Untitled3

November 25, 2019

[57]:
```python
import pandas as pd
from pandas.plotting import autocorrelation_plot, lag_plot
```

[21]:
```python
df_brazil = pd.read_csv("sudeste.csv", usecols=["date", "temp"])
df_madrid = pd.read_csv("weather_madrid_LEMD_1997_2015.csv", usecols=["CET",␣
↪"Mean TemperatureC"])
```

[13]:
```python
def create_final_df(df1, df2):
    df_brazil_no_dup_date = df1.groupby("date").mean().reset_index()
    df_final = pd.merge(df_brazil_no_dup_date, df2, how="inner",␣
↪left_on="date", right_on="CET")
    df_final = df_final[["date", "temp", "Mean TemperatureC"]]
    df_final.columns = ["date", "temp_brazil", "temp_madrid"]
    return df_final
```

[14]:
```python
df_final = create_final_df(df_brazil, df_madrid)
```

[5]:
```python
df_final[["temp_brazil", "temp_madrid"]].corr()
```

[5]:

	temp_brazil	temp_madrid
temp_brazil	1.000000	-0.030652
temp_madrid	-0.030652	1.000000

[22]:
```python
# Brazil and Madrid average daily temperatures have a negative correlation of␣
↪-0.03 but that can be ignored.
# As a result one can say Brazil and Madrid average daily temperatures are␣
↪independent of each other.
```

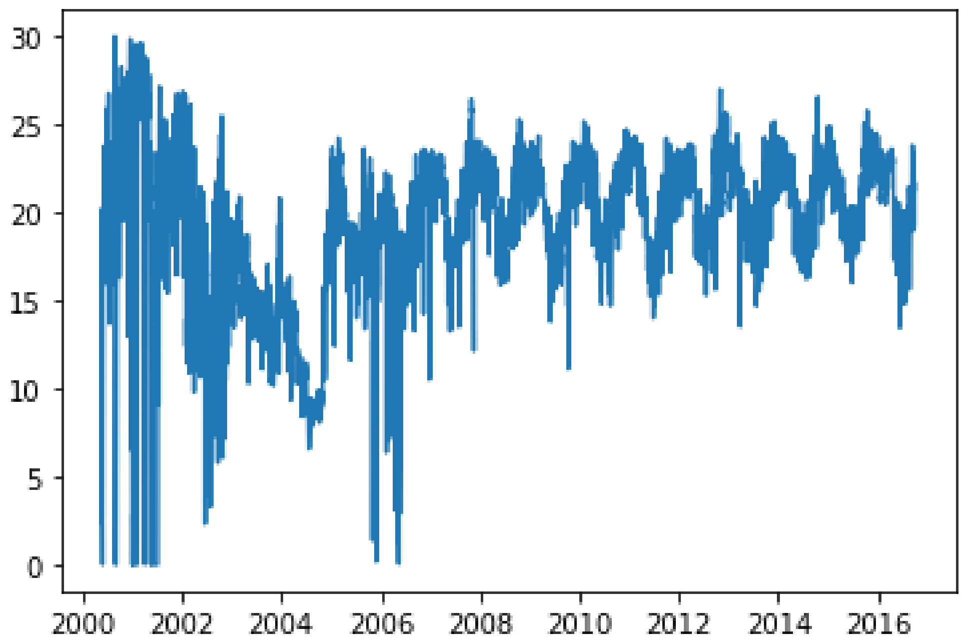
[30]:
```python
def prepare_brazil(df):
    temp = df.groupby("date").mean().reset_index()
    date_series = temp["date"]
    temp_series = temp["temp"]
    temp_series.index =  pd.DatetimeIndex(date_series)

    start_date, end_date = date_series.head(1).values[0], date_series.tail(1).
↪values[0]
    idx = pd.date_range(start_date, end_date)
```

```
    result = temp_series.reindex(idx, fill_value=0)

    return result

def prepare_madrid(df):
    temp = df
    date_series = temp["CET"]
    temp_series = temp["Mean TemperatureC"]
    temp_series.index =  pd.DatetimeIndex(date_series)

    start_date, end_date = date_series.head(1).values[0], date_series.tail(1).
 ↪values[0]
    idx = pd.date_range(start_date, end_date)
    result = temp_series.reindex(idx, fill_value=0)

    return result
```

