Tras realizar la limpieza de los datos, ya teníamos un archivo con las series de valores listas para ser utilizadas. De todas ellas tomamos una, en concreto, la correspondiente a los pedidos de palmeras de chocolate, para analizar y aplicar los modelos.

Da todos los enfoques posibles, decidimos abordar el problema empleando series temporales y modelos de Machine Learning.

Por lo tanto, para poder tratar las series, es necesario realizar primero comprobaciones sobre la serie, y después de aplicar los modelos, comprobaciones sobre los resultados.

Y, en concreto, la comprobación más relevante es la de la estacionariedad.

Comprobar si la serie es estacionaria es especialmente relevante a la hora de aplicar modelos de la familia de los ARIMA, los cuales pretendíamos aplicar.

Realizamos por lo tanto el test de Dickey-Fuller ampliado, y comprobamos que realmente es estacionaria.

En cuanto a propiedades intrínsecas de la serie, comprobamos sus principales estadísticos (como podéis ver, tenemos una varianza muy alta) y descubrimos que la serie presentaba tanto estacionalidad semanal como anual.

También aplicamos las funciones de autocorrelación total y parcial, ahí tenéis los dos correlogramas, donde se puede ver la gran importancia que tienen los lags 1, 6 y 7.

Los correlogramas nos empujan a pensar en modelos autorregresivos o bien de orden 1, o bien de orden 7. También podemos pensar en modelos con orden de diferenciación 0 o 1, y para la parte MA (es más complicado).

Aquí tenéis la estacionalidad anual. El mes con menos demanda de palmeras de chocolate es agosto, lo que creo que no es ninguna sorpresa para los que somos de Madrid.

Finalmente, también realizamos una descomposición de la serie, en donde se puede apreciar de nuevo la estacionalidad semanal de la misma y la gran varianza de los residuos.

Antes de desvelar los resultados, queremos ir un poco paso por paso:

Comentar que de los tests que pasamos a los residuos obtuvimos los siguientes resultados:

Los métodos de la familia de los ARIMA pasaron el test de Jiung-Box, que indica si quedan correlaciones en la serie o no. Tanto el Holt-Winters como el Prophet no, lo cual era previsible del primero pero una sorpresa por parte del segundo.

Los residuos de todos los métodos fallaron en el Shapiro-Wilk, lo que da a entender que el proceso subyacente no es Gaussiano, y nos dificulta la tarea de dar intervalos de confianza.

Y ahora, ya sí, lo que todos estabais esperando: los mejores métodos fueron el SARIMA y Prophet. El Random Forest, entrenado con un año de datos, no fue capaz de superar a los métodos tradicionales de series temporales.

Por este lado, ninguna sorpresa: SARIMA representa una ventaja sobre ARIMA, y dada la clara estacionalidad de nuestra serie, explotar independientemente esta estacionalidad tenía que redundar en una clara ventaja.

En cuanto a la comparativa con el método exponencial, dada la existencia del alisado, no puede ajustarse adecuadamente a la influencia que tiene cada lag. La estacionalidad corrige parcialmente este defecto, pero no lo elimina completamente, y es de destacar el buen resultado que obtiene cuando lo comparamos con el ARIMA clásico pese a ello.

Todos los modelos se comportaron mejor que los de benchmark, como podéis ver.

Y finalmente, comentar que no sólo es importante el error, sino su distribución, que sea estable. Y como se aprecia en la figura inferior, las distribuciones de los distintos métodos también son mucho más estables que las del Naive guess que fue el mejor modelo de referencia.

Por último, comentar que, haciendo la raíz del MSE y dividiendo por la media de la serie, tenemos un valor orientativo del error porcentual medio en nuestras predicciones, y que, para los mejores métodos, este valor es de un 15%, lo que valoramos como aún demasiado alto.

Como herramientas principales empleamos: Python y Tableau. En cuanto a librerías de Python: Pandas, Numpy, Statsmodels, Scikit-learn, Matplotlib, Seaborn, Fbprophet, Tabpy…

Comentar también que empleamos otras herramientas, como Slack y Jira, para gestionar el proyecto, y Github, para el control de versiones y de trabajo en paralelo.

------------------------------------------------

No queríamos terminar sin mencionar los principales problemas que hemos tenido: por un lado, comprobamos que es cierto que limpiar los datos lleva el 80% del tiempo. Nosotros teníamos unos datos, al menos en archivo y formato, limpios, un csv de buena familia sin muchos problemas, y aún así ya habéis visto todas las incidencias que nos encontramos después.

También pudimos comprobar que muchas de las decisiones que íbamos tomando en el proceso de limpieza iban a impactar luego en nuestras series y por lo tanto, en nuestros modelos y predicciones.

En cuanto a problemas técnicos: intentamos acelerar Pandas con Modin, pero nos surgieron muchos problemas que nos hicieron perder varias horas de trabajo y sólo se solucionaron cuando desinstalamos el paquete.

Por otro lado, Prophet nos dio algunos problemas también, y en concreto, inhabilitó el series.plot().

Tampoco fue fácil hacerse con Tabpy a la hora de crear el servidor de Python para Tableau.

Y finalmente, tuvimos también muchos problemas de coordinación y de uso con Github, especialmente a la hora de mergear ramas y de coordinar nuestros esfuerzos.

Pero bueno, de todos estos problemas hemos ido sacando lecciones que seguro que nos servirán en un futuro.

CORRELOGRAMA

Stationarity is an important concept in time series analysis. For a concise (but thorough) introduction to the topic, and the reasons that make it important, take a look at [my previous blog post on the topic](https://towardsdatascience.com/stationarity-in-time-series-analysis-90c94f27322). Without reiterating too much, it suffices to say that:

1. Stationarity means that the statistical properties of a time series (or rather the process generating it) do not change over time.
2. Stationarity is important because many useful analytical tools and statistical tests and models rely on it.