seatleweather

March 23, 2025

importing Libraries

```
[107]: import pandas as pd
       import matplotlib.pyplot as plt
       import numpy as np
       import seaborn as sns
       from statsmodels.tsa.stattools import adfuller
       from statsmodels.tsa.seasonal import seasonal_decompose
       from statsmodels.tsa.arima.model  import ARIMA
```

Loading and Viewing data

```
[108]: sw=pd.read_csv(r"C:\Users\Megha I Angadi\Downloads\seattle-weather.csv")
       sw.head()
```

```
[108]:          date  precipitation  temp_max  temp_min  wind  weather
       0  01-01-2012            0.0      12.8       5.0   4.7  drizzle
       1  02-01-2012           10.9      10.6       2.8   4.5     rain
       2  03-01-2012            0.8      11.7       7.2   2.3     rain
       3  04-01-2012           20.3      12.2       5.6   4.7     rain
       4  05-01-2012            1.3       8.9       2.8   6.1     rain
```

```
[109]: sw.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1461 entries, 0 to 1460
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   date           1461 non-null   object
 1   precipitation  1461 non-null   float64
 2   temp_max       1461 non-null   float64
 3   temp_min       1461 non-null   float64
 4   wind           1461 non-null   float64
 5   weather        1461 non-null   object
dtypes: float64(4), object(2)
memory usage: 68.6+ KB
```

[110]:
```
print(sw[sw['date'].isna()])
```

```
Empty DataFrame
Columns: [date, precipitation, temp_max, temp_min, wind, weather]
Index: []
```

[111]:
```
sw['date']=pd.to_datetime(sw['date'],errors='coerce')
sw.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1461 entries, 0 to 1460
Data columns (total 6 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   date           576 non-null    datetime64[ns]
 1   precipitation  1461 non-null   float64
 2   temp_max       1461 non-null   float64
 3   temp_min       1461 non-null   float64
 4   wind           1461 non-null   float64
 5   weather        1461 non-null   object
dtypes: datetime64[ns](1), float64(4), object(1)
memory usage: 68.6+ KB
```

Setting data index

[112]:
```
sw.set_index('date',inplace=True)
sw.head()
```

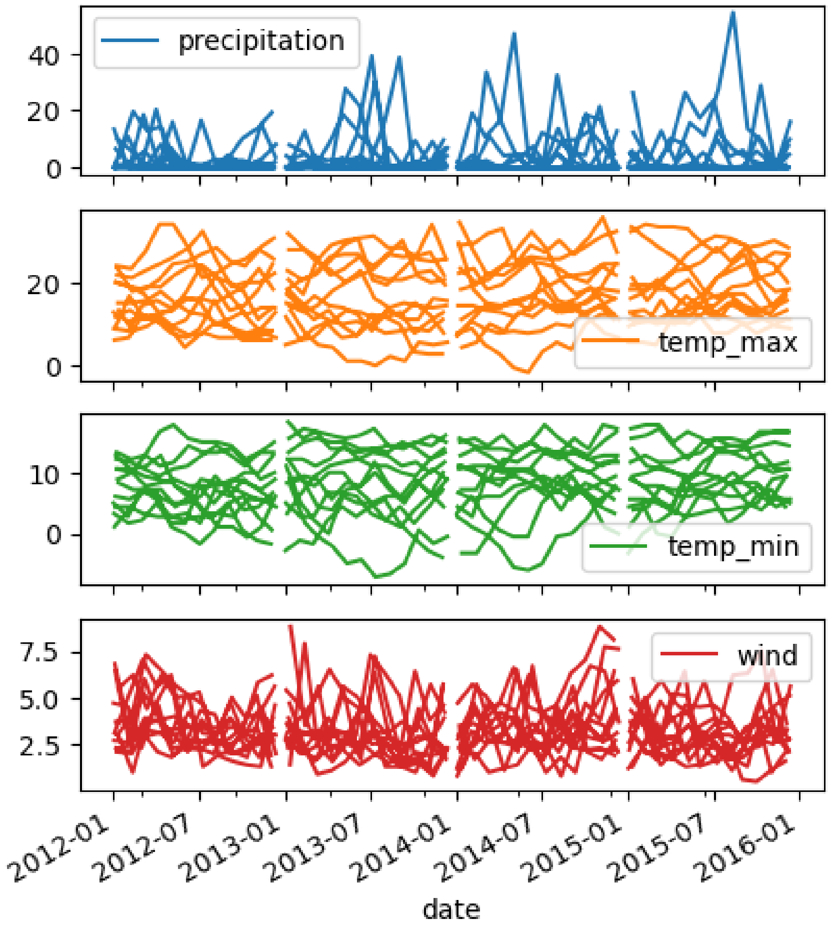
[112]:
```
            precipitation  temp_max  temp_min  wind  weather
date
2012-01-01            0.0      12.8       5.0   4.7  drizzle
2012-02-01           10.9      10.6       2.8   4.5     rain
2012-03-01            0.8      11.7       7.2   2.3     rain
2012-04-01           20.3      12.2       5.6   4.7     rain
2012-05-01            1.3       8.9       2.8   6.1     rain
```

[113]:
```
sw.describe()
```

[113]:
```
       precipitation     temp_max     temp_min         wind
count    1461.000000  1461.000000  1461.000000  1461.000000
mean        3.029432    16.439083     8.234771     3.241136
std         6.680194     7.349758     5.023004     1.437825
min         0.000000    -1.600000    -7.100000     0.400000
25%         0.000000    10.600000     4.400000     2.200000
50%         0.000000    15.600000     8.300000     3.000000
75%         2.800000    22.200000    12.200000     4.000000
max        55.900000    35.600000    18.300000     9.500000
```

1 Visualization

```
[114]: sw.plot(figsize=(5,6),subplots=True)
       plt.show()
```



```
[115]: ard=adfuller(sw["temp_min"])
       print(ard)
       if ard[1]>0.05:
           print("Non-sationary.")
       else:
           print("Sationary.")
```

```
(-2.6056195336348207, 0.09184137745869686, 13, 1447, {'1%': -3.4348772553489617,
'5%': -2.8635394783531085, '10%': -2.5678345067434516}, 5860.06752988114)
Non-sationary.
```

2 Differencing

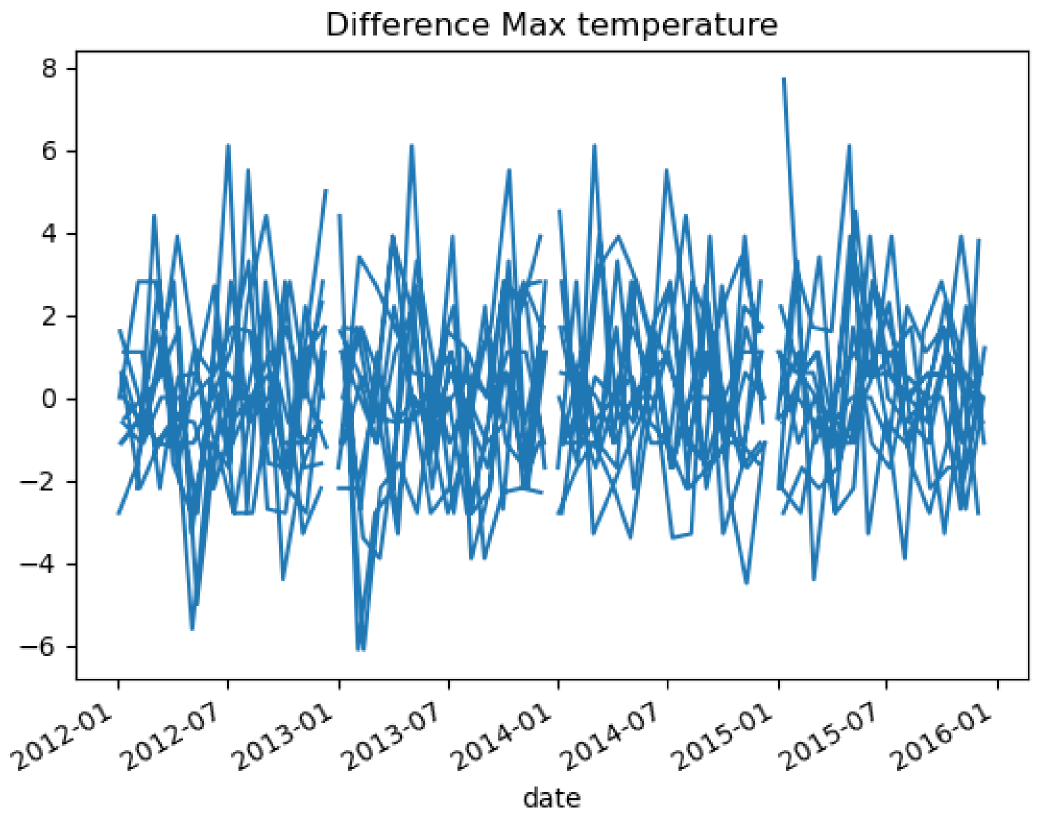
```
[116]: sw["temp_min_d"]=sw["temp_min"].diff()
       sw.head()
```

```
[116]:             precipitation  temp_max  temp_min  wind  weather  temp_min_d
       date
       2012-01-01            0.0      12.8       5.0   4.7  drizzle         NaN
       2012-02-01           10.9      10.6       2.8   4.5     rain        -2.2
       2012-03-01            0.8      11.7       7.2   2.3     rain         4.4
       2012-04-01           20.3      12.2       5.6   4.7     rain        -1.6
       2012-05-01            1.3       8.9       2.8   6.1     rain        -2.8
```

```
[117]: ard=adfuller(sw["temp_min_d"].dropna())
       if ard[1]>0.05:
           print("Non-sationary.")
       else:
           print("Sationary.")
```

