THE BRIDGE DIGITAL TALENT ACCELERATOR

PROYECTO MACHINE LEARNING:
Predicción de ventas

Guillem Pañella Valiente
Tutores: Alberto Romero, Ivan Cordero, Lucas Zamora
Data Science Online Madrid Mayo 2024 - Diciembre 2024

ÍNDICE DE CONTENIDO

1. Introducción y elección de bases de datos

Para este trabajo se ha buscado plantear un proyecto en el que nos permita crear un modelo de Machine Learning que permita predecir las ventas en pedidos online de una empresa a partir de unos históricos de años anteriores. Se trata, pues de un modelado de una serie temporal (ya que las cifras de históricos son diarias) y que nos permita adjudicar esas cifras diarias de cara al próximo año.

El problema de negocio planteado es que la empresa se aproveche de saber las cifras que tendrá para poder organizar las contrataciones, la logística de la empresa para saber cuando habrá un pico y cuando un valle y para, sobre todo, organizar todos los gastos de cara al año que viene (aperturas de tiendas, cerrar tiendas sin beneficios) para saber cuando vendrá la época de contratar a personas y cuando, tal vez, saber controlar muy bien los gastos.

En cualquier empresa, el momento de predecir las cifras, ganancias y pérdidas del próximo año son claves para organizar mucho mejor cómo se va a planificar el año.

Para ello, se ha utilizado una base de datos que concentra los pedidos a la empresa de los años 2011 hasta el 2014, día a día, para que el modelo que vamos a utilizar nos permita saber cómo va a ser día a día el año 2015.

A modo informativo también, las filas del dataset son cada uno de los pedidos que hay y el gasto de cada uno por lo que se ha realizado una suma diaria de la cifra en pedidos para ser más eficiente en la visualización y que se pueda observar

2. Exploración de datos, estacionalidad, tendencia

En esta parte se ha querido analizar cómo era nuestra base de datos con diferentes gráficos.

En primer lugar, a parte de subir los datos con la librería Pandas se ha querido observar la horquilla de fechas en las que teníamos datos. Como ya hemos comentado, los datos van desde 1 de enero de 2011 hasta 31 de diciembre de 2014.

Imagen 1: Como dividimos los datos

Mostramos el rango de fechas

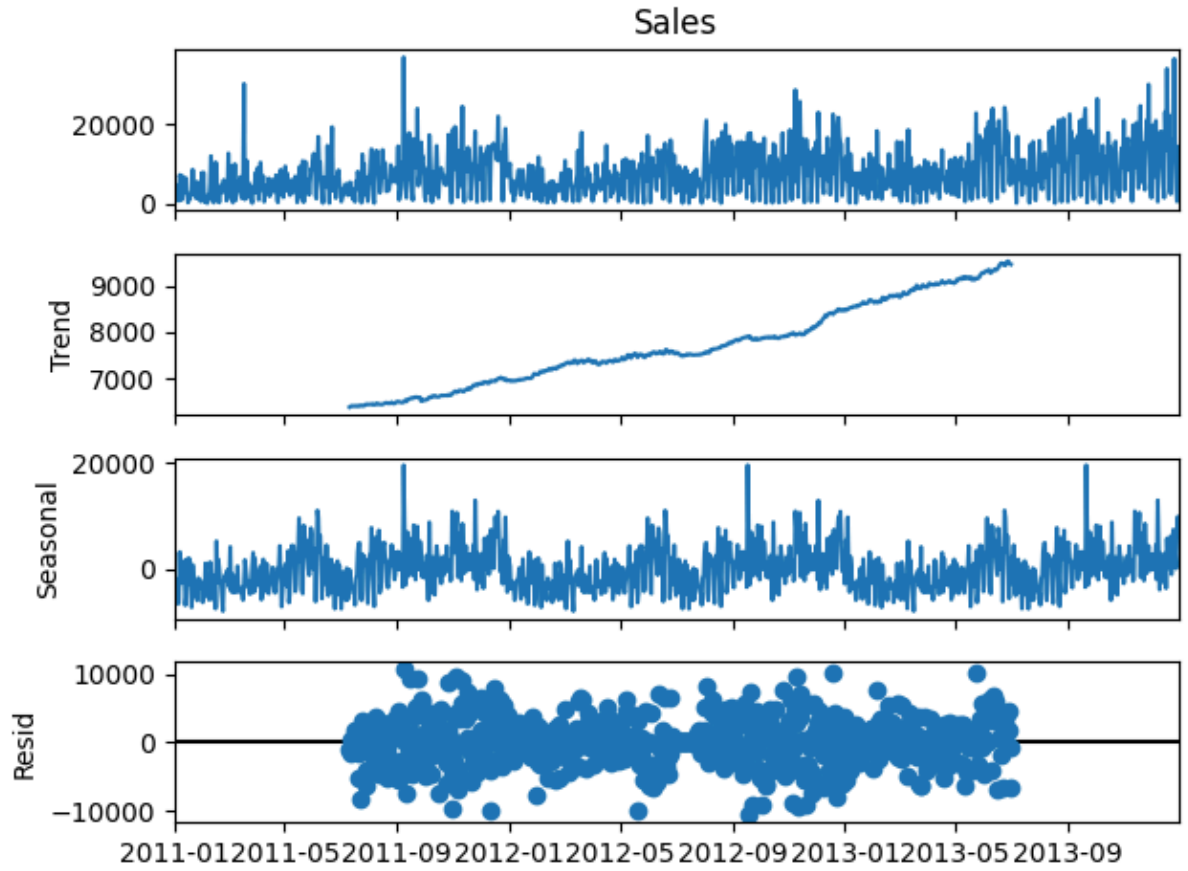
```
# Asegurarnos de que la columna 'Order Date' está en formato datetime
data['Order Date'] = pd.to_datetime(data['Order Date'])

# Mostrar el rango de fechas
start_date = data['Order Date'].min()
end_date = data['Order Date'].max()

print(f"Los datos abarcan desde {start_date} hasta {end_date}.")
```

```
Los datos abarcan desde 2011-01-01 00:00:00 hasta 2014-12-31 00:00:00.
```

