



UNIVERSIDAD  
**AUSTRAL**

Universidad Austral - Maestría en Ciencia de Datos

Laboratorio de Implementación III

Trabajo Final

**Docente**

Gustavo Denicolay

**Alumno**

Matias Tibaldo

# Predicción de Ventas con XGBoost

## Resumen

Se desarrolló un sistema integral de predicción de ventas basado en machine learning, implementando un enfoque de series temporales con XGBoost como modelo principal. La solución aborda la predicción de toneladas vendidas para productos específicos en períodos futuros, incorporando técnicas avanzadas de feature engineering, normalización adaptativa y optimización de hiperparámetros.

## 1. Descripción del Problema

El objetivo principal fue predecir las ventas (en toneladas) de productos específicos para el período 2020-02, utilizando datos históricos de ventas que incluyen información temporal, características de productos, clientes y variables exógenas del mercado.

### Datasets Utilizados:

- **sell\_in:** Datos históricos de ventas por período, cliente y producto
- **tb\_products:** Información categórica de productos (categorías, marcas, tamaños)
- **tb\_stocks:** Niveles de inventario por período y producto
- **product\_id\_aprededir:** Lista de productos objetivo para predicción

## 2. Arquitectura de la Solución

La solución implementa un pipeline completo que incluye:

1. **Preprocesamiento y Feature Engineering**
2. **Normalización Adaptativa por Producto**
3. **Encoding de Variables Categóricas**
4. **Modelado con XGBoost y Sample Weighting**
5. **Optimización de Hiperparámetros**
6. **Predicción con Múltiples Semillas**

## 3. Feature Engineering Avanzado

### 3.1 Features Temporales

Se crearon múltiples características temporales para capturar patrones estacionales y tendencias:

- **Componentes de fecha:** Año, mes, quarter, días del mes
- **Encodings cíclicos:** Transformaciones senoidales y cosenoidales para capturar la ciclicidad mensual y trimestral
- **Variables exógenas:** Cotización del dólar, IPC, períodos de receso escolar, meses con feriados y anomalías políticas

### 3.2 Features de Lag y Diferencias

Implementación de características de rezago temporal para capturar dependencias temporales y cambios en las tendencias de venta

- **Lags:** Valores históricos de 1 a 36 períodos anteriores
- **Delta Lags:** Diferencias entre valores de lag consecutivos (1 a 35 períodos)

### 3.3 Features de Ventanas Deslizantes

Características estadísticas calculadas sobre ventanas temporales que permiten el suavizado de tendencias y detección de patrones de picos y valles

- **Medias móviles:** Promedios de 2 a 36 períodos consecutivos
- **Indicadores de extremos:** Variables booleanas indicando si el valor actual es el mínimo o máximo en la ventana correspondiente

### 3.4 Features de Prophet

Integración del modelo Prophet para capturar componentes de series temporales. Las features de Prophet se normalizaron usando la misma escala que las variables de toneladas

- **Tendencia:** Componente de tendencia a largo plazo
- **Estacionalidad:** Patrones estacionales anuales
- **Predicciones:** Valores estimados y intervalos de confianza

### 3.5 Clustering con Dynamic Time Warping (DTW)

Implementación de clustering avanzado para productos similares (Los clusters aparecieron con importancia relevante en el modelo final):

- **Metodología:** Agrupación de series temporales basada en similitud morfológica
- **Configuración:** 50 clusters usando distancia DTW
- **Propósito:** Identificar productos con patrones de venta similares para mejorar predicciones

## 4. Target Engineering: Enfoque de Ratios

## 4.1 Creación del Target

En lugar de predecir toneladas absolutas, se implementó un enfoque de ratios, evitando problemas de escala entre productos con volúmenes muy diferentes. Al finalizar las predicciones se convierten de vuelta a toneladas usando el último valor conocido del período 2019-12:

- **Fórmula:**  $\text{target\_ratio} = \text{tn\_actual} / (\text{tn\_siguiente\_1} + \text{tn\_siguiente\_2})$

## 5. Normalización Adaptativa

### 5.1 Estrategia de Normalización

Se implementó normalización estándar (media 0, desviación 1) específica por producto:

- **Alcance:** Variables relacionadas con toneladas (lags, delta lags, medias móviles, features de Prophet)
- **Granularidad:** 780 escaladores independientes (uno por cada serie temporal de producto)
- **Prevención de Data Leakage:** La normalización se aplica después del split temporal

Entre las ventajas que se pueden mencionar al utilizar normalización con este enfoque son: **Adaptabilidad**, ya que cada producto mantiene su distribución característica, **Robustez** porque permite que productos de alto volumen dominen el entrenamiento y **Consistencia** porque buscan mantener las relaciones relativas dentro de cada producto

## 6. Encoding de Variables Categóricas

### 6.1 Metodología Implementada

- **Técnica:** Label Encoding para variables categóricas jerárquicas
- **Variables procesadas:** cat1, cat2, cat3, brand
- **Consideración especial:** Inclusión de product\_id debido a su correlación intrínseca con volúmenes de venta

### 6.2 Decisión sobre Product\_ID

Inicialmente se consideró excluir product\_id por alta cardinalidad, pero se identificó una correspondencia significativa entre el código del producto y los volúmenes de venta, justificando su inclusión en el modelo.

