**Métodos de Machine Learning para pronosticar demanda de productos de retail**

**Autor(es)**

**Leonardo Martin Estremadoyro Céspedes**

20160527@aloe.ulima.edu.pe

Universidad de Lima

**Abstract:** El principal problema de toda empresa dedicada a la comercialización de productos es la incertidumbre de cuál será la demanda futura. Un mal pronóstico conlleva a dos principales problemas: Sobreabastecimiento, cuando se tienen más productos en stock que la demanda real, y el desabastecimiento, cuando hay el stock no es suficiente. Estos fenómenos son el resultado de un pobre modelo predictivo implementado en la organización, sea por no contar con recursos necesarios o, peor aún, no reconocerlo como un factor de éxito en una organización. En este artículo, se propondrán diferentes técnicas conocidas de Machine Learning que permitan solucionar esta problemática, como son: Redes Neuronales Artificales (ANN), Support Vector Machine (SVM) y Gradient Boosting Trees (GBT). Para cada técnica, se realizará su respectivo análisis teórico y se explicará la metodología utilizada para poder implementarla. Además, cada técnica será implementada utilizando Algoritmos Genéticos, una técnica de ML que permite obtener los mejores hiperparámetros para cada modelo, utilizando como referencia la teoría de evolución natural de Darwin. Los resultados serán comparados entre ellos mediante diferentes métricas como MSE, RMSE, MAE, MAPE y tiempo de ejecución que permitirá determinar el mejor modelo. Los resultados muestran que SVM ha logrado obtener resultados significativamente superiores a los demás modelos, a pesar de no ser un modelo originalmente pensado para resolver problemas de regresión. El enfoque propuesto será evaluado utilizando un dataset de ventas históricas de una tienda de retail.

**Palabras Clave:** Machine learning, regresión, pronóstico de demanda, algoritmos genéticos, funciones de costo, árboles de decisión, support vector machines, boosting, redes neuronales.

# INTRODUCCIÓN

En la actualidad, distintas organizaciones de diferentes rubros manejan grandes cantidades de inventario de productos terminados, de productos perecibles o no perecibles, y que reflejan una importante cantidad de dinero invertido por parte de la empresa. Sin embargo, a pesar de representar una fuerte suma de dinero, compañías tienden a no destinar la cantidad de recursos necesarios para elaborar un modelo predictivo adecuado a las diferentes variables que se toman en cuenta según la industria, o, peor aún, no reconocerlo como un elemento clave para el éxito empresarial (Mark A. Moon, John T. Mentzer, Carlo D. Smith, and Michael S. Garver, 1998). Mientras más grande sea la industria, y la empresa, mayor será el riesgo de no poder satisfacer la demanda o, en contra parte, contar con inventario superior a tu stock de seguridad que genera gastos de mantenimiento.

Estudios realizados por Harvard Business School a más de 600 puntos de venta en 29 países obtuvieron que un 72% de los problemas por desabastecimiento eran culpa de pobres prácticas de pedidos y reabastecimiento en la organización, dentro de estos problemas se encuentra la incorrecta predicción de demanda. Otro estudio realizado, por la misma entidad, a más de 71000 consumidores alrededor del mundo, indica que, como se aprecia en la Figura 1, alrededor del 31% de clientes dejarán el establecimiento para buscar el mismo producto, pero en la competencia. Este porcentaje puede variar según el producto, como se exhibe en la Figura 2, como son los cosméticos, donde el 43% de consumidores buscarán el mismo producto, pero en un establecimiento diferente. Esas ventas perdidas se traducen en 4% de pérdidas para un típico comerciante. Pero para una organización de un billón de dólares esto significa 40 millones de dólares de pérdida. (Corsten, D., & Gruen, T., 2004).

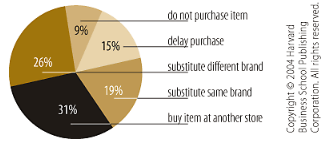


Figura 1. Qué hacen los clientes cuando no encuentran el producto que están buscando. Recuperado de: <https://hbr.org/2004/05/stock-outs-cause-walkouts>

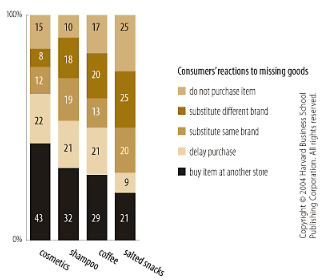


Figura 2. Qué hacen los clientes cuando no encuentran cosméticos, champú, café o bocadillos salados. Recuperado de: <https://hbr.org/2004/05/stock-outs-cause-walkouts>

Por otro lado, el sobre abastecimiento de productos también es un problema grave para las utilidades de un negocio. Esto genera costos de almacenamiento, mantenimiento o reducción de margen de ganancias, que es la respuesta más común por parte de comerciantes para deshacerse de su inventario. Glenn Taylor, billonario hombre de negocios americano, realizó una investigación en el 2015 sobre los 5 factores que más influyen en el exceso de inventario. Los resultados fueron que el factor principal es el fallo en la predicción de demanda que resulta en pérdidas globales de 172.2 billones de dólares en dicho año, seguido por el deterioro de productos perecibles que representa 80.5 billones en pérdidas, problemas de proveedores en 42.5 billones, clima con 39.8 billones y finalmente errores de marketing en pérdidas de 38.4 billones. Por una amplia diferencia, diseñar un modelo adecuado de predicción es, sin duda, el mayor desafío que los comerciantes afrontan hoy en día, especialmente con la proliferación de canales digitales en la última década. (Glenn Taylor, 2015).

Carbonneau realizó una investigación acerca de diferentes técnicas para pronosticar la demanda: Naive Forecast, Average, Moving, Average, Trend, Multiple Linear Regresion, Neural Networks, Recurrent Neural Networks y Support Vector Machines. Dentro de su investigación, categorizó las primeras cinco técnicas mencionadas como tradicionales, y las últimas tres: Avanzadas. Además, indicó que las avanzadas superan a las tradicionales, esto es debido a que incorporan métodos no lineales. Un método no lineal está basado en un comportamiento complejo. Esto hace referencia a la arbitrariedad en la relación de diferentes variables, la relación no es en “línea recta”, puede variar con el tiempo, como es el caso de la demanda (Carbonneau, R., Laframboise, K., & Vahidov, R., 2008).

Esta investigación, propondrá soluciones basadas en métodos de Machine Learning para poder pronosticar la demanda con técnicas avanzadas, como fueron mencionadas anteriormente. Para poder alcanzar este objetivo, se deberá realizará un análisis exhaustivo sobre las tres técnicas de ML que se utilizarán para realizar el pronóstico: Redes Neuronales Artificiales (ANN), Support Vector Machine (SVM) y Gradient Boosting Trees (GBT). Estos modelos serán entrenados utilizando backpropagation junto con stochastic gradient descent. Estos tres métodos serán implementados independientemente utilizando el mismo dataset. Dicho dataset, será pre procesado utilizando la librería Pandas de Python para eliminar datos faltantes, estandarizar los tipos de datos y eliminar los mayores outliers. La selección de los hiperparámetros, para todas las técnicas, será utilizando algoritmos genéticos, los cuales nos permitirán obtener la arquitectura óptima de hiperparámetros. Después del cálculo de todas las técnicas propuestas, estas serán comparadas entre cada una de ellas (ANN, SVM, y GBT) para determinar la que tuvo mayor precisión.

El presente artículo está compuesto de la siguiente forma: El primer punto es el presente, Introducción, el cual ha planteado la problemática que se busca resolver, además de proponer técnicas avanzadas como mejores soluciones sobre las tradicionales; En el segundo puto, Estado del Arte, consiste en una extensa bibliografía sobre trabajos similares que involucran las técnicas de Machine Learning mencionadas y Algoritmos Genéticos; El tercer punto, Antecedentes, describirá exhaustivamente todas las bases teóricas necesarias para poder construir una solución en base a los conocimientos obtenidos; El cuarto punto, Metodología, se explicará qué es lo que se va a realizar para solucionar la problemática, este punto se realizará con un alto nivel de detalle que permita su completa replicación; El quinto punto, Experimentación, se enfocará en demostrar cómo es que se busca llevar a cabo la solución con un alto nivel de detalle; El sexto punto, Resultados, exhibirá los resultados más relevantes obtenidos de la experimentación; El penúltimo punto, Discusión de Resultados, se enfocará en poder justificar los resultados obtenidos mediante la comparación con otras investigaciones; Finalmente, el último punto, Conclusiones, determinará si efectivamente se logró alcanzar el objetivo principal de la investigación, además de lo que se logró obtener.

# ESTADO DEL ARTE

## PRONÓSTICO DE DEMANDA APLICADO EN UN SUPERMERCADO

En el año 2015, se realizó una investigación en un supermercado en Marruecos con el objetivo de implementar un modelo predictivo innovador, para dicho país, basado en redes neuronales artificiales que serviría como herramienta para realizar el pronóstico de la demanda en dicho establecimiento. El pronóstico tuvo un horizonte semanal. (Slimani, I., El Farissi, I., & Achchab, S., 2015).

Para realizar la aplicación del modelo, los investigadores propusieron cuatro pasos:

1. Aprendizaje, en esta etapa se define la metodología de aprendizaje, se determinó la proporción de datos de entrenamiento, validación y prueba. Los datos de entrenamiento serán ingresados en el modelo para comparar los outputs obtenidos con los esperados en periodos determinados. Para poder realizarlo, se calibró la red para determinar el valor apropiado de las conexiones, de forma que el valor de salida pueda ser lo más cercano al output esperado.
2. Extracción de datos, esta etapa consiste en definir los datos que conformarán el conjunto de entrenamiento, validación y prueba. Para esta investigación utilizaron, de la base de datos de un Supermercado de Marruecos, datos históricos semanales de ventas y backorders. Para ello, extrajeron las ventas semanales de un producto por un periodo de seis meses.
3. Elección de la arquitectura adecuada, se definió la capa de entrada y salida. La primera, estuvo compuesta por tres neuronas, la capa de salida por solo una. En la Tabla 1 se muestra un ejemplo de los datos, donde se alimenta el modelo con ventas de tres semanas consecutivas, una semana por neurona, para obtener la venta de la cuarta semana.

|  |  |  |  |
| --- | --- | --- | --- |
| **Input 1** | **Input 2** | **Input 3** | **Output** |
| Quantity of Tuesday,January 1 | Quantity of Tuesday,January 8 | Quantity of Tuesday,January 15 | Quantity of Tuesday,January 22 |
| Quantity of Wednesday, January 2 | Quantity of Wednesday, January 9 | Quantity of Wednesday, January 16 | Quantity of Wednesday, January 23 |
| … | … | … | … |

Tabla 1. Ejemplo de datos de entrada y salida - Slimani, I., El Farissi, I., & Achchab, S. (2015).

1. Pre procesamiento de entradas y salidas, esta etapa final consistió en traducir los valores numéricos de entrada a un rango de -1 a 1 y los valores de salida entre 0 a 1 mediante la regla de tres. Esto es debido que a una red neuronal no se debe alimentar con cualquier tipo de datos, porque los outputs de las funciones de activación estarían muy fuera de su rango definido. Es por ello que los datos deben estar dentro del mismo intervalo. En la Tabla 2, se muestran los datos crudos, antes de ser procesados. Y en la Tabla 3 están los mismos datos, pero escalados entre un rango de -1 a 1.

|  |  |  |  |
| --- | --- | --- | --- |
| **Input 1** | **Input 2** | **Input 3** | **Output** |
| 2208 | 2088 | 2976 | 2352 |
| 2832 | 1968 | 2616 | 2400 |
| 3648 | 3984 | 4656 | 4152 |

Tabla 2. Datos de entrada y salida antes de ser procesados. - Slimani, I., El Farissi, I., & Achchab, S. (2015).

|  |  |  |  |
| --- | --- | --- | --- |
| **Input 1** | **Input 2** | **Input 3** | **Output** |
| -0,74359 | -0,82906 | -0,19658 | 0,179487 |
| -0,29915 | -0,91453 | -0,45299 | 0,196581 |
| 0,282051 | 0,521368 | 1 | 0,820513 |

Tabla 3. Datos de entrada y salida antes después de ser procesados. - Slimani, I., El Farissi, I., & Achchab, S. (2015).

Para la experimentación, los autores utilizaron Neuroph Tool, librería de machine learning de Java, para la programación de las redes neuronales. Los autores indican que no existe una regla definida para determinar las capas oculta que tenga una red, tampoco las neuronas dentro de cada una. Es por ello que se realizaron pruebas con diferentes arquitecturas. Cada una de estas pasó por el mismo proceso de entrenamiento y puesta a prueba. Ejecutaron el entrenamiento con 77 registros, y las pruebas con 60. Se escogieron tres arquitecturas finales, ilustradas en la Tabla 4, su elección se basó en la diferencia entre los datos obtenidos y los resultados esperados, determinados por el valor absoluto del algoritmo Mean Square Error (MSE).

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Capa Entrada** | **Capa Oculta** | **Capa Salida** |
| **ANN1** | 3 | 3,2 | 1 |
| **ANN2** | 3 | 4,4,4 | 1 |
| **ANN3** | 3 | 4,2,10,5 | 1 |

Tabla 4. Arquitectyras de red finalistas - Slimani, I., El Farissi, I., & Achchab, S. (2015).

Como puede observar, las tres arquitecturas escogidas varían en número de neuronas y capas ocultas, esto es debido a que se busca comparar la red más simple, menor número de capas ocultas, con la más compleja o profunda, mayor número de capas. Finalmente, se realizó una comparación final entre las tres arquitecturas finalistas (ANN1, ANN2 y ANN3), también utilizando MSE. En la Figura 3 se muestran los resultados para los datos de entrenamiento, validación y prueba.

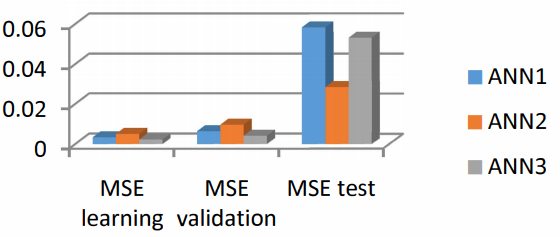


Figura 3. Resultado de comparación entre tres arquitecturas de red neuronal basado en el resultado de MSE - Slimani, I., El Farissi, I., & Achchab, S. (2015).

Después de la experimentación, los investigadores concluyeron que la arquitectura ANN2 es la que mayor reduce la función de costo, MSE. Por lo tanto, es la más precisa y la mejor para poder pronosticar la demanda del supermercado.

## FECHAS ESPECIALES EN EL PRONÓSTICO DE DEMANDA

Uno de los principales desafíos del pronóstico de demanda es poder pronosticar durante días especiales, o series de tiempo que son influenciadas externamente por ciertos días o épocas de calendario. La definición de días especiales (DEs) por los investigadores incluye feriados públicos, días vecinos al feriado, también aquellas fechas que no sean feriados no laborales, sino más bien festividades. En primera instancia, la mayoría de feriados públicos caen en días laborales (Días de semana), y su demanda de productos es similar a los domingos. También, es importante considerar días posteriores y previos al DE, puesto que usualmente las personas suelen tomarse un día libre adicional en el trabajo (Huber, J., & Stuckenschmidt, H., 2020).

Los DEs fueron clasificados en 4 tipos como se muestra en la Tabla 4.

|  |  |
| --- | --- |
| **DE1**: Día especial | Día del feriado o festividad |
| **DE2**: Día anterior | Día anterior al DE1, si el DE1 fuera lunes entonces el DE2 será sábado. |
| **DE3**: Día siguiente | Día siguiente al DE1. |
| **DE4**: Semana siguiente | Una semana después al DE1, sirve para verificar que la demanda haya seguido con normalidad 7 días después del feriado o festividad. |

Tabla 4. Listado de días especiales (DEs) con su respectiva descripción. - Huber, J., & Stuckenschmidt, H. (2020).

