**Feature Engineering for Time Series #3: Lag Features**

Here’s something most aspiring data scientists don’t think about when working on a time series problem – we can also use the target variable for feature engineering!

Consider this – you are predicting the stock price for a company. So, the previous day’s stock price is important to make a prediction, right? In other words, the value at time *t* is greatly affected by the value at time *t-1*. The past values are known as lags, so *t-1* is lag 1, *t-2* is lag 2, and so on.

Here, we were able to generate lag one feature for our series. But why lag one? Why not five or seven? That’s a good question.

The lag value we choose will depend on the correlation of individual values with its past values.

If the series has a weekly trend, which means the value last Monday can be used to predict the value for this Monday, you should create lag features for seven days. Getting the drift?

We can create multiple lag features as well! Let’s say we want lag 1 to lag 7 – we can let the model decide which is the most valuable one. So, if we train a linear regression model, it will assign appropriate weights (or coefficients) to the lag features:

import pandas as pd

data = pd.read\_csv('Train\_SU63ISt.csv')

data['Datetime'] = pd.to\_datetime(data['Datetime'],format='%d-%m-%Y %H:%M')

data['lag\_1'] = data['Count'].shift(1)

data['lag\_2'] = data['Count'].shift(2)

data['lag\_3'] = data['Count'].shift(3)

data['lag\_4'] = data['Count'].shift(4)

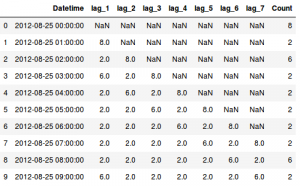
data['lag\_5'] = data['Count'].shift(5)

data['lag\_6'] = data['Count'].shift(6)

data['lag\_7'] = data['Count'].shift(7)

data = data[['Datetime', 'lag\_1', 'lag\_2', 'lag\_3', 'lag\_4', 'lag\_5', 'lag\_6', 'lag\_7', 'Count']]

data.head(10)



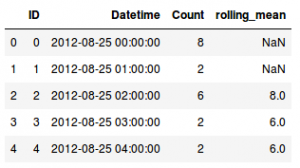
Feature Engineering for Time Series **Rolling Window**

In the last section, we looked at how we can use the previous values as features.

How about calculating some statistical values based on past values? This method is called the rolling window method because the window would be different for every data point.

Now the question we need to address – how are we going to perform feature engineering here? Let’s start simple. We will select a window size, take the average of the values in the window, and use it as a feature. Let’s implement it in Python:

|  |  |
| --- | --- |
|  | import pandas as pd |
|  | data = pd.read\_csv('Train\_SU63ISt.csv') |
|  | data['Datetime'] = pd.to\_datetime(data['Datetime'],format='%d-%m-%Y %H:%M') |
|  |  |
|  | data['rolling\_mean'] = data['Count'].rolling(window=7).mean() |
|  | data = data[['Datetime', 'rolling\_mean', 'Count']] |
|  | data.head(10) |



Exercise

Try to explain the code below, what are we trying to do?

data['MA3'] = data['Value'].shift(1).rolling(window=3).mean()

data['MA9']= data['Value'].shift(1).rolling(window=9).mean()

# Dropping the NaN values

data = data.dropna()

# Initialising X and assigning the two feature variables

X = data[['MA3','MA9']]

By having set-up the explanatory variables, it is time to initialize our dependant variable, y.

Setting-up the dependent variable

y = data['Value']