## 7. Sample Weighting: Balanceando el Impacto de Productos

## 7.1 Problemática Identificada

Los productos con mayores volúmenes de venta tendían a dominar la función de pérdida, sesgando las predicciones.

## 7.2 Solución Implementada

Se implementó un sistema de pesos inversos:

$$\text{peso\_producto} = 1 / (\text{toneladas\_totales\_producto} + \text{epsilon})$$
$$\text{peso\_normalizado} = \text{peso} / \text{suma\_pesos} * \text{num\_productos}$$

## 7.3 Impacto

Esta técnica fue identificada como **FUNDAMENTAL** en el documento original, mejorando significativamente la métrica de error al dar importancia equilibrada a productos de diferentes escalas.

# 8. Modelado con XGBoost

## 8.1 Configuración del Modelo

- **Algoritmo:** XGBRegressor con configuraciones optimizadas
- **Parámetros clave:**
  - **max\_bin=1024:** Maximizado para mejorar precisión en regresión
  - **early\_stopping\_rounds=50:** Prevención de overfitting
  - Parámetros de regularización ajustados

## 8.2 Validación Temporal


- **Split:** División temporal estricta (datos anteriores a 2019-12 para entrenamiento)
- **Objetivo:** Simular condiciones reales de predicción evitando data leakage

# 9. Optimización de Hiperparámetros

Se optó por utilizar optuna con optimización bayesiana ya que permite realizar una búsqueda paralela y distribuida, convergencia eficiente

## 9.1 Espacio de Búsqueda

- max\_depth: (3, 20)

- 
- learning\_rate: (0.01, 0.3)
  - n\_estimators: (100, 1000)
  - min\_child\_weight: (1, 10)
  - subsample: (0.5, 1.0)
  - colsample\_bytree: (0.5, 1.0)
  - reg\_alpha: (0.0, 1.0)

## 10. Estrategia de Predicción con Múltiples Semillas

### 10.1 Metodología

Para capturar la variabilidad por inicialización aleatoria lo que permite reducir la varianza y aumentar la robustez

- **Semillas múltiples:** 5 modelos con diferentes random\_state
- **Agregación:** Promedio de las predicciones individuales

### 10.2 Proceso de Reconstrucción


1. Predicción de ratios usando features engineered
2. Obtención del último valor conocido (2019-12)
3. Conversión de ratios a toneladas absolutas
4. Desnormalización usando escaladores por producto
5. Agregación por producto y período

## 11. Características Técnicas Destacadas

### 11.1 Manejo de Valores Faltantes

Se optó por la preservación de NaN para variables numéricas (como indica la metodología) teniendo como excepción el completado con ceros para combinaciones producto-cliente sin historial.

### 11.2 Prevención de Data Leakage



al considerarse un **punto crítico** hay que tener en cuenta hacer split temporal antes de cualquier transformación, es por esto que Escaladores y Encoders se ajustan sólo con datos de entrenamiento

### 11.3 Escalabilidad y Eficiencia

Para ejecutar los diferentes scripts se eligió trabajar con google cloud con máquinas virtuales de 64gb. A su vez también se aplicaron breakpoints en el código para guardar estados intermedios de ejecución que luego permiten avanzar más rápidamente en caso de error/eliminación/stop de la máquina virtual.

## 12. Resultados y Consideraciones

### 12.1 Output del Sistema

Se realiza las predicciones por producto y período (2020-02) con las toneladas agregadas por producto

### 12.2 Aspectos de Interpretabilidad

El modelo XGBoost permite analizar la importancia de features, identificando:

- Variables de lag más relevantes
- Impacto de características estacionales
- Contribución de clustering DTW
- Relevancia de variables exógenas

## 13. Conclusiones y Lecciones Aprendidas

### 13.1 Factores Clave de Éxito

1. **Sample weighting:** Crucial para balancear productos de diferentes escalas
2. **Feature engineering temporal:** Captura efectiva de patrones complejos
3. **Normalización por producto:** Preserva características individuales
4. **Enfoque de ratios:** Soluciona problemas de escala naturalmente

### 13.2 Innovaciones Implementadas

- Integración de múltiples técnicas de feature engineering
- Clustering DTW para identificación de productos similares
- Sistema de pesos adaptativos por producto
- Pipeline robusto con manejo de dependencias opcionales
- Recuperación en puntos intermedios si hay errores



### 13.3 Extensibilidad Futura

El sistema diseñado permite:

- Incorporación de nuevas variables exógenas
- Ajuste de ventanas temporales
- Experimentación con otros algoritmos de clustering
- Integración de features adicionales de Prophet

## 14. Recomendaciones para mejoras futuras

- Exploración de modelos de deep learning para series temporales
- Desarrollo de modelos específicos por categoría de producto
- Implementación de técnicas de ensemble avanzadas

---

*Este informe documenta la implementación completa del sistema de predicción de ventas, destacando las decisiones técnicas clave y su justificación metodológica.*