mejor puntuación en la métrica MAE y MAPE con respecto al escalador QuantileTransformer, este último logro una puntuación de 0.6104 frente a 0.552

alcanzados con el escalador MinMaxScaler en la métrica R2, permitiendo de este modo que la combinación entre el escalador QuantileTransformer y la función ReLu sea una combinación capaz de aprender el comportamiento de los datos y de elaborar predicciones con márgenes de error del 26,8% aproximadamente.

A partir del presente proyecto y de los resultados expuestos a la jefatura de tecnología y a la dirección de operaciones, actualmente se están adelantando opciones y propuestas para la puesta en marcha de un ambiente de desarrollo para análisis predictivo en tiempo real de la gestión de actividades en los proyectos de Colvaplus V2 y FTTH, que iría de la mano con la herramienta para el análisis de datos y generación de estadísticas Power BI.

Debido a que el desarrollo del presente proyecto, implicó la investigación de una tecnología de la cual, a nivel particular y personal, no poseía conocimiento ni información sobre su aplicación a nivel empresarial, permitió que se desarrollaran tres efectos. El primero consistió sobre el entendimiento alcanzado en temas de machine learning y Deep learning, y sus posibilidades para ser aplicado en un sinnúmero de soluciones software que sean capaces de facilitar las tareas de las personas u organizaciones. El segundo efecto consistió en el aprendizaje de la metodología TDSP para el desarrollo de soluciones analíticas e inteligentes que abre la posibilidad de dar un enfoque diferente a la carrera de ingeniera, más allá del desarrollo de software. El tercer y último efecto consistió en el afianzamiento de conocimientos básicos adquiridos en la academia, tales como la probabilidad, estadística, matemáticas especiales, análisis de algoritmos, etc. Y la integración de estos, con la experiencia adquirida en el sector empresarial.

BIBLIOGRAFIA

1. ColvateL S.A. E.S.P. Quienes Somos [En línea]. Bogotá D.C.: S.N., enero 2015 [Citado en 21 agosto de 2018]. Disponible en Internet: < <http://www.colvateL.com/somos.php> >.
2. ColvateL S.A. E.S.P. Naturaleza jurídica [En línea]. Bogotá D.C.: S.N., enero 2015 [Citado en 21 agosto de 2018]. Disponible en Internet: < <http://www.colvateL.com/juridica.php> >.
3. GONZALEZ AVELLA, J. C.; TUDURI, J. M. y RUL LAN, G.; Análisis de Series Temporales Usando Redes Neuronales Recurrentes. Palma de Mallorca: APSL, 2017. 6 p.
4. BIANCHI, Filippo María; MAIORINO, Enrico; KAMFFMEYER, Michael C.; RIZZI, Antonello; JENSSEN, Robert. An overview and comparative analysis of Recurrent Neural Networks for Short Term Load Forecasting. Norway: Arxiv, 2017. 41 p.
5. WALKER LYSFJORD, MAGNUS J. Modeling and Forecasting the Nord Pool Day-Ahead Power Market through Deep-Learning. Research project submitted for the degree of Masters of Renewable Energy. Norway: University of Agder, 2017. 48 p.
6. LAPTEV, Nikolay; YOSINSKI, Jason; ERRAN, Li y SMYL, Slawek. Time-series extreme event forecasting with neural networks at Uber. San Francisco, California: Semantic Scholar - The Allen Institute for Artificial Intelligence, 2017. 4 p.
7. ZHU, Lingxue y LAPTEV, Nikolay. Deep and confident prediction for time series at Uber. San Francisco, California: Arxiv, 2017. 8 p.
8. Series de tiempo con Python [En línea]. Buenos Aires (Argentina): Raúl E. López Briega, 26 septiembre 2016 [Citado en 22 agosto 2018]. Disponible en

Internet:<<https://relopezbriega.github.io/blog/2016/09/26/series-de-tiempo-con-python>>.

9. BENCARDINO MARTÍNEZ, Ciro. Estadística y muestreo. 13 ed. Bogotá D.C.: ECOE ediciones Ltda., 2012. ISBN:978-958-648-702-3. p. 794-816.

10. CHOLLET, Francois. Deep learning with python. New York: Manning publications Co., 2017. p. 4-5. ISBN 9781617294433.

11. Differences Between AI, Machine Learning and Deep Learning [En línea]. New York (USA): S.N., 29 noviembre 2017 [Citado en 9 septiembre 2018]. Disponible en Internet:<<https://t2conline.com/differences-between-ai-machine-learning-and-deep-learning>>.

12. GOLLAPUDI, Sunila. Practical Machine Learning. Birmingham - Mumbai: Packt Publishing Ltd., 2016. p. 34. ISBN 9781784399689.

