Surakshith Shetty-53026240013

Practical 5 : To analyze the impact of marketing mix variables on sales and develop a predictive time series model using SARIMAX for improved sales forecastin

```
import pandas as pd
import statsmodels.formula.api as smf
from statsmodels.stats.stattools import durbin_watson
import statsmodels.api as sm
from sklearn.preprocessing import StandardScaler
import warnings
warnings.filterwarnings('ignore',category=FutureWarning)

data = pd.read_csv('/content/mktmix.csv')

data.dropna(inplace = True)
data
```

```
{"summary":"{\n  \"name\": \"data\",\n  \"rows\": 100,\n  \"fields\":
[\n    {\n      \"column\": \"Time period\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 29,\n
\"min\": 1,\n        \"max\": 100,\n        \"num_unique_values\":
100,\n        \"samples\": [\n          84,\n          54,\n
71\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Sales\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 1580,\n        \"min\": 17431,\n        \"max\": 24944,\n
\"num_unique_values\": 98,\n        \"samples\": [\n          20153,\n
19588,\n          17926\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Base_Price\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0.5237810979023972,\n
\"min\": 13.73572359,\n        \"max\": 16.2810198,\n
\"num_unique_values\": 17,\n        \"samples\": [\n
15.02927551,\n          14.5850933,\n          15.79984295\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Radio\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 86.99468171368089,\n        \"min\": 0.0,\n        \"max\":
399.0,\n        \"num_unique_values\": 53,\n        \"samples\": [\n
328.0,\n          267.0,\n          201.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"InStore\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
13.671750507690863,\n        \"min\": 10.782,\n        \"max\":
68.119,\n        \"num_unique_values\": 100,\n        \"samples\": [\n
18.074,\n          19.289,\n          37.248\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"TV\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 43.53281043017241,\n
```

```
\"min\": 37.65617447,\n        \"max\": 240.291967,\n
\"num_unique_values\": 100,\n        \"samples\": [\n
109.4686053,\n          130.5990849,\n          167.5356191\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"data"}
```

```
from statsmodels.formula.api import ols

model = ols('Sales~Base_Price+TV+Radio+InStore', data=data).fit()

model.summary()
```

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results

==============================================================================
========
Dep. Variable:                    Sales   R-squared:
0.648
Model:                            OLS   Adj. R-squared:
0.633
Method:                 Least Squares   F-statistic:
43.65
Date:                Mon, 10 Mar 2025   Prob (F-statistic):
9.64e-21
Time:                        09:54:03   Log-Likelihood:
-825.76
No. Observations:                 100   AIC:
1662.
Df Residuals:                      95   BIC:
1675.
Df Model:                           4

Covariance Type:            nonrobust

==============================================================================
========
                 coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
--------
Intercept   4.757e+04   3019.360     15.756      0.000    4.16e+04
5.36e+04
Base_Price -1952.5944    191.249    -10.210      0.000   -2332.271   -
1572.918
TV             7.4444      2.217      3.357      0.001       3.042
11.847
Radio          1.1928      1.109      1.076      0.285      -1.008
```

```
3.394
InStore       35.0531      7.323      4.787      0.000      20.515
49.591
==============================================================================
========
Omnibus:                        3.728   Durbin-Watson:
0.898
Prob(Omnibus):                  0.155   Jarque-Bera (JB):
3.098
Skew:                          -0.332   Prob(JB):
0.212
Kurtosis:                       3.551   Cond. No.
9.63e+03
==============================================================================
========

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is
correctly specified.
[2] The condition number is large, 9.63e+03. This might indicate that
there are
strong multicollinearity or other numerical problems.
"""

exog_data = data[['Base_Price','TV','Radio','InStore']]
scaler = StandardScaler()
scaled_exog_data = scaler.fit_transform(exog_data)
scaled_exog_data = pd.DataFrame(scaled_exog_data,
columns=exog_data.columns,index=exog_data.index)
scaled_exog_data.head()
```


```
{"summary":"{\n  \"name\": \"scaled_exog_data\",\n  \"rows\": 100,\n
\"fields\": [\n    {\n      \"column\": \"Base_Price\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
1.0050378152592123,\n        \"min\": -2.970392187747817,\n
\"max\": 1.9135544692249764,\n        \"num_unique_values\": 17,\n
\"samples\": [\n          -0.48830835385693155,\n          -
1.340610813230131,\n          0.9902662253116309\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"TV\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 1.005037815259212,\n
\"min\": -2.368876090359765,\n        \"max\": 2.3093569987797444,\n
\"num_unique_values\": 100,\n        \"samples\": [\n          -
0.7109493937683404,\n          -0.22311203976548163,\n
0.6296381794091025\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Radio\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 1.005037815259212,\n        \"min\": -2.9655049218752,\n
\"max\": 1.6440882209359917,\n        \"num_unique_values\": 53,\n
\"samples\": [\n          0.8238348045460303,\n
```

0.11911003835183809,\n   -0.6433790529402388\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"InStore\",\n   \"properties\":
{\n   \"dtype\": \"number\",\n   \"std\":
1.005037815259212,\n   \"min\": -1.595425936104774,\n
\"max\": 2.6195319934669197,\n   \"num_unique_values\": 100,\n
\"samples\": [\n   -1.059376830685705,\n   -
0.9700592948981115,\n   0.3501428338700144\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe","variable_name":"scaled_exog_data"}

```
model = sm.tsa.SARIMAX(data['Sales'], exog=scaled_exog_data, order
=(0,0,0),trend='c',error_ar = 1)
results = model.fit()
results.summary()
```

```
<class 'statsmodels.iolib.summary.Summary'>
"""
                                SARIMAX Results

==========================================================================
========
Dep. Variable:                    Sales   No. Observations:
100
Model:                          SARIMAX   Log Likelihood
-825.765
Date:                Mon, 10 Mar 2025   AIC
1663.530
Time:                           10:30:18   BIC
1679.161
Sample:                                0   HQIC
1669.856
                                   - 100

Covariance Type:                    opg

==========================================================================
========
                  coef    std err          z      P>|z|      [0.025
0.975]
--------------------------------------------------------------------------
--------
intercept   2.022e+04    107.340    188.362      0.000       2e+04
2.04e+04
Base_Price -1017.6055    106.716     -9.536      0.000   -1226.764
-808.447
TV           322.4525     95.305      3.383      0.001     135.657
509.248
Radio        103.2478    116.714      0.885      0.376    -125.508
332.003
```

```
InStore       476.8354    106.885      4.461      0.000     267.344
686.327
sigma2        8.71e+05   1.17e+05      7.467      0.000    6.42e+05
1.1e+06
==============================================================================
=============
Ljung-Box (L1) (Q):                  31.23   Jarque-Bera (JB):
3.10
Prob(Q):                              0.00   Prob(JB):
0.21
Heteroskedasticity (H):               1.77   Skew:
-0.33
Prob(H) (two-sided):                  0.11   Kurtosis:
3.55
==============================================================================
=============

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

base_price = [15.64, 15.49]
radio = [279, 259]
instore = [41.50, 20.40]
tv = [103.44, 128.40]
testdf =
pd.DataFrame({'Base_Price':base_price,'TV':tv,'Radio':radio,'InStore':
instore})
mean = data[['Base_Price','TV','Radio','InStore']].mean()
std = data[['Base_Price','TV','Radio','InStore']].std()
x = testdf[['Base_Price','Radio','InStore','TV']]
x_var = (x-mean)/std
x_var
```


```
{"summary":"{\n  \"name\": \"x_var\",\n  \"rows\": 2,\n  \"fields\":
[\n    {\n      \"column\": \"Base_Price\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0.20250065839097342,\n
\"min\": 0.39375199243717063,\n        \"max\": 0.6801311699231664,\n
\"num_unique_values\": 2,\n        \"samples\": [\n
0.39375199243717063,\n          0.6801311699231664\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"InStore\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
1.091297933987541,\n        \"min\": -0.8839343574329994,\n
\"max\": 0.6593939814019196,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          -0.8839343574329994,\n
0.6593939814019196\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Radio\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
```

\"std\": 0.1625632204768092,\n   \"min\": 0.026553347336826096,\n
\"max\": 0.2564524584781773,\n   \"num_unique_values\": 2,\n
\"samples\": [\n   0.026553347336826096,\n
0.2564524584781773\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"TV\",\n   \"properties\": {\n   \"dtype\": \"number\",\n
\"std\": 0.40542719580961206,\n   \"min\": -0.8458698893485193,\n
\"max\": -0.27250925047967345,\n   \"num_unique_values\": 2,\n
\"samples\": [\n   -0.27250925047967345,\n   -
0.8458698893485193\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   }\n]\
n}","type":"dataframe","variable_name":"x_var"}

```
forecast = results.forecast(steps=2,exog=x_var)
forecast
```

```
100    19362.405485
101    19405.838486
Name: predicted_mean, dtype: float64
```