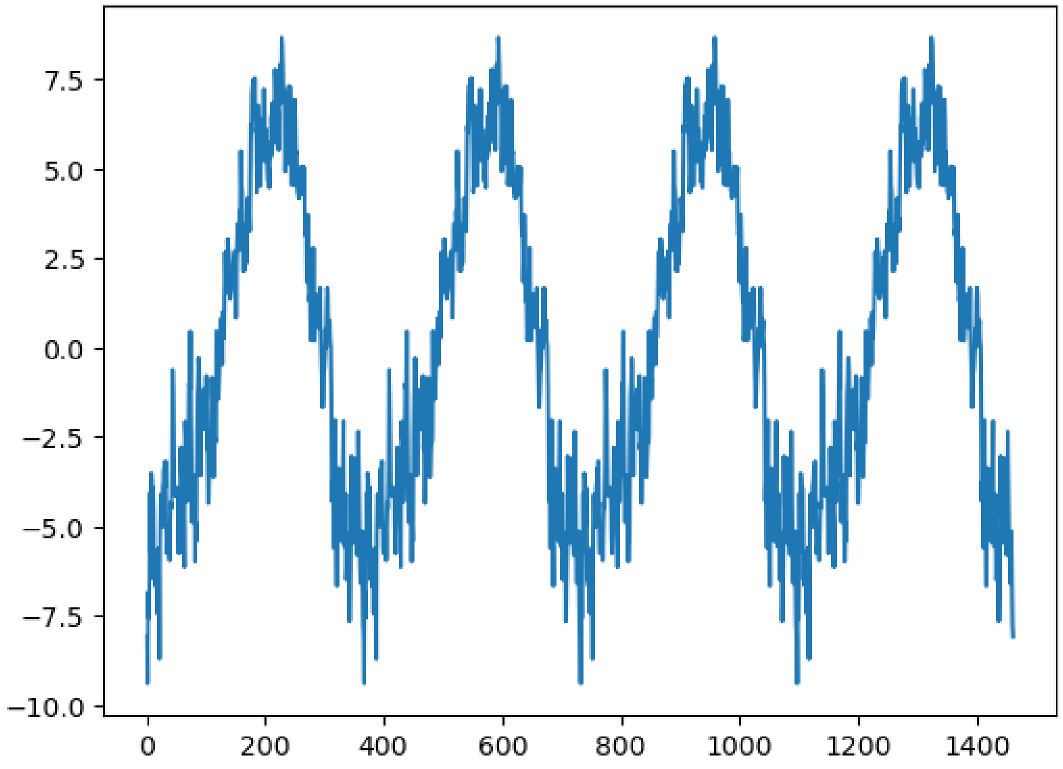
```
Sationary.
```

```
[118]: sw["temp_min_d"].plot(title="Difference Max temperature")
       plt.show()
```