Los autores, realizaron su investigación sobre una panadería situada en Alemania, la cual cuenta con 141 tiendas distribuidas alrededor de todo el país. Cada tienda, tiene un catálogo de 8 productos, esto significa que en total existen 1128 series de tiempo. El dataset fue dividido en entrenamiento, validación y prueba. El conjunto de entrenamiento y validación comprende todos los registros desde octubre del 2014 a enero del 2017. El periodo de prueba, está compuesto por 150 días del 1 de febrero del 2017 al 30 de junio del mismo año. Este rango de fechas fue seleccionado por comprender una cantidad considerable de DEs (39) en Baden-Württemberg, Alemania. Esto incluyó 15 días de DE1, 7 días de tipo DE2, 5 días de tipo DE3 y 14 días de tipo D4 (Huber, J., & Stuckenschmidt, H., 2020).

La implementación fue realizada utilizando dos técnicas conocidas de Machine Learning (Redes Neuronales, y Random Forest) que luego fueron comparadas entre ellas. El 20% del dataset de entrenamiento fue utilizado para validación. Debido a que únicamente se cuenta con el dataset para poder determinar los hiperparámetros de los modelos, se aplicó Cross-Validation sobre el conjunto de entrenamiento para obtener el modelo más preciso que se pueda tener durante esta etapa para luego probarlo con los datos de prueba.

## ARQUITECTURA DE RED Y TÉCNICAS DE APRENDIZAJE

En el año 2011 se publicó el artículo Expert Systems with Applications, donde dos investigadores realizaron estudios acerca del por qué utilizar la inteligencia artificial a pesar que actualmente existen modelos estadísticos tradicionales, como son la regresión lineal o suavisamento exponencial, que son ampliamente conocidos por medianas o grandes organizaciones, que emplean recursos y esfuerzos en la formulación de un efectivo modelo predictivo. (Yu, Y., Choi, T., & Hui, C., 2011).

Dichos modelos estadísticos han sido utilizados masivamente para predecir acontecimientos futuros por décadas. Estos modelos han probado ser efectivos, pero solo bajo prerrequisitos que son el conocimiento del usuario para escoger el modelo y parámetros adecuados. Son conocidos por su velocidad para procesar robustos conjuntos de datos (datasets). Sin embargo, su precisión depende altamente del conocimiento del experto para escoger el modelo y los parámetros. Esto se traduce en una limitación para muchas aplicaciones en la vida real, cuando las personas encargadas de tomar una decisión no cuentan con el conocimiento necesario. Es debido a lo mencionado que las organizaciones empezaron a utilizar Artificial Neural Networks (ANN), porque no requieren que el usuario sea un experto en la industria. En su investigación, los autores también realizaron una comparación entre dos modelos de inteligencia artificial, redes neuronales artificiales tradicionales (Multicapa) y Single-hidden Layer Feedforward Networks (SLFN). Este último se caracteriza por solo contar con 3 capas, capa de entrada, una oculta, y una de salida. Contar con la mínima cantidad de capas posibles simplifica el modelo, incrementa la velocidad del proceso de entrenamiento y la ejecución del mismo. Además, al ser una red Feedforward, el flujo es únicamente en una dirección, sin bucles y sin retroceder a nodos de capas predecesoras. (Yu, Y., Choi, T., & Hui, C., 2011).

Sin embargo, este modelo de red neuronal utiliza Extreme Learning Machine (ELM) como técnica de aprendizaje. Esta técnica es hasta mil veces más rápida que backpropagation, como mencionan los autores. Desafortunadamente, ELM no es perfecto debido a investigaciones señalan salidas inestables del modelo o elevadas diferencias de resultados cuando se aplica MSE para entrenar el modelo. Esto resulta en tener que ejecutar el modelo y procesar la información múltiples veces a comparación de algoritmos convencionales como backpropagation que incrementa los tiempos de ejecución.

Para la implementación, se utilizó un dataset de productos de moda de una compañía en Hong Kong y otro de una tienda de moda online. En dicha investigación se buscó encontrar una relación entre el color, talla, y precio de los productos con la cantidad de ventas y evaluar los tiempos de procesamiento obtenidos con las dos técnicas de aprendizaje mencionadas (ELM y backpropagation), para determinar si ELM es significativamente mejor.

Después de la experimentación se llegó a la conclusión que si bien ELM cuenta con menor tiempo de procesamiento aun así existen las falencias mencionadas, aunque no sostienen que el entrenamiento deba tomar horas o días, se debe buscar un punto medio sin una gran cantidad de capas escondidas, pero tampoco con solo una como son con las SLFN para evitar obtener resultados no alentadores al aplicar MSE (Yu, Y., Choi, T., & Hui, C., 2011).

## ENTRENAMIENTO UTILIZANDO ALGORITMOS GENÉTICOS

Las metodologías de entrenamiento comúnmente utilizadas para generar aprendizaje en una red neuronal son backpropagation junto con stochastic gradient descent, estos algoritmos trabajan alrededor de una gradiente y una serie derivadas sucesivas para mejorar el modelo con el pasar de las épocas de entrenamiento. Sin embargo, la carga computacional puede llegar a ser muy alta cuando se consideran arquitecturas de red profundas (50, 80 o hasta 100 neuronas por capa.). Es por ello que se ha iniciado una tendencia por emplear algoritmos que no necesiten computar una gradiente para poder entrenar un modelo (Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J., 2016).

En Barcelona, España, se realizó una investigación e implementación de un modelo que permita pronosticar la demanda en la red subterránea de tráfico de agua utilizando machine learning. Los investigadores plantearon como solución la implementación de un modelo predictivo utilizando ANN. Sin embargo, su modelo no emplearía una metodología de entrenamiento tradicional, puesto que utilizarían Algoritmos Genéticos (AG). El motivo es que, como indican los autores, entrenar una red neuronal es difícil, tedioso y puede llegar a tomar muchas iteraciones hasta obtener una arquitectura que permita pronosticar con un nivel aceptable de precisión. Inclusive, indican que algoritmos basados en el cálculo de una gradiente no garantizan poder obtener la solución global óptima, puesto que usualmente los problemas de optimización son no convexos (Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J., 2016).

Para implementar su solución, utilizaron una variación de la combinación de las arquitecturas Multi Input Multi Output (MIMO), esta arquitectura está estructura por múltiples inputs y múltiples salidas, y Multi Input-Single Output (MISO), arquitectura con múltiples inputs, pero una sola salida. Esta propuesta está compuesta por 24 modelos, cada modelo será independiente y realizará el pronóstico de consumo de agua de 1 hora en específico. Los datos de entrada para cada modelo serán el histórico de demanda de agua y una variable de comportamiento (Mode). Esta variable tiene la función de reflejar el comportamiento humano con respecto al consumo de agua en el tiempo, esto es calculado mediante la utilización de la técnica k-means para agrupar y segmentar el comportamiento de la demanda medida por 4 válvulas de presión diferentes. Los clústeres obtenidos reflejaran un patrón de demanda que será asignado junto con la demanda de una hora específica a los datos de entrenamiento. En la Figura 4, se muestra la arquitectura de la solución propuesta, donde representa el consumo de agua en la hora (Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J., 2016).

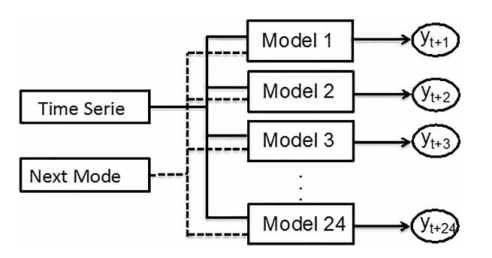


Figura 4. Propuesta de solución compuesta por 24 modelos - Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J. (2016)

Las arquitecturas de cada modelo estaban definidas en 3 capas, pero para determinar las neuronas por cada capa se utilizaron algoritmos genéticos de la siguiente forma: Se generó una población de 250 vectores (También conocidos como cromosomas), cada vector estaba compuesto por una serie de valores entre 0 y 1. Como se muestra en la Figura 5, los dos primeros valores del vector son la cantidad de entradas que tendría el modelo y el número de neuronas en la capa escondida respectivamente. Estos valores permiten definir 2 sets de datos; el primero, determina la cantidad de conexiones, con sus pesos determinados, entre las neuronas de entrada y las de la capa oculta; y el segundo set las conexiones de la capa oculta con la neurona de la capa de salida (Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J., 2016).

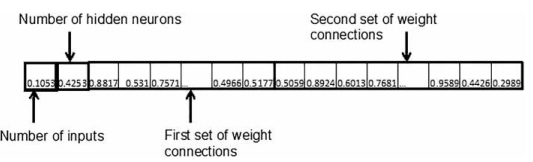


Figura 5. Vector que representa los hiperparámetros de un cromosoma - Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J. (2016)

Estos vectores pasarán por múltiples generaciones a las cuales se efectuarán diferentes operaciones genéticas como selección, mutación, recombinación, etc. Durante cada iteración, determinada por el tamaño de la población, se obtienen un conjunto de padres los cuales tienen la posibilidad de mutar y conseguir estar dentro del criterio de convergencia determinado para poder determinar la mejora arquitectura.

La investigación también tuvo el objetivo de comparar los resultados con otras técnicas de pronóstico también basadas en Machine Learning y otras en modelos estadísticos tradicionales. En la Figura 6, se muestran los resultados de la puesta a prueba en las 4 válvulas (a, b, c y d) de forma graficada.

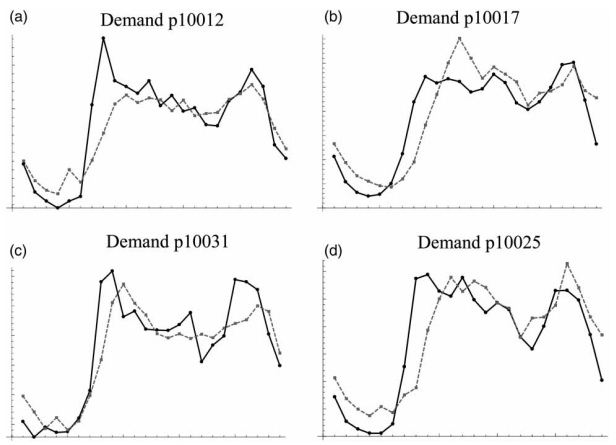


Figura 6. Gráfico de resultados esperados (Línea sólida oscura), y resultados obtenidos (Línea punteada) - Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J. (2016)

Un detalle interesante que rescatar es el comportamiento, en la Figura 6, de la demanda en 24 horas, como inicia muy bajo y luego llega a picos muy altos que pueden llegar a variar.

En la Tabla 5, se muestran los resultados del modelo empleado, Banco de Modelos (BM), para la válvula 1 (p10012) que representa el gráfico (a) de la Figura 6.

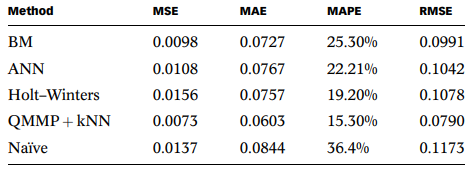


Tabla 5. Comparación de resultados en válvula 1 con otras técnicas predictivas - Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J. (2016).

Como se puede observar el BM tiene uno de los errores porcentuales más altos de las 5 técnicas que se comparan, sin embargo, en la Tabla 5 se observa un comportamiento diferente, con mayor precisión de resultados y una menor función de costo.

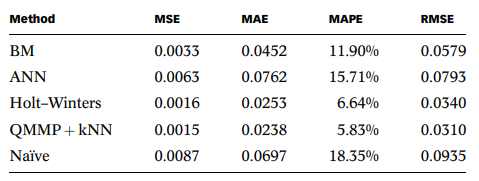


Tabla 6. Comparación de resultados con otras técnicas predictivas - Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J. (2016).

En la Tabla 6, el resultado del error porcentual (MAPE), de la válvula 4 (Gráfico b en la Figura 6) es significativamente menor que su resultado en la válvula 1 (Gráfico a en la Figura 6). Esto se puede deber a que los valores picos de la válvula p10017 son menores a de las demás válvulas, es por ello que la función de costo es menor en todos los modelos. Modelos como Holt-Winters y QMMP + kNN demuestran tener un bajo nivel de error frente a estos escenarios donde los valores máximos no son tan excesivos. **El entrenamiento basado en algoritmos genéticos será implementado en el siguiente curso, Seminario de Investigación II.**

## PRONÓSTICO DE VENTAS CON PROMOCIONES EMPLEANDO SVM

Por muchos años, métodos estadísticos tradicionales se han empleado para pronosticar, como son ARIMA y Suavizamiento Exponencial (SE). Sin embargo, estos modelos pueden fallar o no arrojar resultados precisos si existe una alta irregularidad de ventas, o datos en general, como sucede con las promociones, o alta variabilidad de datos durante días en particular de la semana. Esto es debido a que los modelos tradicionales no están apropiadamente diseñados para modelar el comportamiento no lineal del proceso de ventas. Es por ello que se han empezado a utilizar técnicas de Machine Learning para poder afrontar problemas no lineales (Di Pillo, G., Latorre, V., Lucidi, S., & Procacci, E., 2016).

En este caso de estudio se realizó el pronóstico de ventas diarias de la marca de pasta más popular de dos tiendas de retail de la misma cadena. Se utilizaron registros del año 2007 al 2010 divididos en 13 intervalos de tiempo, por año, para el entrenamiento. El objetivo de dividir los datos en intervalos es poder generar 13 SVM por cada año. El periodo de prueba será el año 2011. También, se estructuraron 2 diferentes modelos de SVM por cada intervalo. Estos modelos variaban en la cantidad y tipo de input de datos. El primer modelo, SVM4i, está conformado por: promoción, horas de apertura al día, precio del producto y número de boletas a emitir en un día (Target, lo que se busca pronosticar). El segundo modelo, SVM12i, comprende los mismos 3 inputs de SVM4i más 9 datos de calendario. Estos datos de calendario están divididos en 7 columnas booleanas que representan el día de ese registro, 1 columna de número de mes (0-11) y 1 columna de día del mes (0-29, dependiendo del mes). El objetivo de tener dos modelos es identificar qué tanto afecta el dato de día de la semana dentro de un modelo de pronóstico de corto alcance, puesto que se busca pronosticar las ventas de un día en específico. Como función de costo se utilizó MSE, para las métricas MSE y Mean Aboslute Perecentaje Error (MAPE). En la Figura 7 se muestra el resultado de la puesta a prueba del modelo para la tienda 1 utilizando el modelo SVM12i, donde el eje de las abscisas son los 4 primeros meses transcurridos del año 2011, este año es el conjunto de datos de prueba, y el eje ordenado el valor de MSE (Di Pillo, G., Latorre, V., Lucidi, S., & Procacci, E., 2016).

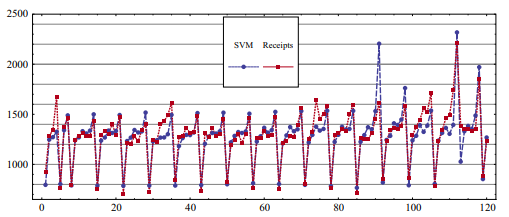


Figura 7. Resultados del pronóstico del modelo SVM y los datos reales. - Di Pillo, G., Latorre, V., Lucidi, S., & Procacci, E. (2016)

Ese modelo obtuvo un MAPE, porcentaje de error, de 4.5% para la tienda 1, y 5.4% para la tienda 2. Estos resultados son alentadores, puesto que es la primera prueba realizada con alguno de los dos modelos de SVM planteados. En la Figura 8, se muestra una comparación de ambos modelos, SE, ARIMA y HWES, por cada uno de los 13 intervalos del año 2011.

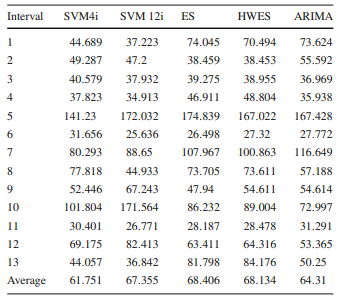


Figura 8. Valores de MSE para cada modelo de SVM y métodos estadísticos - Di Pillo, G., Latorre, V., Lucidi, S., & Procacci, E. (2016)

Como se puede observar en la Figura 5, los resultados de MSE de ambos modelos de SVM consistentemente superan a los métodos estadísticos ES, HEWS y ARIMA. El modelo SVM4i obtiene mejores resultados en 5 de los 13 intervalos y SVM12i en 4 de los 13. Esto significa que los modelos tradicionales tan solo superan en 4 escenarios de los 13 totales a los modelos de Machine Learning. Además, SVM4i prueba ser más preciso que SVM12i a pesar de contar con una menor cantidad de variables de entrada. Pueda que para este último modelo se esté experimentando overfitting. Entonces, se puede concluir que efectivamente los SVM es efectivo para poder pronosticar demanda y que cualquier organización dedicada a la comercialización de productos puede tomar ventaja de esta tecnología (Di Pillo, G., Latorre, V., Lucidi, S., & Procacci, E., 2016).

