UW
DATA SCIENCE
CLUB.

TIME SERIES FORECASTING WORKSHOP

Presented by
Matthew Erxleben

New member sign-up link:

https://bit.ly/dsc-22-signup

Workshop Outline

- What is time series data? What is forecasting?
- Common time series forecasting models
- Seasonality
- Stationarity and Differencing
- Autocorrelation
- Autoregression (AR)
- Moving Average (MA)
- ACF and PACF
- ARIMA Model in code

Prerequisite Knowledge

- Python
- Basic Statistics
- Beginner level data libraries (pandas, matplotlib, numPy)
- Experience with Jupyter Notebooks

What is time series data?

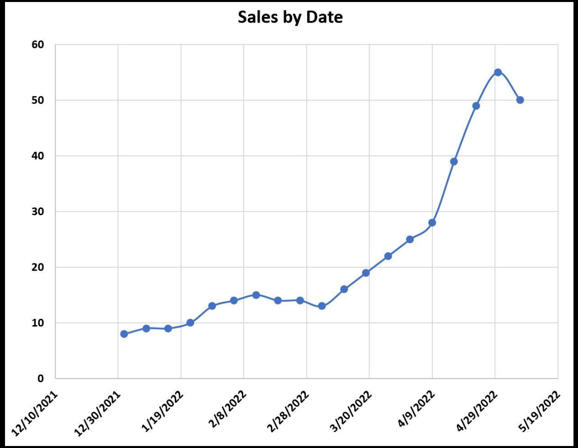
Time series data is historical data that has a singular value per unit of time

E.g:

- Weekly sales of bananas at the grocery store
- There is only 1 number of banana sales the store had on a given week

Common time series datasets:

- Sales
- Stock prices
- Daily temperatures
- An individual's weight
- Etc ...



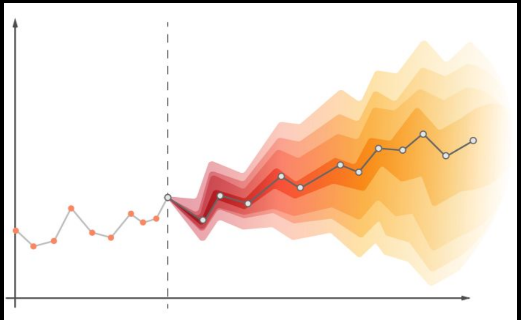
What can we do with time series data?

- The data might have some trends or relationships to each other: Autocorrelation, Seasonality, External factors...

- Based on the data's relationships we can forecast into the future and predict what a value will be in X amount of time

E.g:

- Predicting how many bananas the grocery store will sell next week



Importing Libraries + Data

Lets import some important libraries we will need for this workshop

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.arima.model import ARIMA
```

Lets load in some sample data from Kaggle to do our time series analysis and forecasting with.

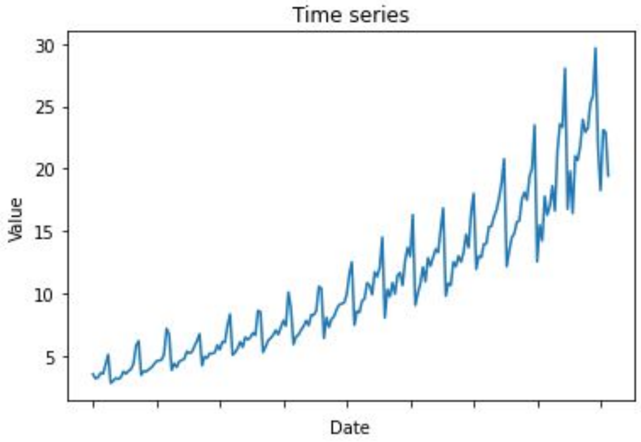
```
df = pd.read_csv('C:/Users/matth/OneDrive/Desktop/UW Data Science Club/Time series forecasting Workshop/data.csv')
```

Here we have a data set that is historical data of a value from 1991 to 2008. As we can see, we have 2 columns. One being the the date, and the other being a value. This is a time series.

Time Series Data

