Analisis Prueba

Análisis Con Series de Tiempo

importamos las librerias necesarias:

```
#!pip install numpy
#!pip install pandas
#!pip install statsmodels
#!pip install matplotlib

import numpy as np
import pandas as pd
from statsmodels.tsa.arima.model import ARIMA
import matplotlib.pyplot as plt
```

Para este ejemplo lo que hacemos es generar datos aleatorios:

```
# Generar datos simulados
np.random.seed(42)
data = np.random.rand(100) * 200  # 100 meses de datos de ventas
meses = pd.date_range('2020-01-01', periods=100, freq='ME')
df = pd.DataFrame(data, index=meses, columns=['Ventas'])
```

En el siguiente paso es importante separa el conjunto de datos en entrenamiento y pruba.

```
# Dividir en train y test
train = df.iloc[:-10]
test = df.iloc[-10:]
```

Ahora creamos el modelo, en este caso ARIMA

```
# Modelo ARIMA
model = ARIMA(train, order=(5,1,0))  # Ejemplo: ARIMA(5,1,0)
model_fit = model.fit()
```

Creamos las predicciones:

```
# Predicción
forecast = model_fit.forecast(steps=10)
```

Los margenes de error:

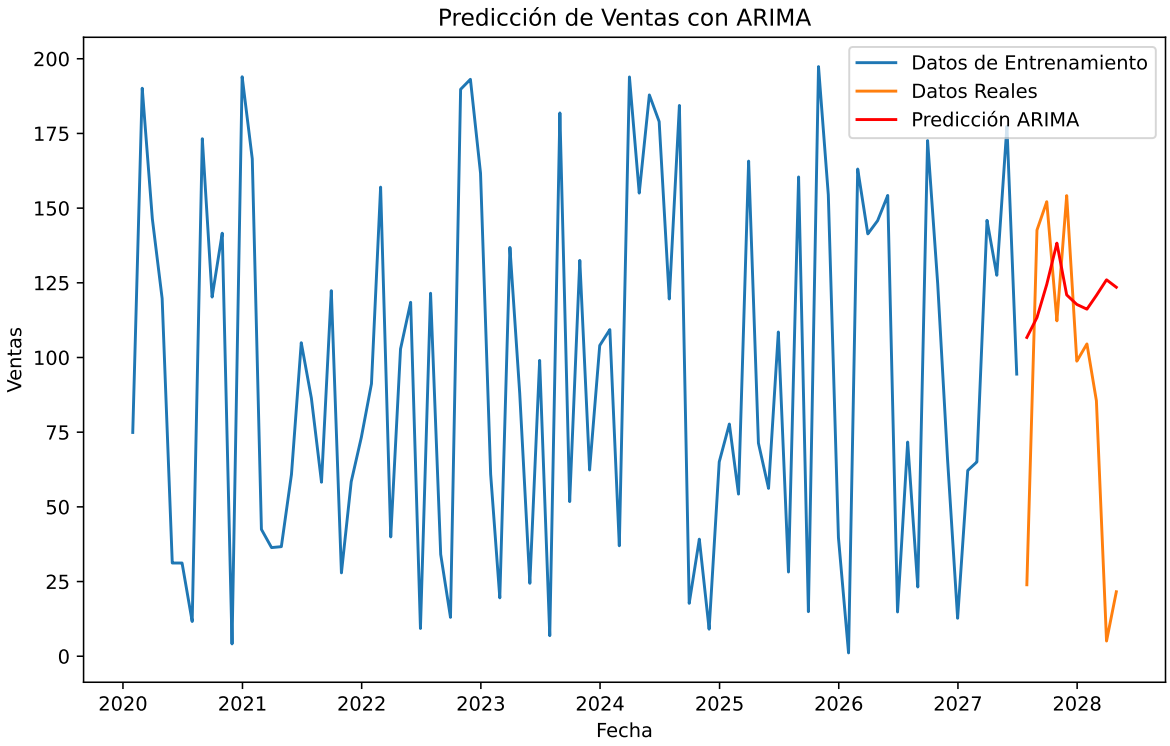
```
# Métricas de error
mae = np.mean(np.abs(forecast - test['Ventas']))
rmse = np.sqrt(np.mean((forecast - test['Ventas'])**2))

print(f'MAE: {mae}')
print(f'RMSE: {rmse}')
```

```
MAE: 48.77863330614092
RMSE: 60.839877541796334
```

Grafica de los valores reales y las predicciones:

```
# Gráfica
plt.figure(figsize=(10, 6))
plt.plot(train.index, train['Ventas'], label='Datos de Entrenamiento')
plt.plot(test.index, test['Ventas'], label='Datos Reales')
plt.plot(test.index, forecast, label='Predicción ARIMA', color='red')
plt.title('Predicción de Ventas con ARIMA')
plt.xlabel('Fecha')
plt.ylabel('Ventas')
plt.legend()
plt.show()
```


Predicción de Ventas con ARIMA
Datos de Entrenamiento
Datos Reales
Predicción ARIMA
Ventas
Fecha
200
175
150
125
100
75
50
25
0
2020
2021
2022
2023
2024
2025
2026
2027
2028