[31]:
```
b, m = prepare_brazil(df_brazil), prepare_madrid(df_madrid)
```

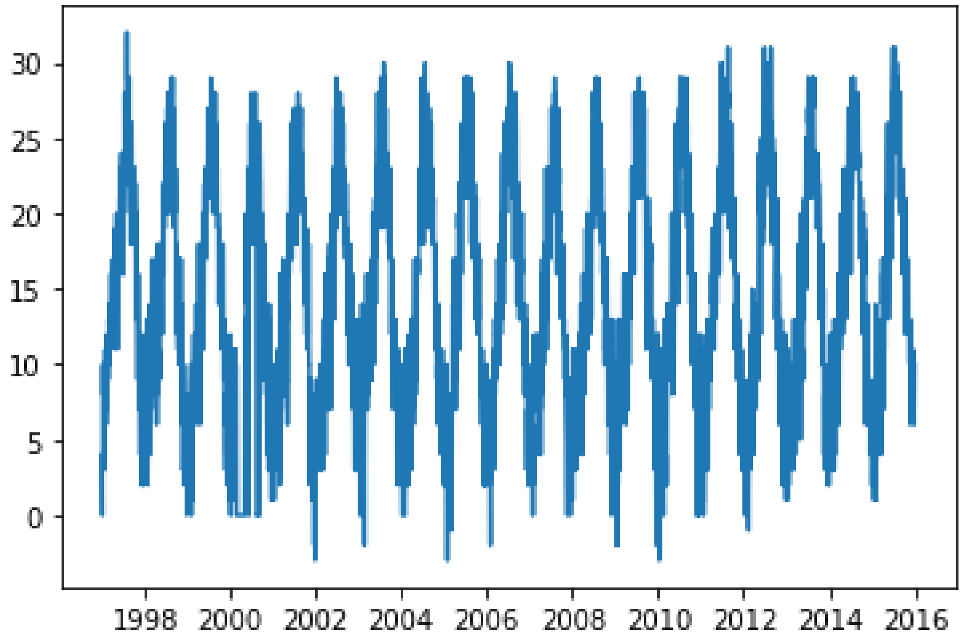
[32]:
```
plt.plot(b)
```

[32]: [<matplotlib.lines.Line2D at 0x7f9eece95bd0>]



[33]:
```
plt.plot(m)
```

[33]: [<matplotlib.lines.Line2D at 0x7f9eece69410>]



[6]:
```
from statsmodels.tsa.stattools import adfuller
```

[43]:
```
adfuller(b.dropna())
```

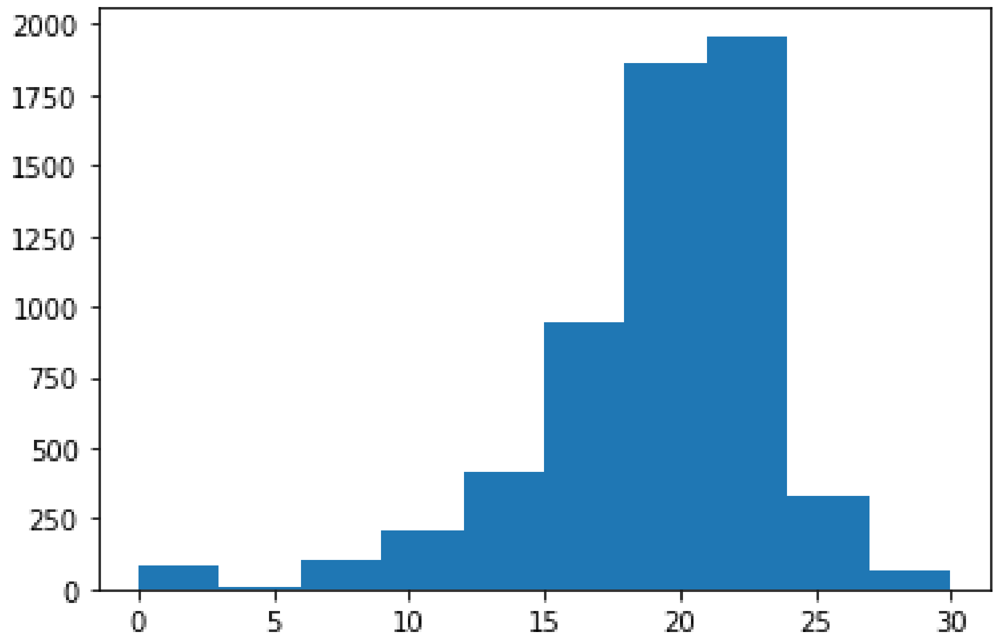
[43]:
```
(-6.589743392006552,
 7.164185170894732e-09,
 21,
 5952,
 {'1%': -3.43144914692048,
  '5%': -2.8620257211840996,
  '10%': -2.5670285470333005},
 23331.562288668385)
```

