

06__consolidado__prescriptivo

November 11, 2025

```
[4]: # =====
# Consolidado Prescriptivo - Integración Final (versión robusta)
# =====
import pandas as pd

# Cargar resultados previos
multi = pd.read_csv("resultados_multi_periodo.csv")
transf = pd.read_csv("resultados_transferencias.csv")
forecast = pd.read_csv("resultados_forecast.csv")

# --- Normalizar nombres ---
multi.rename(columns={"periodo": "semana"}, inplace=True)
forecast.rename(columns={"semana_a_futuro": "semana"}, inplace=True)

# Asegurar consistencia de tipos
forecast["semana"] = forecast["semana"].astype(int)
multi["semana"] = multi["semana"].astype(int)

# --- Ajuste para transferencias ---
if "tienda" not in transf.columns and "destino" in transf.columns:
    transf["tienda"] = transf["destino"]

transf.rename(columns={
    "cantidad_transferida": "transferencia",
    "costo_total": "costo_transferencia"
}, inplace=True)

# --- Verificación previa ---
print("Multi:", multi.shape, "| Transf:", transf.shape, "| Forecast:", forecast.
      ↪shape)
print("Columnas multi:", multi.columns.tolist())
print("Columnas forecast:", forecast.columns.tolist())
print("Columnas transferencias:", transf.columns.tolist())

# --- Unificación ---
consolidado = (
    forecast
```

```

        .merge(multi, how="left", on=["sku", "semana"]) # sin 'tienda'
        .merge(transf[["tienda", "transferencia", "costo_transferencia"]],
        how="left", on="tienda")
    )

# --- Limpieza ---
consolidado.fillna(0, inplace=True)
consolidado.drop_duplicates(inplace=True)

# --- Auditoría rápida ---
print("\n Consolidado generado:")
print(consolidado.head(10))
print("Filas totales:", len(consolidado))

# --- Exportar ---
consolidado.to_parquet("consolidado_prescriptivo.parquet", index=False)
print("\n Archivo 'consolidado_prescriptivo.parquet' generado correctamente.")

```

Multi: (1200, 4) | Transf: (2, 5) | Forecast: (4800, 5)
 Columnas multi: ['sku', 'semana', 'pedido', 'stock']
 Columnas forecast: ['sku', 'tienda', 'fecha_predicha', 'semana',
 'prediccion_ARIMA']
 Columnas transferencias: ['origen', 'destino', 'transferencia',
 'costo_transferencia', 'tienda']

Consolidado generado:

	sku	tienda	fecha_predicha	semana	prediccion_ARIMA	pedido \
0	HM000001	TIENDA001	2024-12-29	1	1.60	19.824658
1	HM000001	TIENDA001	2025-01-05	2	1.77	0.000000
2	HM000001	TIENDA001	2025-01-12	3	1.75	0.000000
3	HM000001	TIENDA001	2025-01-19	4	1.75	0.000000
4	HM000001	TIENDA001	2025-01-26	5	1.75	0.000000
5	HM000001	TIENDA001	2025-02-02	6	1.75	0.000000
6	HM000001	TIENDA001	2025-02-09	7	1.75	0.000000
7	HM000001	TIENDA001	2025-02-16	8	1.75	0.000000
8	HM000001	TIENDA002	2024-12-29	1	1.39	19.824658
9	HM000001	TIENDA002	2025-01-05	2	1.50	0.000000

	stock	transferencia	costo_transferencia
0	18.172603	500.0	2500.0
1	16.520548	500.0	2500.0
2	14.868493	500.0	2500.0
3	13.216438	500.0	2500.0
4	11.564384	500.0	2500.0
5	9.912329	500.0	2500.0
6	8.260274	500.0	2500.0
7	6.608219	500.0	2500.0
8	18.172603	0.0	0.0