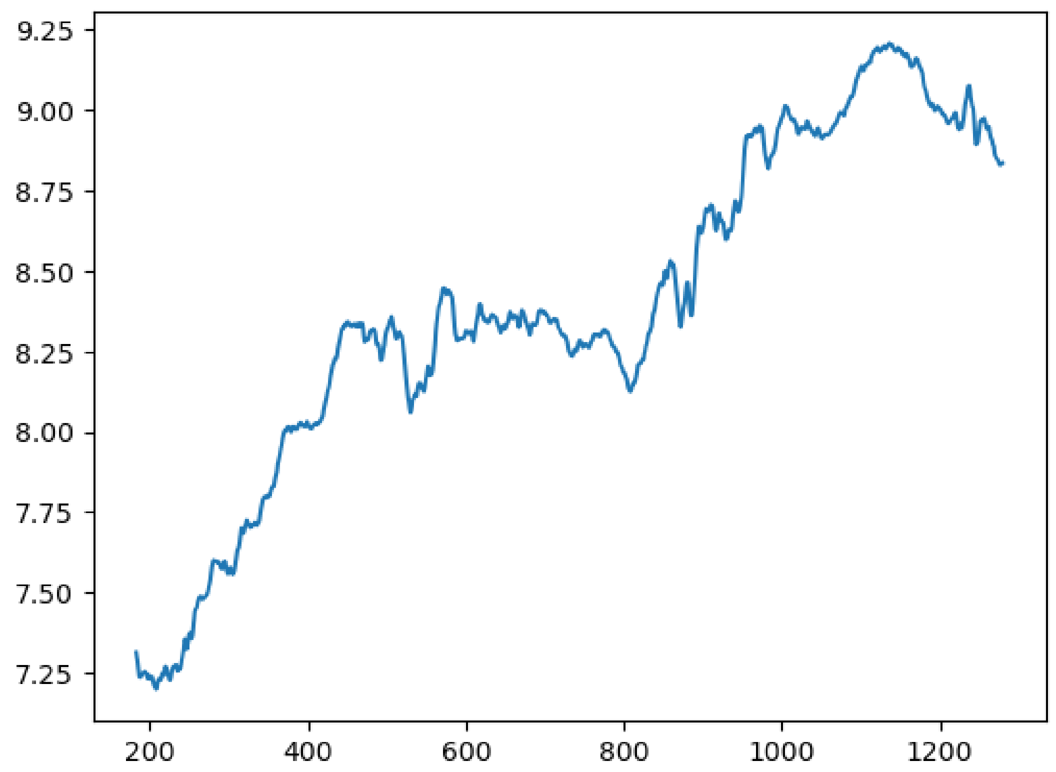


[142]:
```
decomposing=seasonal_decompose(sw["temp_min"],model="additive", period = 365)
```

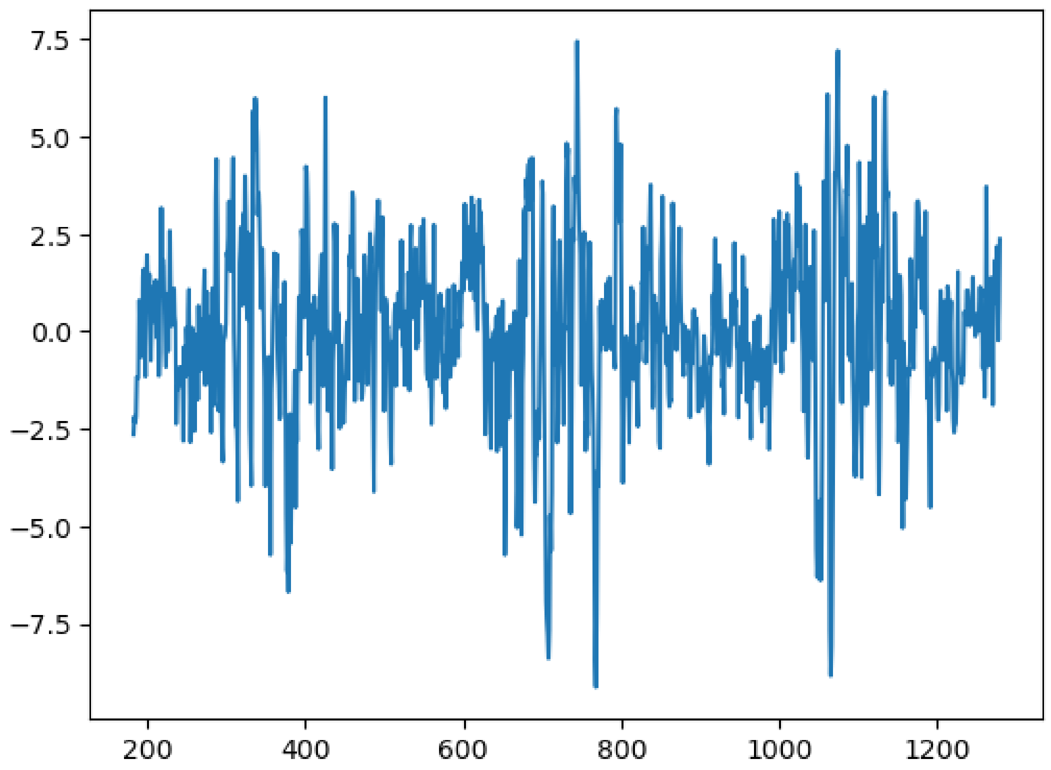
[148]:
```
plt.plot(np.arange(1,1462), decomposing.seasonal)
plt.show()
```



[149]:
```
plt.plot(np.arange(1,1462), decomposing.trend)
plt.show()
```



[151]:
```
plt.plot(np.arange(1,1462), decomposing.resid)
plt.show()
```



3 ARIMA for forecasting

```
[121]: len(sw)
```

```
[121]: 1461
```

```
[122]: print(len(sw)*0.80)
```

```
1168.8
```

```
[123]: train=sw.iloc[0:1169]
       test=sw.iloc[1169:]
       len(test)
```

```
[123]: 292
```

```
[124]: mm=ARIMA(train["temp_min"],order=(1,1,1))
```

```
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be
ignored when e.g. forecasting.
```

```
  self._init_dates(dates, freq)
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it is not monotonic and so will be ignored when e.g.
forecasting.
  self._init_dates(dates, freq)
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be
ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it is not monotonic and so will be ignored when e.g.
forecasting.
  self._init_dates(dates, freq)
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be
ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it is not monotonic and so will be ignored when e.g.
forecasting.
  self._init_dates(dates, freq)
```

[125]:
```
mm=mm.fit()
```

```
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\statespace\sarimax.py:966: UserWarning: Non-stationary
starting autoregressive parameters found. Using zeros as starting parameters.
  warn('Non-stationary starting autoregressive parameters'
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\statespace\sarimax.py:978: UserWarning: Non-invertible
starting MA parameters found. Using zeros as starting parameters.
  warn('Non-invertible starting MA parameters found.'
```

[126]:
```
forecast=mm.forecast(steps=len(test))
print(forecast)
```

```
1169    8.462908
1170    7.803902
1171    7.340458
1172    7.014543
1173    6.785344
          …
1456    6.242190
```

```
1457    6.242190
1458    6.242190
1459    6.242190
1460    6.242190
Name: predicted_mean, Length: 292, dtype: float64
```

```
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:836: ValueWarning: No supported index
is available. Prediction results will be given with an integer index beginning
at `start`.
  return get_prediction_index(
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:836: FutureWarning: No supported
index is available. In the next version, calling this method in a model without
a supported index will result in an exception.
  return get_prediction_index(
```

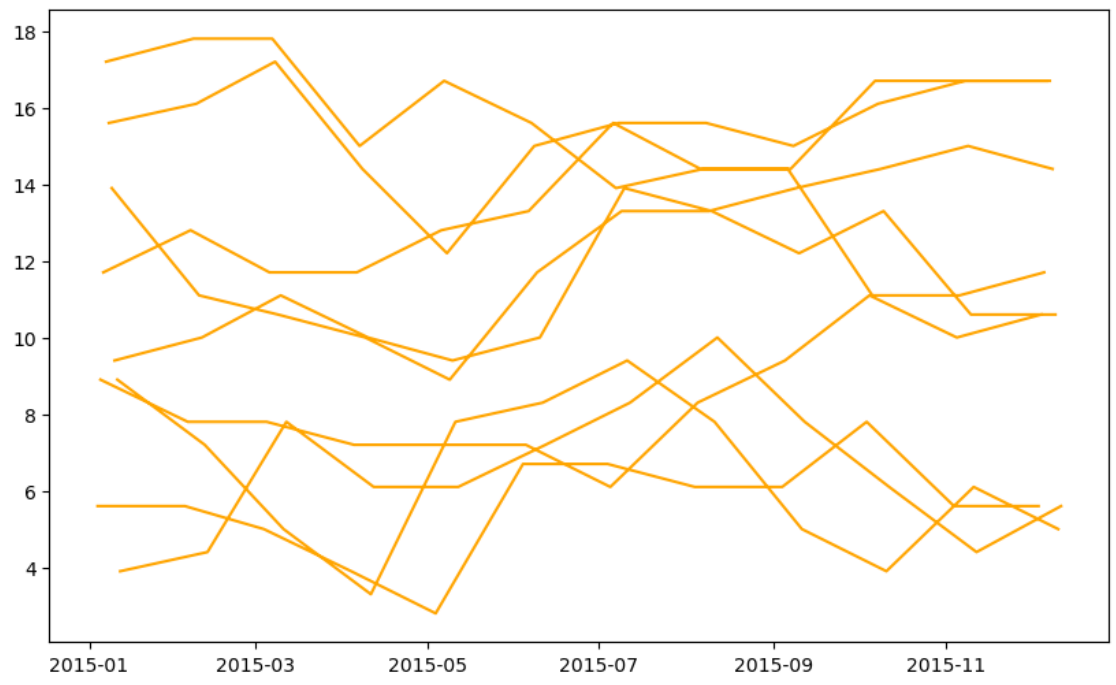
[127]:
```
test.head()
test["forecast"]=forecast
test.head()
```

```
C:\Users\Megha I Angadi\AppData\Local\Temp\ipykernel_8776\2784579413.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  test["forecast"]=forecast
```

[127]:
```
      precipitation  temp_max  temp_min  wind weather  temp_min_d  forecast
date
NaT            55.9      10.6       6.1   4.2    rain        -3.3       NaN
NaT             1.0      13.9       6.1   3.0    rain         0.0       NaN
NaT             0.8      13.3       4.4   2.6    rain        -1.7       NaN
NaT             0.0      15.6       7.2   2.5     sun         2.8       NaN
NaT             0.0      15.6       8.3   1.9     sun         1.1       NaN
```

[128]:
```
plt.figure(figsize=(10,6))
plt.plot(test.index,test["temp_min"],color="orange",label="original")
plt.plot(forecast.index,test["forecast"],color="orange",label="original")
```