[42]:
```
adfuller(m.dropna())
```

[42]:
```
(-5.188311682682534,
 9.288240716475554e-06,
 14,
 6921,
 {'1%': -3.4312951996865126,
  '5%': -2.861957701574514,
  '10%': -2.5669923386600497},
 29511.769274129747)
```

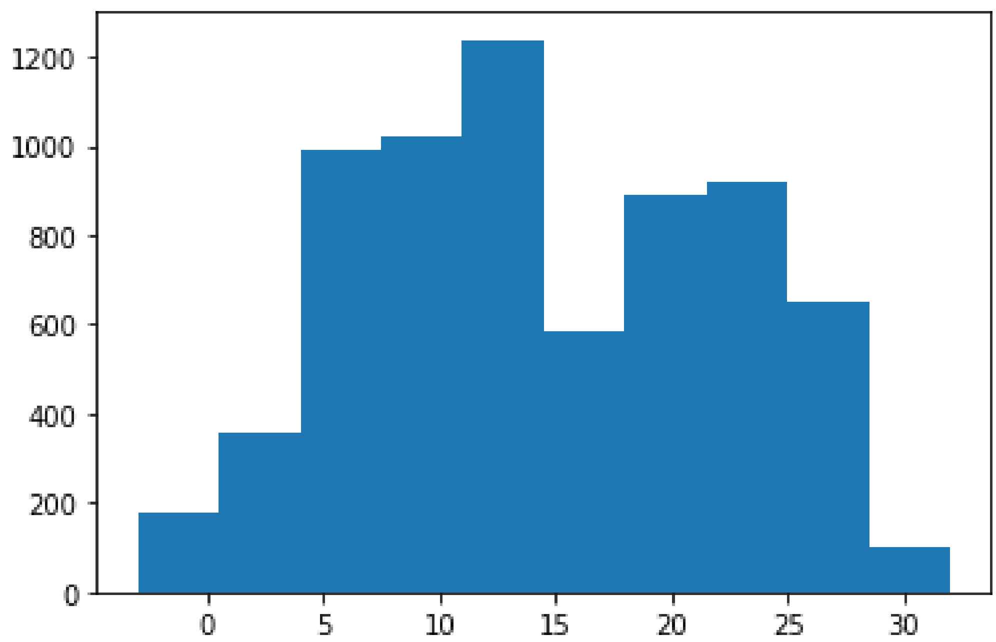
[47]:
```
plt.hist(b.dropna())
```

[47]:
```
(array([  83.,   12.,  100.,  207.,  410.,  941., 1863., 1960.,  330.,
          68.]),
 array([ 0.      ,  2.99875,  5.9975 ,  8.99625, 11.995  , 14.99375,
        17.9925 , 20.99125, 23.99   , 26.98875, 29.9875 ]),
 <a list of 10 Patch objects>)
```



[46]:
```
plt.hist(m.dropna())
```

[46]:
```
(array([ 180.,  358.,  989., 1024., 1238.,  587.,  891.,  917.,  653.,
          99.]),
 array([-3. ,  0.5,  4. ,  7.5, 11. , 14.5, 18. , 21.5, 25. , 28.5, 32. ]),
 <a list of 10 Patch objects>)
```



[50]:
```
X = m.dropna().values
low, high = X[:len(X)//2], X[len(X)//2:]
print (low.mean(), high.mean())
print (low.var(), high.var())
```

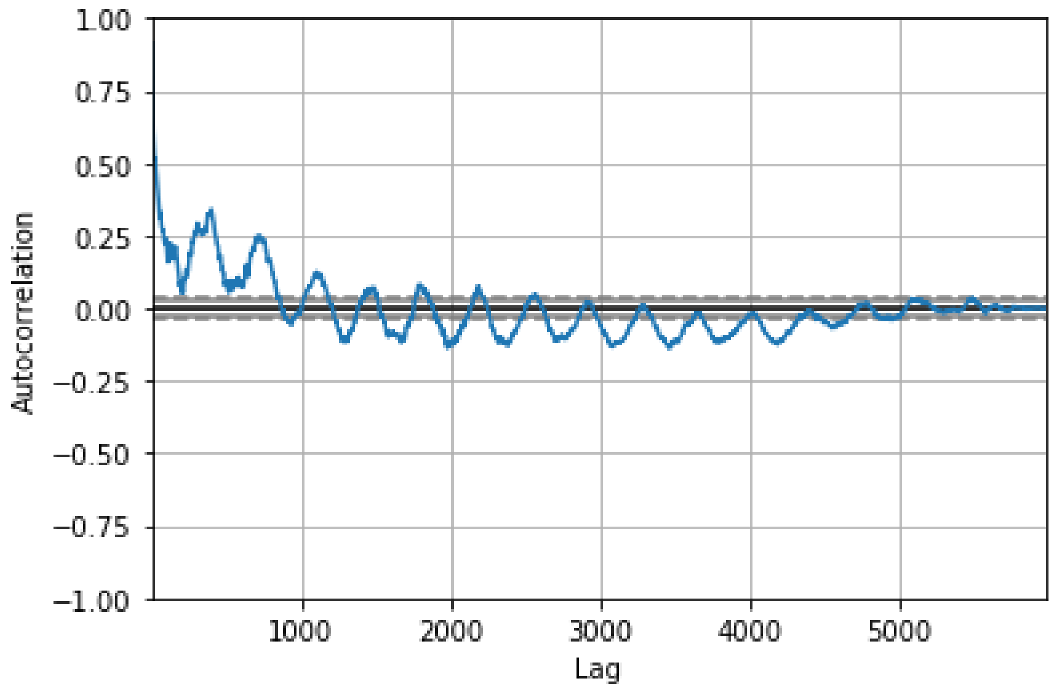
```
13.850346020761245 14.930219146482122
61.389080041745984 58.55856777204402
```

[51]:
```
X = b.values
low, high = X[:len(X)//2], X[len(X)//2:]
print (low.mean(), high.mean())
print (low.var(), high.var())
```

```
17.67876992357058 20.766865734120145
30.21849299020105 5.0722175008291615
```

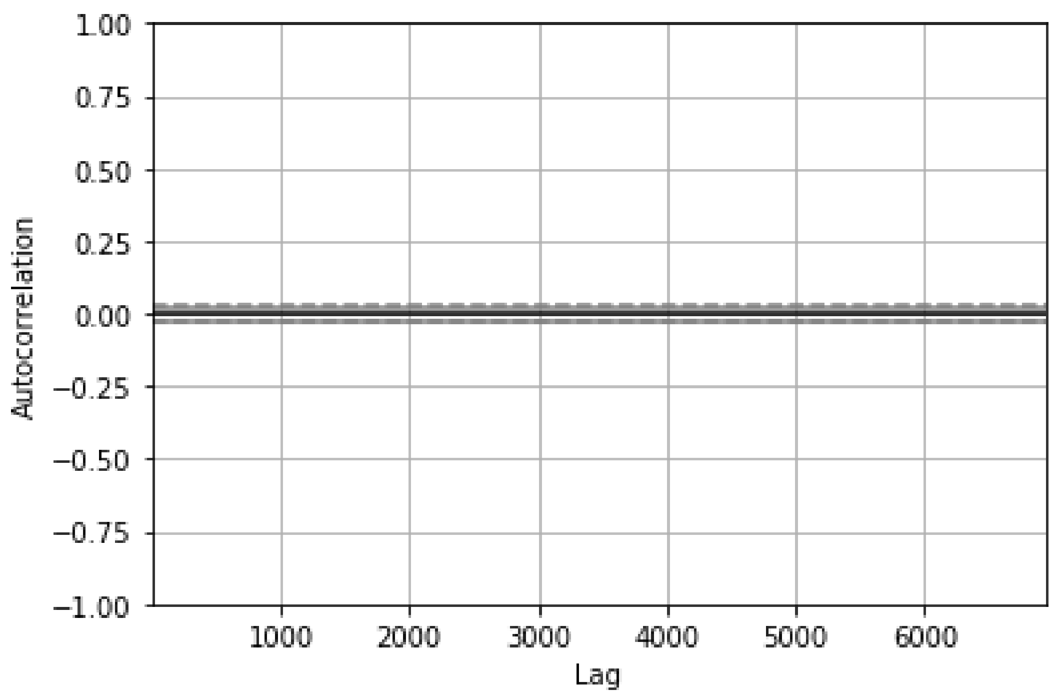
[60]:
```
autocorrelation_plot(b)
```

[60]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9f0c28b310>



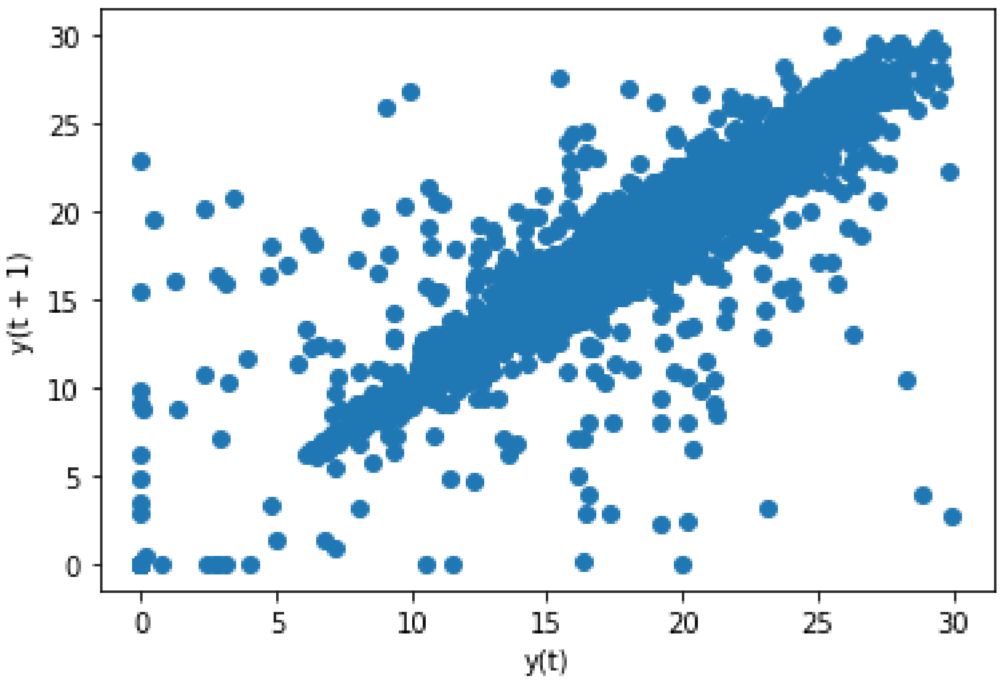

```
[61]: autocorrelation_plot(m)
```

[61]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9f0c3c5390>



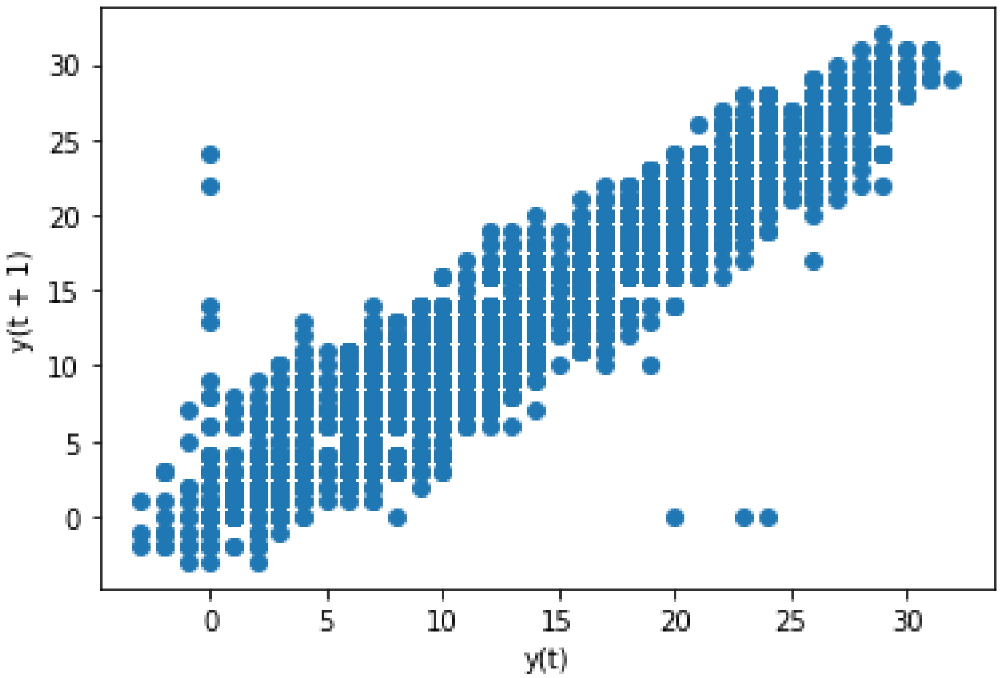
[58]:
```
lag_plot(b)
```

[58]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9f0fdb6050>



[59]:
```
lag_plot(m)
```

[59]: <matplotlib.axes._subplots.AxesSubplot at 0x7f9f0c1b6550>



[65]:
```
# Brazil and Madrid datasets are both stationary and there no increasing trends
# We can reach to this conclusion by checking autocorrelation, lag, histogram
↪and default plottings.
# In addition we also used adfuller test and for both datasets p-value is lower
↪than 0.05
```