9 16.520548 0.0 0.0
Filas totales: 4800

Archivo 'consolidado_prescriptivo.parquet' generado correctamente.

```
[5]: # =====  
# BLOQUE 2 - ANÁLISIS Y VISUALIZACIÓN PRESCRIPTIVA  
# =====  
import pandas as pd  
import matplotlib.pyplot as plt  
  
# Cargar consolidado  
df = pd.read_parquet("consolidado_prescriptivo.parquet")  
  
# --- 1 Resumen general ---  
print("Columnas disponibles:", df.columns.tolist())  
print("Registros totales:", len(df))  
  
# Métricas agregadas por tienda  
resumen_tienda = (  
    df.groupby("tienda")  
    .agg(  
        demanda_total=("prediccion_ARIMA", "sum"),  
        pedido_total=("pedido", "sum"),  
        stock_promedio=("stock", "mean"),  
        costo_transferencia=("costo_transferencia", "sum")  
    )  
    .reset_index()  
)  
print("\nResumen por tienda:")  
print(resumen_tienda)  
  
# --- 2 Top 10 SKU por demanda pronosticada ---  
top_skus = (  
    df.groupby("sku")["prediccion_ARIMA"]  
    .sum()  
    .sort_values(ascending=False)  
    .head(10)  
)  
print("\nTop 10 SKU por demanda pronosticada:")  
print(top_skus)  
  
# --- 3 Visualización: SKU con mayor demanda pronosticada ---  
sku_top = top_skus.index[0]  
ejemplo = df[df["sku"] == sku_top]  
  
plt.figure(figsize=(10,5))
```

```

plt.plot(
    ejemplo["semana"],
    ejemplo["prediccion_ARIMA"],
    label="Demanda pronosticada",
    linestyle="--",
    marker="o"
)
plt.plot(
    ejemplo["semana"],
    ejemplo["pedido"],
    label="Pedido óptimo (s,Q)",
    linewidth=2
)
plt.title(f"SKU {sku_top} - Pronóstico vs Política de Reabastecimiento")
plt.xlabel("Semana")
plt.ylabel("Unidades")
plt.legend()
plt.grid(True)
plt.show()

# --- 4 Visualización: Costo logístico por tienda ---
plt.figure(figsize=(8,4))
plt.bar(resumen_tienda["tienda"], resumen_tienda["costo_transferencia"])
plt.title("Costo total de transferencias por tienda")
plt.xlabel("Tienda")
plt.ylabel("Costo total (S/)")
plt.xticks(rotation=45)
plt.grid(axis="y")
plt.show()

```

Columnas disponibles: ['sku', 'tienda', 'fecha_predicha', 'semana', 'prediccion_ARIMA', 'pedido', 'stock', 'transferencia', 'costo_transferencia']
Registros totales: 4800

Resumen por tienda:

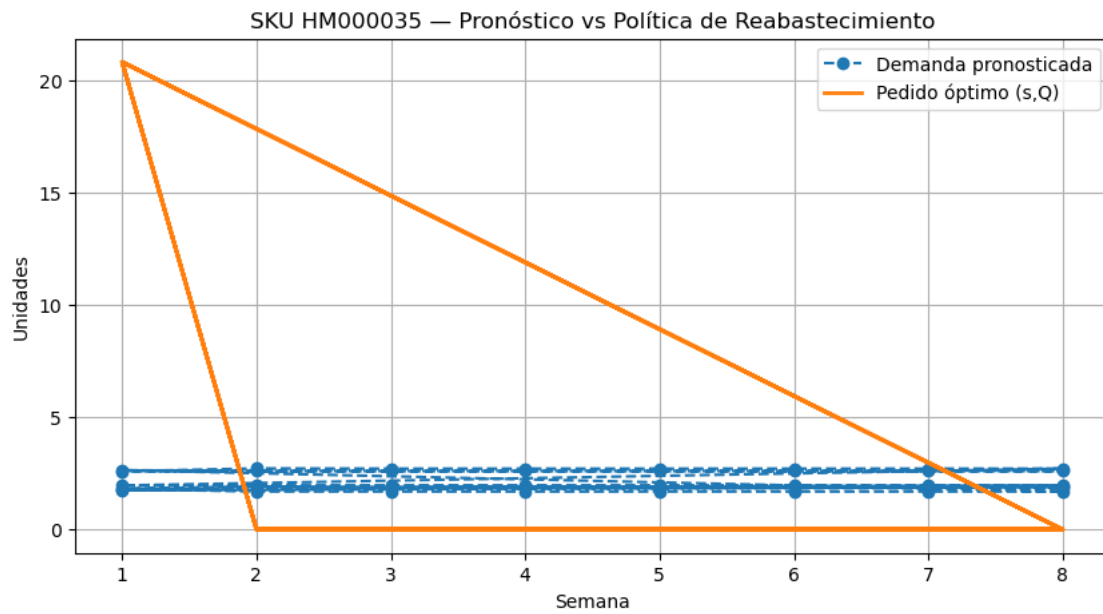
	tienda	demanda_total	pedido_total	stock_promedio	costo_transferencia
0	TIENDA001	1284.34	1898.531513	11.865822	2000000.0
1	TIENDA002	1296.88	1898.531513	11.865822	0.0
2	TIENDA003	1296.17	1898.531513	11.865822	1200000.0
3	TIENDA004	1283.88	1898.531513	11.865822	0.0
4	TIENDA005	1295.03	1898.531513	11.865822	0.0
5	TIENDA006	1331.57	1898.531513	11.865822	0.0