[128]:
```
[<matplotlib.lines.Line2D at 0x193adb8c170>]
```



4 ARIMA for difference value

[129]:
```
len(sw)
```

[129]:
```
1461
```

[130]:
```
print(len(sw)*0.80)
```

```
1168.8
```

[131]:
```
train1=sw.iloc[0:1169]
test1=sw[1169:]
```

[132]:
```
mmd=ARIMA(train["temp_min_d"],order=(1,1,1))
mmd=mmd.fit()
```

```
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be
ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it is not monotonic and so will be ignored when e.g.
forecasting.
```

```
  self._init_dates(dates, freq)
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be
ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it is not monotonic and so will be ignored when e.g.
forecasting.
  self._init_dates(dates, freq)
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it has no associated frequency information and so will be
ignored when e.g. forecasting.
  self._init_dates(dates, freq)
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:473: ValueWarning: A date index has
been provided, but it is not monotonic and so will be ignored when e.g.
forecasting.
  self._init_dates(dates, freq)
```

[133]:
```
forecast_d=mmd.forecast(steps=len(test1))
forecast_d.head()
```

```
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:836: ValueWarning: No supported index
is available. Prediction results will be given with an integer index beginning
at `start`.
  return get_prediction_index(
C:\Users\Megha I Angadi\anaconda3\Lib\site-
packages\statsmodels\tsa\base\tsa_model.py:836: FutureWarning: No supported
index is available. In the next version, calling this method in a model without
a supported index will result in an exception.
  return get_prediction_index(
```

[133]:
```
1169   -0.062054
1170    0.006485
1171    0.003659
1172    0.003776
1173    0.003771
Name: predicted_mean, dtype: float64
```

[134]:
```
test1["forecast_d"]=forecast_d
test1.head()
```

```
C:\Users\Megha I Angadi\AppData\Local\Temp\ipykernel_8776\3136070984.py:1:
SettingWithCopyWarning:
```

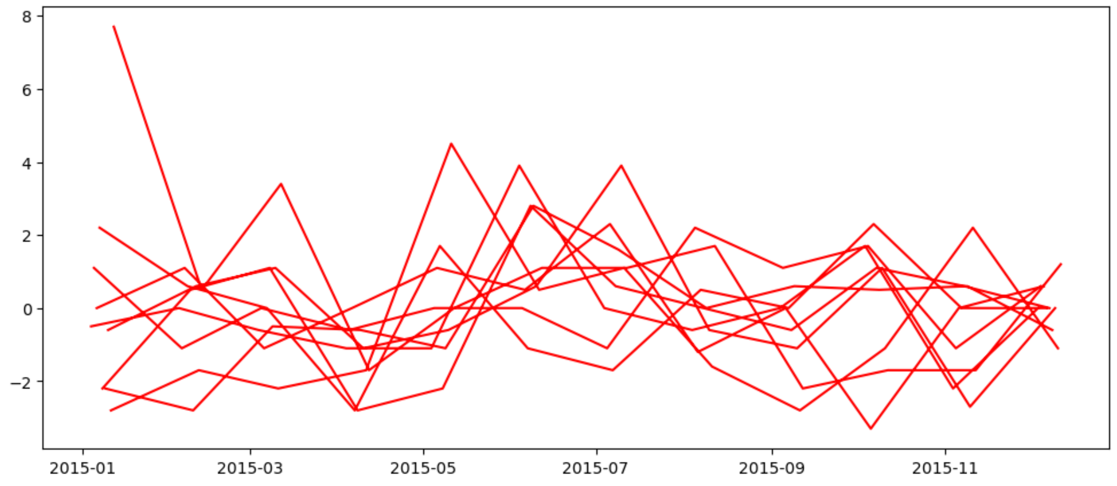
```
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  test1["forecast_d"]=forecast_d
```

[134]:

```
     precipitation  temp_max  temp_min  wind weather  temp_min_d  forecast_d
date
NaT           55.9      10.6       6.1   4.2    rain        -3.3         NaN
NaT            1.0      13.9       6.1   3.0    rain         0.0         NaN
NaT            0.8      13.3       4.4   2.6    rain        -1.7         NaN
NaT            0.0      15.6       7.2   2.5     sun         2.8         NaN
NaT            0.0      15.6       8.3   1.9     sun         1.1         NaN
```

[135]:

```
plt.figure(figsize=(12,5))
plt.plot(test1.index,test1["temp_min_d"],color="red",label="original")
plt.plot(test1.index,test1["forecast_d"],color="green",label="original")
plt.title("")
plt.show()
```



[ ]: