1 

i

from pandas import read\_csv

2

series = read\_csv('/content/daily-minimum-temperatures.csv', header=0, index\_col=0) 3

print(series.describe())

Temp

count 3650.000000

mean 11.177753

std 4.071837

min 0.000000

25% 8.300000

50% 11.000000

75% 14.000000

max 26.300000

1 series.isnull().sum() Temp 0

dtype: int64

1 series=series.dropna() 

1 series.isnull().sum() Temp 0

dtype: int64

1 series.isnull().values.any() False

1 

from pandas import read\_csv

2

from matplotlib import pyplot

3

series = read\_csv('/content/daily-minimum-temperatures.csv', header=0, index\_col=0)

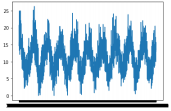
1 

p

pyplot.plot(series)

2

pyplot.show()



1

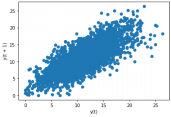
t

from pandas.plotting import lag\_plot

2

lag\_plot(series)

<matplotlib.axes.\_subplots.AxesSubplot at 0x7fcb06120eb8>



1

from pandas import read\_csv

2

from pandas import datetime

3

from matplotlib import pyplot

4

5

def parser(x):

6

  return datetime.strptime('190'+x, '%Y-%m')

7

8

series = read\_csv('/content/shampoo-sales.csv', header=0, parse\_dates=[0], index\_col=0, squeeze=True, date\_parser=parser)

i

9

print(series.head())

10

series.plot()

11

pyplot.show()

/usr/local/lib/python3.6/dist-packages/ipykernel\_launcher.py:6: FutureWarning: The pandas.datetime class is deprecated and will be removed from pandas in a future version. Import from datetime instead

Month

1901-01-01 266.0

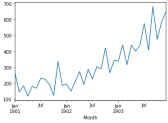
1901-02-01 145.9

1901-03-01 183.1

1901-04-01 119.3

1901-05-01 180.3

Name: Sales, dtype: float64



h

1

# Plot the autocorrelation graph

2

from pandas import read\_csv

3

from pandas import datetime

4

from matplotlib import pyplot

5

from pandas.plotting import autocorrelation\_plot

6

7

def parser(x):

8

  return datetime.strptime('190'+x, '%Y-%m')

9

10

series = read\_csv('shampoo-sales.csv', header=0, parse\_dates=[0], index\_col=0, squeeze=True, date\_parser=parser)

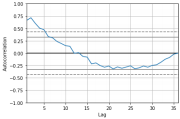
11

autocorrelation\_plot(series)

12

pyplot.show()

/usr/local/lib/python3.6/dist-packages/ipykernel\_launcher.py:8: FutureWarning: The pandas.datetime class is deprecated and will be removed from pandas in a future version. Import from datetime instead

1 

!pip install --upgrade statsmodels

2

# fit an ARIMA model and plot residual errors

3

from pandas import datetime

4

from pandas import read\_csv

5

from pandas import DataFrame

6

from statsmodels.tsa.arima.model import ARIMA

7

from matplotlib import pyplot

8

# load dataset

9

def parser(x):

10

  return datetime.strptime('190'+x, '%Y-%m')

11

series = read\_csv('shampoo-sales.csv', header=0, index\_col=0, parse\_dates=True, squeeze=True, date\_parser=parser)

12

series.index = series.index.to\_period('M')

13

# fit model

14

model = ARIMA(series, order=(5,1,0))

15

model\_fit = model.fit()

16

# summary of fit model

17

print(model\_fit.summary())

18

# line plot of residuals

19

residuals = DataFrame(model\_fit.resid)

20

residuals.plot()

21

pyplot.show()

22

# density plot of residuals

23

residuals.plot(kind='kde')

24

pyplot.show()

25

# summary stats of residuals

26

print(residuals.describe())

Covariance Type: opg

==============================================================================

coef std err z P>|z| [0.025 0.975]

------------------------------------------------------------------------------

ar.L1 -0.9014 0.247 -3.647 0.000 -1.386 -0.417

ar.L2 -0.2284 0.268 -0.851 0.395 -0.754 0.298

ar.L3 0.0747 0.291 0.256 0.798 -0.497 0.646

ar.L4 0.2519 0.340 0.742 0.458 -0.414 0.918

ar.L5 0.3344 0.210 1.593 0.111 -0.077 0.746

sigma2 4728.9608 1316.021 3.593 0.000 2149.607 7308.314

===================================================================================

Ljung-Box (L1) (Q): 0.61 Jarque-Bera (JB): 0.96

Prob(Q): 0.44 Prob(JB): 0.62

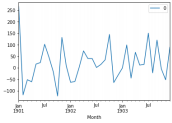
Heteroskedasticity (H): 1.07 Skew: 0.28

Prob(H) (two-sided): 0.90 Kurtosis: 2.41

===================================================================================

Warnings:

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



0

count 36.000000

mean 21.936144

std 80.774430

min -122.292030

25% -35.040859

50% 13.147219

75% 68.848286

max 266.000000

1

from pandas import read\_csv

2

from pandas import datetime

3

from matplotlib import pyplot

4

from statsmodels.tsa.arima.model import ARIMA

5

from sklearn.metrics import mean\_squared\_error

6

from math import sqrt

7

# load dataset

8

def parser(x):

9

  return datetime.strptime('190'+x, '%Y-%m')

10

series = read\_csv('shampoo-sales.csv', header=0, index\_col=0, parse\_dates=True, squeeze=True, date\_parser=parser)

11

series.index = series.index.to\_period('M')

12

# split into train and test sets

13

X = series.values

13

X   series.values

14

size = int(len(X) \* 0.66)

15

train, test = X[0:size], X[size:len(X)]

16

history = [x for x in train]

17

predictions = list()

18

# walk-forward validation

19

for t in range(len(test)):

20

  model = ARIMA(history, order=(5,1,0))

21

  model\_fit = model.fit()

22

  output = model\_fit.forecast()

23

  yhat = output[0]

24

  predictions.append(yhat)

25

  obs = test[t]

26

  history.append(obs)

27

  print('predicted=%f, expected=%f' % (yhat, obs))

28

# evaluate forecasts

29

rmse = sqrt(mean\_squared\_error(test, predictions))

30

print('Test RMSE: %.3f' % rmse)

31

# plot forecasts against actual outcomes

32

pyplot.plot(test)

33

pyplot.plot(predictions, color='red')

34

pyplot.show()#

/usr/local/lib/python3.6/dist-packages/ipykernel\_launcher.py:9: FutureWarning: The pandas.datetime class is deprecated and will be removed from pandas in a future version. Import from datetime instead if \_\_name\_\_ == '\_\_main\_\_':

predicted=343.272180, expected=342.300000

predicted=293.329674, expected=339.700000

predicted=368.668956, expected=440.400000

predicted=335.044741, expected=315.900000

predicted=363.220221, expected=439.300000

predicted=357.645324, expected=401.300000

predicted=443.047835, expected=437.400000

predicted=378.365674, expected=575.500000

predicted=459.415021, expected=407.600000

predicted=526.890876, expected=682.000000

predicted=457.231274, expected=475.300000

predicted=672.914943, expected=581.300000

predicted=531.541449, expected=646.900000

Test RMSE: 89.021

