Reto Atmira Stock Prediction

Realizado por el equipo: **Los Forecasters**

- Eduardo Hugues Gudiño
- Gustavo Adolfo Martínez Pérez
- Manuel Meniño Magán

Marzo, 2021

Índice

1. Resumen

El reto consiste en predecir las ventas futuras de productos en base a datos pasados, así como otros valores conocidos que pueden ayudar a predecir el valor deseado. El conjunto de datos de partida refleja una serie temporal multivariable, donde la unidad de tiempo es el día. Además cada producto tiene un identificador, de esa manera la conjunción de un día y producto determinado indica una serie de valores entre los que están las unidades vendidas de ese producto ese día. Se dispone de 2 conjuntos de datos, uno para entrenar el método de predicción con información de 16 meses, del 01/06/2015 al 30/09/2016, y otro de estimación, sin las unidades vendidas, de 3 meses, del 01/10/2016 al 30/12/2016. Pueden verse los detalles de ambos *datasets* en la página web del reto.

En primer lugar se realiza un análisis exploratorio de datos, para descubrir las características de cada tipo de variable y cómo se relacionan entre ellas. Algunas de las características más importantes que se han visto en los datos es que la distribución de unidades vendidas tiene la mayor parte de sus valores en cero, es decir, para la mayoría de los días y productos no se produce ninguna venta. Es lo que se conoce como una distribución Zero-Inflated. Puede verse más detalle en en el apartado de análisis exploratorio.

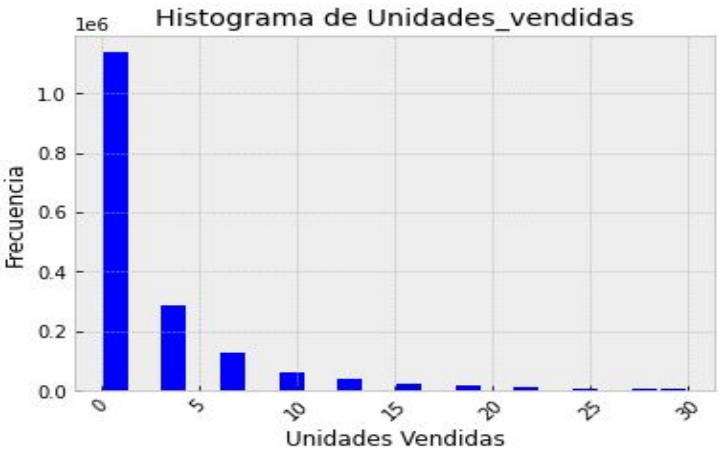
Posteriormente, se procede a una limpieza de variables: eliminando registros duplicados, corrigiendo aquellos valores erróneos, rellenando valores nulos y sincronizando valores entre los dos conjuntos de datos. También se realizan una serie de transformaciones de variables y añadido de nuevas variables, feature engineering, para enriquecer el conjunto de variables que puedan ser procesadas por los algoritmos de predicción. Puede verse más detalle en en el apartado de limpìeza y transformación de datos.

Finalmente se aplica el modelo seleccionado y se realiza la predicción. Aunque se trata de una serie temporal, los algoritmos clásicos de predicción de series temporales (Exponential smoothing, SARIMA, entre otros) no parecen los más adecuados ya que no aprovechan las información de las demás variables. Tras revisión de bibliografía y casos similares se decide cambiar el enfoque a uno de Machine Learning donde todas las variables, incluidas las temporales, se usan para predecir la variable objetivo. En nuestro caso se ha elegido un algoritmo de gradient boosting proporcionado por la librería LightGBM. Se hace uso de la función objetivo *tweedie*, ya que es especialmente efectiva para distribuciones *Zero-Inflated*.

Los resultados obtenidos nos indican que la predicción es bastante buena en general, pero falla en los picos de ventas que se producen en momentos muy puntuales. Una de las causas puede ser que la proporción de datos con ventas muy altas es una parte muy pequeña del dataset, por lo que el algoritmo generaliza bien, pero no tiene datos suficientes para estimar estos casos puntuales. Puede verse más detalle en el apartado de modelado y predicción.

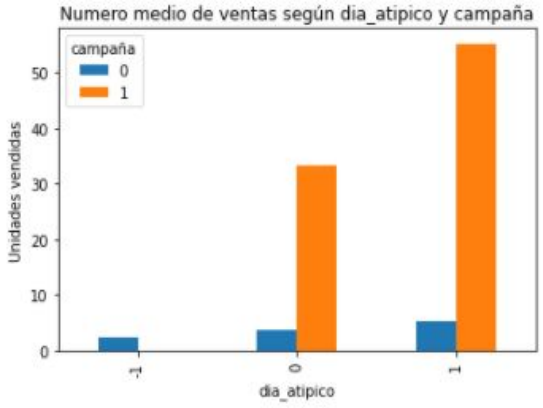
2. Análisis exploratorio

Destacamos información relacionada con la variable de salida que se encuentra en el archivo utilizado para modelar. En primer lugar, se muestra un histograma de dicha variable, que si bien toma valores hasta las 4881 unidades vendidas, se acota a las 30 unidades porque abarcan la mayoría de los datos.



Algo a tener en cuenta es que el valor de la variable unidades_vendidas es siempre cero o múltiplo de 3. Este hecho puede deberse a que el pedido mínimo al proveedor sea un pack de 3 artículos, de tal manera que si vendes un producto tienes que solicitar al menos 3 para poder reemplazarlo. Esta circunstancia se tendrá en cuenta a la hora de hacer la predicción.

Se hace hincapié también en la importancia de las variables dia_atipico y campaña, y su relación con la variable de salida. Como se puede ver, es especialmente importante el impacto que tiene en las ventas el hecho de que un artículo se encuentre en campaña.



En campaña no sólo crece mucho el número de ventas, sino también las visitas, y todo ello a pesar de que el precio de los artículos en media es más elevado:

Campaña	Precio medio	Nº medio de ventas	Nº medio de visitas
0	33.73	3.73	130.06
1	42.36	54.45	1346.63

Otra variable cuyo valor puede influir bastante en el número medio de ventas es la categoría_dos que tenga el artículo. Esta variabilidad se puede observar simplemente si nos fijamos en la media de unidades vendidas de las 5 clases más vendidas de esta categoría:

Categoria_dos	Número medio de ventas
238	48.62
224	23.04
122	14.33
43	10.17
157	8.9

También se presentan diferencias importantes en las ventas según de qué id de artículo se trate y la semana del año en la que se vende. En el notebook "Script exploración" puede encontrarse muchos más detalles de los datos analizados.

3. Limpieza y transformación de datos

Durante el análisis exploratorio se identifica que tanto el *dataset* de Modelar como el de Estimar requerían un preprocesamiento, para poder ser utilizados para entrenar y alimentarse a un modelo para la predicción de la demanda de los meses futuros.

Primero se identifica que el *dataset* Modelar contiene 4,168 productos y para cada producto se cuenta con información de 487 días. Esto indica que en total se debería tener un máximo de 2,029,816 registros dentro del dataset. El número de registros en el dataset original es de 2,015,206 superior al esperado, con un total de 4,045,022. Se observa que el dataset cuenta con registros idénticos duplicados, triplicados y cuadruplicados, provocando que existan 2,004,985 registros a eliminar.

Tras la primera limpieza aún quedan 10,221 líneas por encima de las esperadas por lo que se analizan otros posibles duplicados, dando lugar al descubrimiento de que todos los periodos de campaña se duplican con la única diferencia del valor de la variable "campaña". Se decide eliminar todos los registros con campaña cero cuando se tiene esta condición. Después, se identifica que existen registros duplicados con la única diferencia de contener valor "NaN" en la variable "antigüedad". Tras esta limpieza el dataset llega a los 2,029,816 registros esperados.
De igual forma, se identifica que existe una correlación entre el id del producto y su antigüedad, por lo cual se define sustituir los valores "NaN" a través de una interpolación lineal. De forma similar, se sustituyen los valores "NaN" de la variable "precio" con el precio inmediato anterior disponible. Si no lo tuvieran, se eliminan esos registros del *dataset* de entrenamiento.

Una vez completados los valores "NaN" y habiendo eliminado los duplicados, se procede a asegurar que no existen días sin registro dentro del rango de fechas del dataset Modelar. Se identifica que el 29 de febrero de 2016 no está registrado por lo cual se genera el registro para evitar dañar la serie de tiempo.

El dataset de Estimar pasa por el mismo proceso de preprocesamiento que el dataset Modelar, ya que se unen en un mismo *dataframe* para aplicar las modificaciones.
En cuanto a Feature engineering se realizan los siguientes cambios y se añaden nuevas variables para mejorar la predicción:

- Dividir unidades_vendidas entre 3, para hacer una predicción más precisa. Una vez hecha la predicción se hará la inversa y se multiplicará el resultado por 3.
- Añadir datos referentes a la fecha: mes, número de semana y día de la semana
- Diferencia de precios: La variación de precios entre el valor actual y otros valores de días anteriores.
- Label Enconding: para pasar a ordinal las variables categóricas, categoria_uno y Estado, que son formato string.

Otras variables probadas pero no usadas son:

- Lag y diff de la variable objetivo unidades_vendidas. Es decir, los valores de días anteriores (lag) y diferencia con fechas anteriores (diff). Se prueba en validación que el resultado con estas variables es muy bueno, el problema es al pasar a la predicción real. Hay que realizar one-step forecasting, es decir, realizar una iteración por fecha y en cada una predecir las unidades_vendidas el día siguiente. Al introducir lags el error que se haga en la 1ª predicción se va arrastrando a los siguientes registros. Como el horizonte de predicción es muy grande, 3 meses, al final el error arrastrado también sería muy grande, con lo que la predicción es peor.
- Lag de visitas: valores de las visitas en días anteriores, se prueba pero no mejora el error.
- Mean enconding de unidades_vendidas: la media de las unidades_vendidas agrupadas por alguna otra variable (categoria_uno, dia_atipico...). Sucede algo similar a los lags, es fácil trabajar con ellos en validación, pero complicado en la predicción real.

En el notebook “Script predicción” puede encontrarse todos los detalles.

4. Modelado, validación y predicción

El modelo seleccionado para la predicción es un tipo de gradient boosting implementado en la librería LightGBM. Las razones para elegir este modelo son:

- Enfoque a resolver el problema como uno de Machine Learning, y no como series temporales. Usando todas las variables como variables predictoras de la objetivo. Para ello hay que transformar la variable temporal en otra que pueda ser usada por el algoritmo.
- En la bibliografía y casos estudiados se considera este tipo de algoritmos apropiados para este caso de uso de predicciones de ventas, stocks...
- La implementación de LGBM permite un entrenamiento muy rápido comparado con otros algoritmos de este tipo, lo que permite que en el tiempo disponible probar con diferentes variables y parametrizaciones.
- Facilidad para trabajar con variables categóricas, al estar basadas en árboles de decisión, sin necesidad de recurrir a otras técnicas (Hot Encoding...), necesarias para otros algoritmos como redes neuronales.

En concreto se usará como función objetivo de la regresión la distribución tweedie, que según la bibliografía es especialmente adecuada para casos con muchos ceros como el nuestro. Esta distribución tweedie se puede variar mediante el parámetro *tweedie_variance_power*, entre 1 y 2. Si coge el valor 1, sería una distribución de poisson. Hacia el 2 sería una gamma.

Para validar los resultados se divide el dataset de modelar en 2 conjuntos, uno de entrenamiento y otro de test:

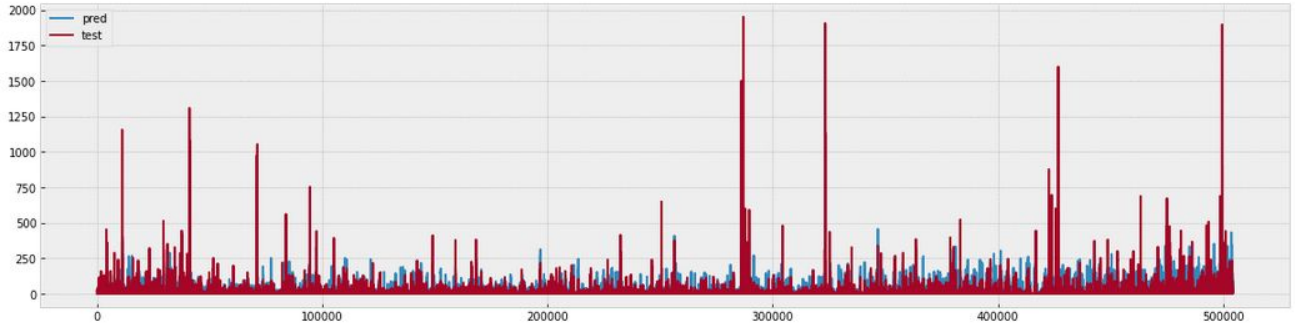
- Entrenamiento: 1 año, del 01/06/2015 al 31/05/2016
- Test: 4 meses, del 01/06/2016 al 30/09/2016

De esta manera se obtiene una proporción entre entrenar y validar del 75%-25% del total de datos de modelar, que es una buena proporción. Ambos conjuntos son una buena representación en cuanto a las características de cada uno de ellos.

Tras realizar diversas pruebas de validación con distintos parámetros del algoritmo LGBM se fijan unos valores, algunos de los más importantes son:

'boosting_type': 'gbdt',
'objective': 'tweedie',
'tweedie_variance_power': 1.9,
'metric': 'rmse',
'num_iterations': 800,

El resultado obtenido sobre el conjunto de validación es de un error de 3.85, con rRMSE igual a 5.49 y CF a 0.97. Se representando el resultado de la predicción sobre el conjunto de validación junto con los valores reales de unidades_vendidas:



Se observa en los resultados que consigue una predicción razonablemente buena para la mayor parte de los registros pero no predice correctamente los picos de ventas. Esto puede ser debido al desbalanceo del dataset respecto a las variables dia_atipico y sobre todo campaña. Se ve que las ventas los días con campaña=1 y/o dia_atipico=1 son sensiblemente mayores que otros días, sin embargo representan un % muy pequeño de registros del dataset, que mayoritariamente tienen estas variables con valor a cero. Esto puede hacer que para estos valores el modelo funcione peor.

Una opción probada es dividir el dataset en función de estas variables, dia_atipico y campaña, y entrenar cada uno de esos conjuntos por separado, obteniendo modelos distintos. Los resultados obtenidos en validación no mejoran con este método, posiblemente sea debido a que el número de datos en estos casos son tan pocos que el algoritmo no es capaz de entrenar correctamente. Otra opción pensada para este problema pero no probada en esta primera fase sería hacer un oversampling de estos registros con un método SMOTE o similar, para poder tener una representación mayor y que el algoritmo funcione mejor en estos casos.

Finalmente se realiza el entrenamiento del modelo con el conjunto de datos completo de modelar, y se realiza la predicción teniendo como entrada el dataset estimar tras haber pasado la limpieza y transformación. Tras este proceso se obtiene el fichero de resultado final, con el formato solicitado en el reto. También se puede observar la importancia que el modelo otorga a cada variable.

En el notebook “Script predicción” puede encontrarse más detalles.

5. Referencias y Bibliografía

1. Zero-Inflated distribution: https://en.wikipedia.org/wiki/Zero-inflated_model
2. LightGBM: https://lightgbm.readthedocs.io/en/latest/index.html
3. He Zhou, Yi Yang, Wei Qian. “Tweedie Gradient Boosting for Extremely Unbalanced Zero-inflated Data.” 2019.
4. Tweedie distribution:: https://www.statisticshowto.com/tweedie-distribution/