Reporte: Sistema de Predicción de Ventas usando LSTM

Motivación: Mi enfoque laboral actual es en el área de la manufactura 4.0, también llamada cuarta revolución industrial. Los proyectos en los que estoy participando buscan implementar herramientas computacionales tales como Optimización lineal, ML, DL y cualquier herramienta digital con el objetivo de optimizar procesos de manufactura de diversas empresas.

El área en la que participo actualmente se enfoca en generar planeaciones de producción optimizando ganancias y tiempos productivos, esto generando el que, cuanto, cuando, donde, para quien y como se deben producir los productos de una planta manufacturera.

Uno de los puntos necesarios para lograr generar un plan de producción adecuado, es contar con un Forecast robusto que permita visualizar las tendencias de demanda de los productos, es por esta razón que mi propuesta radica en partir del ejemplo del generador de texto a nivel de carácter usando un LSTM, pero modificándolo con el objetivo de no generar una secuencia de texto a partir de una oración inicial si no generar los valores de tendencia de la demanda de un producto basado en los últimos registros de demanda del producto que se tienen.

1. Descripción General

El proyecto implementa un sistema de predicción de ventas utilizando redes neuronales LSTM (Long Short-Term Memory), diseñadas para capturar patrones temporales complejos en datos de ventas.

2. Arquitectura del Sistema

Debido a la complejidad del proyecto y el tiempo de elaboración, se opto por enfocar los esfuerzos en construir una base solida sobre la cual ir mejorando el modelo para obtener predicciones cada vez mejores y que sean aplicables a casos reales. Para ello el primer desarrollo mostrado aquí, consiste en la generación de datos sintéticos que buscan representar patrones de datos de ventas como lo son los de ciclo estacional subiendo y bajando a lo largo de los años, o los que tienen patrones mas locales como mayores ventas los fines de semana.

2.1 Generación de Datos Sintéticos

- Se implementaron dos métodos de generación de datos:
- generate_senoidal_sales_data(): Genera datos con patrones estacionales

- generate_weekend_sales_data(): Genera datos con patrones semanales y picos en fines de semana

Como punto de mejora, se decidió incluir datos temporales de los registros creados, e indicadores de interés tales como que día de la semana se trata, la temporada, etc. Esto con el objetivo de construir una base que permita más adelante agregar más características como podrían ser de tipo ambiental o histórico.

2.2 Características Temporales

El sistema utiliza las siguientes características:

- Año
- Mes
- Día de la semana
- Día del año
- Indicador de fin de semana
- Temporada (Invierno, Primavera, Verano, Otoño)

3. Metodología

Basándose en los ejemplos presentados en la página de Keras, se crearon las secuencias de registros que serían usadas como conjunto de entrenamiento, las funciones se generaron de manera que sea sencillo adaptarlas y mejorar para lograr crear estas secuencias de registros sin mayor inconveniente al agregar más características a los mismos.

3.1 Preparación de Datos

1. **División de Datos**:

- 90% para entrenamiento (20% validación)
- 10% para prueba

2. **Preprocesamiento**:

- Normalización de datos usando MinMaxScaler
- Creación de secuencias para el modelo LSTM
- Manejo de características temporales

Se implementaron tres modelos partiendo de una base muy simple como la que se mostraba en el ejemplo de Keras y se fue añadiendo complejidad y funcionalidades para explorar mas alternativas de cara a buscar un mejor modelo para predecir datos más caóticos o complejos.

3.2 Modelos LSTM Implementados

Se implementaron tres variantes del modelo:

1. **Modelo Simple**:

- Una capa LSTM
- Capa de salida densa

2. **Modelo Deep 1**:

- Tres capas LSTM (128, 64, 32 unidades)
- Capas de Dropout (0.2)
- Capas densas intermedias

3. **Modelo Deep 2**:

- Tres capas LSTM (256, 128, 64 unidades)
- Regularización L2
- Dropout más agresivo (0.3)
- Más capas densas intermedias

3.3 Entrenamiento

- Optimizador: Adam con learning rate de 0.001
- Función de pérdida: Huber (Por ser recomendada para LSTM en casos similares)
- Métrica: MAE (Error Absoluto Medio)
- Early Stopping con paciencia de 10 épocas
- Batch size: 32
- Validación: 20% de los datos de entrenamiento

4. Sistema de Predicción

Se desarrolló una función que permita construir la secuencia de predicciones dada una cantidad de ellas definida por el usuario, estas predicciones serán analizadas y visualizadas más adelante.

4.1 Proceso de Predicción

1. **Preparación de Secuencias**:

- Uso de la última secuencia conocida
- Incorporación de características temporales futuras

2. **Predicción Recursiva**:

- Predicción día a día
- Actualización de la secuencia con cada predicción
- Manejo de características temporales para fechas futuras

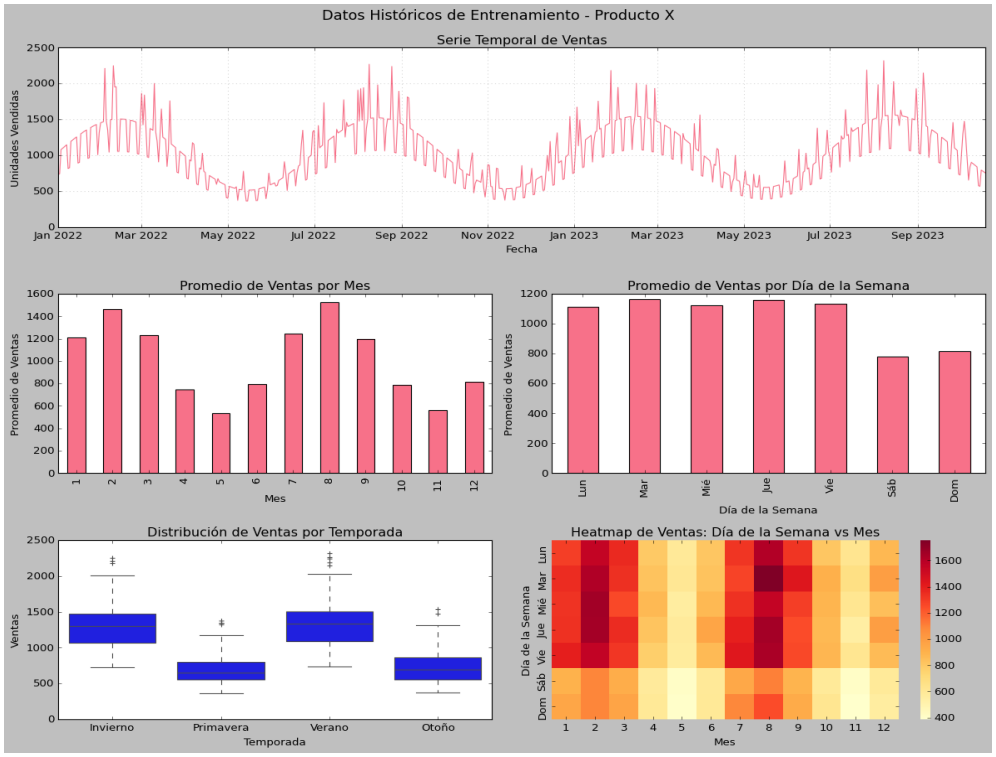
4.2 Métricas de Evaluación

- MAE (Error Absoluto Medio)
- MSE (Error Cuadrático Medio)
- RMSE (Raíz del Error Cuadrático Medio)
- MAPE (Error Porcentual Absoluto Medio)

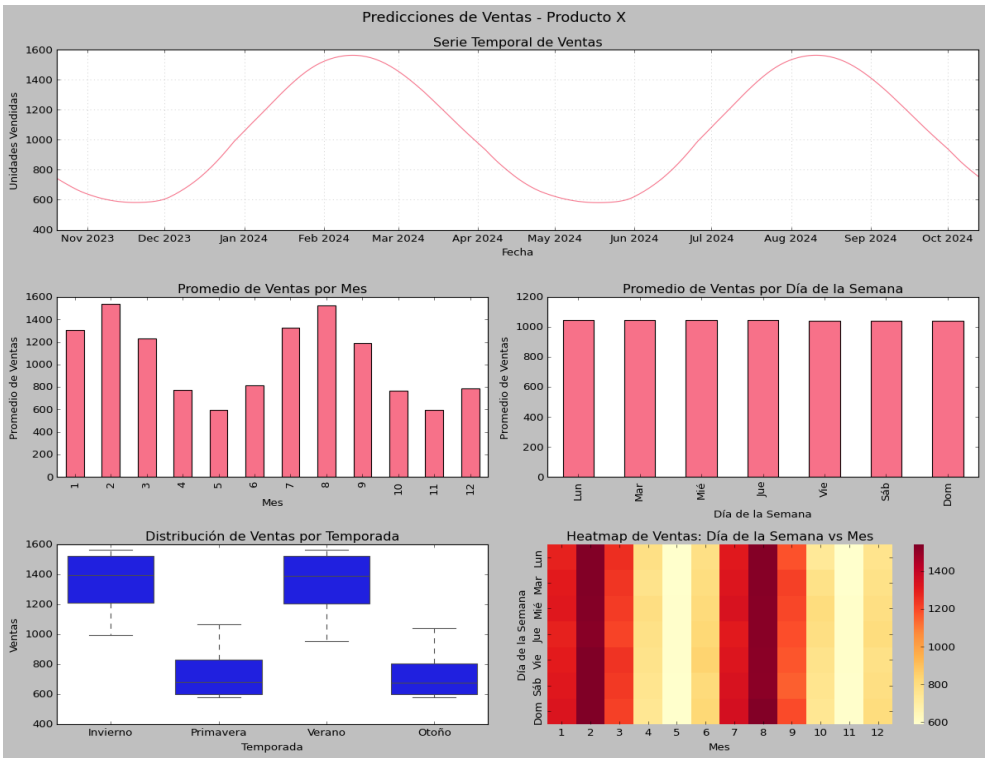
Se desarrollo una función para mostrar un conjunto de gráficos de interés sobre los dataframe de datos.

5. Visualización y Análisis

Con las funciones de visualización es posible generar gráficos de interés de manera estandarizada.



Gráficos de los datos de entrenamiento.



Gráficos de las predicciones de ventas.

5.1 Gráficos Implementados

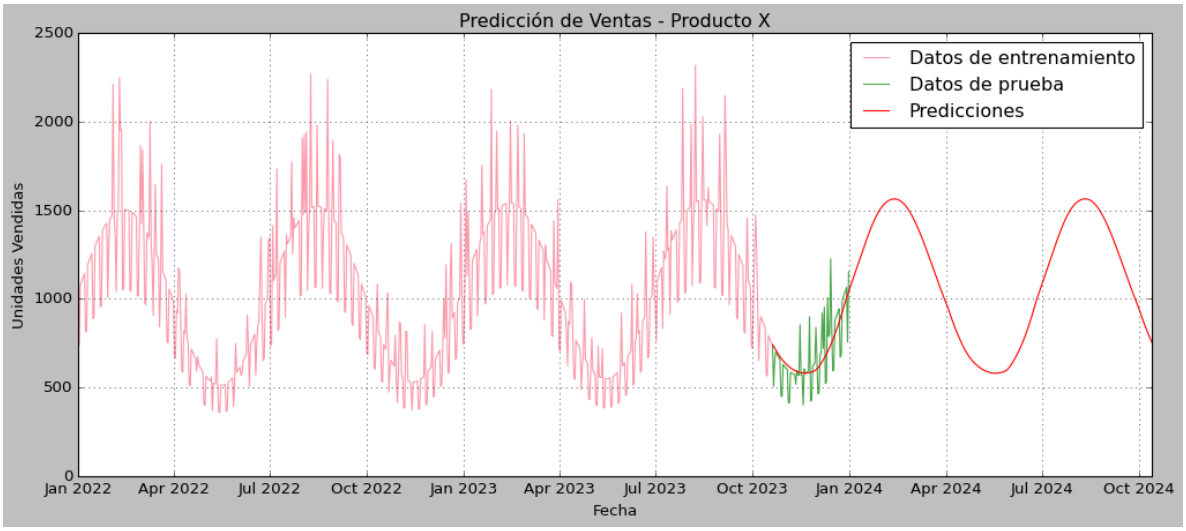
- Curvas de pérdida durante el entrenamiento
- Datos históricos de ventas
- Predicciones vs datos reales
- Comparación de patrones estacionales

