Sweet, sweet Data Science

Aplicación de modelos predictivos a la demanda de un obrador

**Autores:**

Miguel Ruiz Nogues

Illán Lois Bermejo

# ÍNDICE DE CONTENIDOS:

[1. Introducción 3](#_Toc28867990)

[2. Herramientas y modelos empleados 3](#_Toc28867991)

[2.1. Herramientas 3](#_Toc28867992)

[2.2. Modelos 3](#_Toc28867993)

[3. Datos 4](#_Toc28867994)

[3.1. Obtención 4](#_Toc28867995)

[3.2. Limpieza y preparación 4](#_Toc28867996)

[4. Análisis y modelos 5](#_Toc28867997)

[4.1. Introducción 5](#_Toc28867998)

[4.2. Modelos de series temporales 5](#_Toc28867999)

[4.3. Modelos de Machine Learning 6](#_Toc28868000)

[4.4. Redes neuronales 6](#_Toc28868001)

[5. Resultados 7](#_Toc28868002)

[5.1. Introducción 7](#_Toc28868003)

[5.2. Análisis 7](#_Toc28868004)

[5.3. Conclusiones 7](#_Toc28868005)

[6. Visualización 8](#_Toc28868006)

[6.1. Herramientas y métodos empleados 8](#_Toc28868007)

[7. Conclusiones 9](#_Toc28868008)

# Introducción

Nuestro objetivo es la aplicación de métodos estadísticos predictivos a datos de demanda de las diferentes tiendas de un obrador de pastelería.

La idea de este proyecto surgió debido a la necesidad de dicho obrador de poder mejorar sus predicciones en cuanto a la demanda de las diferentes tiendas a las que sirve, tanto para poder adaptarse mejor a las mismas como para redistribuir las futuras cargas de trabajo y así evitar picos.

En este sentido, los métodos empleados hasta la fecha se basaban en el conocimiento experto de sus empleados, los cuales no empleaban en ningún caso las series históricas de pedidos que nuestro cliente sí almacenaba.

Con esta información a nuestra disposición, nuestra intención es, por lo tanto, optimizar las predicciones de la demanda, empleando para ello métodos de series temporales y machine learning.

# Herramientas y modelos empleados

## Herramientas

De la multitud de herramientas a nuestra disposición, decidimos emplear Python tanto para el proceso de limpieza y depuración de los datos, como para su modelado.

Los motivos que nos empujaron a optar por Python fue la comodidad que sentimos programando en este lenguaje, y la multitud de herramientas disponibles en el mismo tanto para tratamiento de datos (Pandas) como visualización (Matplotlib, Seaborn), y sobre todo, modelos, tanto de series temporales (StatsModels, FbProphet) como de Machine Learning (Scikit-learn, TensorFlow, Keras, PyTorch).

Para la visualización decidimos emplear Tableau, dada la comodidad y sencillez de manejo que ofrece.

## Modelos

En cuanto a modelos, decidimos emplear tanto los modelos clásicos de series temporales (en concreto, modelos exponenciales y ARIMA) como modelos de Machine Learning.

Nuestra idea es realizar un estudio comparativo de los resultados ofrecidos por ambas técnicas y seleccionar en cada caso aquella estrategia que mejores resultados arroje.

# Datos

## Obtención

Los datos nos fueron suministrados por el cliente, en un archivo de tipo csv de 220MB, que contenía una tabla de 23 columnas y 1607908 filas. Los datos estaban organizados por pedidos por fecha y producto, existiendo para cada producto un código identificativo y una descripción. Cada línea de pedidos presentaba varios valores, siendo estos las cantidades pedidas por cada una de las tiendas de cada producto en cada día.

Nuestra primera decisión consistió en convertir estos datos en transacciones, introduciendo la tienda que realiza el pedido en una columna adicional.

El primer problema que nos encontramos al cargar los datos tuvo que ver con el formato de fecha, que solucionamos rápidamente introduciendo la función apropiada en el parseador.

La siguiente incidencia consistió en conseguir los nombres o identificativos de cada una de las tiendas para nombrar las columnas, dado que los datos no incluían en el encabezado su descripción.

Finalmente, el último problema evidente tenía que ver con la notación empleada para las cantidades. Estas empleaban la convención continental de la coma como separador decimal (a pesar de que todos los números eran enteros, las cantidades se expresaban como “X,00”.

Por último, comentar que en el csv encontramos la presencia de algunos NaN, así como líneas duplicadas, sin ser ninguno de estos problemas de gran impacto en nuestros datos.

# Limpieza y preparación

Dado que, de todos los productos, se iban a tratar solamente 10, nuestra primera tarea consistiría en el filtrado de los datos. Para ello, el cliente nos suministró una lista con los nombres de los 10 productos en los que estaba interesado.

El primer problema con el que nos encontramos consistió en que los nombres de esa lista no coincidían con las descripciones de los problemas.

Ahondando en el problema, descubrimos la existencia de una gran disparidad de designaciones empleadas en las descripciones. Asimismo, también encontramos que la correspondencia entre los códigos asignados a los productos y las descripciones era, en muchos casos, muy débil.

Así, existían muchos códigos que tenían un gran número de descripciones asociadas. Asimismo, encontramos descripciones aparentemente de un producto, bajo varios códigos diferentes.