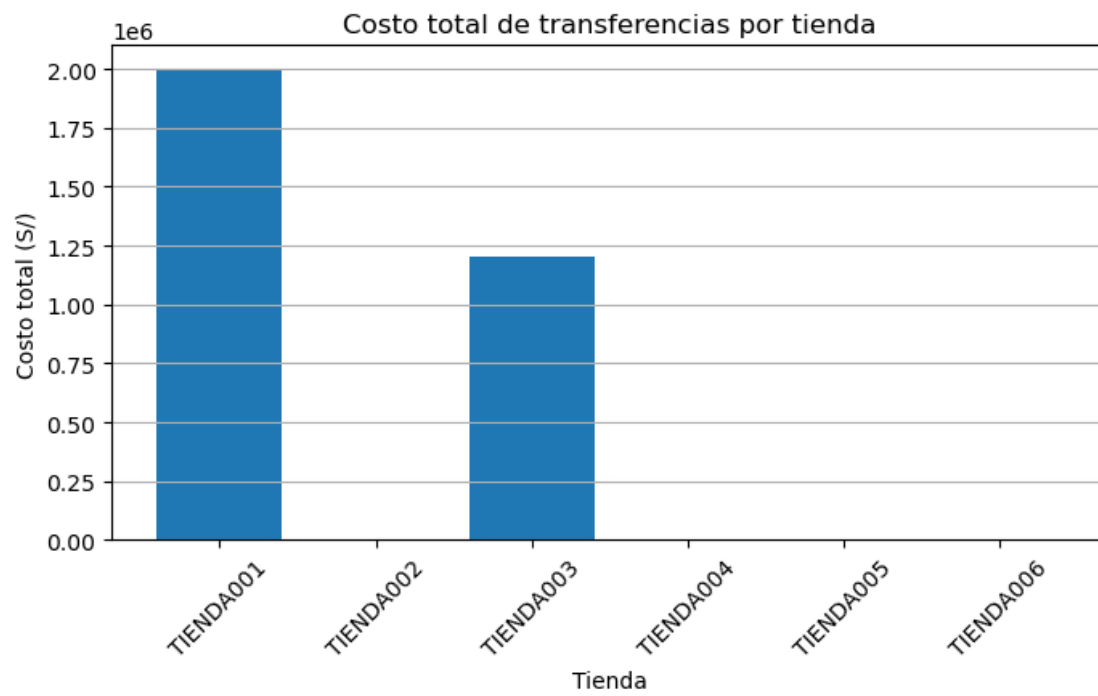
Top 10 SKU por demanda pronosticada:

sku	
HM000035	101.52
HM000051	94.51
HM000014	93.93

HM000092	93.17
HM000057	91.02
HM000006	90.85
HM000100	89.18
HM000084	88.79
HM000003	88.78
HM000095	88.43

Name: prediccion_ARIMA, dtype: float64





[]: