In [20]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
get_ipython().run_line_magic('matplotlib', 'inline')
import warnings
warnings.filterwarnings('ignore')
import seaborn as sns
from pmdarima.arima import auto_arima
```

In [25]:
```
sales_data = pd.read_excel("champagne.Sales.xlsx")
```

In [26]:
```
sales_data.head()
```

Out[26]:

	Month	Champagne sales
0	1964-01	2815
1	1964-02	2672
2	1964-03	2755
3	1964-04	2721
4	1964-05	2946

In [27]:
```
#Making sure there are no null values
sns.heatmap(sales_data.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[27]:
```
<AxesSubplot:>
```

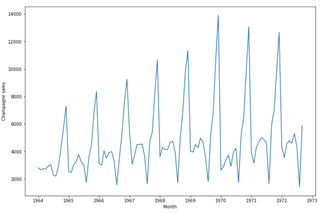

In [28]:
```
#Converting the month column to datetime
sales_data['Month']=pd.to_datetime(sales_data['Month'])
```

In [29]:
```
sales_data.head()
```

Out[29]:

	Month	Champagne sales
0	1964-01-01	2815
1	1964-02-01	2672
2	1964-03-01	2755
3	1964-04-01	2721
4	1964-05-01	2946

In [30]:
```
# To understand the pattern
plt.figure(figsize=(12,8))
sns.lineplot(data=sales_data, x='Month', y= 'Champagne sales')
```

Out[30]:
```
<AxesSubplot:xlabel='Month', ylabel='Champagne sales'>
```


In [31]:
```
#Setting the index of the Month
sales_data.set_index('Month',inplace=True)
```

In [34]:
```
#Testing for stationarity
from pmdarima.arima import ADFTest
adf_test = ADFTest(alpha = 0.05)
adf_test.should_diff(sales_data)
```

Out[34]:
```
(0.01, False)
```

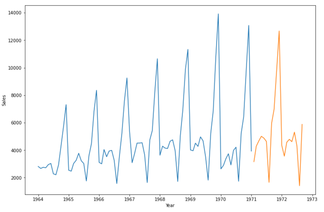
In [35]:
```
#Splitting the dataset into train and test
train = sales_data[:85]
test = sales_data[-20:]
```

In [36]:
```
train.tail()
```

Out[36]:

	Champagne sales
Month	
1970-09-01	5221
1970-10-01	6424
1970-11-01	9042
1970-12-01	13076
1971-01-01	3934

In [37]:
```
plt.figure(figsize=(12,8))
plt.plot(train)
plt.plot(test)
plt.xlabel('Year')
plt.ylabel('Sales')
```

Out[37]:
```
Text(0, 0.5, 'Sales')
```


In [38]:
```
arima_model =  auto_arima(train,start_p=0, d=1, start_q=0,
                          max_p=5, max_d=5, max_q=5, start_P=0,
                          D=1, start_Q=0, max_P=5, max_D=5,
                          max_Q=5, m=12, seasonal=True,
                          error_action='warn',trace = True,
                          suppress_warnings=True,stepwise = True,
                          random_state=20,n_fits = 50 )
```

```
Performing stepwise search to minimize aic
 ARIMA(0,1,0)(0,1,0)[12]             : AIC=1203.853, Time=0.03 sec
 ARIMA(1,1,0)(1,1,0)[12]             : AIC=1192.025, Time=0.09 sec
 ARIMA(0,1,1)(0,1,1)[12]             : AIC=1176.246, Time=0.16 sec
 ARIMA(0,1,1)(0,1,0)[12]             : AIC=1174.731, Time=0.06 sec
 ARIMA(0,1,1)(1,1,0)[12]             : AIC=1176.034, Time=0.18 sec
 ARIMA(0,1,1)(1,1,1)[12]             : AIC=1176.700, Time=0.29 sec
 ARIMA(1,1,1)(0,1,0)[12]             : AIC=1175.054, Time=0.08 sec
 ARIMA(0,1,2)(0,1,0)[12]             : AIC=1174.769, Time=0.09 sec
 ARIMA(1,1,0)(0,1,0)[12]             : AIC=1194.721, Time=0.03 sec
 ARIMA(1,1,2)(0,1,0)[12]             : AIC=1174.564, Time=0.28 sec
 ARIMA(1,1,2)(1,1,0)[12]             : AIC=inf, Time=0.38 sec
 ARIMA(1,1,2)(0,1,1)[12]             : AIC=inf, Time=0.39 sec
 ARIMA(1,1,2)(1,1,1)[12]             : AIC=1176.631, Time=0.71 sec
 ARIMA(2,1,2)(0,1,0)[12]             : AIC=1176.127, Time=0.23 sec
 ARIMA(1,1,3)(0,1,0)[12]             : AIC=1176.124, Time=0.32 sec
 ARIMA(0,1,3)(0,1,0)[12]             : AIC=1176.458, Time=0.16 sec
 ARIMA(2,1,3)(0,1,0)[12]             : AIC=1176.656, Time=0.13 sec
 ARIMA(2,1,3)(0,1,0)[12] intercept   : AIC=1180.619, Time=0.49 sec
 ARIMA(1,1,2)(0,1,0)[12] intercept   : AIC=inf, Time=0.18 sec

Best model:  ARIMA(1,1,2)(0,1,0)[12]
Total fit time: 4.347 seconds
```

In [39]:
```
#Summary of the model
arima_model.summary()
```

Out[39]:

SARIMAX Results

Dep. Variable:	y	**No. Observations:**	85
Model:	SARIMAX(1, 1, 2)x(0, 1, [], 12)	**Log Likelihood**	-583.282
Date:	Thu, 20 Jan 2022	**AIC**	1174.564
Time:	17:59:42	**BIC**	1183.670
Sample:	0	**HQIC**	1178.189
	- 85		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.8412	0.152	-5.542	0.000	-1.139	-0.544
ma.L1	0.0513	0.167	0.306	0.759	-0.275	0.378
ma.L2	-0.8673	0.086	-10.133	0.000	-1.035	-0.700
sigma2	5.862e+05	7.03e+04	8.342	0.000	4.48e+05	7.24e+05

Ljung-Box (L1) (Q):	0.05	**Jarque-Bera (JB):**	8.55
Prob(Q):	0.83	**Prob(JB):**	0.01
Heteroskedasticity (H):	2.61	**Skew:**	-0.10
Prob(H) (two-sided):	0.02	**Kurtosis:**	4.68

Warnings:

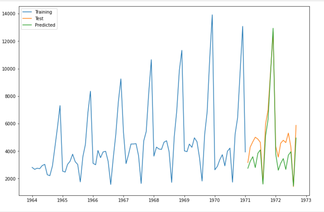
[1] Covariance matrix calculated using the outer product of gradients (complex-step).

In [40]:
```
prediction = pd.DataFrame(arima_model.predict(n_periods = 20),index=test.index)
prediction.columns = ['predicted_sales']
prediction
```

Out[40]:

Month	predicted_sales
1971-02-01	2746.710907
1971-03-01	3247.908814
1971-04-01	3592.507516
1971-05-01	2800.874094
1971-06-01	3841.901364
1971-07-01	4088.019322
1971-08-01	1596.302697
1971-09-01	5089.999423
1971-10-01	6284.001747
1971-11-01	9709.570240
1971-12-01	12937.203919
1972-01-01	3800.559019
1972-02-01	2606.765426
1972-03-01	3113.752342
1972-04-01	3455.163877
1972-05-01	2666.211376
1972-06-01	3704.983560
1972-07-01	3952.996408
1972-08-01	1459.686184
1972-09-01	4954.725067

In [44]:
```
plt.figure(figsize=(12,8))
plt.plot(train,label="Training")
plt.plot(test,label="Test")
plt.plot(prediction,label="Predicted")
plt.legend(loc = "best")
plt.show()
```

```


```

In [45]:
```
from sklearn.metrics import r2_score
test['predicted_sales'] = prediction
r2_score(test['Champagne sales'], test['predicted_sales'])
```

Out[45]:
```
0.8114741564009944
```

In [ ]:
```

```