Fuente: Capturado directamente del notebook del modelo

En lo que se refiere a los diferentes indicadores que queremos remarcar de la base de datos serán la tendencia, la estacionalidad y los residuos. Los ponemos a continuación:

Gráficos 1: Visualización de la tendencia, estacionalidad y residuo de la base de datos (train diaria)



Fuente: Obtenida directamente del código del notebook. Librería matplotlib.pyplot y statsmodels.tsa.seasonal.

3. Machine Learning

En este punto empezamos a hablar del modelado de los datos para que nos ayuden a predecir los datos del futuro, siempre teniendo en cuenta los datos que tenemos que, como hemos visto ya, son del 2011 al 2014 completos.

Aquí veremos dos vertientes de entrenamiento de datos:

a) Datos diarios y entrenamiento con datos diarios. Será un modelo menos preciso en cuanto a captar la gran estacionalidad que tienen los datos

b) Datos semanales y entrenamiento con datos semanales. Será un modelo donde permitirá tener una foto más parecida y que nos ayude a captar algo mejor la estacionalidad, que no es tan marcada como en los datos diarios y así escapar un poco de las efemérides que pueden haber durante el año.

3.1. Modelos Machine Learning diarios

Separación del train y test

En este punto es importante explicar bien como se han separado los datos para que se entienda como se ha entrenado el modelo que se ha utilizado en este proyecto.

Lo interesante de tener datos de series temporales es que nos permite entrenar con tendencias de años enteros para luego ver como funciona ese modelo con un año entero completo y evaluar los resultados con los datos reales del año de test. Es decir, los modelajes de timeseries permiten evaluar el modelo primero con un año que tenemos datos para luego que pueda predecir datos de un año en el que no tenemos datos de forma adecuada.

Por ello, se han sumado las diferentes Orders diarias que había para que hubiera una cifra diaria y se han agrupado los datos en años (2011-2014). Una vez se ha sumado el dato diario se han escogido los años 2011, 2012 y 2013 como los datos para entrenar el modelo (datos train) y el año 2014 como los datos para testear y evaluar el modelo (datos test).

Test de ADF

Una de las características para trabajar con modelos de series temporales es si los datos son estacionarios o no. Por ello se ha querido añadir el test de ADF (Augmented Dickey-Fuller) y se ha hecho con los datos del train y de test

Train ADF Test Statistic: -3.2858857408160698

p-value: 0.015521291890962964

Critical Values: {'1%': -3.436617132314828, '5%': -2.864307097893787, '10%': -2.568243313067353}

Test ADF Test Statistic: -1.7731849449769657

p-value: 0.3938535201782304

Critical Values: {'1%': -3.4495033946549123, '5%': -2.8699787979596136, '10%': -2.5712658305164955}

Como se puede observar, los datos de train son estacionarios (p-value menor a 0,05) pero los de test no lo son. Para que el modelo pueda ser útil planteamos la

transformación de los datos de test y hacemos la misma operación con los de test para que trabajen en las mismas unidades y para que ambas sean estacionarias.