5.2 Análisis de Resultados

- Evaluación de la precisión de las predicciones
- Análisis de patrones estacionales
- Identificación de tendencias

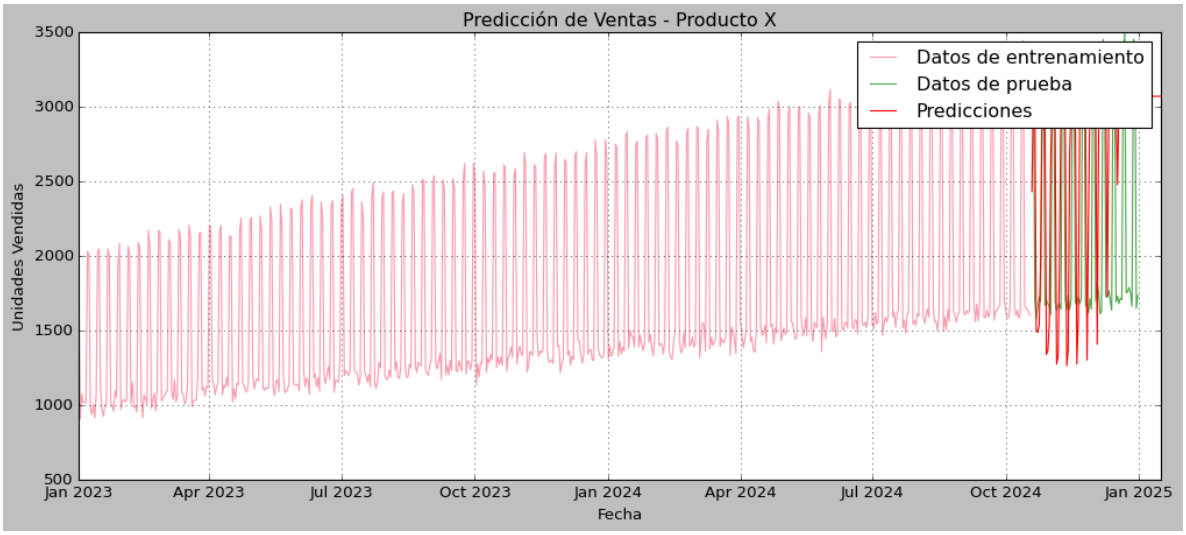
6. Conclusiones

Con el caso de los datos senoidales se logro demostrar que el modelo es capaz de reconocer patrones estacionales a lo largo del tiempo. Claramente se puede observar la reconstrucción del ciclo de valores de ventas. Esto nos dice que es posible predecir tendencias de largo plazo. Sin embargo, el modelo actual logra esto al usar hiperparametros que permitan observar estos comportamientos, como lo es una máxima longitud de las secuencias grande, en este caso al usar una longitud de 180 se lograba tener una resolución adecuada para observar estos patrones, pero a costo de perder precisión en las variaciones de corto plazo.



Con el caso de picos en fines de semana se ataco el caso contrario, aquí se mostro que el modelo es capas de interpretar y reconstruir patrones de tendencia mas corta, como lo es semanal, para ello se utilizó una longitud de cadena mas corta, con solo

14 días, con esta resolución el resultado refleja el comportamiento con picos cada cierto tiempo. Sin embargo, flaquea al interpretar tendencias a largo plazo, como lo es la tendencia de aumento en ventas al pasar el tiempo.



El sistema implementado demuestra:

- Capacidad para capturar patrones temporales complejos
- Flexibilidad en la configuración del modelo
- Robustez en el manejo de datos

8. Recomendaciones para Mejoras Futuras

El siguiente objetivo será mejorar el modelo para lograr incluir ambas propiedades, una capacidad de detección de patrones de largo plazo y picos y variaciones de corto plazo, para ello se pone como principales puntos los siguientes:

1. **Mejoras Técnicas**:

- Experimentación con arquitecturas más complejas
- Optimización de hiperparámetros

2. **Mejoras Funcionales**:

- Incorporación de más características externas
- Implementación de análisis de incertidumbre