# ANTECEDENTES

## SUPPORT VECTOR MACHINE (SVM)

Support Vector Machines, o Máquinas de Vectores Soporte en español, fueron introducidas inicialmente en el 1979, pero su primera documentación formal data de 1995. Es un método de aprendizaje relativamente nuevo que antiguamente era solo utilizado para resolver problemas de clasificación. Pero que, actualmente, también ha sido extendido a resolver problemas de regresión, donde la salida es de valor numérico. Su metodología consiste en encontrar un hiperplano que separe los datos de d-dimensiones en dos clases. Sin embargo, no todos los datos son linealmente separables, es por ello que SVM introduce el concepto de Kernel o Núcleo, este pasa los datos a un espacio donde la dimensión es mayor, compuesta por características donde los datos sí son separables (Boswell, D., 2002). Pero, tampoco se debe incurrir en prácticas más complejas que incrementan dimensiones, puesto que no todos los modelos necesitan un alto nivel de complejidad para poder ser efectivos, más bien, esto puede generar resultados adversos. Una característica particular de SVM, a diferencia de RNA, es que permiten calcular una Dimesión-VC, este parámetro permite medir la probabilidad de que un modelo tenga un buen desempeño con datos no conocidos, como son los datos de prueba. SVM busca maximizar el margen de separación de datos, cuando es un problema de clasificación, para que la diferenciación sea apropiada (Garrido, W. L., 2012). Esto se puede observar claramente en la Figura 9.

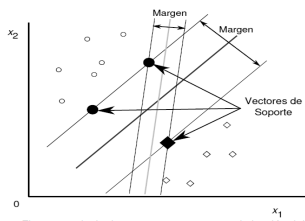


Figura 9. Representación del margen de diferenciación de datos - Garrido, W. L. (2012)

Pero hay escenarios donde los casos no son linealmente separables, es entonces cuando se tienen que introducir un espacio de características específicas junto con un Kernel para poder realizar la diferenciación apropiada como se muestra en la Figura 10.

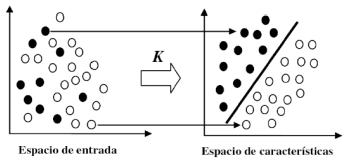


Figura 10. Espacio de entrada y de características - Garrido, W. L. (2012)

Sin embargo, cuando se busca utilizar SVM para pronosticar un valor, problema de regresión, también se utiliza un espacio de características, pero no se busca maximizar un margen, sino, emplear regresión lineal en dicho espacio para poder aproximar valores (Samsudin, R., Shabri, A., & Saad, P., 2010).

Los SVM, tanto en su modalidad de clasificación como regresión, cuentan con un alto nivel de personalización para poder adaptarlo a una gran cantidad de escenarios, esto es debido a su extensa cantidad de hiperparámetros que se pueden modificar. En SVM existen tres hiperparámetros que tienen una alta influencia en el modelo y sus resultados, estos son conocidos como hiperparámetros de diseño: C, E, G. El primero, es el encargado de controlar y regular los errores de entrenamiento con la complejidad del modelo, si *C* es un valor muy grande el modelo tendrá altas penalidades sobre por puntos no separables en el hiperplano y almacenará muchos vectores de soporte que podrán generar overfitting, por otro lado, si *C* es un valor muy pequeño el modelo manifestará underfitting; El segundo, conocido como “epsilon”, involucra lo ancho de la zona en el plano que permite entrenar el modelo, mientras mayor sea *E*, menor serán la cantidad de vectores de soporte serán seleccionados, esta misma relación es aplicada cuando el valor *E* sea muy pequeño; Finalmente, *G*, o “gamma”, controla el coeficiente del kernel, lo que se traduce en la habilidad de generalización de la regresión SVM, este último solo aplica para kernels de tipo RBF (Radial Basis Function), Poly (Polynomial) y Sigmoid (Sigmoideal) (Tsirikoglou, P., Abraham, S., Contino, F., Lacor, C., & Ghorbaniasl, G., 2017).

## REDES NEURONALES

Una red neuronal artificial (RNA) es la traducción del comportamiento cerebral, de aprendizaje humano, en una computadora, esta red está compuesta por capas que contienen neuronas conectadas con otras, de capas subsiguientes o predecesoras, que tienen la habilidad de aprender, calcular o en este caso pronosticar. En términos generales, una RNA es capaz de solucionar problemas con ayuda de data histórica, aprendizaje y una estructura de red definida. Las redes neuronales pueden tener múltiples componentes dependiendo del tipo de red, pero todas ellas comparten conceptos básicos como son los siguientes: Capa de entrada, capas ocultas, capa de salida, conexiones, con sus respectivos pesos, funciones de activación y reglas de aprendizaje. Estos componentes mencionados serán explicados con mayor detalle en los párrafos siguientes (Slimani, I., El Farissi, I., & Achchab, S., 2015).

Mishra y Gupta en su artículo Deep Machine Learning and Neural Networks: An Overview, grafican las redes neuronales. Las neuronas son representadas por nodos, y su conexión entre capas está diagramada con flechas como se muestra en la Figura 11. Estas conexiones tienen diferentes pesos, que son multiplicados por el valor de la neurona para pasar como input de la siguiente capa escondida.

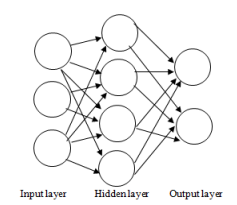


Figura 11. Arquitectura de una ANN – Mishra, C., & Gupta, D. (2017)

Los pesos son coeficientes que determinan la intensidad de la señal de entrada a una neurona desde otra capa, estos pesos serán modificados durante la etapa de entrenamiento de una red.

## ARQUITECTURA DE RED NEURONAL

Una arquitectura básica de una red neuronal está compuesta por una capa de entrada, una oculta, y una de salida. Los autores, recalcan el hecho que no existe un parámetro fijo para la cantidad de capas ocultas, así que se pueden utilizar las que se consideren convenientes a costa de volver el modelo más complejo según amerite el problema (Slimani, I., El Farissi, I., & Achchab, S., 2015).

Como mencionado, existen diferentes arquitecturas, de redes neuronales, siendo las Redes Neuronales Prealimentadas (Feedforward Neural Network) las más comunes por su simplicidad. La principal característica de una FNN es que no existen bucles, en otras palabras, el flujo es únicamente en una dirección, a la siguiente capa oculta o capa de salida. Dentro de la categoría FNN las arquitecturas más comunes son los Perceptrones de Una Capa (SLP), conformados por solo 2 capas, una de entrada y salida; y el Perceptrón Multicapa (MLP), que a diferencia del anterior este posee una o más capas ocultas. En la Figura 12, se muestra el flujo de un MLP aplicado al pronóstico de que un alumno apruebe o repruebe una asignatura según las horas de estudio (Hinkelmann, K., 2016).

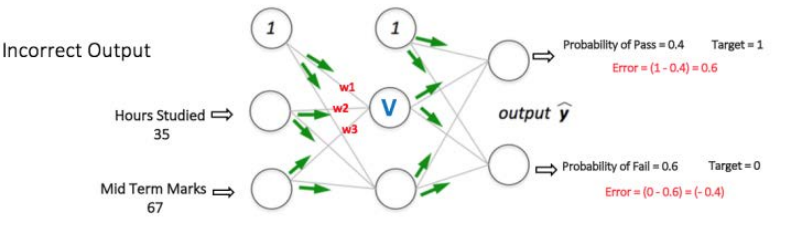


Figura 12. Flujo de un Perceptrón Multicapa aplicado al resultado final de un alumno en una asignatura según sus horas de estudio. - Hinkelmann, K. (2016).

## APRENDIZAJE Y CÁLCULO DEL ERROR

Mishra, C., y Gupta, D dividen los tipos de aprendizaje en dos, supervisado y no supervisado. En el aprendizaje supervisado existen dos técnicas: Regresión y clasificación; mientras que el tipo no supervisado se utilizan técnicas de clusterización. Las técnicas mencionadas, como se muestra en la Figura 13, implementan diferentes tecnologías para cada una de ellas. Para regresión: Redes neuronales, árbol de decisión, Ensembles Methods, Lineales y No-lineales. Clasificación: Support Vector Machine (SVM), Análisis discriminativo y Naive Bays, Vecino más cercano. Clustering: K-Means, Jerárquico, Red neuronal, Gaussian Mixtures y Hidden Markov Model. Para la presente investigación se utilizará aprendizaje supervisado debido a que es un problema de regresión (Mishra, C., & Gupta, D., 2017).

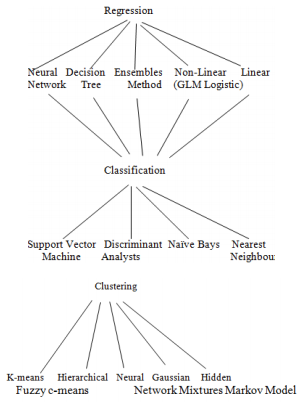


Figura 13. División de tecnologías que se utilizan según las técnicas de aprendizaje – Mishra, C., & Gupta, D. (2017)

La principal característica de las redes neuronales es la capacidad de aprender, esto se da durante el proceso de entrenamiento, donde la red neuronal es alimentada con data histórica y corrida múltiples veces. Esta fase consiste en poner a correr el modelo con una configuración por defecto y comparar sus salidas con las salidas esperadas, que vienen a ser las salidas reales ocurridas en periodos anteriores. Existen diferentes algoritmos para medir la diferencia entre los resultados, estos algoritmos también son conocidos como funciones de costo, como son Mean Square Error (MSE), Mean Absolute Error (MAE), Root Mean Square Error (RMSE), entre otros. Esta diferencia permite determinar cuántas corridas son necesarias para estar dentro del rango de aceptación previamente determinado por el usuario. Para la experimentación de este trabajo, se utilizará el algoritmo MSE al ser el utilizado por Slimani, El Farissi y Achchab en su trabajo previamente exhibido. (Slimani, I., El Farissi, I., & Achchab, S., 2015).

Fórmula 1. Mean Square Error (MSE) – Slimani, I., El Farissi, I., & Achchab, S., (2015).

En la Fórmula 1 se detalla la fórmula MSE, el error está representado por la diferencia entre , e , valor obtenido y valor esperado, respectivamente, y n es la cantidad de corridas realizadas.

## ALGORITMOS GENÉTICOS

## Los algoritmos genéticos son una técnica de optimización inspirada por la teoría de evolución de Darwin. Estos algoritmos replican una versión más simple de la evolución biológica, la cual consiste en crear una población de individuos, donde cada uno representa una posible solución del problema que se busca resolver, estos se reproducen y crean descendientes mejores que los predecesores. El mejor individuo, la mejor solución, es aquella que tenga el mejor valor de función de costo. Esta técnica modifica la población utilizando tres operadores genéticos como son: Selección, crossover y mutación, etc.

## El operador de Selección consiste en escoger una estrategia de selección de individuos, los seleccionados serán comparados entre sí basándose en su peso o función de costo. Las estrategias más utilizadas son Roulette Wheel Selection y Tournament Selection. La primera, consiste en calcular los costos de cada uno de los individuos de la población, para luego seleccionar de forma preferente a aquellos con un menor costo y compararlos; Tournament, por otro lado, tiene un criterio de selección completamente aleatorio, sin considerar el costo de los individuos hasta el momento de tener a los 2 o 3 seleccionados y realizar la comparación. Los dos mejores individuos resultantes de la comparación serán utilizados como padres para la operación de Crossover, pero solo el mejor de ellos dos será categorizado como gen predominante.

## La operación de Crossover consiste en la reproducción de los dos individuos seleccionados, se escogen 2 características, de forma aleatoria, del Cromosoma A para ser pasadas a su sucesor (Hijo), manteniendo su posición en el cromosoma hijo, luego el Cromosoma B pasa una característica al sucesor y así de forma iterativa hasta completar con todas las características del nuevo cromosoma. Cabe recalcar que la posición de cada hiperparámetro se mantiene, para evitar que los hijos tengan valores duplicados del mismo hiperparámetro. Este proceso debe realizarse hasta tener la misma cantidad de población original.

La última operación, Mutación, consiste en la probabilidad de los cromosomas hijos de manifestar mutación, esta mutación consiste en seleccionar un hiperparámetro aleatorio de uno de los hijos y multiplicar su valor por un factor aleatorio, este factor debe estar en el mismo rango que se utilizó para crear los hiperparámetros de forma aleatoria en la Etapa 1.

## Estas operaciones son realizadas de forma iterativa (Generaciones) hasta alcanzar un threshold de la función de costo o iterar una cantidad de generaciones predeterminada, esto es con el objetivo que no entre en un bucle sin fin. Es importante mencionar que las operaciones de Crossover y Mutación son susceptibles a una probabilidad de ocurrencia pre definida en cada modelo, por ello es que pueden existir generaciones solo con Selección. Al final del proceso, el resultado es la mejor solución (Individuo) que se pudo obtener en base a la población creada y las operaciones realizadas, cabe recalcar que siempre existe un nivel de aleatoriedad que influye en el resultado (Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J., 2016).

## FUNCIÓN DE ACTIVACIÓN

Las funciones de activación son el determinante de la salida de una neurona, estas definen la línea base (Threshold) necesaria para activar una neurona. Existen diferentes tipos de funciones como son: Función Uni-Polar Sigmoidea, Bipolar Sigmoidea, Tangente Hiperbólica, Sección Cónica, Relu, entre otras. Para este trabajo, se utilizarán las funciones Tangente Hiperbólica, Relu y Sigmoidea las cuales pasarán por un modelo de optimización de hiperparámetros que determinará la mejor función para cada capa de la red. En primera instancia, la función Tangente Hiperbólica es la mayormente utilizada en modelos no lineales, como es el pronóstico de demanda, además de ser un problema de regresión y no clasificación, donde funciones como Relu tienen un mejor resultado como indican Eckle, K., & Schmidt-Hieber, J. (2018). Esta función, consiste en comparar el resultado de aplicar tanh(x), la descomposición de la función se indica en la Fórmula 2, al input del flujo entrante, donde x es la entrada. Este resultado es comparado entre un rango de aceptación definido, como se muestra en la Figura 14 (Karlik B., Olgac V. A., 2011).

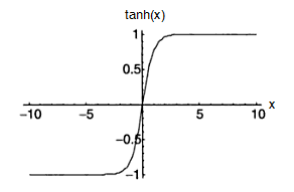


Figura 14. Gráfico de la Función Tangente Hiperbólica - Karlik B., Olgac V. A., (2011)

Fórmula 2. Función Tangente Hiperbólica - Karlik B., Olgac V. A., (2011)

## PESOS DE LAS CONEXIONES

Los pesos de las conexiones son factores que, junto con el input de la data, ingresan a la función de activación, explicada en el punto 3.6, que determina la salida de la neurona. Estos pesos, son constantemente modificados durante la fase de aprendizaje, porque permiten alterar la activación de neuronas según sea requerido basándose en la función de costo, en este caso MSE. En la Fórmula 3, se indican las operaciones necesarias para determinar el output de una neurona, según la arquitectura presentada en la Figura 15, y se incluye el rol de los pesos de las conexiones.

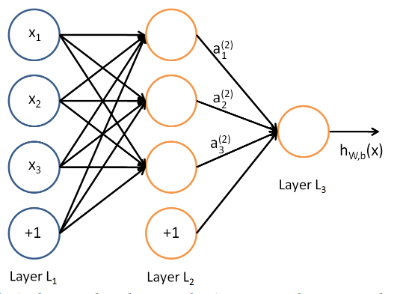


Figura 15. Red neuronal con arquitectura de Perceptrón Multicapa - Stanford University. Multi Layer Neural Networks

Esta red neurona está compuesta por 3 capas, entrada, una oculta y capa de salida, la primera capa está compuesta por tres neuronas denotadas por y +1, este último es conocido como bias, denotado por , donde el índice superior representa la capa y el inferior el número de neurona de la segunda capa de esta arquitectura, . La tarea del bias es ser una constante que permita incrementar o disminuir los valores como input que recibe una función de activación, de este modo se asegura que esté dentro de los límites, superior e inferior, de la función. La segunda capa () está conformada por , (2) hace referencia a y 1,2,3 al número de neurona.

