Predicción de la demanda para empresa del sector retail

Realizado por:
Carlos Andrés Cuartas Murillo.
cacuartasm@eafit.edu.co

Carlos Alberto Cerro Espinal
cacerroe@eafit.edu.co

Daniel Román Ramírez
dromanr@eafit.edu.co

Daniel Enrique Pinto Restrepo
dpintor1@eafit.edu.co

Santiago Mejía Chitiva
smejiac3@eafit.edu.co

Proyecto integrador
Maestría en Ciencia de los Datos y Analítica
Universidad EAFIT
Medellín

Mayo 31 del 2020

1. Introducción

El proceso de cálculo de la previsión de la demanda en retail tiene por naturaleza una gran cantidad de datos que provienen de los puntos de venta y que deben depurarse antes de calcular las previsiones. En la actualidad algunas empresas han creado sus propios mecanismos para el cálculo de la demanda, sin embargo no siempre responden con la rapidez y exactitud que exige el mercado, perdiendo así oportunidades de venta, aumentando costos y afectando la calidad del servicio al cliente.

Por lo tanto este proyecto busca desarrollar por medio de **metodologías estadísticas y de aprendizaje automático**, modelos que realicen el pronóstico de la demanda con una mayor confianza.

Palabras clave: Pronosticar, modelo,ventas, demanda, aprendizaje, automático

2. Justificación

Se entiende que a partir de una correcta planeación de la demanda se pueden obtener optimizaciones de negocio como: reducción de ventas perdidas, disminución de inventarios obsoletos, definición de estratégica de precios, aumento del flujo de caja, satisfacción del cliente, entre otros. En ese sentido es importante que una empresa pueda implementar modelos confiables que permitan desarrollar estrategias a niveles tácticos y operativos que le brinden una ventaja competitiva en el sector que se desarrolla.

2.1. Objetivo general

Desarrollar un modelo para pronosticar de manera precisa la demanda de productos de una empresa del sector retail.

2.2. Objetivos Específicos

- Identificar que modelos son los recomendados para realizar el pronóstico de la demanda.
- Realizar un estudio descriptivo que permita identificar tendencias en los datos para la selección de las variables a utilizar en la matriz de covariables.
- Proponer al menos un modelo capaz de pronosticar la demanda con mayor precisión que los benchmark propuestos por la competencia de Kaggle.

3. Contexto general

El conjunto de datos utilizado para realizar el proyecto fue proporcionado por Walmart Inc., una compañía multinacional del sector de venta al detal para la competencia "M5 Forecasting"de **kaggle**. La base de datos brindada contiene información sobre las **ventas diarias de 3.049 referencias para 10 tiendas diferentes, ubicadas en los estados de California, Texas y Wisconsin**. Para cada referencia, se especificó la categoría a la que pertenece **(HOBBIES, FOODS ó HOUSEHOLD)** y dentro de dicha categoría el departamento correspondiente.

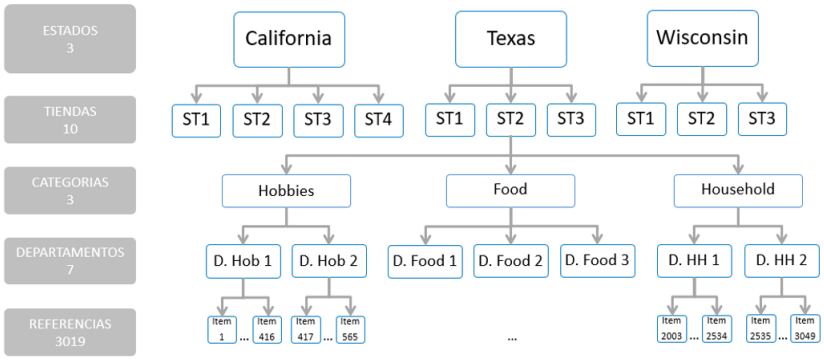


Figura 1: Esquema objetivo de estudio

El estudio descriptivo se realizó partiendo del entendimiento más general y agrupado del conjunto de datos, para luego bajar progresivamente a subconjuntos que permitieran evidenciar comportamientos más específicos para determinadas agrupaciones. Este entendimiento inicial proporcionará información relevante para el diseño de los modelos predictivos realizados en este proyecto, ya que mostrará tendencias relevantes en los datos y sugerirá variables de entrada importantes para ser utilizados en los mismos.

4. Análisis exploratorio

4.1. Análisis general del conjunto de datos agrupado

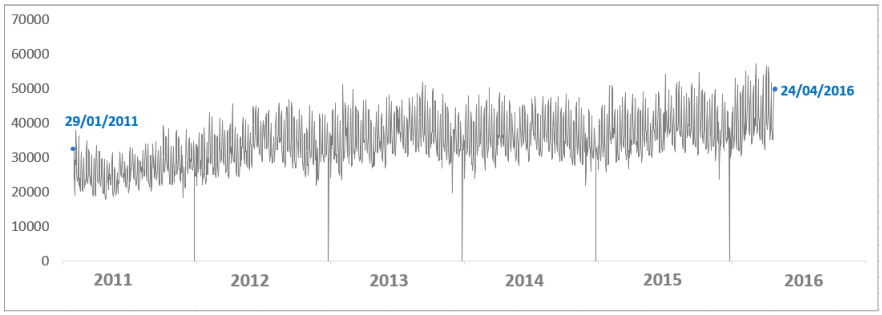


Figura 2: Demanda diaria

Los 1.913 días del conjunto de datos evidencian que la demanda promedio de unidades para la sumatoria de las 10 tiendas del estudio es de 34.341 unidades. Adicionalmente, se debe destacar que hay un día por cada año que presenta un valor atípico, donde la demanda total obtiene valores iguales o cercanos al cero. Esto sucede el 25 de diciembre, donde las tiendas están cerradas debido a las fiestas navideñas (En análisis posteriores se mostrará como ciertos eventos impactan positiva y negativamente en las ventas de las tiendas).

Por último, se evidencia un comportamiento cíclico a partir de las ventas de cada año, donde se puede ver un aumento progresivo de las unidades vendidas hasta aproximadamente septiembre, mes a partir del cual comienza un descenso hasta llegar a enero. A continuación, se muestra una gráfica suavizada con media móvil de 90 días, de forma que se pueda apreciar más fácilmente el fenómeno descrito.

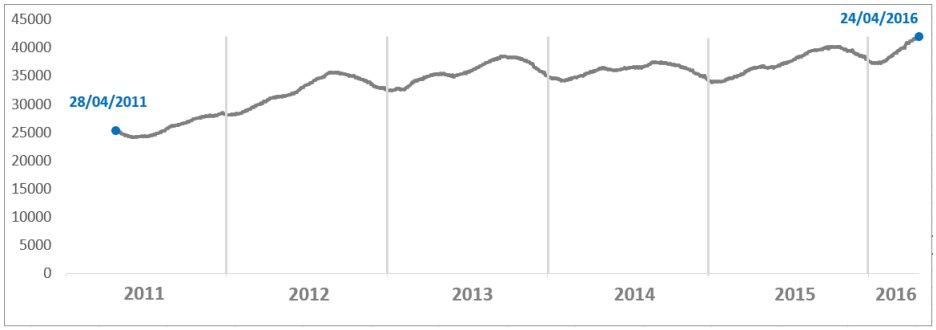


Figura 3: Gráfica suavizada con media móvil de 90 días

4.2. Análisis descriptivo por categoría

Boxplot de la demanda diaria por Categoría de referencia.

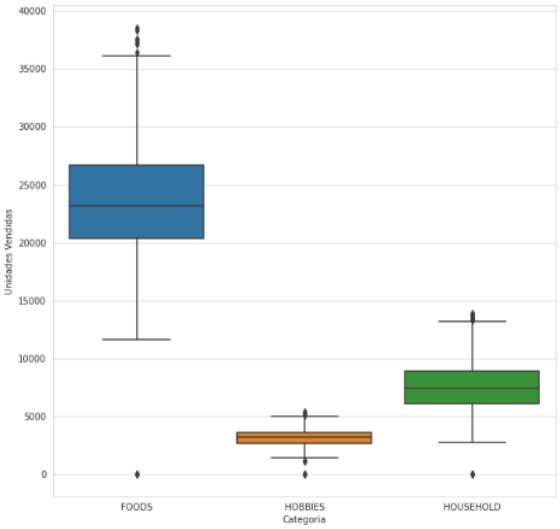


Figura 4: Demanda diaria por categoría

Se evidencia que la categoría con mayor demanda promedio por día y mayor varianza a su vez es la categoría FOODS. En segundo lugar, se encuentra la categoría "HOUSEHOLD" con una demanda promedio de 7.569 unidades vendidas por día para el acumulado de todas las tiendas del estudio. Por último, se encuentra la categoría HOBBIES, la cual tiene una demanda promedio diaria de 3.201 unidades. Del resultado anterior deriva información que es relevante para los modelos a implementar, ya que HOBBIES al ser la categoría con menor demanda promedio, será la que tenga mayor proporción de referencias con demanda intermitente (Referencias que tienen gran cantidad de días en las que no se venden unidades). En los capítulos siguientes de este trabajo, se explicarán las dificultades de realizar modelos de pronóstico para referencias que presentan este tipo de características.

Comportamiento de ventas en variables temporales.

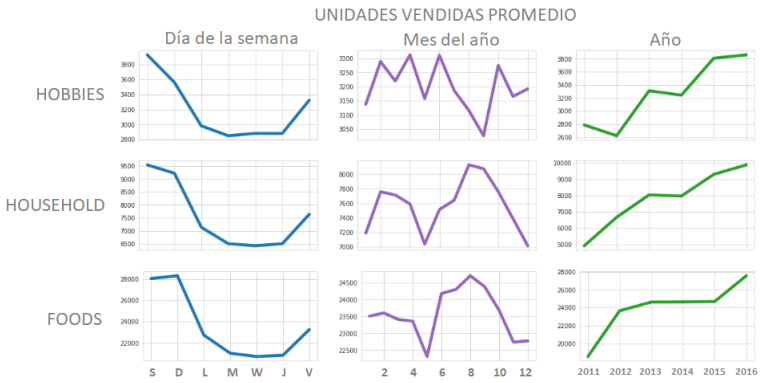


Figura 5: Comportamiento variables temporales

Como se observará en el desarrollo de los modelos propuestos en el proyecto, las variables temporales tendrán gran significancia como datos de entrada para el pronóstico de la demanda. Es por ello por lo que es de gran importancia entender como es el comportamiento de las categorías de los productos en diferentes ventanas de tiempo. A continuación, los hallazgos más relevantes:

- Día de la semana: Se observa que las tres categorías tienen un comportamiento muy similar para los días de la semana. Como era de esperarse, las ventas tienen su pico más elevado en los días del fin de semana, mientras que para los días de la semana la demanda se mantiene en los niveles más bajos (Excepto por el viernes donde la demanda empieza su crecimiento por ser la víspera del fin de semana). Por lo tanto, se evidencia que para gran cantidad de referencias la demanda se verá afectada dependiendo del día de la semana que se evalúe.

- • Mes del año: Se evidencia un comportamiento muy similar para las categorías FOODS y HOUSEHOLD, en la que se observan descensos importantes de la demanda para mayo, octubre y noviembre, y picos altos de demanda para los meses comprendidos entre junio y septiembre. Sin embargo, lo más llamativo es que para los meses de agosto, septiembre y octubre, el comportamiento de la demanda es totalmente contrario para la categoría HOBBIES. Esto será relevante en el momento de evaluar el comportamiento de los modelos predictivos agrupados por categoría, ya que el resultado debe modelar estas realidades adecuadamente para los resultados

- Año: El comportamiento más notorio en las gráficas de las ventas en unidades promedio por año, es el crecimiento progresivo para las tres categorías en la mayoría de los años. Esto será relevante para los modelos a elaborar, ya que se debe tener en cuenta la proporcionalidad del crecimiento en la demanda para los mismos días de cada mes en los diferentes años.

Análisis descriptivo por tienda

- Demanda móvil de 90 días de las unidades vendidas por tienda.

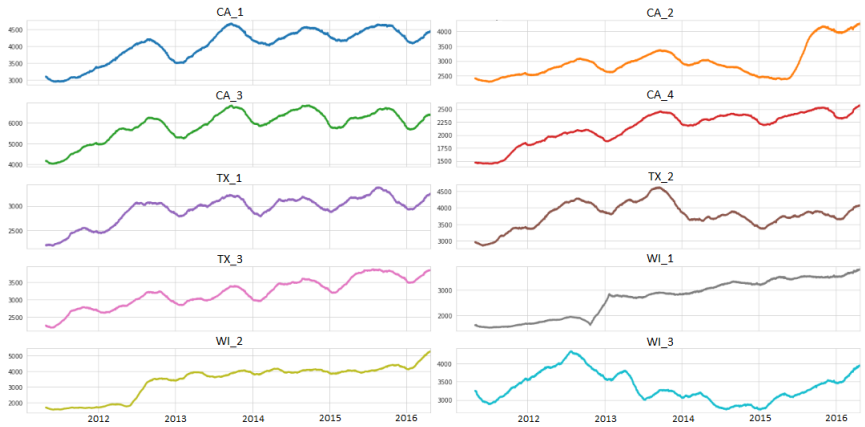


Figura 6: Demanda móvil de 90 días por unidades vendidas en cada tienda

Se utilizó una media móvil de 90 días para la demanda analizada por tienda, de modo que se pudiera suavizar la curva que describía el comportamiento de las unidades vendidas. Se pueden evidenciar varios aspectos interesantes a ser tenidos en cuenta para la ejecución de los modelos:

- Hay tiendas con cambios abruptos positivos en el comportamiento de la demanda. Tiendas como WI_2, CA_2 y WI_1 tuvieron un aumento de las unidades vendidas poco usual si se compara con el resto de las tiendas. Esta evidencia podría ser un factor relevante para decidir el conjunto de datos con el que se entrenará los modelos de pronóstico.

- Contrario al fenómeno explicado en el punto anterior, hay dos tiendas (WI_3 y TX_2) que reflejan disminuciones abruptas en el comportamiento de la demanda. Similar al caso anterior esta será información relevante al momento de diseñar los modelos.

- Se evidencia un claro comportamiento estacional para las tiendas que no tienen cambios abruptos en el comportamiento de la demanda. Se puede ver que las tiendas CA_21, CA_23, CA_24, TX_21, y TX_22 se presenta un crecimiento continuo de las unidades vendidas desde el mes de febrero hasta aproximádamente los meses de agosto y septiembre, donde se llega al pico más alto de ventas. Luego de ellos en el cuarto trimestre se nota un decrecimiento continuo de las misma (Este comportamiento es similar al identificado en la gráfica de ventas generales).

-Demanda por tienda en el rango de tiempo total del conjunto de datos.

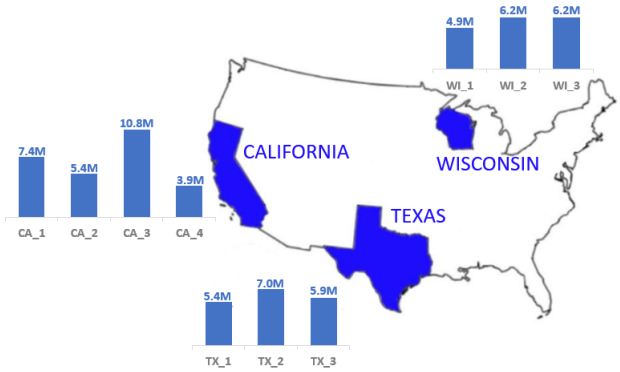


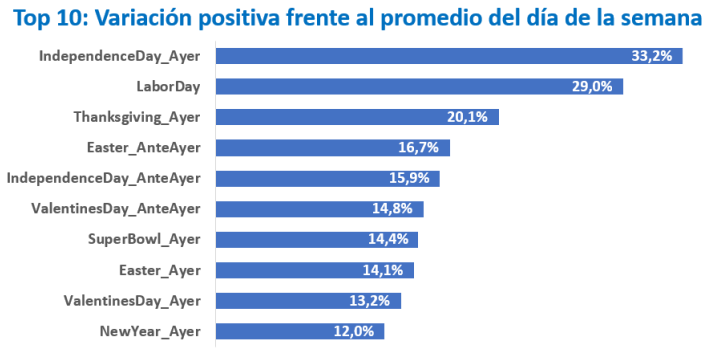
Figura 7: Demanda en cada una de las tiendas en el país

El gráfico anterior evidencia rápidamente que las tiendas con mayor cantidad de demanda pertenecen al estado de California. Contrariamente, se puede notar que el promedio de ventas por tienda en el estado de Wisconsin es el más bajo de los tres Estados y a su vez el que mayor cantidad de unidades vendidas similares tiene para sus tiendas, ya que es el que menor variación refleja (La desviación estándar es de 0,75 frente 0,81 de Texas y 2,98 de California). Por otro lado, el gráfico anterior expone las ventas por tienda destacando la distancia geográfica que hay entre los estados. Esto debido a que dicha diferencia territorial puede impactar en la demanda de ciertos productos específicos. Por ejemplo, es probable que la venta de bebidas frías sea más alta para las tiendas del Estado de Texas que para las tiendas del estado de Wisconsin, ya que la temperatura promedio de Wisconsin es menor a la del estado de Texas.

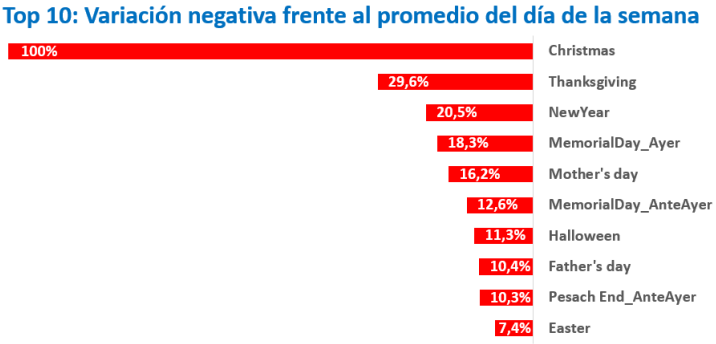
Análisis del impacto en la demanda por eventos

Se realizó un enriquecimiento de la información histórica de la demanda a partir de la asociación de eventos relevantes para los días del estudio. Para ellos se agregaron eventos de cuatro categorías diferentes: Eventos Deportivos, Culturales, Nacionales y Religiosos. El estudio descriptivo evaluó la influencia en la demanda para dos días anteriores al día del evento, para el día anterior al día del evento y para el día actual del día del evento. De esta forma, se comparó para cada uno de los tiempos anteriormente descritos, cuanto porcentaje se aumentaba o disminuía la demanda de dicho día respecto al promedio del consumo histórico del mismo. A continuación, se muestran los resultados de los eventos que mayor aumento porcentual generaron y también para los que mayor disminución porcentual impactaron las ventas:

Se evidencia que los impactos más relevantes en las ventas se dan por lo



general en el día previo al evento. También se observa que en este Top 10 hay eventos de las cuatro categorías (Religioso, Deportivo, Nacional y cultural) por lo que no necesariamente el tipo de evento sea la razón principal que impacte el aumento de la demanda, sino más bien el evento específico.



En cuanto a los eventos que influencian negativamente la demanda de unidades de las ventas, se puede observar que gran proporción de dichos casos ocurren el mismo día en que se realiza el evento. El caso más extremo, como se evidenció en el primer gráfico del estudio descriptivo, es "Christmas" (Navidad) día en el que las tiendas son cerradas al público.

5. Procedimiento ETL

5.1. Extracción

Los datos de insumo están almacenados en **KAGGLE** y desde allí se realizó el proceso de extracción:

Ubicación de los insumos

https://www.kaggle.com/koralturkk/m5-comptetition/data

Data insumo

- *calendar.csv* :Contiene información sobre las fechas en que se venden los productos. Contiene 10 filas por 14 columnas
- *sales_train_validation.csv* : Contiene los datos históricos de ventas de unidades diarias por producto y tienda [d_1 - d_1913]. Contiene 10 filas y 1919 columnas
- *sample_train_validation.csv* : Contiene el formato correcto para los envíos. Contiene 10 filas por 29 columnas
- *sell_prices.csv* :Contiene información sobre el precio de los productos vendidos por tienda y fecha. Contiene 4 filas por 4 columnas

Los datos fueron descargados por medio de herramientas analíticas **R studio y Python,** y allí construyeron algoritmos para la obtención de los datos desde su instancia.

5.2. Transformación

Procesos de JOIN:

Luego de tener los insumos en el directorio de trabajo se empieza con el proceso de transformación de los datos.

Para el desarrollo de transformación se realizan los algoritmos en Python y R- studio ya que en cada aplicación se realizan diferentes metodologías de análisis

sales _train_validation y calendar:

Estos dos insumos se cruzan por medio del id de las semanas y los días con el fin de crear **eventos X demanda**, es decir la demanda ofertada por cada día y semana del año

eventos X demanda y sell _prices:

Luego de tener la demanda por cada evento y por día - semana se quiere traer la información de los estados y los nombres de las tiendas para consolidar toda la información y crear la base final para la aplicación de las metodologías analíticas.

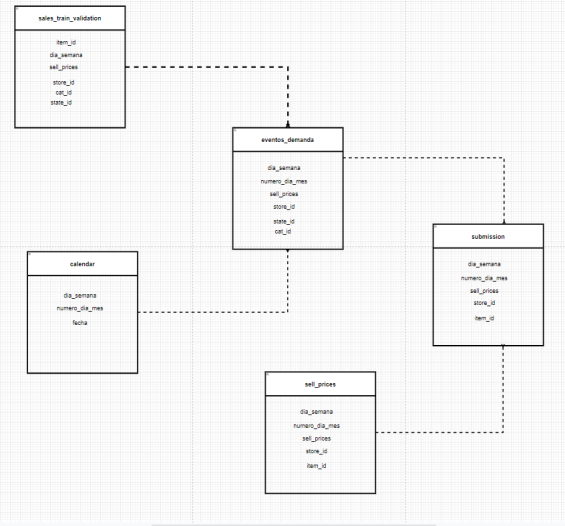


Figura 8: Transformacion

5.3. Carga de datos y resultados

Luego de tener la base preparada se aplican todas las metodologías de analítica en las aplicaciones ya mencionadas y finalmente los resultados se cargan en la plataforma de kaggle para recibir los resultados

Diagrama de ETL

El siguiente diagrama expone todo el proceso de ETL del proyecto

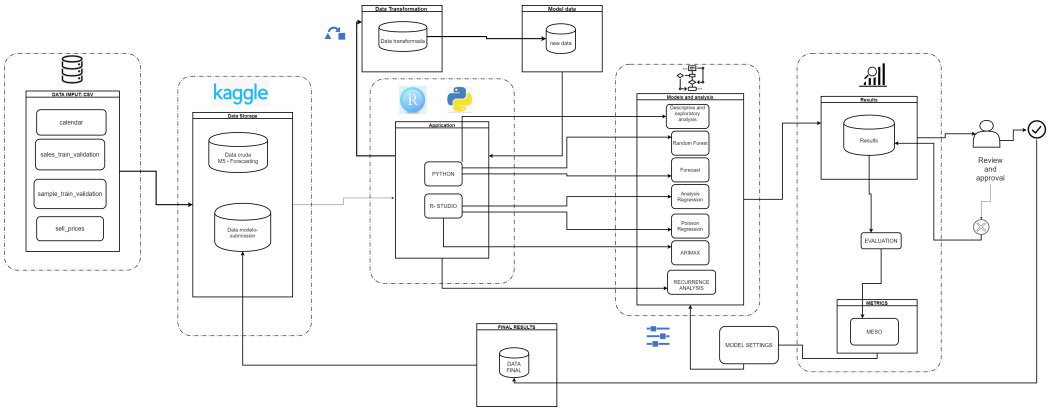


Figura 9: Diagrama ETL

6. Metodologías aplicadas

Para tener mejores resultados, inicialmente se aplica las metodologías a unas referencias con el fin de ver su comportamiento y si finalmente los resultados son óptimos se despliegan al resto de las referencias.

6.0.1. Regresión lineal

La regresión lineal es un modelo matemático usado para aproximar la relación de dependencia entre una variable dependiente Y, las variables independientes X_i y un término aleatorio ϵ (1) es decir:

$$Y_t = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p + \epsilon$$

Se tiene para este caso:

- Y_i : *Demanda* Variable dependiente
- $X_1, X_2, ..., X_p$: Variables explicativas que inciden en la oferta de demanda en cada sede de la tienda para cada artículo
- $\beta_0, \beta_1, \beta_2, ..., \beta_p$: Parámetros, miden la influencia que las variables explicativas tienen sobre la variable dependiente

β_0 es la intersección, las $\beta_i, (i > 0)$ son los parámetros respectivos a cada variable independiente, y p es el número de parámetros independientes a tener en cuenta en la regresión.

Desarrollo

Se requiere analizar el efecto de variables transaccionales y de tiempo que tiene sobre la demanda para el departamento de comida para la compañia comercializadora Walmart

Para llevar acabo los análisis se hacen los siguientes pasos:

- Extracción de los datos
- Exploración de datos
- Limpieza de los datos
- Estructuración de covariables (base para el modelo)
- Análisis de correlación
- Análisis del comportamiento de la variable respuesta
- Aplicación del modelo de regresi'on
- Iteraciones para la selección del mejor modelo aplicando conceptos de ingeniería de características

- Comprobación de pruebas de normalidad
- Predicción de la demanda
- Comparación de la prediccón resultante vs los valores reales

Inicialmente se propone el modelo de regresión pronósticando la demanda para el mes de febrero del año 2012, por la razón de que se quiere observar el comportamiento inicial del modelo con una muestra de la base.(2)

Tendencia de la demanda para el mes de febrero

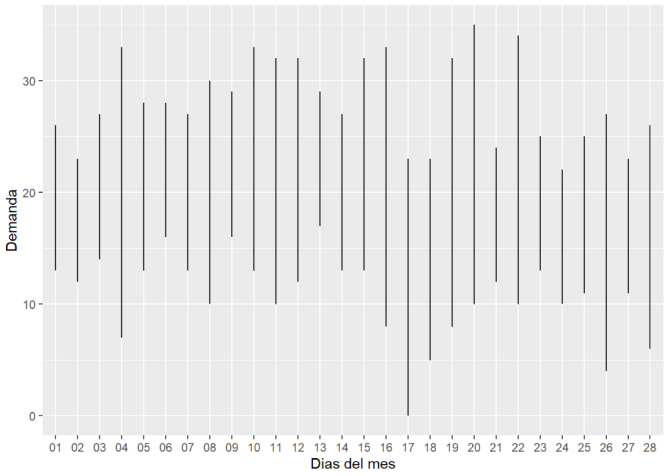


Figura 10: Tendencia febrero

Análisis de correlación

Se realiza un análisis de correlación uno a uno con toda la matriz de covariables y se evidencia que no hay relaciones entre ella significativas y se procede aplicar el modelo de regresión.

	y_demand_target	wm_yr_wk	year_2012	year_2013	year_2014	year_2015	year_2016	day_month_01
y_demand_target	1.0000000	-0.0011675	-0.0067545	-0.0145482	-0.0197440	0.1127486	-0.0717019	0.0327977
wm_yr_wk	-0.0011675	1.0000000	-0.7072834	-0.3522763	-0.0013148	0.3531865	0.7076879	-0.0025499
year_2012	-0.0067545	-0.7072834	1.0000000	-0.2500000	-0.2500000	-0.2500000	-0.2500000	0.0000000
year_2013	-0.0145482	-0.3522763	-0.2500000	1.0000000	-0.2500000	-0.2500000	-0.2500000	0.0000000
year_2014	-0.0197440	-0.0013148	-0.2500000	-0.2500000	1.0000000	-0.2500000	-0.2500000	0.0000000
year_2015	0.1127486	0.3531865	-0.2500000	-0.2500000	-0.2500000	1.0000000	-0.2500000	0.0000000
year_2016	-0.0717019	0.7076879	-0.2500000	-0.2500000	-0.2500000	-0.2500000	1.0000000	0.0000000
day_month_01	0.0327977	-0.0025499	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	1.0000000
day_month_02	-0.0287980	-0.0022774	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	-0.0370370
day_month_03	0.0719949	-0.0022774	0.0000000	0.0000000	0.0000000	0.0000000	0.0000000	-0.0370370

Figura 11: Vista previa extraída de (2)

Aplicación del modelo de regresión en R

Se sabe que la variable respuesta es la demanda, por lo tanto se desea conocer las posibles variables que podrían afectar el comportamiento de esta en todos los períodos de cada año (3)

Los modelos se ejecutan en R studio.

Modelo 1

El primer modelo de regresión se ejecuta por medio de la función en R (4)

lm los resultados fueron los siguientes:

- $R^2 = 0{,}65$

Posteriormente se ejecutan las **pruebas de normalidad del modelo:**
Gráficamente se evidencia que cumple con las pruebas de normalidad.

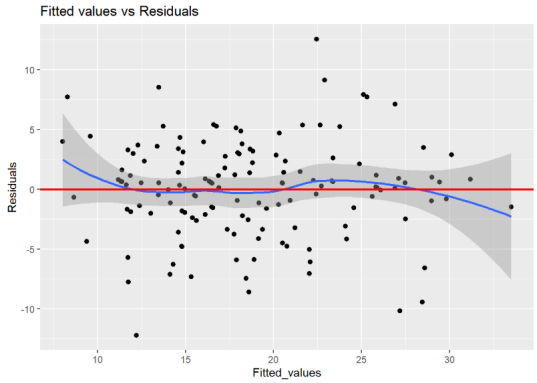


Figura 12: Valores ajustados

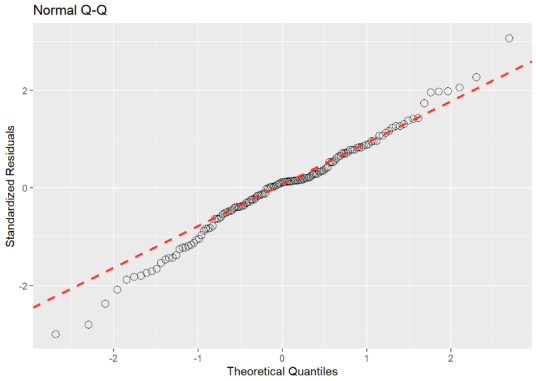


Figura 13: Normal QQ

"Backward elimination"por AIC

Para seleccionar el mejor modelo y tener una prediccón mas efectiva realizamos "backward elimination":(2)

- En este caso se comienza con el modelo completo y en cada paso se va eliminando una variable.
- Si todas las variables predictoras son importantes, es decir tienen p-value pequeños para la prueba t, entonces el mejor modelo es el que tiene todas las variables predictoras disponibles.
- En caso contrario, en cada paso la variable que se elimina del modelo es aquella que satisface cualquiera de los siguientes requisitos equivalentes entre sí.

- Aquella variable que tiene el estadístico de t, en valor absoluto, más pequeño entre las variables incluidas aún en el modelo
- Aquella variable que produce la menor disminución en el R^2 al ser eliminada del modelo.
- Aquella variable que tiene la correlación parcial (en valor absoluto) más pequeña con la variable de respuesta, tomando en cuenta las variables aún presentes en el modelo. El proceso termina cuando se llega a un modelo con un número prefjado p de variables predictoras.

Aplicando esta metodología se obtiene finalmente el **modelo resultante** con los siguientes resultados(3)

- $R^2 = 0{,}65$

Se evidencia que el R^2 no cambió mucho con respecto al primer modelo dado que la variable respuesta tiene muchos ceros y no es una variable continua.

Predicción de la demanda vs real

Se realiza la predicción de la variable respuesta con la función **predict** y posteriormente la comparamos con respecto a los valores reales (2)

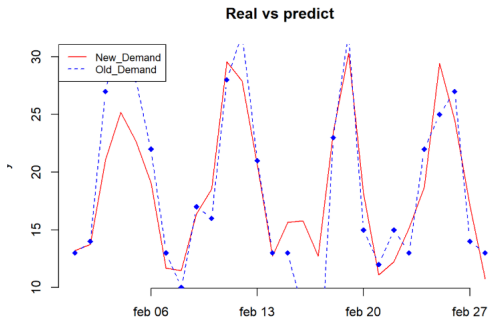


Figura 14: Real vs predicción

Aunque se podría mal interpretar como una buena predicción se observa que aún el modelo se ve sobreajustado y tiene problemas con la variable respuesta dado al gran cantidad de ceros que tiene esta, el modelo de regresión lineal no es óptimo para este tipo de variable es por esto que se aplicarán otras metodologías que serán explicadas a lo largo del documento

6.0.2. Aplicación de regresión de Poisson inflada a cero

La regresión de Poisson inflada a cero se utiliza para modelar datos de conteo que tienen un exceso de conteos cero. Además, la teoría sugiere que los ceros en exceso se generan mediante un proceso separado de los valores de conteo y que

los ceros en exceso se pueden modelar de forma independiente. Por lo tanto, el modelo zip tiene dos partes, un modelo de conteo de Poisson y el modelo logit para predecir el exceso de ceros (5).

Modelo Poisson para predicción de la demanda para empresa del sector retail
Se requiere analizar el efecto de variables transaccionales y de tiempo que tiene sobre la demanda para la compañia Walmart, el efecto de este estudio es poder tener la demanda necesaria para las temporadas durante el año de ventas. En este caso tenemos un conteo en la variable respuesta demanda se evidencia una gran existencia de zeros que indica los días que no se tuvo demanda.

La distribución de Poisson es una distribución de probabilidad discreta que expresa, a partir de una frecuencia de ocurrencia media, la probabilidad de que ocurra un determinado número de eventos durante cierto período de tiempo. Concretamente, se especializa en la probabilidad de ocurrencia de sucesos con probabilidades muy pequeñas, o sucesos raros".(6)

Propiedades

A continuación se identifica la función de densidad de probabilidad de Poisson.(6)

$$f(k, \lambda) = \frac{e^2 \lambda^k}{k!}$$

- k es el número de ocurrencias del evento o fenómeno (la función nos da la probabilidad de que el evento suceda precisamente k veces).
- λ es un parámetro positivo que representa el número de veces que se espera que ocurra el fenómeno durante un intervalo dado. Por ejemplo, si el suceso estudiado tiene lugar en promedio 4 veces por minuto y estamos interesados en la probabilidad de que ocurra k veces dentro de un intervalo de 10 minutos, usaremos un modelo de distribucińn de Poisson con $\lambda = 10x4 = 40$
- e es la base de los logaritmos naturales ($e = 2,71828...$)

La función generadora de momentos con valor esperado λ es:

$$\mathrm{E}\left(e^{tX}\right) = \sum_{k=0}^{\infty} e^{tk} f(k;\lambda) = \sum_{k=0}^{\infty} e^{tk} \frac{\lambda^k e^{-\lambda}}{k!} = e^{\lambda(e^t-1)}.$$

Figura 15: Función generadora de momentos

las variables aleatorias de Poisson tienen la propiedad de ser infinitamente divisibles.

La divergencia Kullback-Leibler (5)desde una variable aleatoria de Poisson de parámetro λ_0 a otra de parámetro λ es

$$D_{\mathrm{KL}}(\lambda||\lambda_0) = \lambda\left(1 - \frac{\lambda_0}{\lambda} + \frac{\lambda_0}{\lambda}\log\frac{\lambda_0}{\lambda}\right).$$

Figura 16: Divirgencia de Kullback-leiber

Definición de intervalos de confianza

Dada una serie de eventos k (al menos el 15-20) en un periodo de tiempo T, los límites del intervalo de confianza para la frecuencia vienen dadas por:

$$F_{low} = \left(1 - \frac{1.96}{\sqrt{k-1}}\right)\frac{k}{T}$$
$$F_{upp} = \left(1 + \frac{1.96}{\sqrt{k-1}}\right)\frac{k}{T}$$

Figura 17: Intervalos de confianza

entonces los límites del parámetro λ están dados por:
$\lambda_l ow = F_{low}T; \lambda_{upp} = F_{upp}T \lambda_{low} = F_{low}T; \lambda_{upp} = F_{upp}T$

Desarrollo

Inicialmente se quiere observar el comportamiento de la demanda en una referencia **FOOD** ya que si se evidencia un buen comportamiento en el modelo se puede desplegar para todas las referencias de las tiendas.
$y = Demanda$

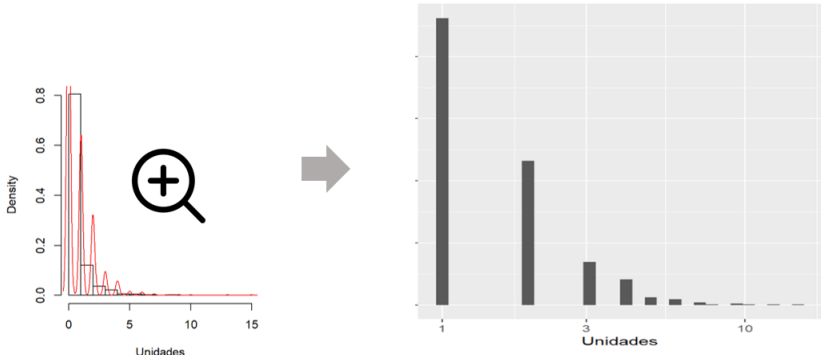


Figura 18: Comportamiento de la demanda

Aplicación del modelo Poisson

Para el la aplicación del modelo Poisson a la referencia FOOD se utilizó R-studio y la función **glm, family = poisson** y la librería **pscl** (6)

finalmente se ajustaron 4 modelos, los cuales se calificaron por AIC y MASE (7):

Se aplicó esta metodología inicialmente a una referencia: FOODS

	AIC	Residual deviance	MASE
Modelo 1	5025	2933	1.1269
Modelo 2	4977	2947	1.1269
Modelo 3	4977	2947	1.1269
Modelo 4	4977	2947	1.1269

Figura 19: Calificación

Comparación de modelos

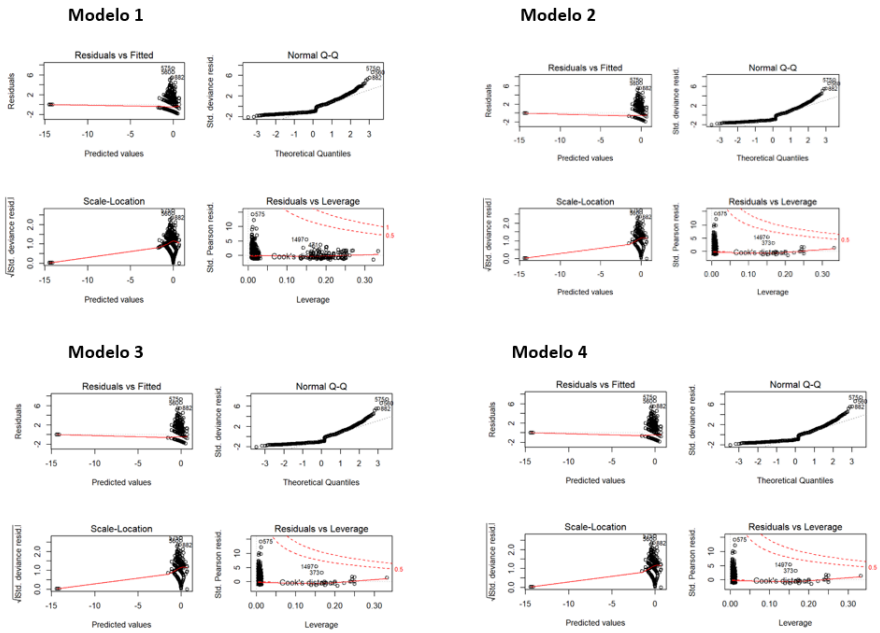


Figura 20: Comparación modelos

Como se evidencia no hay mejoramiento en ninguno de los modelos a partir de la primera iteración, por lo tanto se concluye que esta metodología no es tan óptima para la predicción de la demanda.

6.0.3. ARIMAX

Contexto

El problema a resolver tiene una clara aproximación, series de tiempo jerárquicas, esto quiere decir que nos encontramos ante un problema de series de tiempo en diferentes niveles de agregación. Lo primero que hay que aclarar es que en la tienda de retail se venden 3049 referencias, dichas referencias pertenecen a 7 departamentos (sub-categorías) y dichos departamentos pertenecen a 3 categorías (Household, Hobbies y Foods). Ahora bien, dichas referencias se venden en 10 tiendas diferentes, distribuidas en los Estados de California(4), Texas(3) y Wisconsin(3). Esto genera que las 3049 referencias a pronosticar se multipliquen por 10, esto nos deja con un total de 30490 series de tiempo.

Hay 3 aproximaciones para las series de tiempo jerárquicas; la primera es la de **bottom-up**, que se centra en pronosticar las series a su nivel más bajo de desagregación, esto quiere decir que tendríamos que estimar 30490 series de tiempo; la segunda es la de **top-down** que se encarga de predecir el nivel más alto de la jerarquía y luego se desagrega por unas proporciones históricas a los niveles más bajos, en este sentido si utilizáramos este método estaríamos prediciendo las ventas totales de retail y repartiendo por una proporción histórica a cada uno de los items; la última aproximación, y es la que utilizamos para desarrollar el modelo Arimax, es la de **midle-out** que combina las dos anteriores, en este caso se escoge un nivel intermedio, en nuestro caso los departamentos, y lo que se hace es que se pronostica ese nivel intermedio y se utiliza tanto para desagregar a los niveles más bajos como para sumar y llegar a la serie total.

Al tomar la agregación por departamento, esto nos deja un total de 70 series para pronosticar, 7 departamentos por 10 tiendas. La metodología utilizada para modelar cada una de las series fue la de Arimax. A continuación haremos una explicación del modelo.

Metodología

Primero es importante recalcar el concepto de estacionariedad. Una serie estacionaria es en la cual sus propiedades no depende en el tiempo que se observe la serie, por lo tanto series con tendencia o con estacionalidad, no son series estacionarias. Transformaciones como los logaritmos pueden ayudar a estabilizar la varianza de las series y la **diferenciación** (**?**) puede ayudar a estabilizar la media de las series de tiempo removiendo cambios en el nivel de la series de tiempo, y por lo tanto ayuda a eliminar o reducir la tendencia o la estacionalidad.

Luego nos centraremos en el concepto de modelos autoregresivos, en dichos modelos, pronosticamos la variable de interés usando una combinación lineal de los valores pasados de la misma. El termino autoregresivo indica que es una regresión de la variable vs ella misma. Así, un modelo autoregresivo de orden p

puede ser escrito como:

$$y_t = c + \phi_1 y_{t-1} + \phi_2 y_{t-2} + \cdots + \phi_p y_{t-p} + \epsilon_t$$

Donde ϵ_t es ruido blanco. Esto nos indica que el modelo autoregresivo es como una regresión múltiple pero con valores rezagados de la variables y_t como predictores. Así, nos referimos a un modelo AR(p), como un modelo autoregresivo de orden p

El paso siguiente es describir los modelos de media móvil, en este caso en vez de utilizar los valores pasados de la variable en cuestión en una regresión, un modelo de media móvil usa los errores de pronostico pasado en un modelo de regresión.

$$y_t = c + \epsilon_t + \theta_1 \epsilon_{t-1} + \theta_2 \epsilon_{t-2} + \cdots + \theta_q \epsilon_{t-q}$$

Nos referimos a un modelo MA(q), como un modelo de media móvil de orden q. Por último, se puede observar que el valor de y_t puede ser pensado como un una media móvil ponderada de los errores de pronóstico pasados.

Obsérvese que los modelos AR, MA y la integración componen las primeras partes del modelo ARIMAX. AR es la parte autoregresiva del modelo, MA es la parte de media móvil, y la I representa el orden de integración que debe tener la serie para que sea estacionaria. De esta forma, solo faltaría describir el componente X, dicho componente representa variables adicionales en el modelo, por ejemplo, como estamos pronosticando la demanda una variable relevante es el precio y dicha variables se puede agregar en un modelo ARIMAX. En nuestro caso, debido a la agregación de las series, no se pudo incluir la variable precio ya que, la serie no hablaba de una sola referencia, sin embargo, pudimos agregar variables de las fechas especiales, los días de la semana, los meses del año y los días del mes.

Dicha metodología ARIMAX fue realizada para cada una de las 70 series de tiempo agregadas. Es pertinente mencionar que para realizar los modelos, utilizamos el comando auto.arima() en el software estadístico R, dicha función usa una variación del algoritmo de Hyndman-Khandakar (**?**), que combina pruebas de raíces unitarias (Para corroborar estacionariedad de la series) y una minimización de AICc y de la máxima verosimilitud.

Luego de pronosticar las 70 series de tiempo de los departamentos por tienda, pasamos a pronosticar el nivel más bajo de la jerarquía de la siguiente manera. Para cada uno de los items, obtuvimos la participación histórica de cada uno dentro del departamento y la tienda correspondiente y, multiplicamos la serie pronosticada por dicha proporción para obtener el pronóstico a nivel de referencia.

Resultados

Para revisar los resultados utilizamos 3 métricas de calidad, la primera es el **MAE** que se refiere a la desviación absoluta y es muy útil cuando se tiene buen conocimiento de las magnitudes de la series, la segunda es el **MAPE** que es la devisación porcentual absoluta media y da una noción en que porcentaje de error estuvo el pronóstico vs los valores reales; por último, se encuentra el **MASE** que mide el error absoluto escalado, el uso de esta métrica es muy útil precisamente cuando tenemos series de tiempo intermitentes.

Debido a la magnitud de los resultados, solo se observaran algunos resultados generales. Los primeros se refieren a los 70 modelos generados y los segundos a los resultados a nivel de item.

- Resultados a nivel de departamento y tienda:
  - 56 de los 70 modelos están entre un 5 y un 20 por ciento de MAPE.
  - El mejor ajuste fue para el departamento de household_1 en la tienda California 3.
  - El peor ajuste fue para el departamento foods_2 de California 2.
- Resultados a nivel de item por tienda:
  - No se puede calcular el MAPE para un total de 29420 series, lo que podría indicar que este es el nivel de series intermitente en los datos
  - Dentro de las métricas de interés se encuentra el MASE, la interpretación de esta métrica es que si MASE >1 implica que el pronóstico real es peor fuera de la muestra que un pronóstico ingenuo en la muestra, en términos de error absoluto medio.
  - Hay 18580 series que tienen un MASE <1.
  - El mejor desempeño en términos de MAPE es para el Item FOODS_3_377 en la tienda de Texas 3 con un resultado del 13 % de error porcentual medio.

6.1. Redes neuronales

En esta parte del trabajo se exploran dos acercamientos diferentes al uso de redes neuronales recurrentes (RNN) para el pronóstico de las ventas, el primero es un híbrido entre los modelos LSTM (redes neuronales recurrentes) y Random Forest (Aprendizaje supervisado) y el segundo es un modelo LSTM más simple que incluye dentro de los datos de entrada variables explicativas externas

Método híbrido LSTM + RF

Esta metodología está basada en "Deep learning with long short-term memory networks and random forests for demand forecasting in multi-channel retail"(8), en la cual los autores proponen un método para pronosticar series

de tiempo que incluyen variables explicativas, como promociones, eventos, fechas especiales, entre otras, y además prueban los resultados con datos reales de una empresa del sector retail, lo cual se adapta bien al trabajo que estamos desarrollando. La propuesta consiste de 3 partes:

1) Realizar pronósticos con LSTM utilizando solo los datos históricos de ventas, es decir la serie de tiempo como tal, sin tener en cuenta variables explicativas.

2) Calcular los errores entre las predicciones realizadas en el paso 1 y los valores reales de la demanda. Estos errores en teoría representan la variabilidad no explicada por el modelo LSTM. Luego, a través de un modelo supervisado de Random Forest, tratar de pronosticar estos errores usando las variables explicativas excluídas en el paso 1, como los eventos, promociones, cambios de precio, fines de semana, etc.

3) Finalmente, para obtener las predicciones finales, se realiza la agregación de estos dos pronósticos, es decir, se suman el pronóstico de ventas por LSMT realizado en el paso 1 y el pronóstico de los errores realizado con Random Forest en el paso 2.

LSTM (Long-Short Term Memory)

Las redes LSTM han mostrado recientemente resultados promisorios en tareas de predicción de series de tiempo (9). Las redes LSTM son capaces de trabajar bien en series de tiempo lineales y no lineales (10). Por lo tanto no se requiere la descomposición de la serie de tiempo en componentes lineales y no lineales.

Las redes LSTM pertenecen a la clase de redes neuronales recurrentes (RNNs). RNNs tienen la propiedad de persistencia de la información, reteniendo el estado de las variables a través de los "time steps" (11) RNN solo pueden manejar dependencias de corto plazo porque sufren del problema "vanishing gradient". Las redes LSTM, por otro lado, tienen la capacidad de aprender dependencias de largo plazo (12)

La arquitectura de las redes LSTM tienen tres tipos de capas:

1. Una capa de entrada con un número de neuronas igual al número de variables de entrada.

2. Una o múltiples capas ocultas.

3. Una capa de salida con un número de neuronas igual a las variables de salida.

Las capas ocultas de una red LSTM consisten en una célula de memoria. Las redes LSTM son superiores a las RNN estándar debido a la presencia de esta

célula de memoria, la cual ayuda a retener información a través de los "time steps", ya que esto no era posible en redes neuronales previas.

La estructura de la célula de memoria tiene tres tipos de compuertas:
1. Forget(ft)
2. Input(it)
3. Output(ot)

En la siguiente imagen, se puede apreciar la estructura de una célula de memoria.

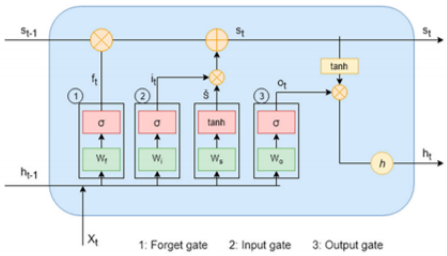


Figura 21: tomada de: As more layers using certain activation functions are added to neural networks, the gradients of the loss function approaches zero, making the network hard to train

En la célula de memoria, en cada "time step" t, la entrada consiste en un elemento de la secuencia de entrada (Xt) y la salida del "time step" previo (ht-1). En el estado de la célula "t":

a) La compuerta "forget" toma estas entradas y decide sobre qué información será removida de la memoria.

b) La compuerta "input" decide que información deberá adicionarse a la memoria.

c) La compuerta "output" decide la salida del bloque de memoria. Las redes LSTM son entrenadas en múltiples iteraciones conocidas como "epochs". Durante estas iteraciones, sesgo y pesos cambian para minimizar la función objetivo a través de los sets de entrenamiento.

Implementación del modelo LSTM

Para implementar el LSTM a través de la librería "Keras", debemos convertir los datos de ventas históricas a arreglos de 3 dimensiones [muestras, timesteps, atributos] en este caso las muestras son los días que tenemos de ventas observadas (1913), timesteps son los pasos que queremos que recuerde nuestra red, en

este caso lo definimos en 14, y los atributos serían las 30490 diferentes series de tiempo que queremos predecir. De igual manera, se deben normalizar los datos debido a que Keras trabaja mejor de esta forma.

Creamos los datos de entrenamiento "X_trainz z_train". Para cada artículo X_train, se incluyen 14 días pasados de ventas. Entonces un elemento de X_train es de tamaño (14, 30490). Para y_train estamos prediciendo un día de ventas de 30490 artículos entonces un elemento de y_train es de tamaño (1, 30490).

Se inicializa el regresor con la clase “Sequential” el cual es adecuado dado que el presente problema tiene un solo input de datos (demanda) y un solo output. Se agrega al regresor la primera capa al LSTM con 40 unidades, la segunda capa con 300 unidades, y la tercera capa también con 300 unidades. Para cada una se aplica un “Dropout” de 0.2, que es una selección de nodos aleatoria para evitar el sobre ajuste o especialización de los nodos, y se añade finalmente la capa de salida de 30490 unidades.
Luego se pasa a compilar el regresor utilizando como parámetro de optimización ‘adam’. Adam es una actualización del Optimizador RMSProp (Root Mean Square Propagation) que se basa en un ratio de aprendizaje adaptativo y como función de pérdida el MSE (Mean squared error)

Luego se fijan los siguientes paráámetros:

-Batch: Es el número de partes en las que se va a dividir el dataset, se establece 44.
-Epoch: Es el número de veces que se van a pasar los datos en las redes neuronales (normalmente se necesita pasar el dataset completo múltiples veces), se establece 32.

Una vez entrenado el modelo, procedemos a crear X_test. Se extraen del dataset los últimos 14 días (timesteps) de datos y se normalizan, luego se transforman en array. A través de un ciclo “for” que va desde “timesteps” hasta “timesteps” más los días totales a predecir (en este caso 28), luego se utiliza el atributo “.predict” y se recorre el ciclo. Una vez hechas las predicciones, se invierte la normalización para que queden en la escala real del problema. Finalmente se organizan las predicciones en el formato deseado para exportar. Arriba se describe el proceso para la predicción de los 28 días que van desde el 25 de abril hasta el 22 de mayo, que es el objetivo de la competencia, sin embargo, se realizaron 14 iteraciones más, con ventanas móviles de predicción de un mes, desde hasta el 24 de abril de 2016, con el fin de tener todo 2015 y hasta el 24 de abril de 2016 para entrenamiento del modelo de predicción de errores con Random Forest que se ejecuta en el paso 2.

Implementación de Random Forest con LSTM En retail, la información “no-temporal” o que no pertenece a la serie de tiempo, representa información “no cíclica” y variaciones irregulares en los datos de demanda, debido

al impacto de promociones y otros eventos de rebajas o festivos. Es posible implementar diferentes tipos de regresiones, pero los autores del método híbrido seleccionan Random Forest debido a su superioridad en precisión sobre otros métodos y amplia aplicabilidad en el manejo de retail.(13) Además random forest presenta un desempeño de predicción muy competitivo en evaluaciones empíricas recientes.

Para la implementación de este modelo, se tomó como periodo de entrenamiento los datos comprendidos entre el 1ro de enero de 2015 y el 30 de marzo de 2016. Los primeros 24 días de abril como período de validación, y finalmente del 25 de abril al 22 de mayo como datos de testeo. Con estos datos se calcularon los errores como: Demanda real – Predicción LSTM, entre el 1ro de enero de 2015 y el 24 de abril de 2016, que es hasta donde contábamos con datos de la demanda real, los últimos 28 días de predicciones se usan en el paso 3, donde se realiza la agregación de estos con las predicciones de los errores. Al igual que en el modelo anterior donde solo se usó Random Forest para predecir la demanda, tomamos variables extraídas del calendario, como meses del año, si es fin de semana o no, día de evento o promoción, etc. De igual forma se tuvo en cuenta la variable precio, pero a diferencia de ese modelo, se excluyeron las variables construídas a partir de la serie de tiempo como la media móvil de la demanda, desviación estándar, entre otras, dado que la información proporcionada por la serie de tiempo de ventas ya se tuvo en cuenta para las predicciones del LSTM. Para entrenar el modelo y realizar las predicciones de los errores se hizo necesario normalizar las variables regresoras, ya que algunas como el precio, y el número de semana del año, estaban tomando una relevancia exagerada debido a su mayor escala sobre las demás. Una vez hecho esto, se realiza una optimización de híper-parámetros a través de la técnica Random Grid, donde se establecen unos rangos de prueba para los parámetros deseados, 50 iteraciones en total y como función de optimización, minimizar el MSE. De donde se obtiene:

- Bootstrap = True
- max= 10
- max= 9
- min= 2
- min= 1
- n= 75

Con estos híper-parámetros se realizaron 30 iteraciones, dadas las limitaciones computacionales, para poder tomar datos de entrenamiento desde principio de 2015 y para tratar de ser más precisos en la predicción de los errores, se dividieron las 30490 series de tiempo por tienda (10) y categoría (3).

Agregación LSTM + RF

Una vez calculadas las predicciones en los pasos anteriores, se realiza la agregación siguiendo la metodología del artículo, que consiste en sumar las predicciones realizadas a través del LSTM utilizando solo la serie de tiempo pura, y las predicciones de los errores por Random Forest utilizando otras variables explicativas como se mencionó anteriormente, para el periodo comprendido entre el 25 de abril y el 28 de mayo, es decir 28 días.
Con esta agregación se obtienen las predicciones finales del modelo híbrido y se procede a validar los resultados

Resultados método híbrido LSTM + RF

Desafortunadamente, los resultados no fueron lo esperados, en la competencia de Kaggle el modelo LSTM por si solo obtuvo un **score de 0.85143**, mientras que el modelo híbrido, luego de realizar varias pruebas, solo logró **0.94882**, es decir, que en lugar de mejorar la calidad de las predicciones, las empeoró.

Esto puede deberse a la limitación en cuanto a tiempo y poder computacional, que en un tipo de trabajo tan empírico como este, se vuelve un gran problema, ya que sé quedan sin explorar un gran rango de variaciones posibles sobre la manera de entrenar y testear los modelos. Otra posible causa es que haya algún error metodológico, debido a que los autores del paper no entran en algunos detalles, como las ventanas de tiempo usadas para entrenar el modelo de errores o el comportamiento y distribución de los mismos, lo cual hace difícil replicar los resultados obtenidos por ellos para datasets con características diferentes e intermitencia como el que tenemos en este caso. También podría ser que hay un factor estadístico de fondo, con respecto a la distribución de los errores, que sería interesante analizar en detalle en un trabajo posterior. Se observó que para algunos items en particular el modelo logra capturar algo de la dinámica de los errores, generalmente los que tienen demanda más constante, pero en algunos otros, en lugar de ajustarse al comportamiento de los errores, pareciera aumentar más el ruido, lo cual explicaría la desmejora en los resultados generales.

Debido a lo anterior, este modelo no se tuvo en cuenta para el ensamble final de modelos y se optó por buscar alguna forma de mejorar los resultados del LSTM individual.

LSTM FINAL

Al no haber podido obtener los resultados deseados con el modelo híbrido, se realizó otra búsqueda de métodos sencillos de aplicar, que pudieran ayudar a incluir armónicamente la información de las variables explicativas. Se probó uno que consistía en incluir las predicciones del LSTM como variable regresora junto con las demás variables explicativas de calendario y precio, pero esto estaba generando cierto tipo de sobre ajuste, y las métricas de validación eran inferiores a las del modelo RF individual, por lo tanto se desechó rápidamente. Así las cosas, se optó por probar incluyendo algunas variables explicativas directamente en el modelo LSTM, que no aumentaran mucho el tamaño del dataset y el consumo de RAM (que estaba cerca de alcanzar la capacidad máxima). Luego de realizar varias pruebas, fueron incluidas finalmente las variables: **"Evento día siguiente", "Evento hoy", "Snap_CA", Snap_TX", "Snap_WI"**. Todas de tipo binario, las primeras dos representan si hay evento en "t+1", y en "t", respectivamente, las variables Snap valen 1 cuando hay Snap day en el correspondiente estado (California, Texas o Wisconsin) en el día t. Con esta configuración de logó un **score de 0.74662 en la competencia**, mejorando los resultados obtenidos por el LSTM sin incluir variables explicativas.

6.1.1. Implementación sólo con Random Forest

El tercer modelo utilizado para predecir la demanda de las referencias de los productos fue el de Random Forest Regresor (RFR). La principal ventaja de este algoritmo frente a otros de machine learning radica en que con pocos ajustes de los hyperparámetros se puede obtener un modelo que puede competir con otros como Redes Neuronales, Support Vector Machine y Support Vector Regressor.(14)
Antes de ejecutar el modelo de Random Forest, se implementó una estrategia con el objetivo de volver más eficiente su proceso de construcción. Lo anterior debido a que las exigencias computacionales requeridas para poder llevar a cabo este proceso excedían los recursos computacionales con los que se contaban. A continuación se describen cada una de estas:

1) Limitación en el número de modelos construidos: inicialmente el problema de investigación plantea el pronóstico de 30,490 series de tiempo. El caso ideal sería la implementación de un modelo que pudiese ser entrenado para cada una de las series de tiempo de manera individual; sin embargo, se determinó realizar un modelo para cada categoría de cada tienda. En este sentido, se construyeron 30 modelos para abordar el pronóstico de las series de tiempo.

2) Limitación de datos de entrenamiento: para reducir la capacidad computacional tiempo de entrenamiento de 1940 días a aproximadamente 365 días; es decir, a un año.

3) Selección de variables relevantes: otro aspecto que permitió reducir la cantidad de datos en la matriz de covariables fue la selección de variables relevantes a través de un análisis de correlación de Spearman; de esta manera se definieron para dicha matriz las variables con mayor significancia para la variable respuesta.

4) Reducción del espacio en los tipos de datos del dataframe: se redujo el espacio de almacenamiento de algunas columnas que tenían como tipo de datos float32 a un formato float16. Con este ajuste se logró reducir el tamaño en memoria de los datos en un 19 %.

5) Curvas de Entrenamiento: por medio de esta metodología se pudo hacer un ajuste a los hyperparámetros basados no solo en el mejoramiento de la métrica de evaluación, sino también en el impacto en los tiempos de entrenamiento.

6) Selección de plataforma para construir el modelo: gracias a la plataforma que ofrece Kaggle para la construcción de notebooks -en lenguaje phyton-, se logró extender la capacidad de memoria RAM a 16 GB.

Implementación del modelo

El modelo de RFR se llevó a cabo en cuatro etapas: en la primera se realizó

un análisis para identificar las variables relevantes para la variable respuesta; en la segunda, se construyó un modelo base con los parámetros que tiene el algoritmo por defecto; posteriormente, se hizo un ajuste de hyper parámetros, y por último se construyó el modelo final ajustado con los parámetros óptimos encontrados.

1. **Identificación de variables relevantes:**

Lo primero que se llevó a cabo fue un análisis de correlación con el objetivo de identificar las variables relevantes para el pronóstico de la variable respuesta (demanda). La métrica utilizada en este proceso fue la correlación de Spearman, puesto que, esta es recomendable en escenarios en donde existen relaciones no lineales, con distribuciones asimétricas y con valores extremos (Liu, Li, Wanga, Sheperd, 2018). Adicionalmente, dicho análisis de correlación de variables permitió analizar la multicolinealidad entre las variables explicativas, para de esta forma descartar aquellas que pudiesen generar distorsiones en la estimación del modelo.

2.**Creación del modelo**

Una vez seleccionadas las variables más relevantes para el modelo, se procedió a crear el RFR por medio del algoritmo Random Forest Regressor de la librería Scikit learn. Es importante resaltar que en este primer acercamiento se llevó a cabo el entrenamiento del modelo con los parámetros que vienen establecidos por defecto. Este primer modelo sirvió como base para evaluar los resultados de los ajustes de los hyper parámetros en la siguiente etapa.

Ajuste de parámetros del modelo

Con el modelo base definido, se procedió a ajustar y a encontrar los parámetros para el modelo. Los que se tomaron como referencia de estudio fueron: max_pth, max_features, min_samples_split,min_samples_leaf,bootstrap y n_estimators.
Dado que la capacidad computacional era muy limitada, se optó por realizar una búsqueda de hyper parámetros óptimos a través del algoritmo Random Grid, el cual por medio de búsqueda de combinaciones aleatorias encuentra unos parámetros óptimos. En la imagen 1 se puede observar los resultados de los hyper parámetros óptimos encontrados por el algoritmo de Random Grid para uno de los modelos implementados.

- best n depth: 20
- best n trees: 145
- best features: 9
- best n min samples split: 3

- best min samples leaf
- best boostrap

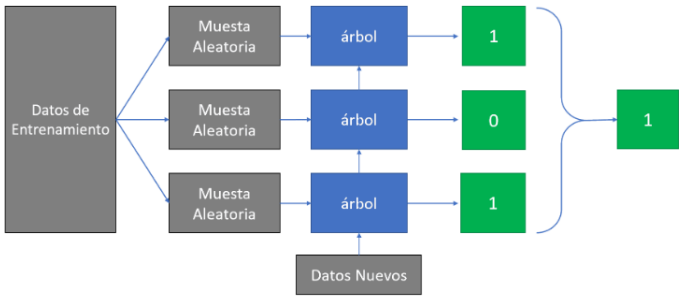


Figura 22: Random forest funcionamiento

Sin embargo, como la selección de combinaciones es aleatoria, se determinó complementar el ajuste de los parámetros utilizando el algoritmo de Grid Search. Este algoritmo permitió evaluar todas las combinaciones cercanas a los valores encontrados por Random Grid y de esta forma ajustar aún más los hyper parámteros del modelo. En la imagen 2 se puede observar los hyper parámetros óptimos encontrados para el modelo.(15)

- best n depth: 19
- best n trees: 120
- best features: 5
- best n min samples split: 5
- best min samples leaf:2
- MSE : 19.20
- R^2 0.53

Resaltar que la estrategia de la selección de hyper parámetros en estas dos etapas se debió principalmente a la búsqueda de una metodología que fuera eficiente para la gran cantidad de datos que se tenían en el set de entrenamiento y las limitaciones computacionales del proyecto; de lo contrario, si se buscaran por medio de un algoritmo como el de Grid Search para evaluar todas las combinaciones posibles en un intervalo de parámetros amplio, el entorno de ejecución de Python se reiniciaría dado el exceso de uso de RAM.

Por último, se llevó a cabo un estudio de las curvas de entrenamiento de los principales parámetros con el objetivo de poder comprender visualmente

el comportamiento de la variación de estos con el tiempo de entrenamiento, y los respectivos resultados obtenidos con la métrica de evaluación. En esta etapa se pudo evaluar hasta qué punto justificaba sacrificar el desempeño de la métrica de evaluación con el fin de buscar una mayor eficiencia en los tiempos de entrenamiento. En la imagen 3 se puede observar las curvas de entrenamiento del hyper parámetro n_estimators asociado al número de árboles para construir el modelo, esta imagen tiene a su izquierda una comparativa entre el resultado de la métrica a medida que se van aumentando el número de árboles en el modelo y en la derecha un comparativo entre el tiempo de entrenamiento y el número de árboles. Este análisis permitió ajustar los parámetros de tal forma que se tuviera en cuenta tanto el rendimiento de la métrica como el tiempo de entrenamiento

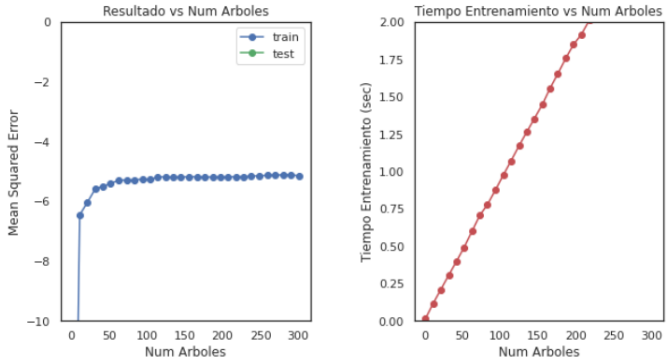


Figura 23: Curvas de entrenamiento del hyper parámetro de número de árboles

Creación del modelo final

Finalmente, con base en los hyper parámetros óptimos encontrados y en análisis de curvas de entrenamiento se definieron los parámetros de entrenamiento y se ejecutaron en paralelo los 30 modelos de Random Forest para cada uno de las categorías de productos de cada una de las tiendas.

Análisis de resultados

En la imagen 4 se pueden observar los resultados de las 3 métricas seleccionadas para cada uno de los modelo, con el objetivo de poder comparar los resultados de manera más general se realizó un análisis del comportamiento de cada métrica para cada categoría (ver imagen 5). En este se puede observar que la categoría que más precisión tuvo fue FOODS, fue la que mejor se pudo explicar (tuvo un R cuadrado de 0,69) y la tuvo el menor error bajo la métrica MASE. Recordar que la métrica MSE al no estar estandarizada no es comparable entre categorías.
La categoría que tuvo peores resultados fue la de HOBBIES, por un lado obtu-

vo los errores más altos bajo la métrica MASE y por el otro solo logró explicar 0,33 de la variable predictora. A pesar de que la métrica no es tan comparable entre categorías, en este caso llama la atención de que aun siendo HOBBIES una categoría con mucho menor volumen de ventas que HOUSEHOLD, su error cuadrático medio fue más (MSE) alto.
El modelo de Random Forest de manera generalizada obtuvo un rendimiento en la métrica del MASE de 0,54, logró explicar la variable predictora en un 64 % y obtuvo una calificación en la competencia de Kaggle de 0,77. En general fue un modelo que a pesar de que en cuanto información estuvo muy condicionado (se limitaron la cantidad de features, datos entrenamiento, entre otros.) buscando la eficiencia computacional, obtuvo buenos resultados.

RESULTADOS MODELOS POR CATEGORÍA Y POR TIENDA				
Tienda	Categoría	MSE	R2	MASE
TX1	FOODS	4.54	0.72	0.5
TX2	FOODS	5.48	0.76	0.46
TX3	FOODS	5.38	0.71	0.47
WI1	FOODS	4.56	0.56	0.55
WI2	FOODS	17.98	0.6	0.57
WI3	FOODS	8.04	0.71	0.52
CA1	FOODS	6.64	0.66	0.5
CA2	FOODS	6.55	0.58	0.59
CA3	FOODS	10.99	0.73	0.45
CA4	FOODS	2.79	0.46	0.63
CA1	HOBBIES	4.58	0.38	0.59
CA3	HOBBIES	5.24	0.37	0.59
WI1	HOBBIES	2.52	0.39	0.62
CA4	HOBBIES	0.24	2.62	0.64
TX3	HOBBIES	2.38	0.48	0.68
TX2	HOBBIES	2.16	0.26	0.69
CA2	HOBBIES	2.94	0.28	0.71
TX1	HOBBIES	2.39	0.13	0.75
WI3	HOBBIES	2.02	0.20	0.75
WI2	HOBBIES	1.65	0.26	0.79
CA3	HOUSEHOLD	5.20	0.61	0.50
TX3	HOUSEHOLD	1.92	0.47	0.57
WI2	HOUSEHOLD	3.23	0.61	0.57
TX1	HOUSEHOLD	1.90	0.53	0.58
CA1	HOUSEHOLD	2.11	0.52	0.59
WI3	HOUSEHOLD	1.68	0.53	0.62
TX2	HOUSEHOLD	2.69	0.34	0.65
CA2	HOUSEHOLD	2.23	0.42	0.65
WI1	HOUSEHOLD	1.34	0.39	0.68
CA4	HOUSEHOLD	0.81	0.29	0.78

Figura 24: Resultados modelos por categoría y por tienda

	MÉTRICAS POR CATEGORÍA		
	MSE	R2	MASE
FOODS	6.02	0.69	0.51
HOBBIES	2.38	0.33	0.68
HOUSEHOLD	2.02	0.49	0.61

Imagen 5: Métricas por categoría

	MSE	R_CUADRADO	MASE	KAGGLE
Rendón Forest	4,79	0,64	0,54	0,77

Figura 25: Resultados Globales del Modelo Random Forest

6.1.2. Modelo Stacking

Stacking es un algoritmo de ensamble que está diseñado para mejorar las predicciones de los modelos combinando las predicciones de múltiples algoritmos de machine learning (Sill,Tacs,Mackey,Lin, 2009). Para este caso de estudio, integra las predicciones realizadas por medio de los algoritmos de: Regresión Lineal, Random Forest y redes neuronales Long Short Term Memory (LSTM). Este modelo en esencia se entrena con base en las predicciones de cada uno de estos algoritmos y realiza una nueva predicción. Existen diversas metodologías de stacking, para la implementación de este modelo se tomó como base el Regresión Lineal en donde los coeficientes representan el peso de las predicciones de los algoritmos previamente mencionados.

Metodología

Para llevar a cabo su implementación, en primer lugar se agrupó los vectores de pronóstico de cada uno de los modelos construidos, y con estos se conformó la matriz de datos con la cual entrenar el algoritmo de Stacking. Es decir, la primera columna contenía las predicciones del modelo de regresión lineal, la segunda las predicciones del modelo de Random Forest y la tercera las predicciones de LSTM.(16)

- $\beta_0 = -0{,}06$
- $\beta_1 = 1,04$
- $\beta_2 = 0{,}0023$

Los resultados de dichos coeficientes fueron coherentes con los resultados obtenidos por cada uno de los modelos individualmente, esto dado que finalmente al modelo que más peso le asignó fue al de Regresión Lineal. Por medio de ésta metodología se obtuvieron los mejores pronósticos de la demanda, se logró explicar la variable dependiente en un 65 %, obtuvo 0,53 de error bajo la métrica de MASE y obtuvo una calificación de 0,68 bajo la métrica de la competencia de Kaggle.

7. Resultados finales

En esta tabla se observan los resultados finales de cada uno de los modelos usados. Por medio de las métricas MSE, R^2 y MASE se compara cada uno de ellos y finalmente se realiza la calificación final dada por Kaggle.

El Stacking superó en cada una de las métricas de evaluación a los demás modelos planteados ya que fué el que tuvo mayor explicación de la variabilidad, el menor error cuadrático medio, el menor puntaje en MASE lo que derivó en la mejor calificación de los modelos realizados en este proyecto para la competencia de Kaggle. Es interesante que el modelo de regresión lineal presenta un desempeño superior comparado con los demás modelos individuales sobre los datos de testeo de la competencia de Kaggle a pesar de tener peores métricas sobre el período de validación, esto podría deberse a que los demás modelos estan sobre ajustados.

	MÉTRICAS DE EVALUACIÓN			
MODELOS	MSE	R CUADRADO	MASE	KAGGLE
Stacking	4,79	0,65	0,53	0,68
LR	6,54	0,52	0,60	0,72
Random Forest	4,79	0,64	0,54	0,77
LSTM	5,66	0,58	0,55	0,75

Figura 26: Evaluación de modelos 1

7.0.1. Anexo gráficas de los modelos resultantes

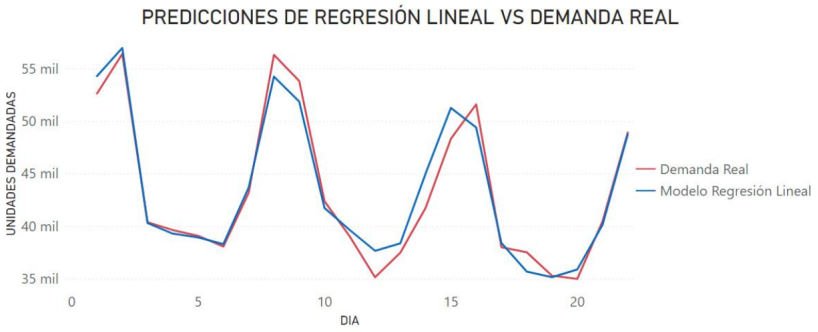


Figura 27: Evaluación Regresión lineal

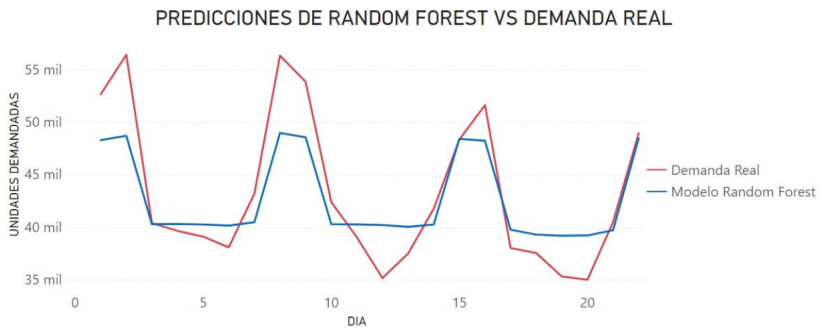


Figura 28: Evaluación Random forest

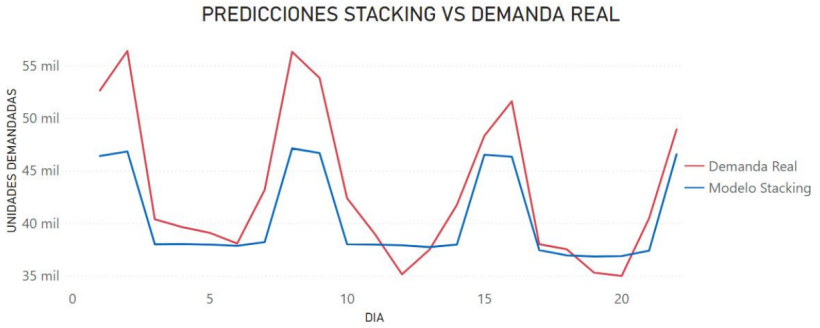


Figura 29: Evaluación Stacking

Figura 30: Evaluación LSTM

7.1. Conclusiones

- No hay un consenso definitivo sobre el mejor modelo para el pronóstico de la demanda en el sector retail. Sin embargo, se encontraron varios autores que recomendaron el uso de LSTM, Random Forest y Regresión Lineal

Múltiple para el pronóstico de la demanda.

- Se evidenció que la intermitencia afecta el desempeño de la precisión de los modelos, ya que aquellas categorías de productos con mayor proporción de ceros en su demanda fueron las que presentaron el score más bajo de MASE.
- El modelo de Stacking demostró ser una alternativa importante para el pronóstico de la demanda. La integración de las predicciones de los modelos propuestos permitió generar pronósticos más precisos que los modelos individuales, logrando superar los benchmark de la competencia.
- De acuerdo al conjunto de variables que describen el comportamiento del mercado, se evidencia que tienen una fuerte relación con la demanda, tales como: día de la madre, eventos deportivos, conciertos, navidad, día de gracias, entre otros.
- Se tiene que realizar una iteración manual con las variables para identificar anomalías
- Es difícil generalizar el conjunto de variables significativas para todos los modelos que expliquen el comportamiento de la demanda.

8. Proyecto en GitHub - Anexo códigos

Con el fin de tener toda la documentación del proceso técnico y teórico de las metodologías usadas en el proyecto, se crea un workspace en github donde es libre de revisión y extracción de información.

Allí se encuentran documentados los algoritmos usados en Python y R- studio

Link: https://github.com/mcdya2020/Proyecto-integrador-2020

Referencias

[1] “introduction to linear regression, curvefit.com,” 08 2019.

[2] "https://github.com/mcdya2020/Proyecto-integrador 2020", “"proyecto integrador 2020",”

[3] “Apunte sobre rectas de regresión,” *Ministerio de educación y ciencia. Gobierno de España.*

[4] M. C. L. de castilla Vásquez", “Regresión lineal,” 2011.

[5] B. S. .Everitt and Hothorn", “regression models for categorical and limited dependent variables. thousand oaks, ca",” 1997.

[6] S.-D. Poisson", “recherches sur la probabilité des jugements en matiéres criminelles et matiére civile.”,” 2005.

[7] R. J. Hyndman", “.another look at forecast-accuracy metrics for intermittent demand",” 2006.

[8] S. P. S. J. K. K. L. "Sushil Punia, Konstantinos Nikolopoulos, “"deep learning with long short-term memory networks and random forests for demand forecasting in multi-channel retail, international journal of production research",” 2020.

[9] "Fischer and Krauss", “"lstm (long-short term memory)",” 2018.

[10] Çhollet", “"series de tiempo lineales y no lineales ",” 2017.

[11] "Graves and S. ", “"time steps",” 2005.

[12] "Hochreiter and S. ", “"vanishing gradient”,” 1997.

[13] m. t. s. ÏnteractiveChaos, “"stackingr",” 2015.

[14] A. Lahouar and J. B. H. Slama", “random forests model for one day ahead load forecasting,ïrec2015 the sixth international renewable energy congress, sousse”,” 2015.

[15] L. C. W. V. . S. B. E. "Liu, Q., “çovariate-adjusted spearman’s rank correlation with probability-scale residuals. biometrics”,” 2018.

[16] T. G. M. L. . L. . "Sill, J., “"feature-weighted linear stacking”,” 2009.