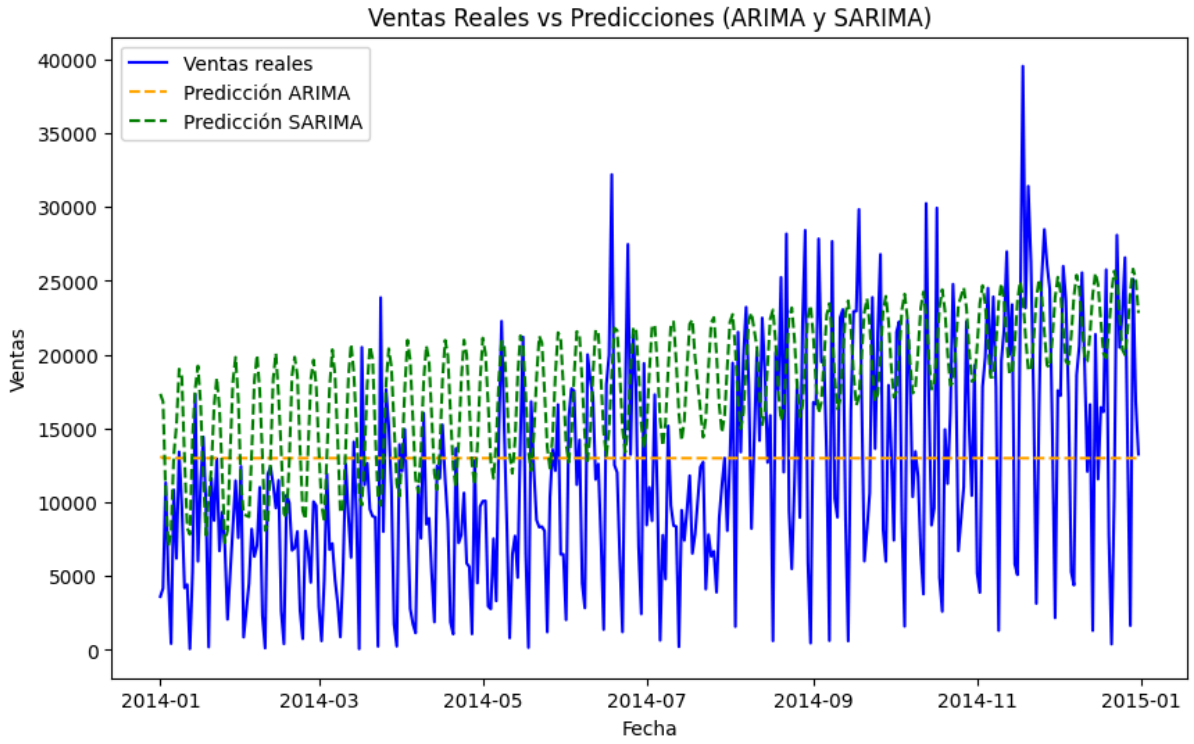
Habiendo hecho esto, el modelo más obvio a entrenar sería el SARIMA debido a la estacionalidad. Esto se verá a continuación en los siguientes apartados.

Train de datos

En este caso, se ha querido entrenar los datos de train con ARIMA y SARIMA para ver que ambos modelos funcionan y ver cómo se comportan con nuestra base de datos.

Evaluación de datos vs test

Gráfico 2: datos diarios de 2014 reales enfrentados a los que daría el modelo entrenado.



Fuente: Obtenido a partir de utilización de librería matplotlib

Como se puede observar en el gráfico 2, el modelo ARIMA no capta la estacionalidad de los datos porque no está entrenado para ello, entonces nos da el valor medio durante el año entero sin percibir esa estacionariedad. El modelo SARIMA si que capta esa estacionariedad pero no la capta totalmente, como se puede observar en el gráfico.

ARIMA - RMSE: 7736.2748428369105, MAE: 6444.526627700249

SARIMA - RMSE: 9776.02693169478, MAE: 8208.95909788409

Los resultados del modelo y de sus errores son los siguientes. Veremos cual es el resultado cuando ajustemos los parámetros.

Mejora de los parámetros con pmdarima

A partir de la librería pmdarima, ajustamos el modelo con los mejores parámetros posibles y que capte mejor los datos de entrenamiento.

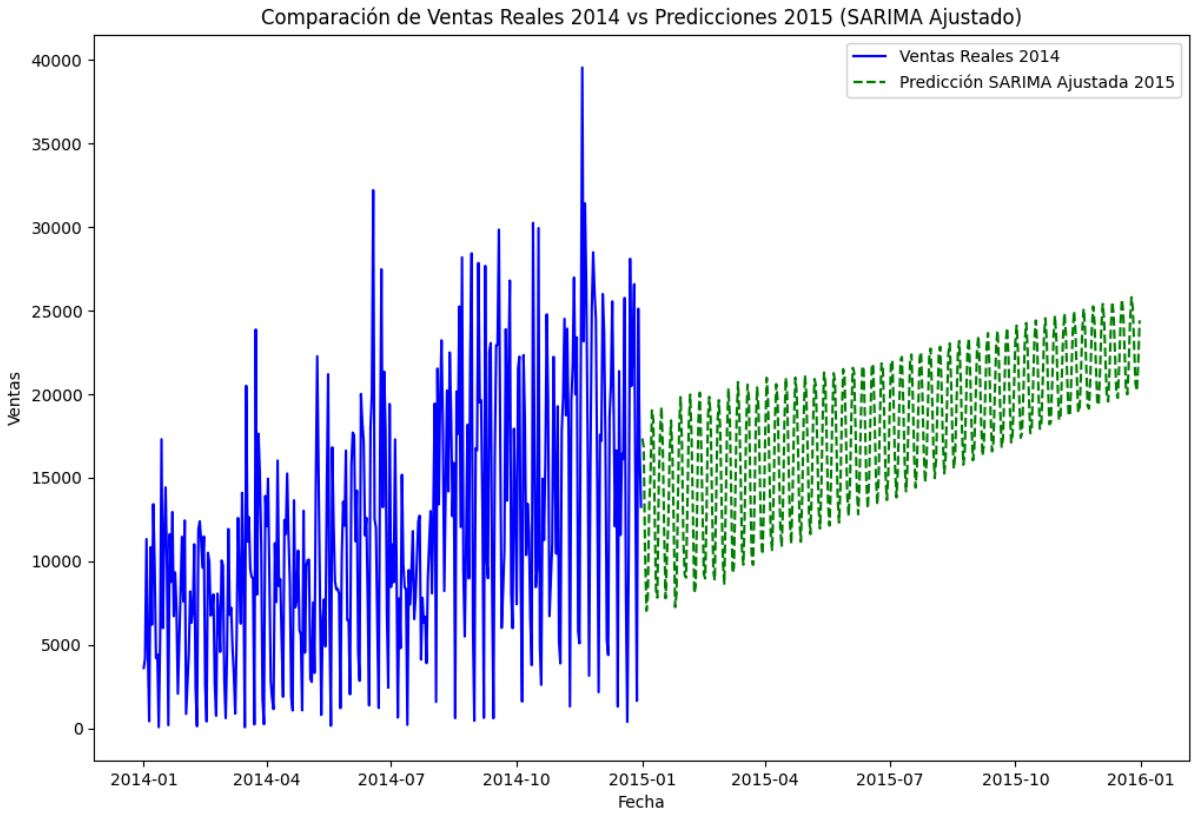
Best model: ARIMA(2,1,5)(2,0,2)[12] intercept

Es cierto que capta mejor la tendencia de los datos pero sigue costándole captar la fuerte estacionalidad de los datos.

Predicción de cifra diaria de 2015

Siguiendo con la predicción de ventas para este proyecto el principal motivo por el que se creaba era por el hecho de poder predecir datos que no tenemos, encarados a las ventas de un año futuro entero. Es por ello que la prueba final del modelo es ver como se comporta para predecir un año entero en el que no tenemos datos. Estos son los resultados:

Gráfico 3: ventas reales de 2014 y predicción con el modelo SARIMA ajustado con los mejores parámetros calculados



Fuente: A partir de la librería matplotlib

Como se puede observar, el modelo capta la tendencia que tienen las cifras diarias del año 2014 de crecimiento y capta parte de la estacionariedad pero no captan en gran medida esos picos y valles diarios que pueden haber en un año. En parte, debido a las posibles efemérides que pueden haber en un año y a las diferencias entre días festivos y días laborales, el modelo no acaba de captar esas grandes curvas diarias.

No obstante, nos gustaría destacar que cuando calculamos el error medio del modelo con el RMSE y el MAE nos dan resultados bastante aceptables en cuanto al modelo como coge la cifra diaria, concretamente el SARIMA.

SARIMA - RMSE: 7977.5578, MAE: 6717.7790

3.2. Modelos Machine Learning semanales

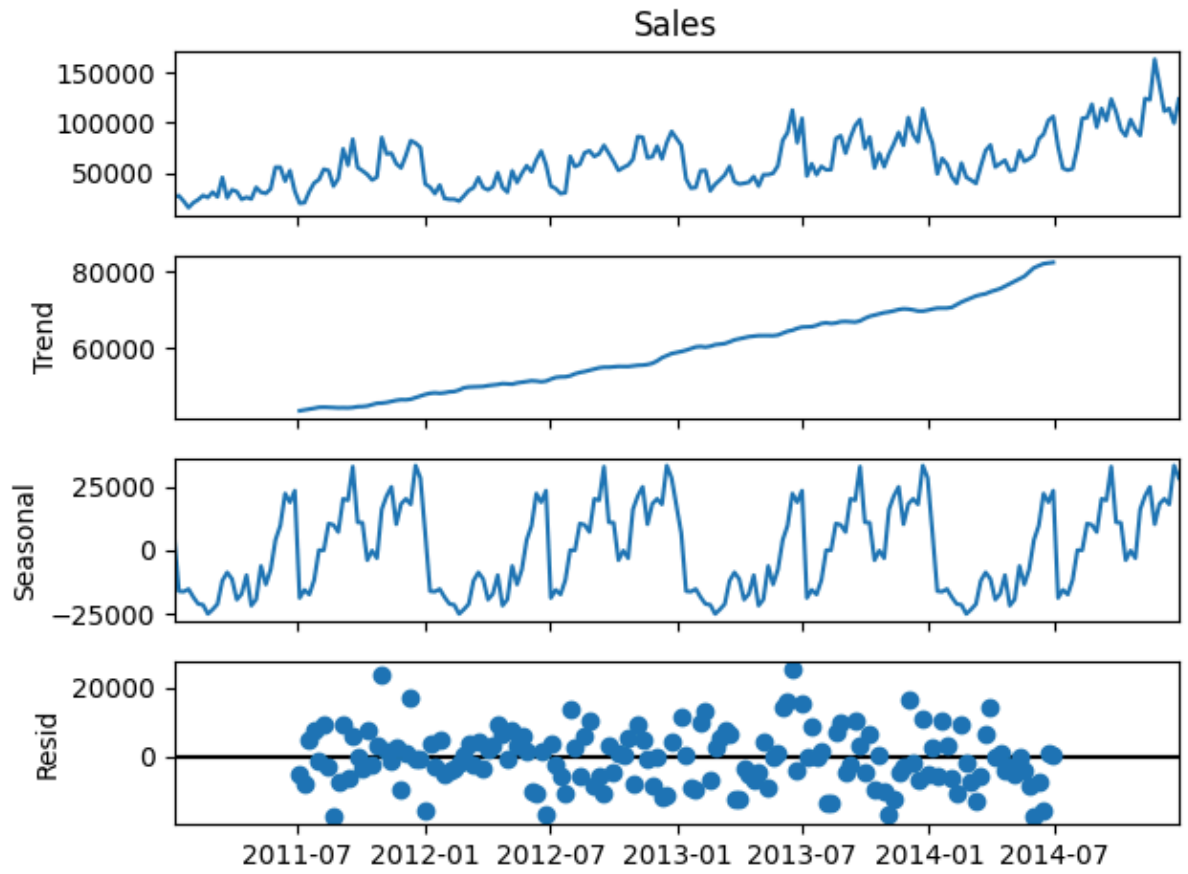
Viendo las dificultades que teníamos para captar la alta estacionariedad que tienen los datos de forma diaria y que, escuchando un poco al negocio, vemos que la mayoría de veces el negocio no necesita una predicción diaria sino más semanal para que los recursos que se puedan aplicar en el negocio sean aplicados con tiempo de maniobra y no para un dia sí y para otro no.

Es por ello que hemos planteado un modelado muy parecido al diario pero con los datos ya de forma semanal.

Separación del train y test

En este inicio de modelado, como en el caso anterior, separaremos los años 2011, 2012 y 2013 para el train y el 2014 para el test. No obstante, haremos el paso previo de calcular la cifra semanal de cada uno de los años para poder ver la curva de ventas y los diferentes indicadores de los datos, en este caso particionados por semana:

Gráfico 4: Todos los datos de la base de datos utilizados pero particionados por semana en los diferentes indicadores para entender cómo se comportan (tendencia, estacionalidad y residuo)



Fuente: a partir de la librería de matplotlib (todos los datos de la base de datos)

Como se puede observar en este gráfico 4, los datos se pueden observar más claramente y nítidamente viendo bien las diferencias entre semanas y esas diferencias entre ellas.

Test de ADF

ADF Test Train (diferenciado): estadístico=-6.416730176361734, p-valor=1.8334407808957478e-08

ADF Test Test (diferenciado): estadístico=-4.390049403457799, p-valor=0.00030910284479903017

Como en el caso de los datos diarios, se ha planteado el test de ADF pero ya directamente se ha hecho el diferenciado de los datos para que ambos casos (train y test) sean estacionarios.

Train de datos

Se entrenan, como en el caso anterior, los datos de 2011 hasta 2013 pero en este caso con los sumatorios por semanas de los modelos ARIMA y SARIMA aunque ya hemos visto que nos interesará mucho más el SARIMA debido a la estacionariedad

de los datos (aunque semanalmente haya menos estacionariedad, la sigue habiendo notoria ya que hay diferencias grandes entre semanas).

Evaluación de datos vs test

En la evaluación de los modelos con el test, lo primero que ponemos de manifiesto es el RMSE y el MAE como se ha hecho en el día a día.

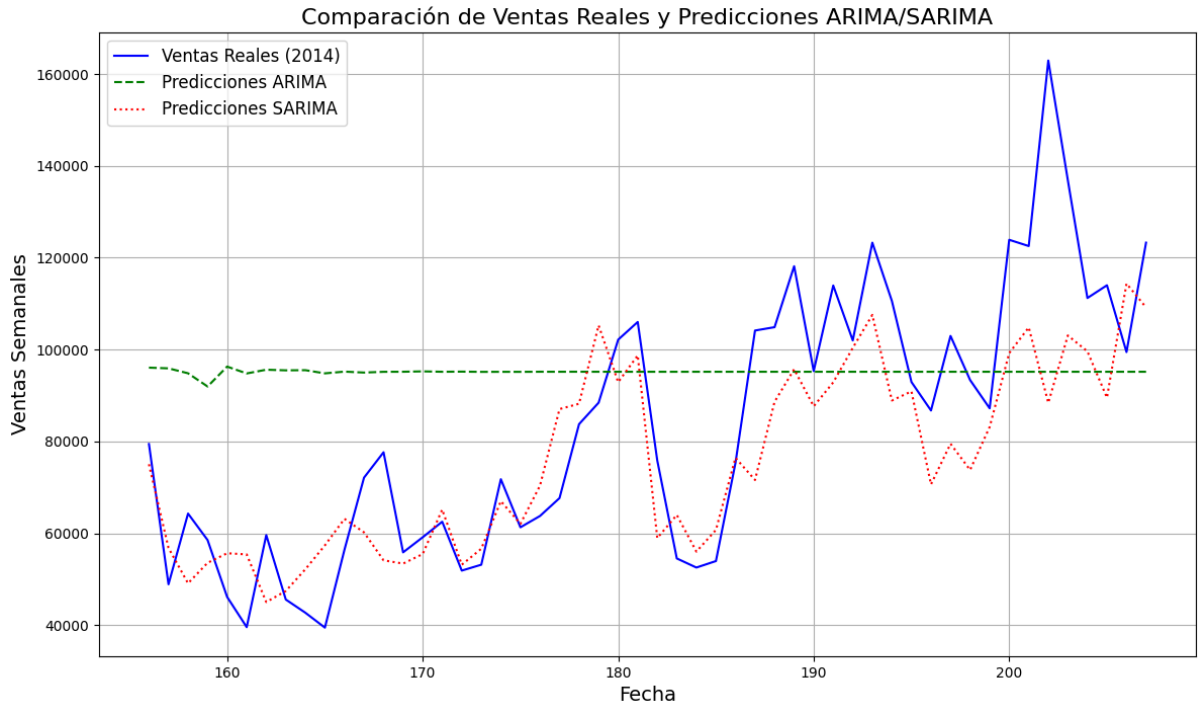
ARIMA RMSE: 31312.638874985998, MAE: 26675.77968794753

SARIMA RMSE: 17778.956443943218, MAE: 13159.393788824253

Como se puede detectar, el modelo de SARIMA aquí tiene notoriamente menos errores y está más cercano a 0 por lo que el modelo es más adecuado y pertinente.

Esto, gráficamente se verá más claro:

Gráfico 5: ventas semanales de 2014 reales enfrentadas a las predicciones de los dos modelos entrenados



Fuente: A partir de la librería de matplotlib con los datos del test

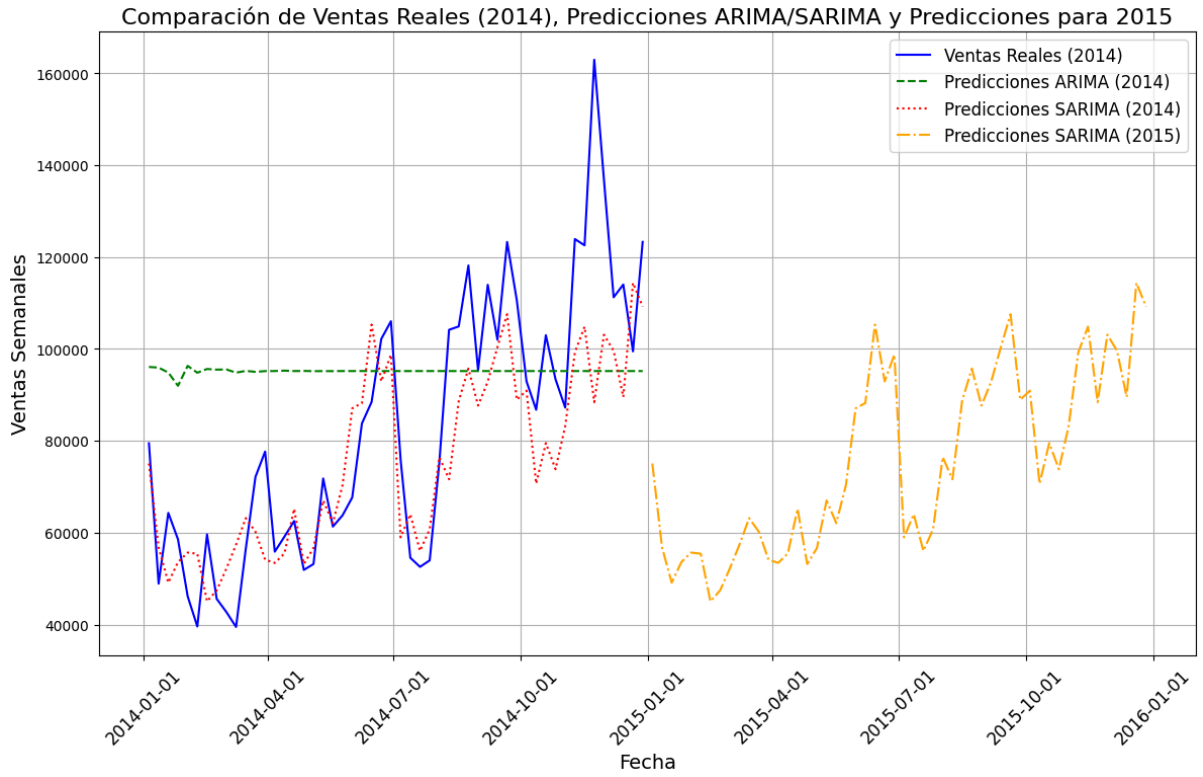
Como se puede observar en el gráfico 5 los datos de test con lo que se le ha aplicado el modelo entrenado se ajustan bastante más a los datos reales de 2014. Como en el caso de los datos diarios, el modelo ARIMA no recoge esa estacionariedad que semanalmente también tiene los datos y, por lo tanto, mantiene la línea en la media aritmética de la cifra anual que tiene previsto hacer el modelo.

Es interesante detectar también que el pico que tiene casi al final en los datos reales de 2014 no es detectado por el modelo porque en los datos de train no había

detectado y por lo tanto, no plantea que la empresa vaya a tener una subida tan fuerte.

Predicción 2015 y mejores parámetros

Gráfico 6: Predicción de 2015 con las ventas reales y las predicciones de 2014



Fuente: A partir de la librería matplotlib y ajustando bien el rango de años dentro de los ejes utilizados

Como se puede observar en el gráfico 6 el modelo predice bastante parecido a lo que hacia en 2014, ajustándose bastante a los datos con los que ha entrenado y captando bien la estacionalidad que tienen los datos. Lo que no acaba de captar bien el modelo para ser más adecuado es la tendencia de las cifras. Esto puede ser causado a que la tendencia que está cogiendo es de los años de train y el crecimiento que se da en 2014 no lo está cogiendo. No obstante, este modelo capta bastante mejor las diferencias entre semanas y los picos/valles que tienen los datos.

Se han buscado los parámetros que mejor funcionan para el modelo:

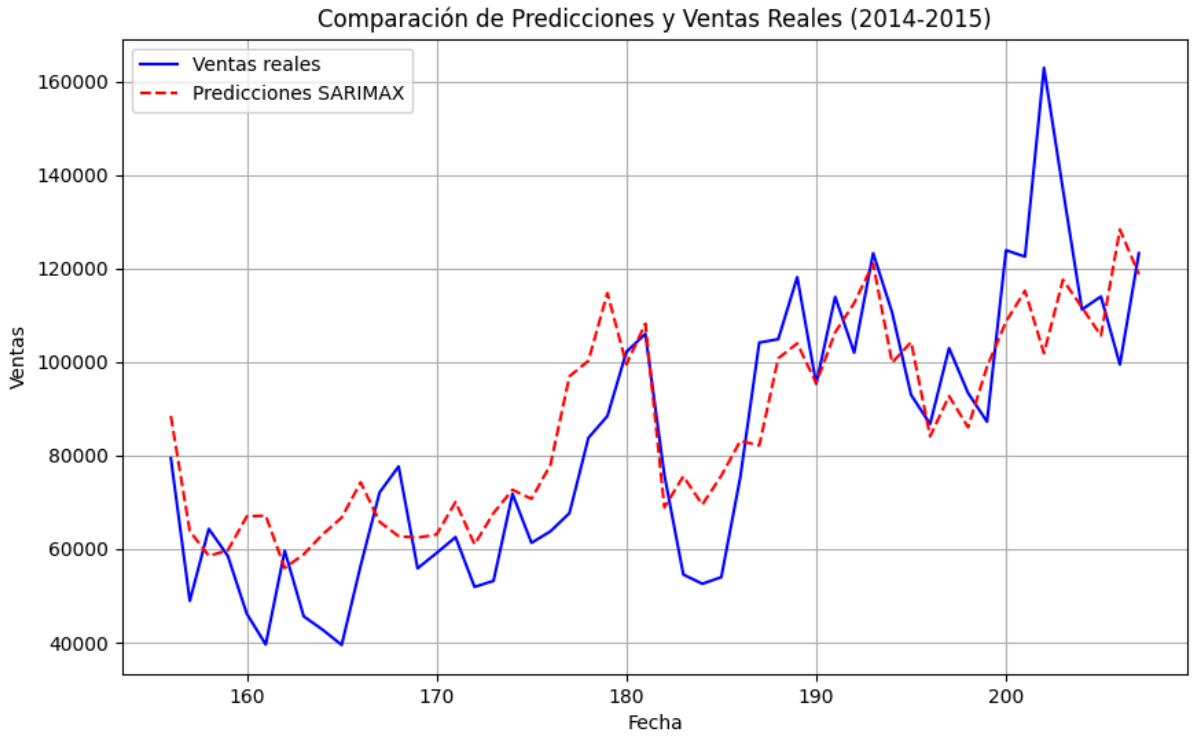
Mejores parámetros: ((2, 2, 2), (0, 1, 1, 52))

Mejor AIC: 1043.807924158313

Con estos parámetros se ha vuelto a entrenar el modelo y se ha vuelto a reevaluar:

RMSE para los datos de 2014 y 2015: 16428.42

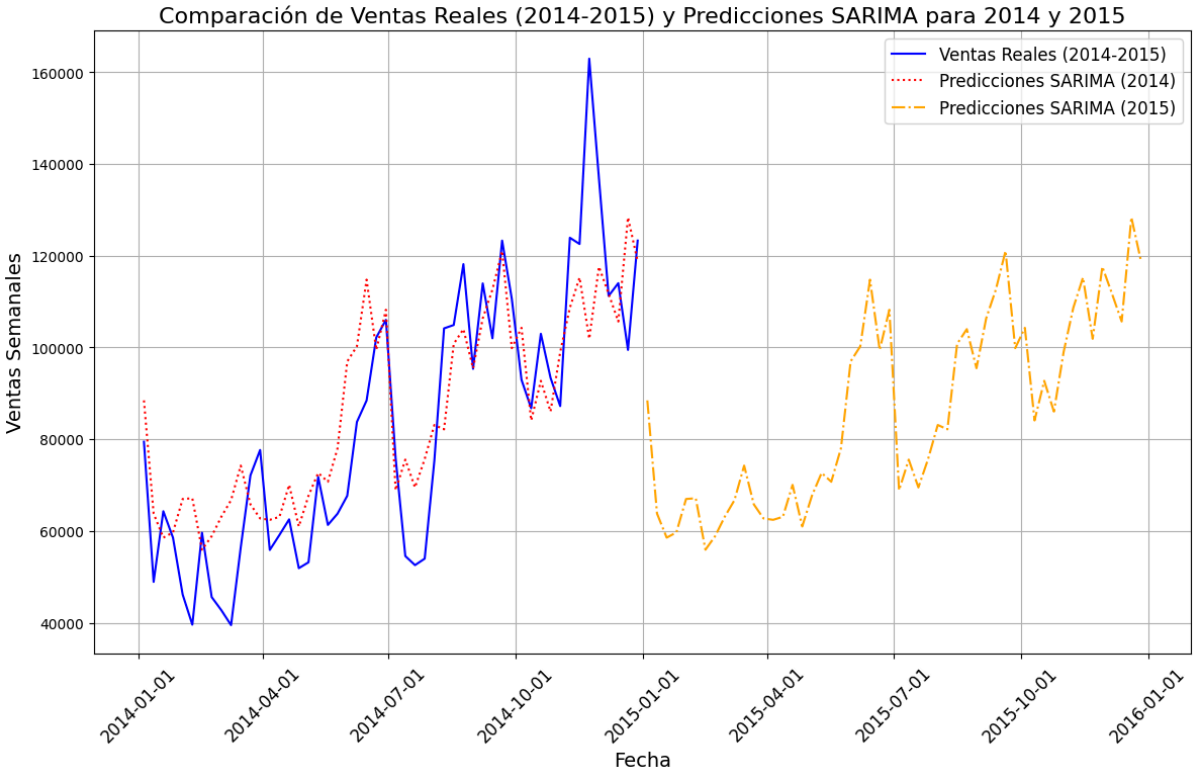
Gráfico 7: Datos de 2014 reales enfrentados a predicción de 2014



Fuente: A partir de la librería de matplotlib y las predicciones del modelo

También hemos vuelto a plantear el mismo gráfico para que el 2015 lo cogiera igual y lo comparara con las predicciones de 2014 y los datos reales de 2014.

Gráfico 8: Predicciones con los mejores parámetros de 2014 y 2015



Fuente: A partir de la librería matplotlib

Como se puede observar parece que el modelo sin acabar de recoger la tendencia a la alza que tiene el 2014 en cuanto a ventas para que se aplique también a los picos/valles que tienen las ventas.

4. Conclusiones y mejoras del modelo

Durante el proyecto de modelado de machine learning que nos ha atendido hemos ido aprendiendo las diferentes características que tenía nuestra base de datos sobre ventas. Se trataba de una base de datos con muchos desniveles diarios y semanales que hacía que el modelado para predecir las ventas no fuera fácil de captar por nuestros modelos de ARIMA y SARIMA.

No obstante, ha quedado de manifiesto que tanto el negocio como los datos se verían beneficiado de que el modelo plantearía dar los datos en formato semanal. No sólo porque visualmente y gráficamente era mucho más entendible sino que también porque daba mucha más capacidad de acción y de maniobra al negocio para poder adaptarse a las diferencias efemérides que día a día cuesta más captar y mes a mes es más facil saber que meses van a ser más fuertes y que meses van a ser más tranquilos en cuanto a facturación. Por lo tanto, también podrán adaptar los gastos de personal semana a semana y los momentos en lo que se pueden hacer diferentes actividades que permitan a los trabajadores establecer la organización y que momentos el trabajador tiene que estar 100% por lo que pueda pedir el cliente.

Una de las dificultades más notorias que se han tenido de cara a plantearlo semanal era a la hora de evaluar los datos con los datos reales por la diferencia de semanas que se tienen de un año a otro.

Finalmente, tal vez con más tiempo y la posibilidad de poder trabajar el modelado con mayor profundidad hubiera planteado otros modelos de Machine Learning de timeseries como prophet o ETS (Error, Trend, Seasonality) que tal vez hubiera dado diferentes resultados y más provechosos para el negocio.