13. Introducción al Deep Learning [En línea]. Buenos Aires (Argentina): Raúl E. López Briega, 13 junio 2017 [Citado en 22 agosto 2018]. Disponible en Internet:<<https://relopezbriega.github.io/blog/2017/06/13/introduccion-al-deep-learning>>.

14. BONNIN, Rodolfo. Building Machine Learning Projects with TensorFlow. Engaging projects that will teach you how complex data can be exploited to gain the most insight. Birmingham - Mumbai.: Packt Publishing, 2016. p. 112. ISBN 9781786466587.

15. TELEMAGEMENT FORUM. Business Process Framework (eTOM) - Concepts and principles. GB921CP. Version 14.5.2. Morristown, New Jersey, United States: TM Forum, 2015. p. 22.

16. WANG, Jie; WANG, Jun; FANG, Wen y NIU, Hongli. Financial Time Series Prediction Using Elman Recurrent Random Neural Networks. Beijing, China: Hindawi Publishing Corporation, 2016. p. 2.

17. ICANN: INTERNATIONAL CONFERENCE ON ARTIFICIAL NEURAL NETWORKS, 17th (9-13 september, 2007, Porto, Portugal.). Comparison of Echo State Networks with Simple Recurrent Networks and Variable-Length Markov Models on Symbolic Sequences. Berlin: Springer Berlin.

18. What is an intuitive explanation of Echo State Networks? [En línea]. Salt Lake City, UT, USA: Adam Conkey, 21 Julio 2015 [Citado en 18 septiembre 2018]. Disponible en internet: <<https://stats.stackexchange.com/questions/140652/what-is-an-intuitive-explana>>.

19. GRAVES, Alex. Supervised Sequence Labelling with Recurrent Neural Networks. Berlin, Springer-Verlag GmbH, 2012. p. 32. ISBN 9783642247972.

20. CHUNG, Junyoung; GULCEHRE, Caglar; CHO, KyungHyun y BENGIO, Yoshua. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. Montreal, Canada: Arxiv, 2014. p. 4.

21. Recurrent Neural Network Tutorial, Part 4 – Implementing a GRU/LSTM RNN with Python and Theano [En línea]. San Francisco (Estados Unidos): Denny Britz, 27 octubre 2015 [Citado en 18 septiembre 2018]. Disponible en Internet:<<http://www.wildml.com/2015/10/r>>.

22. Team Data Science Process Documentation [En línea]. Redmond, WA, USA: Microsoft - Debraj GuhaThakurta, 20 octubre 2017 [Citado en 17 octubre 2018]. Disponible en internet: <https://docs.microsoft.com/en-us/azure/machine-learning/team-data-science-process/>.

23. GARCÍA, Salvador; RAMÍREZ GALLEG0, Sergio; LUENGO, Julián y HERRERA, Francisco. Big Data: Preprocesamiento y calidad de datos. En: Novática. Julio-octubre, 2016, no. 237, p.17-23.

24. Team Data Science Process Documentation - Data acquisition and understanding [En línea]. Redmond, WA, USA: Microsoft - Debraj GuhaThakurta, 20 octubre 2017 [Citado en 17 octubre 2018]. Disponible en internet: <<https://docs.microsoft.com/en-us/azure/machi>.

25. Análisis Confirmatorio vs Exploratorio [En línea]. Austin, TX, USA: Daniel Legorreta, 12 marzo 2015 [Citado en 17 octubre 2018]. Disponible en internet: <<https://dlegorreta.wordpress.com/2015/03/12/analisis-confirmatorio-vs-exploratorio/>>.

26. Team Data Science Process Documentation - Modeling [En línea]. Redmond, WA, USA: Microsoft - Debraj GuhaThakurta, 20 octubre 2017 [Citado en 17 octubre 2018]. Disponible en internet: < <https://docs.microsoft.com/en-us/azure/machine-learning/team-data-sci>.

27. Evaluate Model [En línea]. Redmond, WA, USA: Microsoft - Roope Astala, 10 octubre 2018 [Citado en 17 octubre 2018]. Disponible en internet: < <https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/evaluate-model> >.

28. Score Model [En línea]. Redmond, WA, USA: Microsoft - Roope Astala, 24 enero 2018 [Citado en 17 octubre 2018]. Disponible en internet: < <https://docs.microsoft.com/en-us/azure/machine-learning/studio-module-reference/score-model> >.

29. Team Data Science Process Documentation - Deployment [En línea]. Redmond, WA, USA: Microsoft - Debraj GuhaThakurta, 04 noviembre 2017 [Citado en 17 octubre 2018]. Disponible en internet: < <https://docs.microsoft.com/en-us/azure/machine-learning/team-data>.

30. Team Data Science Process Documentation - Customer acceptance [En línea]. Redmond, WA, USA: Microsoft - Debraj GuhaThakurta, 04 noviembre 2017 [Citado en 17 octubre 2018]. Disponible en internet: < <https://docs.microsoft.com/en-us/azure/machine-learning/>.