Time Series Forecasting of Champagne Sales

1 Introduction

This report analyzes the monthly Champagne sales dataset and applies different time series techniques such as decomposition, stationarity testing, ARIMA, and SARIMAX modeling. The goal is to forecast future sales.

2 Data Loading and Preprocessing

We begin by loading the dataset and preparing it for time series analysis.

```
import pandas as pd
df = pd.read_csv("/content/perrin-freres-monthly-champagne.csv")
df = df.dropna()
df.columns = ["month", "sales"]
df["month"] = pd.to_datetime(df["month"])
df.set_index("month", inplace=True)
```

Here, the dataset is loaded, missing values are removed, column names are renamed, and the month column is converted to a datetime index.

3 Time Series Decomposition

The time series is decomposed into trend, seasonality, and residual components.

```
from statsmodels.tsa.seasonal import seasonal_decompose
decom = seasonal_decompose(df["sales"])

plt.figure(figsize=(14,6))
plt.plot(df["sales"], label = "Original")
plt.legend(loc = "best")

plt.figure(figsize=(14,6))
plt.plot(decom.seasonal, label="Seasonality")

plt.figure(figsize=(14,6))
plt.plot(decom.resid, label="Residual")
```

This step shows the seasonal patterns and residual fluctuations in Champagne sales.

4 Stationarity Check (ADF Test)

We check whether the time series is stationary using the Augmented Dickey-Fuller (ADF) test.

```
from statsmodels.tsa.stattools import adfuller

def adfuller_test(data):
    result = adfuller(data)
    print("Test Statistics: " ,result[0])
    print("p value: " ,result[1])
    if(result[1] < 0.05):
        print("Stationary")
    else:
        print("Non Stationary")

adfuller_test(df["sales"])
```

The output indicates whether the time series is stationary or requires differencing.

5 Rolling Statistics

We plot rolling mean and standard deviation to further check stationarity.

```
rolling_mean = df.rolling(window=11).mean()
rolling_std = df.rolling(window=11).std()
df["ma_sales"] = rolling_mean["sales"]
df["rolling_std_sales"] = rolling_std["sales"]

df.plot(figsize=(14,6))
```

This visualization helps to see trends and variance changes over time.

6 Differencing

To remove seasonality and trends, differencing is applied.

```
df_diff_ma_sales = df["sales"] - df["ma_sales"]
adfuller_test(df_diff_ma_sales.dropna())

df_diff_12 = df["sales"] - df["sales"].shift(12)
df_diff_12.plot(figsize=(14,6))
```

This makes the data more stationary for ARIMA modeling.

7 ACF and PACF Plots

We plot autocorrelation and partial autocorrelation to identify ARIMA parameters.

```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf

plot_acf(df_diff_12.dropna())
plot_pacf(df_diff_12.dropna())
```

The plots help in selecting the AR and MA terms.

8 ARIMA Modeling

We build and fit an ARIMA model.

```
from statsmodels.tsa.arima.model import ARIMA

arima_model = ARIMA(df["sales"], order=(1,1,1))
arima_model_fit = arima_model.fit()
print(arima_model_fit.summary())

arima_predict = arima_model_fit.predict(start=80, end=105, dynamic=True)

plt.figure(figsize=(14,6))
plt.plot(df["sales"], label="Original")
plt.plot(arima_predict, label="Predict")
plt.legend(loc="best")
```

The ARIMA model provides in-sample predictions compared with actual values.

9 SARIMAX Modeling

Next, we apply a seasonal ARIMA (SARIMAX) model.

```
import statsmodels.api as sm

sarimax_model = sm.tsa.statespace.SARIMAX(
    df["sales"], order=(1,1,1), seasonal_order=(1,1,1,12)
)
sarimax_model_fit = sarimax_model.fit()
print(sarimax_model_fit.summary())

sarimax_predict = sarimax_model_fit.predict(start=80, end=105, dynamic=
   True)

plt.figure(figsize=(14,6))
plt.plot(df["sales"], label="Original")
plt.plot(sarimax_predict, label="Predict")
plt.legend(loc="best")
```

The SARIMAX model captures both seasonality and trend.

10 Forecasting

Finally, we forecast future Champagne sales using the SARIMAX model.

```
sarimax_forecast = sarimax_model_fit.forecast(steps=24)

plt.figure(figsize=(14,6))
plt.plot(df["sales"], label="Original")
plt.plot(sarimax_forecast, label="Forecast")
plt.legend(loc="best")
```

This forecast predicts Champagne sales for the next 24 months.

11 Conclusion

We successfully analyzed Champagne sales data, checked for stationarity, and applied ARIMA and SARIMAX models. SARIMAX performed better due to seasonality. The forecast provides useful insights for future sales trends.