Tras consultarlo con el cliente, decidimos ir examinando cada uno de los diferentes productos de nuestra lista, analizando su situación particular y filtrando en consecuencia. Así, para algunos de nuestros productos filtramos por código, tomando uno o varios diferentes. Para otros, nos basamos en descripciones, todo ello siempre tras analizar nuestros datos.

Una vez realizado el filtrado, hubimos de realizar el tratamiento de los datos: efectivamente, como comentábamos al comienzo, teníamos datos duplicados y nulos, pero preferimos realizar el filtrado de la información previamente a su limpieza tanto para no correr el riesgo de pervertir los datos antes de aplicar los filtros, como también para evitar realizar la limpieza de toda la tabla, pudiendo en el segundo caso realizar la limpieza de los datos de nuestro interés.

Así, realizamos a continuación el análisis y limpieza de datos nulos y de duplicados, para luego realizar un estudio de los propios valores de la serie en sí: identificación de outliers y análisis de completitud de las series.

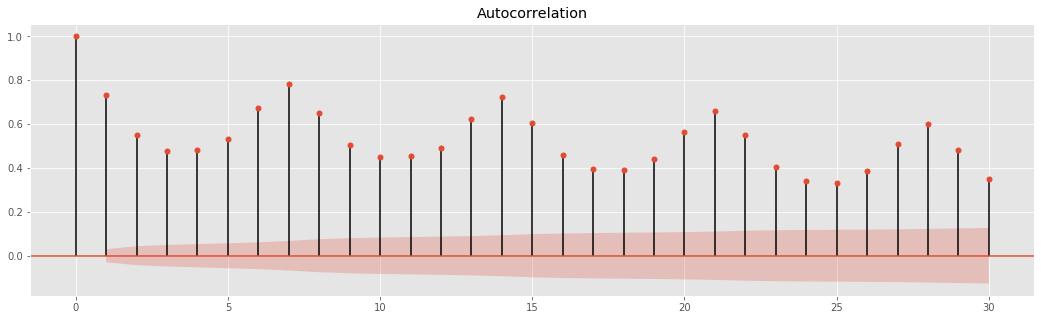
# Análisis y modelos

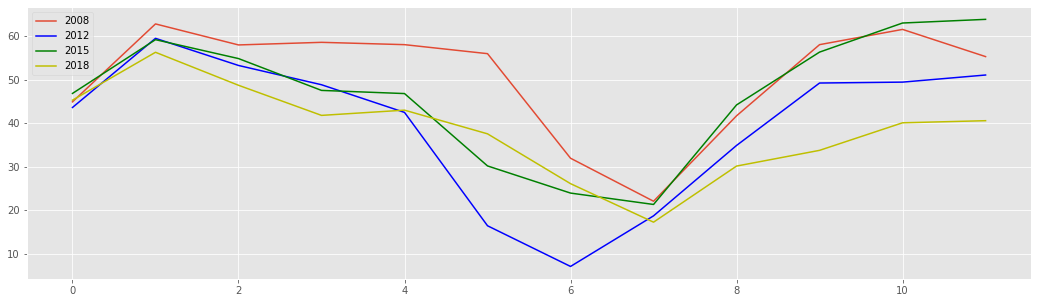
# Introducción

Como se comentó al comienzo, el enfoque del proyecto consistía en comparar los resultados arrojados por los modelos de análisis de series temporales tradicionales con los de los modelos de Machine Learning y con los obtenidos de la aplicación de redes neuronales.

Comenzamos realizando un análisis de la información contenida en las series, empleando para ello el test de Dickey-Fuller para comprobar la estacionariedad de la serie y la función de autocorrelación, la función de autocorrelación parcial e hicimos una primera descomposición de la serie.

En este análisis encontramos que la serie es estacionaria. También encontramos que existe una gran influencia de los 7 días anteriores sobre cualquier día de la serie, como se desprende del acf (que nos sugiere apostar por modelos AR(7) ). Asimismo, encontramos que la serie es doblemente estacionaria, con estacionalidad semanal y anual.





Para comparar los modelos entre sí, el criterio que emplearemos el error cuadrático medio y el criterio de información de Akaike.

Como benchmark emplearemos primero un modelo basado en el valor medio de la serie, para luego contemplar, dada la estacionalidad semanal de la misma, primero un modelo basado en los valores medios por día de la semana, y después, un naîve guess basado en el último valor del día equivalente de la semana anterior.

El horizonte a predecir será de una semana para todos los métodos empleados.

# Modelos de series temporales

Como modelos de series temporales hemos empleados tanto modelos exponenciales como modelos ARIMA y SARIMA (*“Seasonal ARIMA”,* modelos ARIMA adaptados para permitir introducir la estacionalidad).

También hemos aplicado Facebook Prophet, una librería desarrollada por los científicos de datos de Facebook y que aplica modelos de series temporales, pero que se emplea (en el sentido de que se entrena y predice) como los modelos de Machine Learning de Scikit-learn.

## Modelos de Machine Learning

En cuanto a modelos de Machine Learning, hemos aplicado un Random Forest, adaptando para ello los datos convenientemente para introducir la información de la serie decalada de 1 a n intervalos de la manera adecuada.

# Redes neuronales

# Resultados

# Introducción

# Análisis

# Conclusiones

# Visualización

# Herramientas y métodos empleados

# Conclusiones