Fórmula 3. Operaciones de los pesos de las conexiones para ingresar la función de activación. - Stanford University. Multi Layer Neural Networks

Como se mencionó, en la Fórmula 3 se muestran las operaciones realizadas con los pesos de las conexiones entre neuronas de las capas de la red. Donde representa el peso de la conexión entre en la capa , con en la capa , notar la posición de los índices (Stanford University. Multi Layer Neural Networks).

## ALGORITMO DE APRENDIZAJE

Para poder aprender, las redes neuronales son capaces de utilizar diferentes técnicas de aprendizaje como es backpropagation junto con stocahstic gradient descent para disminuir la función de costo mediante la modificación de los pesos de las conexiones de neuronas entre capas. El objetivo de backpropagation es computar una gradiente, esta es calculada utilizando los pesos de las conexiones entre neuronas, obtenidas siguiendo un flujo contrario, de derecha a izquierda, he de allí el nombre backpropagation. Pero esta técnica no es suficiente para poder generar aprendizaje, puesto que solo calcula la gradiente. Stochastic gradient descent, tiene el trabajo más complicado, porque tiene que utilizar la gradiente computada para generar aprendizaje mediante una fórmula de pérdida (Loss Function). Para lograrlo se debe realizar un forward propagation del error de cada capa de la red, se calculan derivadas sucesivas que permitan calcular un valor numérico que modifique el peso de conexión entre neuronas, haciendo que los valores que ingresan a la función de activación sean más cercanos para obtener en output esperado (Goodfellow, I., Bengio, Y., & Courville, A., 2016).

## GRADIENT BOOSTED TREES

Los Árboles de Decisión (DT) implementan una función específica en el espacio de características de d-dimensiones. Los DT están compuestos por nodos los cuales están asociados con una función booleana que evalúa la comparación de un vector de característica con un threshold determinado. Cada nodo tiene ligado por lo menos output, el cual es determinado según el resultado de la función booleana. Dependiendo si la evaluación resulta en 0 o 1 se toma la rama izquierda o derecha, este proceso se repite hasta alcanzar un nodo hoja (Nodo final), el cual representa la salida del árbol de decisión. La profundidad de un árbol de decisión está determinada por la longitud del camino más largo desde el nodo raíz (Nodo inicial) al nodo hoja (Wu, D. J., Feng, T., Naehrig, M., & Lauter, K., 2016).

Gradient Boosted Trees (GBT) es un método muy popular de Boosting junto con Adaptive Boosting (AdaBoost). El concepto de Boosting, originalmente llamado Hypothesis Boosting, hace referencia a cualquier método que pueda combinar una gran cantidad de predictores débiles, en uno confiable con mayor nivel de precisión. El predictor más común utilizado en Boosting son los DT. Similarmente con SVM este método originalmente solo estaba enfocado a resolver problemas de clasificación, pero que eventualmente se extendió a problemas de regresión. El objetivo de la mayoría de métodos de Boosting es entrenar predictores de forma secuencial, uno después del otro, donde cada uno busca corregir los errores del predictor predecesor. Esto se logra mediante la modificación de pesos al pasar por cada modelo predictor. A cada modelo se le computa un peso, el cual puede variar desde valores negativos a positivos. Mientras más preciso es el predictor, mayor será su peso, de forma contraria cuando falta de precisión. (Krauss, C., Do, X. A., & Huck, N., 2017). GBT, en vez de mejorar cada predictor en base a la modificación de pesos por cada iteración como lo hace AdaBoost, este método intenta mejorar el predictor basándose en los errores residuales generados por el predictor predecesor. En la Figura 16 se muestra el comportamiento de los errores residuales, columna izquierda, junto el predictor mejorado, columna derecha (Géron, A., 2019).

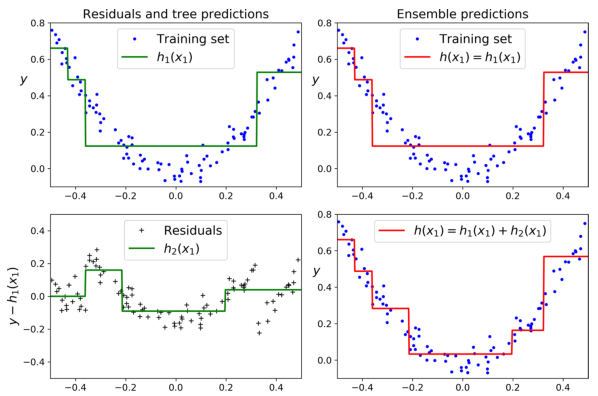


Figura 16. Errores residuales y predictor mejorado - Géron, A. (2019).

Como se puede visualizar en la Figura 16, las gráficas de la primera fila, error residual y el predictor, son iguales, esto se manifiesta durante la primera iteración, puesto que aún no se ha calculado el primer output de error para pasarlo a un predictor sucesor.

Esta técnica cuenta con una amplia cantidad de hiperparámetros para poder modificarlos como input al modelo, de esta forma poder encontrar los mejores valores para un modelo en específico, una lista de los hiperparámetros que se utilizarán en esta investigación son los que se muestran en la Tabla 7.

|  |  |
| --- | --- |
| **Hiperparámetro** | **Descripción** |
| learning\_rate | Ratio de aprendizaje del modelo por iteración. |
| n\_estimators | Número de estimadores que serán utilizados consecutivamente para mejorar el modelo. |
| max\_depth | Profundidad máxima del árbol de boosting, mientras mayor sea el valor mayor será la probabilidad de experimentar overfitting en el modelo. |
| min\_child\_weight | Costo mínimo del valor de un modelo hijo para poder existir y peticionarse. |
| subsample | Porcentaje de datos que se utilizarán para entrenar por cada iteración de los modelos. |
| colsample\_bytree | Numero de columnas que serán utilizadas para entrenar por cada iteración de los modelos |
| gamma | Threshold del costo por modelo que se debe alcanzar para generar un modelo hijo (Un nuevo y mejorado estimador) |

Tabla 7. Listado de hiperparámetros que ingresarán al modelo de algoritmos genéticos - Xie, Y., Zhu, C., Lu, Y., & Zhu, Z. (2019).

# METODOLOGÍA

Para poder lograr pronosticar la demanda de un artículo en particular o categoría de productos, primero se debe definir cuál será el horizonte de predicción. En el estado del arte se exhibieron trabajos que tenían un horizonte semanal, diario y hasta por hora. En el caso de una tienda de retail, para esta investigación, lo que se busca es poder pronosticar la demanda diaria de un producto, puesto que el dataset que se utilizará cuenta con histórico de ventas diarias, agruparlo de forma semanal o mensual reduciría drásticamente la cantidad de datos históricos que el modelo utilizará para ser entrenado. Sin embargo, utilizar únicamente el histórico de ventas para pronosticar la demanda en tiendas sería una solución sesgada, puesto que existen otras variables de entorno relevantes que permiten explicar la demanda. Es por lo mencionado que la propuesta de solución empleará Días Especiales (DEs), estos días representan fechas festivas, laborales o no laborales, que, según lo revisado en la sección 2.2 del Estado del Arte, tienen un impacto en la demanda. Se tomarán en cuenta tanto la fecha del DE, días aledaños y la fecha una semana después al DE que servirá como verificación. También se aplicará el concepto de Ingeniería de Características, el cual consiste en crear variables en base a la data actual para que ayuden a explicar la demanda.

Lo que se va a realizar es una comparación de 3 técnicas de ML (ANN, SVM y GBT), cada una de las técnicas contará con su respectiva metodología de entrenamiento y, especialmente, la ANN utilizará algoritmos genéticos para definir la arquitectura de red óptima con sus hiperparámetros. Posteriormente, cada una de estas técnicas serán comparadas por su función de costo (MAPE) para poder determinar qué técnica tuvo mayor precisión.

## MODELAMIENTO DE DIAS ESPECIALES

Como se elaboró en la sección de estado del arte, las investigaciones de Huber, J., & Stuckenschmidt, H. (2020) y Di Pillo, G., Latorre, V., Lucidi, S., & Procacci, E. (2016) indican la importancia y criticidad de tomar en cuenta tanto las fechas como los días de la semana en que se realiza una transacción, además de contar con fechas donde la demanda será alterada por tratarse de una festividad o día especial. Los Días Especiales (DE), son aquellas fechas que se consideran como festividades laborales o no laborales. Estas fechas tienen alto nivel de relevancia al momento de realizar el pronóstico de productos de retail, debido a que el público, de un establecimiento de venta, puede ser niños, jóvenes o adultos, mas no empresas. Este público mencionado, es susceptible a fechas, puesto que su nivel de consumo, para ciertos productos o industrias, se incrementa o disminuye según un día en específico. Para afrontar esta incertidumbre, es necesario extraer todos los DE del contexto demográfico de la ciudad o país donde se encuentre el establecimiento sobre el cual se aplicará la solución. Sin embargo, no solo se utilizará únicamente la fecha del DE, sino también la de los días aledaños (Previos y posteriores al DE) y la fecha del día una semana después (Ejemplo: Si el DE es lunes 08 de julio, el siguiente DE será lunes 15 de julio). Los 4 tipos de DE que se han determinado, y su respectiva descripción, se encuentran en la Tabla 8.

|  |  |
| --- | --- |
| **Día Especial (DE)** | **Descripción** |
| **DE1**: Día especial | Día del feriado o festividad |
| **DE2**: Día anterior | Día anterior al DE1, si el DE1 fuera lunes entonces el DE2 será sábado. |
| **DE3**: Día siguiente | Día siguiente al DE1. |
| **DE4**: Semana siguiente | Una semana después al DE1, sirve para verificar que la demanda haya seguido con normalidad 7 días después del feriado o festividad. |

Tabla 8. Tipos de días especiales (DEs) con su respectiva descripción. - Huber, J., & Stuckenschmidt, H. (2020).

Estos 4 tipos de DE serán ingresados como variable dentro del dataset, se creará una columna llamada DE donde se indicará qué tipo de día es (Del 1 al 4). Para esta investigación, se tomaron en cuenta los DE de Inglaterra, puesto que el centro de operaciones y ventas, de donde proviene el dataset, residen en ese país. Debido a que el dataset, cuenta con registros que van desde noviembre del 2017 a diciembre del 2019, se deberán tener en cuenta aquellos días que no tienen una fecha específica, sino que son cada cierto número de semanas. En la Tabla 8, se presenta un listado de todos los DE de dicho país, ordenados desde enero a diciembre.

|  |  |
| --- | --- |
| **Fecha** | **Descripción** |
| 01-01 | Año Nuevo |
| 02-01 | Feriado bancario |
| 17-03 | Día de San Patricio |
| 10-04 | Viernes Santo**\*** |
| 13-04 | Pascua**\*** |
| 08-05 | Early May Bank Holiday |
| 25-05 | Spring Bank Holiday |
| 12-07 | Batalla de Boyne |
| 31-08 | Summer Bank Holiday |
| 30-11 | St. Andrew’s Day |
| 25-12 | Feriado de Navidad |
| 26-12 | Feriado de Navidad |

Tabla 9. Listado de días especiales (DE) – Recuperado de https://www.gov.uk/bank-holidays.json

Todos los días listados en la Tabla 9 son del Año 2020, la fecha de la mayoría de estos DE se mantiene la misma para todos los años, sin embargo, los días **\*** son aquellos que su fecha cambia según el año. Sin embargo, la fuente de los datos tiene registros desde el 2015 hasta el 2021.

## INGENIERÍA DE CARACTERÍSTICAS

Los atributos de entrada de los modelos de ML son sumamente importantes para poder encontrar patrones entre las variables y los registros del dataset. La ingeniería de características nos permite generar nuevos atributos, en base a los que ya tenemos, que mejoren el modelo (Que ayuden a explicar mejor la demanda). Por otro lado, generar una gran cantidad de atributos puede hacer que se manifieste overfitting (Cuando el modelo deja de aprender al tener demasiados atributos). Para los beneficios de esta investigación, se han construido 3 atributos: Histórico de nivel de demanda, cambio absoluto y cambio relativo. La descripción de cada uno de los atributos, y cómo se construyen, se muestran en la Tabla 10.

|  |  |
| --- | --- |
| **Atributo** | **Descripción** |
| Promedio histórico de demanda | El promedio de ventas histórico para cada día específico de la semana. |
| Cambio absoluto | La diferencia absoluta de ventas en un DE con el promedio histórico del día de la semana. |
| Cambio relativo | La diferencia relativa de ventas en un DE con el promedio histórico del día de la semana. |

Tabla 10. Listado de atributos creados – Elaboración propia

Los atributos mostrados en la Tabla 10 van a permitir que el modelo pueda aprender y obtener explicaciones numéricas de los cambios en la demanda de los DE, además de las diferencias ocasionadas por días de semana o fin de semana.

## ALGORITMOS GENÉTICOS

Los algoritmos genéticos (GA), permiten obtener la mejor solución de una problemática en base a una población de posibles soluciones, esta estrategia está inspirada en la teoría de selección natural como lo indican (Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J., 2016). Los GA reflejan el proceso de selección natural donde las soluciones o individuos más aptos son seleccionados para reproducción con el objetivo de que los hijos hereden las cualidades positivas de los padres. Un individuo o gen representa los componentes o características de una posible solución. Las generaciones son realizadas de forma iterativa hasta alcanzar un número de iteraciones determinado o, un threshold de aceptación previamente determinado.

Los GA son una técnica que permite ser adaptada por diferentes soluciones de ML para resolver problemas de clasificación o, como en este caso, de regresión. Para esta problemática, cada individuo representará un conjunto de hiperparámetros, esta estructura será diferente para cada una de las técnicas, pero se escogerán aquellos que tengan mayor influencia en la solución. Para ello, se deben definir las diferentes estructuras de GA para cada técnica de ML que se vaya implementar (ANN, GBT y SVM).

### **ESTRUCTURAS DE GA POR CADA TÉCNICA**

Para cada técnica de ML se va a definir su propia estructura. Empezando por ANN, está técnica de ML es una de las que contiene la mayor cantidad de hiperparámetros que pueden ser modificados para obtener diferentes soluciones, sin embargo, en este caso es necesario seleccionar solo los más relevantes. Se tomará como referencia la investigación realizada por Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J. (2016) para definir qué hiperparámetros conformarán cada individuo. Par esta investigación se utilizará un MLP (Multi Layer Perceptron), el cual estará compuesto por tres capas: Entrada, una oculta y la capa salida. En la capa de entrada se encontrarán todas las variables que sirvan para poder explicar la demanda del producto seleccionado: “White hanging heart t-light holder”, las neuronas de la capa oculta son las que van a cambiar por cada individuo, para ello se tiene que determinar una cantidad máxima de neuronas que se puedan generar, en este caso estará delimitado a 20, y la capa de salida contará con el pronóstico de unidades de venta para un día en específico. La estructura del MLP se encuentra graficado en la Figura 16.

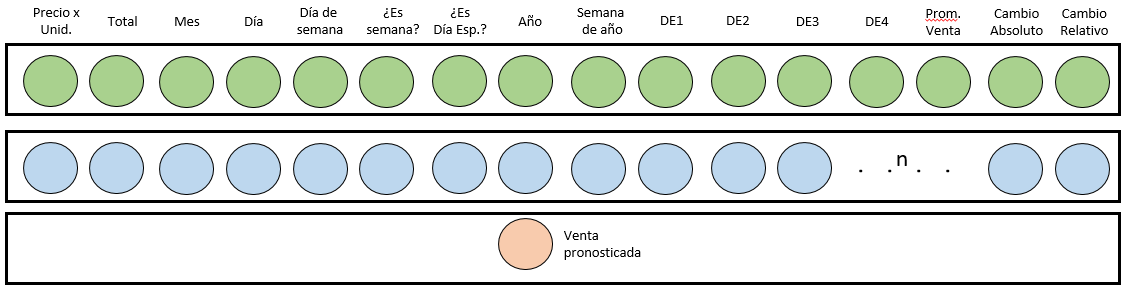


Figura 17. Estructura de capas del MLP – Elaboración propia

Como se observa en la Figura 17, la capa de entrada, la que contiene las neuronas en verde, está conformada por todas las entradas que se utilizarán para entrenar al modelo, la capa oculta, contiene las neuronas en celeste, tiene una cantidad de neuronas que será diferente por cada individuo con un máximo de 20. Ya la última capa, capa de salida, solo está conformada por la salida que es la cantidad de venta pronosticada para un día en específico.

Cada individuo de este modelo tendrá la estructura de vector, la primera posición identificará el número de neuronas en la capa oculta, la segunda y tercera, tendrán los valores enteros de entre [0-2], cada uno representará una función de activación: Relu, Tangente Hiperbólica y Sigmoideal, estas funciones de activación fueron seleccionadas según la investigación de Eckle, K., & Schmidt-Hieber, J. (2018) que selecciona estas como las principales. Debido a que el modelo cuenta con 3 capas se utilizarán 2 funciones de activación (Una por cada conexión entre capas). El resto de valores del vector serán 2 sets de pesos, el primer set contará con todos los pesos de las conexiones de las neuronas de la capa de entrada a la capa oculta, y en el segundo set se contará con los pesos de cada conexión de la capa oculta a la capa de salida. En la Figura 17 se exhiben gráficamente los hiperparámetros seleccionados para ser “tuneados”, la estructura de cada individuo y cómo ingresarán a cada cromosoma, que cumple con formato de vector.

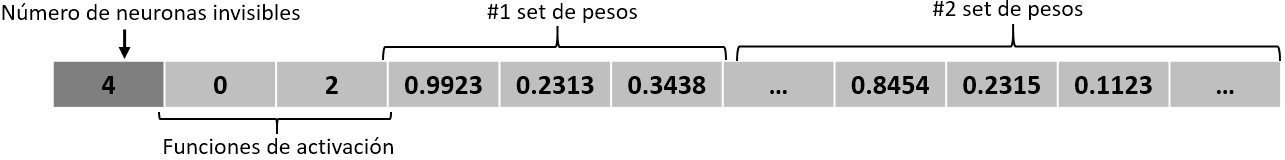


Figura 18. Estructura de GA para la ANN – Elaboración propia

Como se muestra en la Figura 18, la primera posición es el número de neuronas ocultas que tiene la arquitectura de red, la segunda y tercera son las funciones de activación, y las siguientes son los 2 sets de pesos mencionados. Todo el vector que se muestra en la Figura 17 viene a ser la estructura completa de un cromosoma (Individuo) para la técnica de optimización (GA).

La siguiente técnica que será implementada será SVM en su versión de regresión. El tipo de kernel que se utilizará será RBF (Radial Basis Function), puesto que es el más común para regresión en SVM por su buena capacidad de generalización (Zhang, D., Xiao, J., Zhou, N., Zheng, M., Luo, X., Jiang, H., & Chen, K., 2015). Los hiperparámetros escogidos serán *C*, *E*, *G,* al ser conocidos como hiperparámetros de diseño, mencionado en los Antecedentes. En la Figura 18 se diagrama la arquitectura del cromosoma para esta técnica.

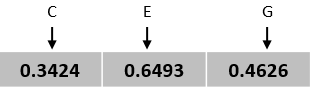


Figura 19. Estructura de GA para SVM Regresión – Elaboración propia

La Figura 19 muestra la estructura del individuo que será utilizado para la técnica de SVM, cada vector estará compuesto por los 3 hiperparámetros C, E, G.

Finalmente, los hiperparámetros que se escogerán para GBT serán 7 en total, puesto que esta técnica también cuenta con una gran cantidad hiperparámetros que pueden ser modificados para obtener una mejor solución. El cromosoma estará compuesto por los siguientes elementos: ratio de aprendizaje, número de estimadores, profundidad máxima, peso mínimo del descendiente, subsample, colsample bytree y gamma. En la Figura 19 se muestra su estructura del cromosoma además de todos los hiperparámetros que lo componen.



Figura 20. Estructura de GA para GBT – Elaboración propia

En la Figura 20 se muestra en detalle la estructura del vector, las sentencias utilizadas son los nombres de los hiperparámetros utilizados en el modelo de GBT.

### **ETAPAS Y METODOLOGÍA DE GA**

Una vez que tenemos completamente definido la estructura de los cromosomas para cada una de las técnicas de ML que se van a aplicar, ya se puede definir el proceso o etapas por la cual atravesarán cada una de las poblaciones que se definan. En la Figura 21 se muestran cada una de las etapas secuenciales que se deben realizar para poder implementar la Algoritmos Genéticos sobre los modelos ML, estas etapas se encuentran propiamente detallas en la sección de Antecedentes.

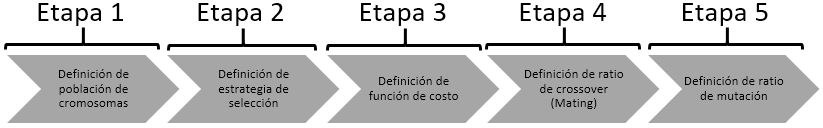


Figura 21. Etapas del GA – Elaboración propia

En la Etapa 1, se debe definir la población de individuos que van a ser parte del proceso de selección natural, para ello se generarán 20 individuos de forma aleatoria para crear los hiperparámetros de cada uno de los cromosomas.

La Etapa 2 consiste en determinar la estrategia de selección de individuos para ser puestos a prueba entre ellos. Si bien Roulette Wheel conlleva un mayor trabajo computacional, esta permite generar mejores descendientes, puesto que los individuos con menor costo (Los mejores) tendrán prioridad de selección, lo que permitirá reproducirse en un hijo que combine sus características positivas para el modelo (Höschel, K., & Lakshminarayanan, V., 2018).

La Etapa 3 consiste en definir la función de costo que se va a utilizar para evaluar a cada uno de los individuos de la población, es una métrica que nos permitirá medir numéricamente que tan buena solución puede ser un cromosoma. Se empleará el Mean Square Error (MSE) como métrica, al ser la más conocida y utilizada por autores en Estado del Arte presentado, esta métrica obtiene el promedio de las diferencias de los resultados obtenidos con los esperados, también es conocido como el nivel de error del modelo, mientras menor sea el valor mejor será la solución.

En la Etapa 4, luego de haber seleccionado 2 individuos de forma aleatoria y haber calculado el costo de cada uno, se inicia el proceso de Crossover, el acontecimiento de este proceso está sujeto a un ratio de ocurrencia que será definido en 50%. En la Figura 22 se muestra gráficamente en qué consiste la Etapa 4 sobre la estructura de cromosoma de SVM definida previamente.

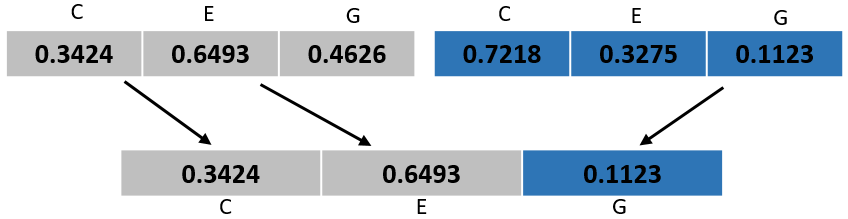


Figura 22. Proceso de Crossover del GA para SVM – Elaboración Propia

En la Figura 22 se ve el proceso de Crossover con dos individuos seleccionados aleatoriamente, el Cromosoma A está resaltado de color gris, que al tener una menor función de costo tiene prioridad de pasar sus características al hijo sobre el Cromosoma B resaltado de color azul.

Finalmente, en la Etapa 5 se realizará la operación de Mutación. La probabilidad de mutación que se utilizará será 20% parar todas las técnicas que se utilizarán. En la Figura 22 se grafica la manifestación de la mutación sobre un cromosoma descendiente.

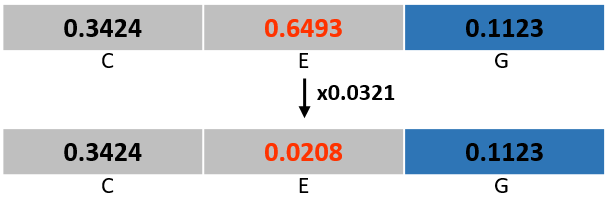


Figura 23. Mutación del cromosoma hijo – Elaboración Propia

En la Figura 23, se muestra una representación visual del proceso de mutación cuando un individuo la manifiesta. El hiperparámetro E es seleccionado aleatoriamente para ser mutado, luego este es multiplicado por un factor aleatorio para dar paso a una mutación del cromosoma, siendo este el nuevo resultante para la primera generación.

### **FLUJO DE EJECUCIÓN DE LOS GA**

Ahora que se tienen todas las estructuras de cromosomas definidas para RNN, SVM y GBT, a la vez que todas las etapas que moldean la estructura de trabajo de los Algoritmos Genéticos, lo último que se necesita es establecer un flujo de ejecución e interacción de los GA para luego poder establecer las rutinas de trabajo del código que será programado. En la Figura 24 se muestra el Flujo de Trabajo propuesto para llevar a cabo la implementación de los GA.

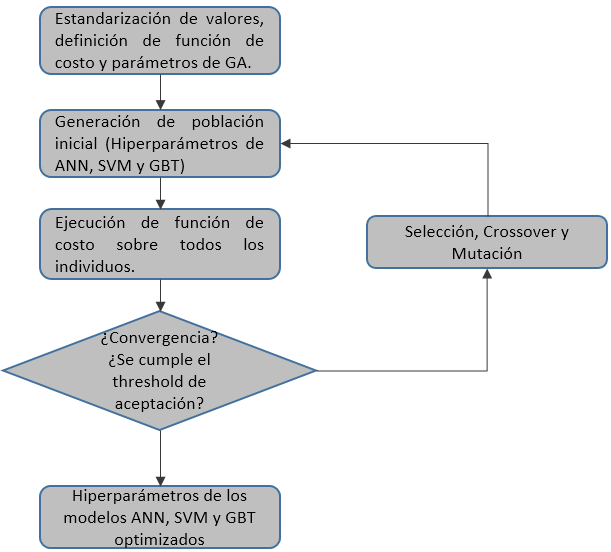


Figura 24. Flujo de ejecución de GA para todas las técnicas – Elaboración propia

En la Figura 24 se muestra el flujo de ejecución del modelo, incluyendo una casuística que permite determinar cuántas veces deberá ser ejecutado los modelos. Este flujo es asumiendo un número de generaciones ilimitado, puesto que la condición de salida del modelo es cuando se cumple con el threshold de aceptación ajustado al 5% error, siendo este el objetivo que todos los modelos deberían apuntar. Sin embargo, si el modelo no llegara a converger, y se quedara estancado, este seguiría ejecutándose de forma continua sin límite alguno. Por ello, también se debe determinar una cantidad de generaciones máxima a ejecutar para que, en el caso que no se manifieste convergencia en el modelo, este deberá dejar de correr después de 100 generaciones.

# EXPERIMENTACION

Para aproximar la problemática planteada, se llevarán a cabo las estrategias mencionadas en la metodología. La implementación de estas estrategias se realizará utilizando el lenguaje de programación Python 3, además de utilizar las librerías de TensorFlow, para la creación de los modelos de ANN, SVM y GBT; Pandas, para el preprocesamiento del dataset, además de la implementación de la incorporación de los Días Especiales (DE) y construcción de nuevas variables; finalmente se utilizó la librería Deap que permitió la creación de los modelos de GA (Algoritmo Genético) para cada una de las técnicas de ML y la estructuración de la población, los cromosomas, las operaciones de selección, crossover y mutación. Luego de obtener los resultados para cada una de las técnicas de procederá a realizar un proceso de comparación de resultados, donde se buscará determinar la técnica que obtuvo la mayor precisión de las 3.

La base de datos para realizar el entrenamiento y pruebas será un dataset libre obtenido de Kaggle, el cual comprende los registros de ventas y datos adicionales de productos de una tienda virtual de recuerdos ubicada en Inglaterra. El producto seleccionado para realizar el pronóstico es ‘White hanging heart t-light holder’. El horizonte está comprendido por un periodo de 26 meses, desde enero de 2017 hasta diciembre de 2019. El dataset cuenta con 5574 registros, está compuesto por 8 columnas: Número de boleta, código de producto, descripción de producto, cantidad comprada ese día, fecha de boleta, precio de unidad, identificador de cliente y país de venta. En la Figura 25 se muestra la venta agregada semanalmente del producto ‘White hanging heart t-light holder’ desde el 2017 al 2019.



Figura 25. Ventas agregadas semanalmente del producto ‘White hanging heart t-light holder’ – Elaboración Propia

Como se puede observar en la Figura 25, el producto experimenta altos picos de venta dentro de un rango de semanas, el modelo planteado buscará poder identificar esos días y explicar los outliers mediante los Días Especiales.

La implementación de la solución planteada será dividida en cuatro etapas: La primera etapa, Preprocesamiento de Datos, en esta etapa inicial se realizarán pruebas de calidad de datos sobre el dataset, dentro de estas pruebas se buscan identificar valores vacíos, nulos, registros repetidos y además de realizar una depuración de los picos de valores, siendo; Segunda etapa, Hiperparámetros de los Modelos, en esta etapa se van a definir los hiperparámetros relevantes para ser tuneados e ingresados al modelo de Algoritmos Genéticos que se definirán en la siguiente etapa; tercera etapa, Modelamiento del modelo de Algoritmos Genéticos, esta etapa consiste en la estructuración del modelo de GA que permita calibrar los hiperparámetros seleccionados mediante operaciones de selección, crossover, y mutación; En la etapa final, Evaluación de Resultados, se realizará la comparación de los resultados de todos los modelos, además de generación de escenarios con diferentes valores de hiperparámetros del modelo de GA.

El flujo de actividades para cada etapa está diagramado detalladamente en la Figura 26.

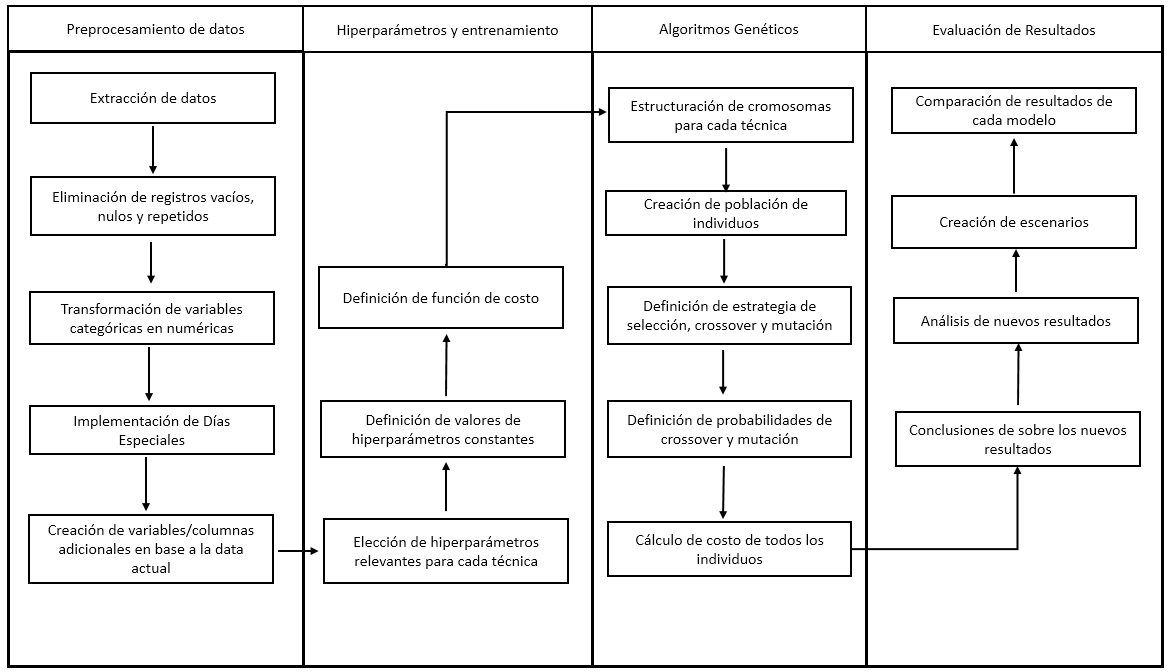


Figura 26. Flujo de metodología de implementación de solución – Elaboración propia

La primera etapa, Pre procesamiento de datos, consiste en extraer la fuente de datos del repositorio de Kaggle, luego se definen cuáles serán las columnas que se utilicen para realizar el entrenamiento. Se realiza la eliminación de registros vacíos, formatos inválidos, columnas innecesarias y estandarización de columnas en formatos adecuados. Además, se incorporan las columnas de Días Especiales (DE) con un formato de fecha estándar y se crearán las variables adicionales como se indicó en la sección de Metodología. Finalmente se genera la nueva base la cual se usará para el entrenamiento de los modelos. Para este dataset, se contabilizó todos los días especiales existentes dentro del mismo, en la Tabla 11 se puede ver su distribución y el total.

|  |  |
| --- | --- |
| **DE (Día Especial)** | **Cantidad** |
| DE1 | 125 |
| DE2 | 205 |
| DE3 | 89 |
| DE4 | 188 |
| **Total** | 607 |

Tabla 11. Distribución de DE en el dataset – Elaboración propia

Como se puede observar en la Tabla 11, se muestra la frecuencia de cada uno de los tipos de DE en el dataset, contabilizándose un total de 607 DE, este es un número alentador puesto que mientras mayor sea este valor mejor se podrá explicar la demanda, al tener más registros especiales de donde los modelos puedan aprender.

La segunda etapa, Hiperparámetros y entrenamiento, se seleccionaron los hiperparámetros que serán tuneados por el modelo de Algoritmos Genéticos. Para la ANN los hiperparámetros que serán modificados son el peso de las conexiones y la cantidad de neuronas que existen en la capa oculta, para SVM, se seleccionarán 3 hiperparámetros, el encargado de regular los errores de entrenamiento, el ancho de la zona en el plano, y el coeficiente, del Kernel que se utilizará (Radial Basis Function). Para todos los hiperparámetros, excepto el número de neuronas, se utilizará una función que determine su valor de forma aleatoria en un rango de [0.001-0.01], para las neuronas se determinará un máximo de neuronas en la capa oculta y se seleccionará aleatoriamente la cantidad por individuo.

En la tercera etapa, Algoritmos Genéticos, se definirá el modelo y estructura de los Algoritmos Genéticos, primero se estructurarán los cromosomas (Individuos) que se serán utilizados de forma diferenciada para cada técnica de ML. La cantidad de individuos en la población serán 100, se definirán las operaciones que serán utilizadas. El operador de selección utilizará la estrategia de Roulette Wheel Selección. Las operaciones de crossover y mutación tendrán una probabilidad de ocurrencia de 20% y 50% respectivamente. El modelo será ejecutado iterativamente empleado todos los operadores indicados hasta alcanzar el threshold de precisión definido en 5% de error, o llegar a un límite de generaciones corridas que estará establecido en 100, luego el modelo seleccionará al mejor individuo como la solución óptima al problema.

La etapa final, Evaluación de Resultados, se obtendrán los resultados de las 3 técnicas implementadas (ANN, SVM, GBT), basándose en la métrica MSE y tiempo de ejecución. Luego de seleccionada la técnica que obtuvo la mayor precisión.

Para la experimentación se utilizarán las técnicas, algoritmos, conceptos desarrollados en la sección de Marco Teórico y Metodología. Se utilizará el lenguaje de programación Python 3 en Anaconda, junto con las librerías Keras, Sklearn y TensorFlow-GPU. La ejecución del modelo será de forma local utilizando la tarjeta de video Nvidia RTX 2070 con 8GB de VRAM y 2304 CUDA cores, pero también algunas pruebas serán realizadas en Google Colab con su ambiente por defecto. El código implementado se puede encontrar en el siguiente repositorio en Github. <https://github.com/Estremadoyro/Tesis>

## EXTRACCIÓN Y PRE PROCESAMIENTO DE DATOS

Lo primero que se debe realizar es un control de calidad sobre los datos de la base o “cleansing”, el objetivo de este proceso es detectar y eliminar datos o registros que tengan valores nulos o que no cumplan con el tipo de dato de la columna. No realizar esta etapa conllevaría a problemas al momento de ejecutar el modelo, puesto que esos valores sesgarían el resultado o directamente arrojarían un error. Un paso muy importante para modelos de ML, cuando hay gran variabilidad en sus datos, es la estandarización de datos numéricos. La estandarización de datos consiste en escalar los datos a un intervalo de [0-1], esto hace que los modelos de ML utilicen todos los datos con el mismo nivel de prioridad, puesto que se trabajan con magnitudes, de este modo cada registro es utilizado y no se priorizan valores de una mayor magnitud. También, es vital realizar una traducción de variables categóricas a numéricas, puesto que los modelos funcionan únicamente con valores numéricos. Esto se realiza mediante la asignación de un valor numérico a uno categórico para ser reemplazado. Finalmente, es necesario establecer un formato estándar para las fechas del dataset, y así poder ser interpretado correctamente por Python, para ello se convierte la columna de periodo en formato americano.

Durante esta primera etapa se extraerán los datos en formato CSV. Se analizarán las columnas de la base de datos para escoger las que se utilizarán para realizar el pronóstico, en esta investigación se utilizarán, para cada producto, unidades vendidas en un día, descripción del producto, fecha de boleta, precio de unidad de producto y país de venta. Además, se realizará una depuración de datos en busca de valores vacíos, incompletos o de una unidad diferente. La variable Fecha será pre procesada en día, mes y año. El conjunto de días especiales de Inglaterra y Gales, desde el 2015 al 2021, se puede encontrar en la siguiente API del gobierno de Reino Unido <https://www.gov.uk/bank-holidays.json>. Este API facilita altamente la extracción de los días especiales, puesto que utilizando el lenguaje Python se pueden procesar fácilmente. También se realizó la creación de las tres variables propuestas: Promedio histórico de demanda por día, cambio absoluto y cambio relativo.

Finalmente, se distribuirán los datos en dataset de entrenamiento (60%), validación (20%) y prueba (20%), los datos pertenecientes a cada grupo serán escogidos aleatoriamente.

## ARQUITECTURA DE RED NEURONAL

En esta etapa se definirá la arquitectura de capas que se utilizará para evaluar y comparar durante el proceso de entrenamiento y prueba. La ANN estará compuesta por 3 capas, una capa de entrada, una oculta y una de salida.

Para realizar el aprendizaje, se utilizarán el 60% de datos, proporción mencionada anteriormente, se utilizará backpropagation, para calcular la gradiente, junto con stochastic gradient descent para poder modificar los pesos de las conexiones entre neuronas y permitir que la red aprenda. Los pesos iniciales, funciones de activación, y cantidad de neuronas en la capa oculta, ingresarán al modelo de Algoritmos Genéticos. Para el aprendizaje, se utilizará un ratio de 0.01 para ser más preciso en la optimización de la red, el entrenamiento, constará de 100 épocas con tamaño de batch de 500MB para cada uno de los modelos. Los hiperparámetros mencionados se pueden observar en la Fórmula 4.

Fórmula 4. Hiperparámetros de ANN – Elaboración propia

## HIPERAPRÁMETROS DE SVM

Para el modelo de SVM se seleccionaron 3 hiperparámetros para que formen parte de la estructura de individuo en la población de los GA, los cuales fueron: El regulador de errores de entrenamiento, el ancho de la zona en el plano, y la amplitud del Kernel. Por otro lado, el tipo de kernel que se deberá utilizar es RBF, como se muestra en la Fórmula 5, estos hiperparámetros seleccionados fueron tomados como referencia del trabajo de Zhang, D., Xiao, J., Zhou, N., Zheng, M., Luo, X., Jiang, H., & Chen, K. (2015).

Fórmula 5. Hiperparámetros de SVM Regresión – Elaboración propia

## HIPERAPRÁMETROS DE GBT

Finalmente, para GBT, se seleccionaron aquellos hiperparámetros que no serán optimizados en el proceso de Algoritmos Genéticos, pero serán utilizados dentro del modelo como input. Estos hiperparámetros son ilustrados en la Fórmula 6.

Fórmula 6. Hiperparámetros de GBT – Elaboración propia

Mientras que los hiperparámetros utilizados para la construcción del cromosoma, para el modelo de optimización (GA), fueron: learning\_rate, n\_estimators, max\_depth, min\_child\_weight, subsample, colsample\_bytree y gamma, los cuales fueron explicados en la sección 3.9 de Antecedentes.

# RESULTADOS

Todas las técnicas de ML fueron acopladas dentro de la estructura del modelo que emplea GA bajo los mismos hiperparámetros, esto se puede reflejar en la Tabla 12.

|  |  |
| --- | --- |
| **Hiperparámetro** | **Valor** |
| Estrategia de selección | Roulette Wheel Selection |
| Individuos por selección | 3 |
| % de ocurrencia de Crossover | 20% |
| % de ocurrencia de Mutación | 50% |
| Número máximo de generaciones | 100 |
| Threshold | 5% de error |
| Población | 100 |

Tabla 12. Hiperparámetros de GA para todas las técnicas – Elaboración propia

En la Tabla 12 se muestran todos los hiperparámetros que se han utilizado para correr todas las técnicas (ANN, SVM y GBT).

Al finalizar la ejecución de todos los modelos se obtuvieron los mejores individuos para cada técnica. En la Figura 27 se muestra el mejor individuo de la ANN.

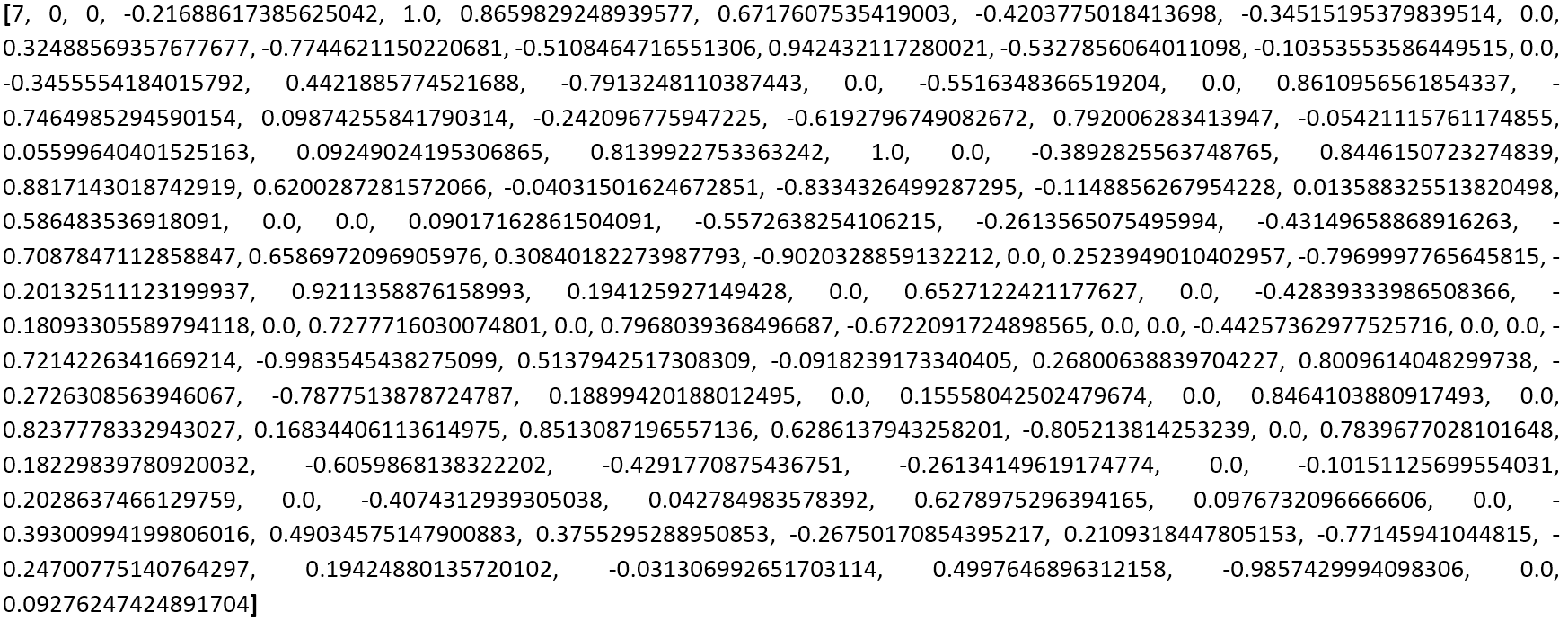


Figura 27. Mejor individuo de ANN – Elaboración propia

Algo interesante que recalcar es que el mejor individuo de ANN contempla un modelo con solo 7 neuronas en la capa oculta, puesto que la población inicial contenía soluciones de entre 0 y 20 neuronas es la capa oculta, siendo los demás valores los pesos de conexiones de capa de entrada a oculta y de oculta a salida.

En la Figura 28 se muestra el mejor resultado para la técnica de SVM.



Figura 28. Mejor individuo de SVM – Elaboración propia

Con respecto a SVM, podemos ver que la Gamma obtuvo un valor entero, siendo esto algo interesante que rescatar puesto que la población tenía valores que rondaban el 0.001 a 0.1, pero la mutación permitió obtener este valor entero. También se puede observar que el valor de E es muy cercano a 0, al igual que C pero en este caso se acerca más al entero 1.

Finalmente, en la Figura 29 se muestra el mejor individuo para GBT.



Figura 29. Mejor individuo de GBT – Elaboración propia

Para GBT podemos observar los valores óptimos para cada uno de los hiperparámetros utilizados, es interesante resaltar el número de estimadores óptimo, 85, puesto que el valor por defecto que se utilizaría normalmente es 100, entonces el modelo requiere una menor cantidad de árboles, o modelos para mejorar, de lo contrario, probablemente ocasione overfitting.

Después de la ejecución de todos los modelos se seleccionaron métricas de comparación, para esta investigación se utilizaron 5 métricas: MSE, RMSE, MAE, MAPE y tiempo de ejecución. En la Tabla 13 se muestra la tabla comparativa de resultados para todas las técnicas.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **ANN** | **SVM** | **GBT** |
| **MSE** | 27.74 | 530.56 | 812.51 |
| **RMSE** | 5.26 | 23.03 | 28.50 |
| **MAE** | 1.56 | 2.27 | 2.74 |
| **MAPE** | 25.37% | 7.66% | 13.17% |
| **Precisión** | **74.62%** | **92.34%** | **86.83%** |
| **Tiempo de ejecución** | 8 h 12min. | 1 h 12 min. | 12 min. |

Tabla 13. Tabla de resultados de ejecución de modelos – Elaboración propia

Como se puede observar en la Tabla 13, se obtuvieron resultados para cada uno de los modelos planteados. SVM fue el modelo que obtuvo el mayor nivel de precisión con un 92.34% sobre el test de prueba. Por otro lado, ANN es el modelo con la menor precisión (74.62%) y a su vez el que demora más en ejecutar completamente (8 horas con 12 minutos). GBT obtuvo una precisión de 86.83%, lo cual se podría categorizar como alta; sin embargo, lo más resaltante de este modelo es que su tiempo de ejecución fue de tan solo 12 minutos, menos del 10% del tiempo de SVM, y mucho menos de ANN.

Se realizó un análisis de las curvas de pérdidas de cada uno de los modelos finales, esto es con el objetivo de determinar si existe overfitting o si el modelo deja de aprender progresivamente durante el entrenamiento y validación. Cada uno de los modelos utilizó MSE como su métrica objetivo de minimización. En la Figura 30, se muestra la curva de pérdida de la ANN óptima.

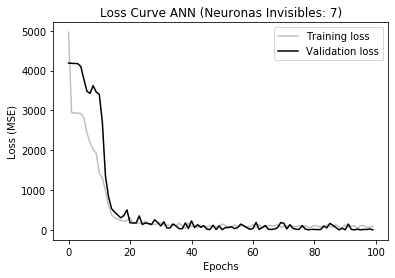


Figura 30. Curva de pérdida de ANN – Elaboración propia

Como se puede ver en la Figura 30, se han graficado dos curvas de pérdida, la del set de entrenamiento y la de validación. En ambas curvas, podemos ver que el modelo aprende drásticamente hasta la época 20, a partir de esta época el modelo deja de converger.

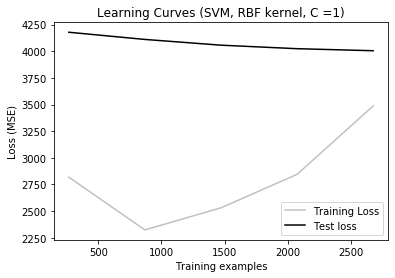
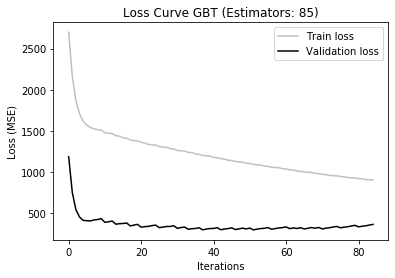


Figura 31. Curva de pérdida de SVM – Elaboración propia

En la Figura 31 se muestra la curva de pérdida para el modelo de SVM, previamente seleccionado como el mejor modelo en los resultados mostrados en la Tabla 12. La curva de pérdida del entrenamiento muestra varios picos de precisión, durante las primeras 1000 etapas de entrenamiento se ve cómo el modelo mejora drásticamente hasta que empieza a empeorar a tal punto que su precisión disminuye de la inicial. En la iteración 2000, el MSE incrementa aún más llegando a 3600. Por otro lado, la curva de pérdida, para el dataset de validación, tiene muchos mejores valores, puesto que tienen una tendencia a disminuir de forma constante y no experimentan picos positivos o negativos. Pese a ello, si bien el aprendizaje es constante, esta curva tiene valores muy altos, pero con tendencia a mejorar. Es por ello, que se debería entrenar el modelo con una mayor cantidad de iteraciones para verificar si se podrían obtener mejores resultados.

Figura 32. Curva de pérdida de GBT – Elaboración propia



La Figura 32 se muestra la curva de pérdida de GBT, se observan resultados esperados, iniciando con un alto valor de MSE y disminuyendo hasta dejar de converger por la tercera iteración. Sin embargo, el modelo muestra un alto nivel de aprendizaje dentro de las iteraciones iniciales, pero rápidamente deja de converger y mejorar los resultados.

Adicionalmente a las curvas de pérdida, se realizaron gráficos que permitan visualizar los pronósticos realizados por cada modelo, frente a los picos de demanda ocasionados por los días espaciales. Además, en cada gráfico se incluyó el valor esperado, la demanda real. El periodo utilizado fue desde mayor del 2018 a noviembre del mismo año, se escogió este rango de meses al tener la mayor cantidad de picos de ventas en el año. En la Figura 33, se muestran 7 meses de predicción del modelo SVM.

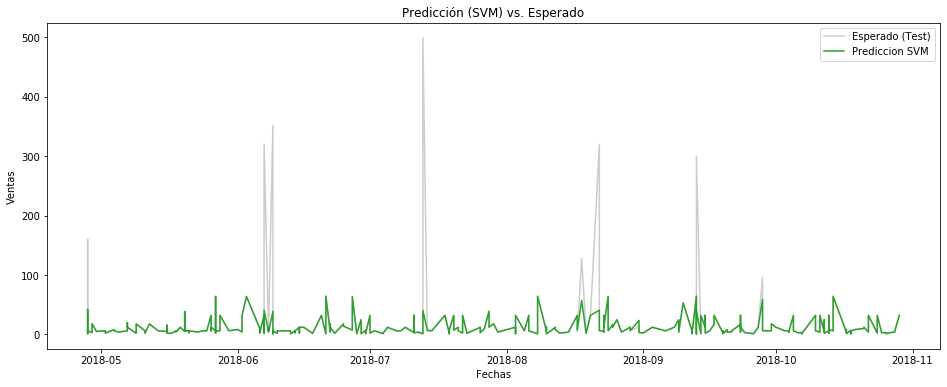


Figura 33. Predicción del modelo SVM vs. valor esperado – Elaboración propia

Como se muestra en la Figura 33, para el modelo SVM las predicciones son altamente precisas cuando los picos de demanda son pequeños y no fluctuan de gran manera, sin embargo, cuando está frente a picos muy altos, como a mediados de año del 2018, el modelo no intenta predicir dicho portamiento de demanda y mas bien se mantiene constante. Lo mismo sucede meses adelante.

En la Figura 34 se muestra el grafico de predicción versus esperado para el modelo GBT, en este caso es posible identifcar una diferencia significativa entre el comportamiento de este modelo con SVM.

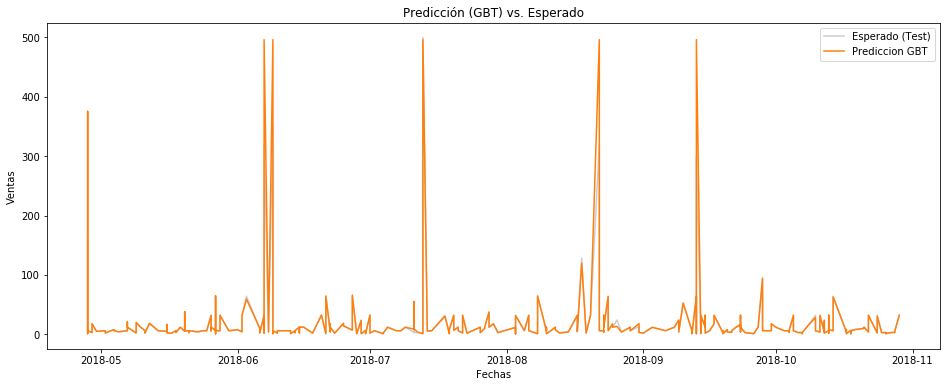


Figura 34. Predicción del modelo GBT vs. valor esperado – Elaboración propia

A diferencia de SVM, este modelo sí intenta predecir las fechas en que las ventas incrementan drasticamente, como se muestra a lo largo del año. Sin emgaro, GBT al intentar pronosticar los picos de venta, genera predicciones muy por encima del valor real, pero esto involucra un riesgo, puesto que sacrifica precisión para series constantes, por un mejor pronóstico para los outliers, que al final ese era el objetivo de la incorporación de Días Especiales (DE).

Finalmente, para ANN, el modelo con la menor precisión, se realizó su grafico respectivo en la Figura 35.

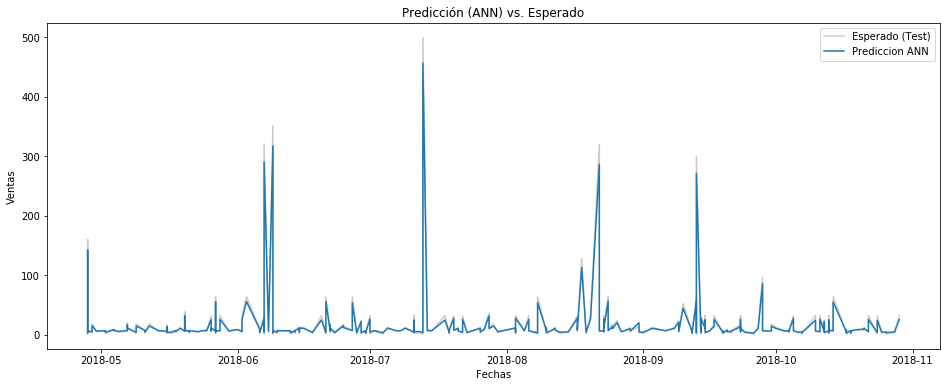


Figura 35. Predicción del modelo ANN vs. valor esperado – Elaboración propia

La Figura 35 muestra los pronosticos del modelo ANN, como se puede ver, las predicciones realizadas son un intermedio entre los modelos de SVM y GBT, puesto que sí buscan pronosticar los altos picos de demanda, sin sobrepasarlos, pero en picos menores o series constantes de ventas es donde su precisión decae. Por lo tanto, el modelo tiene mayores inprecisiones a lo largo del periodo.

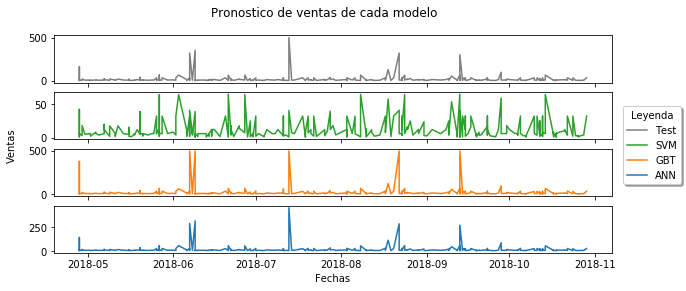


Figura 36. Pronostico de ventas de cada modelo – Elaboración propia

Finalmente, el la Figura 36 se muestra un gráfico con todos los pronósticos de cada modelo de forma resumida, si bien SVM parece que estuviera prediciendo valores similares a los otros modelos, la gráfica muestra que su mayor valor es 60. Cada uno de los modelos han demostrado tener un nivel alto de precisión, del 63% hasta un 92% por parte de SVM. Tanto ANN como GBT han sido capaces de pronosticar los picos de demanda generados por DE, lo cual permite que estos modelos sean versátiles para otras series de tiempo que sean afectadas por factores estacionales como este es el caso. En la siguiente sección, Discusión de Resultados, se realizarán comparaciones de los resultados obtenidos en esta investigación, con los resultados de la bibliografía revisada, además se realizarán pruebas con diferentes hiperarpámetros del modelo de algoritmos genéticos, para poder verificar si se podrían obtener mejores resultados.

# DISCUSIÓN DE RESULTADOS

Esta sección tiene como objetivo justificar los resultados obtenidos durante la experimentación y ejecución de cada uno de los modelos. En esta investigación se determinó que el modelo más preciso fue SVM, en su modalidad de regresión, obteniendo una precisión del 92.34%; seguido de GBT con una precisión del 86.22%, y finalmente la ANN con 74.62%. En primer lugar, uno de los principales motivos por el cual se logró obtener resultados superiores al 75%, aproximadamente, en todos los modelos, es consecuencia de haber tomado en consideración la estacionalidad del producto, tanto a nivel semanal como mensual, como factor relevante que permita explicar la demanda durante ciertos periodos.

Si bien en la presente investigación no se realizó la comparación del modelo actual con uno sin considerar el factor estacional, en la investigación de Di Pillo, G., Latorre, V., Lucidi, S., & Procacci, E. (2016) realizan dicha comparación con un modelo de SVM. Su investigación tuvo como objetivo realizar el pronóstico de demanda de productos de pasta en dos tiendas de la misma cadena, para ello elaboraron dos modelos de SVM, uno que no considera la estacionalidad como variable influyente en la demanda, y otro que sí. En la Tabla 14 se muestran los resultados obtenidos en su investigación, donde SVM4i representa el modelo que tan solo utiliza el precio de producto, horas de apertura al día y un booleano que representa si existe o no promoción, mientras que SVM12i sí considera estacionalidad, por lo tanto, no solo utiliza las variables mencionadas, si no, también los días de la semana de forma booleana y el número de mes en que se encuentra.

|  |  |  |
| --- | --- | --- |
| **Intervalo** | **SVM4i** | **SVM12i** |
| **1** | 44.689 | 37.223 |
| **2** | 49.287 | 47.2 |
| **3** | 40.579 | 37.932 |
| **4** | 37.823 | 34.913 |
| **5** | 141.23 | 172.032 |
| **6** | 31.656 | 25.636 |
| **7** | 80.293 | 88.65 |
| **8** | 77.818 | 44.933 |
| **9** | 52.446 | 67.243 |
| **10** | 101.804 | 171.564 |
| **11** | 30.401 | 26.771 |
| **12** | 69.175 | 82.413 |
| **13** | 44.057 | 36.842 |

Tabla 14. Valores de MSE para cada modelo de SVM - Di Pillo, G., Latorre, V., Lucidi, S., & Procacci, E. (2016)

En la Tabla 14, se puede ver el valor de MSE (Mean Squared Error) para cada uno de los intervalos, donde cada intervalo representa una fracción del año 2011, el cual fue utilizado como set de prueba. Si comparamos el error obtenido en cada intervalo por ambos modelos, podemos ver que en total 8 de 13 intervalos obtuvieron mejores resultados utilizado en modelo SVM12i, el cual efectivamente toma en cuenta la estacionalidad del producto.

Adicionalmente, el modelo utilizado en la presente investigación no solo utiliza el factor de estacionalidad de productos, sino que también incorpora el concepto de Días Especiales (DE), además de la creación de variables adicionales como son el promedio de ventas relativo y absoluto como bien indican Huber, J., & Stuckenschmidt, H. (2020) que estas variables y los DE tienen alta relevancia en el volumen de ventas de un producto, especialmente cuando se trata de retail y no ventas al por mayor o a nivel industrial. También mencionan que el factor de promociones y ofertas tienen un gran impacto sobre la demanda, y que se deberían utilizar en caso se cuenta con la información.

Sin embargo, se realizó una comparación de resultados entre los modelos de los investigadores con el modelo del presente artículo. Para ello, se obtuvo el promedio de cada uno de los intervalos de SVM4i y SVM12i. En la Tabla 15 se muestran los resultados, pero cabe aclarar que cada investigación utilizó datasets diferentes, por lo tanto, no se debería generalizar esta comparación, debería ser tomada a nivel referencial.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **SVM4i** | **SVM12i** | **SVM** |
| **MSE** | 61.63 | 67.18 | 530.56 |

Tabla 15. Resultados de los modelos – Elaboración propia

Como se puede ver en la Tabla 15, están los resultados de ambos modelos además del utilizado en esta investigación. Pero debido a que los autores únicamente utilizaron MSE como métrica de comparación, estos valores son directamente proporcionales a la cantidad de registros que se utilizan en el set de prueba. Esta amplia diferencia permite determinar que sus modelos fueron puestos a prueba con una menor cantidad de datos, puesto que la diferencia de SVM es sustancial. Pero, lo importante de indicar es que el promedio SVM4i es mejor que SVM12i a pesar de que el primero no toma en cuenta la estacionalidad. Probablemente, su investigación necesite utilizar variables adicionales que puedan potenciar las variables de estacionalidad. Como es el caso en la investigación presente, en la que se utilizaron DE y promedios anuales.

Por otro lado, ahora es necesario definir por qué SVM (92.34%) ha sido el mejor modelo, a nivel de precisión de pronósticos, sobre GBT (86.22%) y especialmente sobre ANN (74.62%) al presentar una diferencia de casi el 20% de precisión. En primera instancia, estos resultados parecieran no ser los esperados, debido a que SVM obtuvo los mejores resultados, a pesar de no ser un modelo originalmente pensado para resolver problemas de regresión, como bien lo indica Garrido, W. L. (2012). Sin embargo, la investigación realizada por Demir, L., & Akkaş, S. (2018), indicaría lo contrario, puesto que ellos afirman que SVM tiene una ventaja significante sobre ANN. Dado que la última puede ser afectada negativamente por tener más de una mínima local, mientras que las soluciones de SVM son únicas y globales, y por ende es mejor para pronosticar la venta de productos. En su investigación, buscan pronosticar la demanda de una empresa de alimentos para 5 productos. En la Tabla 16 podemos ver los resultados de su experimentación.

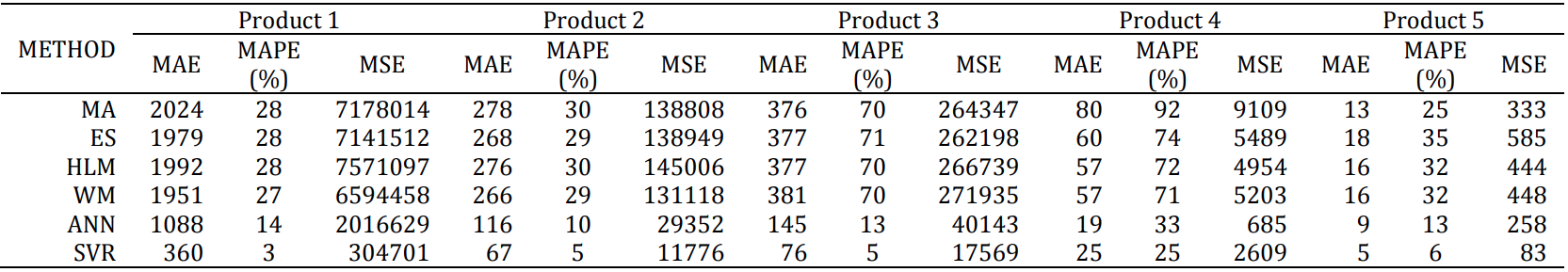


Tabla 16. Errores porcentuales de pronóstico para cada producto - Demir, L., & Akkaş, S. (2018)

Como podemos ver en la Tabla 15, los resultados de ANN y SVR (SVM) se exhiben en la parte inferior para cada uno de los productos con su modelo respectivo. Podemos apreciar como SVR constantemente obtiene mejores resultados que ANN, rondando el 8-11% de diferencia, siendo que en el mejor escenario SVR obtiene un error de tan solo 3% y ANN de 10%. Pero, también cabe mencionar que su ANN no utilizaba Gradient Deschent como su función de optimización, más bien, indicaban que Levenberg-Marquardt era la más utilizada en la literatura de ANN, esto podría afectar tanto positiva como negativamente los resultados obtenidos, al incrementar o disminuir la diferencia de precesión entre SVM y ANN. En la Tabla 17 y 18 se muestran los resultados de ANN y SVM respectivamente, para cada uno de sus productos, y se comparan con los resultados de esta investigación.

Es importante mencionar que estas comparaciones deben ser tomadas únicamente a nivel superficial, puesto que los autores utilizan un dataset diferente el cual no se encuentra disponible de forma pública.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Producto 1** | **Producto 2** | **Producto 3** | **Producto 4** | **Producto 5** | **ANN** |
| **MAPE** | 14% | 10% | 13% | 33% | 13% | 25% |

Tabla 17. Comparación de errores porcentuales de cada modelo ANN – Elaboración propia

Como podemos ver en la Tabla 17, con ANN, los resultados de su experimentación fueron mejores que la presente, si bien en ambos modelos se tiene la misma cantidad de capas ocultas, los autores utilizan una diferente cantidad de neuronas ocultas por cada producto: 5, 10, 10, y 5 para cada producto respectivamente. A comparación con el modelo óptimo de este artículo que utiliza 7 neuronas ocultas al encontrar que esta cantidad fue la mejor al realizar la optimización con GA.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Producto 1** | **Producto 2** | **Producto 3** | **Producto 4** | **Producto 5** | **SVM** |
| **MAPE** | 3% | 5% | 5% | 25% | 6% | 8% |

Tabla 18. Comparación de errores porcentuales de cada modelo SVM – Elaboración propia

En la Tabla 18, se muestran los resultados por cada uno de sus productos y el obtenido en esta investigación, como podemos observar, el Producto 1 (3%) es el mejor de todos los resultados, mientras que el Producto 4 (25%) es el peor. La principal diferencia con su modelo SVM es que utilizan un kernel Gaussiano a comparación del RBF que se emplea en este artículo. Esta diferencia de kernels podría ocasionar diferentes resultados, aunque Zhang, D., Xiao, J., Zhou, N., Zheng, M., Luo, X., Jiang, H., & Chen, K. (2015) indican que RBF es el mejor para problemas de regresión, debido a su buena capacidad de generalización.

En su investigación, el dataset que utilizaron fue dividido de la siguiente forma: 70% entrenamiento, 15% validación y 15% de prueba. Mientras que, en la presente, se utilizó 60% de los datos como entrenamiento, y 20% para validación y prueba respectivamente. Esta diferencia de distribución de dataset puede generar un modelo mejor entrenado; sin embargo, los autores no presentan curvas de pérdida o aprendizaje, donde se pueda evidenciar que el modelo efectivamente está aprendiendo y no experimentando overfitting. Si ese fuera el caso, entonces una mayor cantidad de datos de entrenamiento podría ser perjudicial para el modelo, puesto que aprendería los errores y los reflejaría al pronosticar sobre nuevos datos.

Con respecto a GBT, este modelo obtuvo resultados mejores a ANN, pero menores a SVR. Sin embargo, GBT y el concepto de boosting en general, es considerado una de las mejores y más poderosas técnicas de ML según Vineeth, V. S., Kusetogullari, H., & Boone, A. (2020). Por lo tanto, se tenían mayores expectativas con respecto a sus resultados. A pesar de ello, en su investigación, realizan una comparación entre múltiples técnicas de ML (Entre ellas SVR) de las cuales GBT resulta ser la peor de las cuatro técnicas utilizadas. Su trabajo, buscó realizar el pronóstico de ventas de partes de camiones, en la Figura 37 se muestran los resultados de sus diferentes modelos.

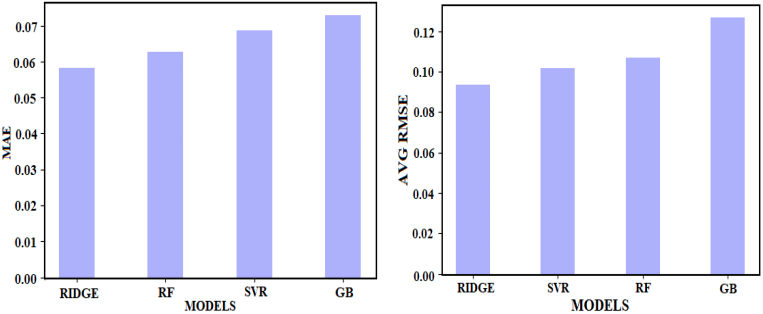


Figura 37. Errores MAE y RMSE para cada modelo - Vineeth, V. S., Kusetogullari, H., & Boone, A. (2020).

En la Figura 37, podemos ver los resultados para cada uno de los modelos: RIDGE, RF, SVR y GBT. En ambos gráficos GB (GBT) es el modelo que tiene el mayor error a comparación de los demás, y siendo RIDGE el mejor modelo en la experimentación que realizaron. Los autores también indicaron que sorpresivamente SVR tuvo un buen desempeño, mucho mejor que GB y Random Forest (RF), lo cual no esperaban. Más bien, las expectativas eran que GB y RF obtuvieran los mejores resultados, el cual no fue el caso, en realidad, todo lo contrario. Ellos indican que este comportamiento es consecuencia de problemas de overfitting y que en general es difícil seleccionar los mejores hiperparámetros si es que no se tiene un modelo dedicado a ello, como lo tiene el presente artículo utilizando Algoritmos Genéticos.

Durante esta sección se lograron encontrar diferentes razones por las cuales un modelo es mejor que otro, de la mano de diferentes investigadores que también experimentaron con resultados y modelos similares. Si bien cada una de las investigaciones expuestas, utilizaron un dataset diferente, la comparación y determinación de los mejores modelos es congruente con los de la presente investigación. Lo cual permite, hasta cierto grado, justificar los resultados obtenidos.

**CONCLUSIONES**

Las empresas, pequeñas, medianas o grandes, que se dediquen al comercio de productos son sumamente importante para la sociedad en la que vivimos actualmente. Puesto que no solo genera empleo para personas de todas las clases sociales, al representar el 25.8% de la Población Económicamente Activa (PEA), sino que también representa el 10.7% del PBI de la republica peruana al año 2019 según Bueno, D. (2019) para la página Perú Retail. Debido a esta gran importancia del retail en la sociedad, es que es necesario que estas organizaciones sepan afrontar el mayor problema de este rubro, el cual es conocido como la incertidumbre de la demanda. Esta incertidumbre, puede generar pronósticos o, peor aún, modelos predictivos defectuosos. Como Taylor, G. (2015) indica, esta es la principal causa del sobreabastecimiento, lo cual contrae costos de mantenimiento.

Los resultados obtenidos en esta investigación nos permiten afirmar que efectivamente se pueden utilizar técnicas de Machine Learning para realizar pronósticos adecuados de demanda, que puedan satisfacer el comportamiento fluctuante del consumidor. El cual es afectado por los Días Especiales y factores estacionales.

Adicionalmente, podemos concluir que existen múltiples hiperparámetros, o variantes a los modelos presentados, que se pueden implementar para obtener diferentes resultados. Si bien la estrategia de optimización utilizada (Algoritmos Genéticos) permitió obtener los mejores valores para ciertos hiperparámetros, existieron dificultades para seleccionarlos. Esto es debido a que no existe una regla que permita indicar qué hiperparámetros son más relevantes que otros, más bien, esto recae sobre el juicio del investigador. Es por ello que se tomó como referencia los hiperparámetros utilizados en la sección de Estado del Arte y Antecedentes.

Una futura investigación podría determinar si algunos hiperparámetros son más relevantes que otros, de modo que se pueda saber con certeza cuáles valen la pena optimizar y cuáles no, por su bajo nivel de impacto en el modelo.

Cada uno de los modelos implementados (SVM, ANN, GBT) presentó diferentes falencias al momento de pronosticar la demanda. El primero, SVM, no pronosticó los picos altos, pero sí tuvo un alto nivel de precisión cuando la demanda fue constante; El segundo, ANN, sí pronosticó los picos altos, pero no logró alcanzar su valor, para todos los picos siempre predijo un valor menor; El tercer modelo, GBT, presentó un comportamiento contrario a ANN, puesto que su pronóstico sobrepasaba los valores de picos, predijo valores más altos de lo real.

En una futura investigación se podría ejecutar los modelos utilizando diferentes atributos, aplicar Data Augmentation para incrementar la cantidad de datos con la que se entrenen los modelos, implementar una Red Neuronal de tipo Deep Learning, puesto que en esta investigación se utilizó una ANN con solo una capa oculta, una mayor cantidad de capas y neuronas, probablemente añaden el nivel de profundidad necesario para mejorar el modelo.

Finalmente, en un trabajo futuro, se podría desarrollar un Sistema de Soporte de Decisiones (SSD) empleando los modelos presentados, que permitan pronosticar la demanda de ciertos productos para una organización. Esto también permitiría evaluar el desempeño de los modelos frente a diferentes series de tiempo.

**REFERENCIAS**

Barrow, D. K., & Crone, S. F. (2016). Cross-validation aggregation for combining autoregressive neural network forecasts. *International Journal of Forecasting*, *32*(4), 1120-1137.

Boswell, D. (2002). Introduction to support vector machines. Departement of Computer Science and Engineering University of California San Diego.

Bueno, D. (2019, September 9). Negocio de retail representa el 10.7% del PBI del Perú. Perú Retail. https://www.peru-retail.com/negocio-de-retail-representa-el-10-7-del-pbi-peru/#:%7E:text=Actualmente%2C%20seg%C3%BAn%20el%20gremio%20Retail,Producto%20Bruto%20Interno%20(PBI).

Carbonneau, R., Laframboise, K., & Vahidov, R. (2008). Application of machine learning techniques for supply chain demand forecasting. European Journal of Operational Research, 184(3), 1140–1154. doi:10.1016/j.ejor.2006.12.004

Corsten, D., & Gruen, T. (2004). Stock-Outs Cause Walkouts. Retrieved 2 October 2019, from <https://hbr.org/2004/05/stock-outs-cause-walkouts>

Demir, L., & Akkaş, S. (2018). A comparison of sales forecasting methods for a feed company: A case study. Pamukkale University Journal of Engineering Sciences, 24(4).

Di Pillo, G., Latorre, V., Lucidi, S., & Procacci, E. (2016). An application of support vector machines to sales forecasting under promotions. 4OR, 14(3), 309–325. doi:10.1007/s10288-016-0316-0

Eckle, K., & Schmidt-Hieber, J. (2018). A comparison of deep networks with ReLU activation function and linear spline-type methods. Neural Networks. doi:10.1016/j.neunet.2018.11.005

Garrido, W. L. (2012). Máquinas De Soporte Vectorial (svm) Para La Detección De Nódulos Pulmonares En Tomografía Axial Computarizada (TAC). Bogotá: Pontificia Universidad Javeriana

Géron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow* [Ebook] (2nd ed.). Estados Unidos: O'Reilly Media.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT press.

Gov.UK (2020). *Working, jobs and pensions* [Listado ordenado de todos los días festivos de Inglaterra] Recuperado de https://www.gov.uk/bank-holidays.json.

Hinkelmann, K. (2016). *Neural Networks* [Ebook] (pp. 1-21). Suiza: University of Applied Sciences Switzerland. Retrieved from <http://didattica.cs.unicam.it/lib/exe/fetch.php?media=didattica:magistrale:kebi:ay_1718:ke-11_neural_networks.pdf>

Höschel, K., & Lakshminarayanan, V. (2018). Genetic algorithms for lens design: a review. Journal of Optics, 48(1), 134–144. doi:10.1007/s12596-018-0497-3

Huber, J., & Stuckenschmidt, H. (2020). Daily retail demand forecasting using machine learning with emphasis on calendric special days. International Journal of Forecasting. doi:10.1016/j.ijforecast.2020.02.005

Karlik, B., & Olgac, A. V. (2011). Performance analysis of various activation functions in generalized MLP architectures of neural networks. International Journal of Artificial Intelligence and Expert Systems, 1(4), 111-122.

Krauss, C., Do, X. A., & Huck, N. (2017). Deep neural networks, gradient-boosted trees, random forests: Statistical arbitrage on the S&P 500. *European Journal of Operational Research*, *259*(2), 689-702.

Kumar, M. K. (2020). *Online Retail Data v3* [Transacciones históricas diarias de clientes que compraron un obsequio de una tienda virtual.]. Recuperado de <https://www.kaggle.com/coldperformer/online-retail-data-v3/version/1>

Mishra, C., & Gupta, D. (2017). Deep Machine Learning and Neural Networks: An Overview. IAES International Journal Of Artificial Intelligence (IJ-AI), 6(2), 66. doi: 10.11591/ijai.v6.i2.pp66-73

Moon, M., Mentzer, J., Smith, C., & Garver, M. (1998). Seven keys to better forecasting. Business Horizons, 41(5), 44-52. doi: 10.1016/s0007-6813(98)90077-5

Rangel, H. R., Puig, V., Farias, R. L., & Flores, J. J. (2016). Short-term demand forecast using a bank of neural network models trained using genetic algorithms for the optimal management of drinking water networks. Journal of Hydroinformatics, 19(1), 1–16. doi:10.2166/hydro.2016.199

Samsudin, R., Shabri, A., & Saad, P. (2010). A comparison of time series forecasting using support vector machine and artificial neural network model. *Journal of applied sciences*, *10*(11), 950-958.

Slimani, I., El Farissi, I., & Achchab, S. (2015). Artificial neural networks for demand forecasting: Application using Moroccan supermarket data. 2015 15Th International Conference On Intelligent Systems Design And Applications (ISDA). doi: 10.1109/isda.2015.7489236

Stanford University. (s. f.). Multi Layer Neural Networks. Recuperado 14 de mayo de 2020, de <http://ufldl.stanford.edu/tutorial/supervised/MultiLayerNeuralNetworks/>

Taylor, G. (2015). Overstock Losses To Cost Retailers $471 Billion In 2015 - Retail TouchPoints. Retrieved 2 October 2019, from https://www.retailtouchpoints.com/topics/inventory-merchandising-supply-chain/overstock-losses-to-cost-retailers-471-billion-in-2015

Tsirikoglou, P., Abraham, S., Contino, F., Lacor, C., & Ghorbaniasl, G. (2017). A hyperparameters selection technique for support vector regression models. Applied Soft Computing, 61, 139–148. doi:10.1016/j.asoc.2017.07.017

Van Gerven, M., & Bohte, S. (2017). Artificial neural networks as models of neural information processing. Frontiers in Computational Neuroscience, 11, 114

Vineeth, V. S., Kusetogullari, H., & Boone, A. (2020, August). Forecasting Sales of Truck Components: A Machine Learning Approach. In 2020 IEEE 10th International Conference on Intelligent Systems (IS) (pp. 510-516). IEEE.

Wu, D. J., Feng, T., Naehrig, M., & Lauter, K. (2016). Privately evaluating decision trees and random forests. Proceedings on Privacy Enhancing Technologies, 2016(4), 335-355.

Xie, Y., Zhu, C., Lu, Y., & Zhu, Z. (2019). Towards Optimization of Boosting Models for Formation Lithology Identification. Mathematical Problems in Engineering, 2019, 1–13. doi:10.1155/2019/5309852

Yu, Y., Choi, T., & Hui, C. (2011). An intelligent fast sales forecasting model for fashion products. Expert Systems With Applications, 38(6), 7373-7379. doi: 10.1016/j.eswa.2010.12.089

Zhang, D., Xiao, J., Zhou, N., Zheng, M., Luo, X., Jiang, H., & Chen, K. (2015). A Genetic Algorithm Based Support Vector Machine Model for Blood-Brain Barrier Penetration Prediction. BioMed Research International, 2015, 1–13. doi:10.1155/2015/292683