```
In [80]: plt.plot(df.value)
         plt.title('Time series')
         plt.ylabel('Value')
         plt.xlabel('Date')
         ax = plt.gca()
         ax.axes.xaxis.set_ticklabels([])
         plt.show
```

Out[80]: <function matplotlib.pyplot.show(close=None, block=None)>



Commonly used time series forecasting models

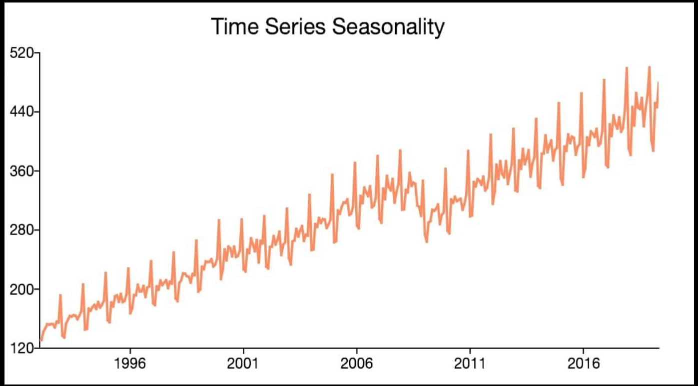
- AR (Autoregression)
- MA (Moving average)
- ARMA (Autoregressive moving average)
- ARIMA (Autoregressive integrated moving average)

What is Seasonality?

- Seasonality is a trend that occurs in time series data
- This trend is a predictable change that recur every calendar year

E.g.

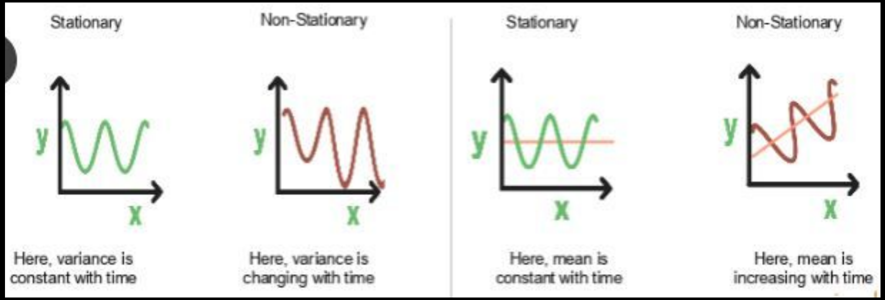
- Sales at retail stores increase every December due to Christmas shopping



What is Stationarity?

Stationarity requires:

- Constant Mean
- Constant Standard Deviation
- No seasonality



We need a time series to be stationary to do forecasting

How do we make a time series stationary?

What is Differencing?

Differencing:

- Taking the difference between consecutive values in a series
- E.g.
  - Banana sales last week = 10
  - Banana sales current week = 15
  - Difference: 15 - 10 = 5

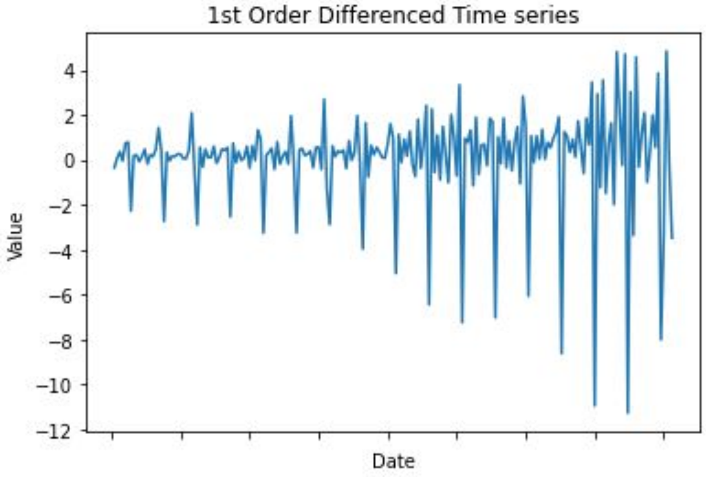
- Used to make the time series stationary

- **Order of differencing:** number of times differencing needs to occur to a time series for it to become stationary

1st Order Differencing

```
plt.plot(df.value.diff())
plt.title('1st Order Differenced Time series')
plt.ylabel('Value')
plt.xlabel('Date')
ax = plt.gca()
ax.axes.xaxis.set_ticklabels([])
plt.show
```

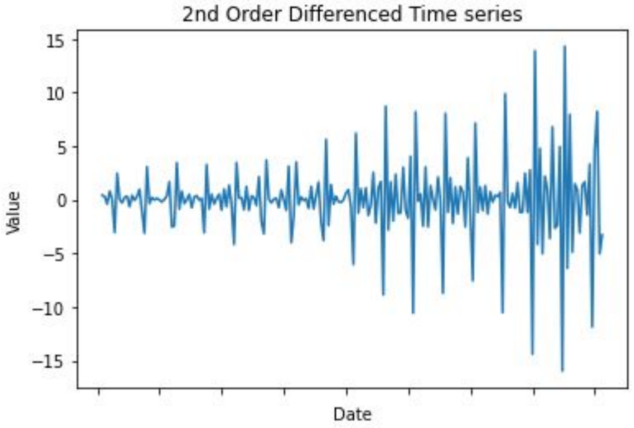
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



2nd Order Differencing

```
plt.plot(df.value.diff().diff())
plt.title('2nd Order Differenced Time series')
plt.ylabel('Value')
plt.xlabel('Date')
ax = plt.gca()
ax.axes.xaxis.set_ticklabels([])
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



The time series is now near stationary as we can see from the graph

What is Autocorrelation?

- Autocorrelation means that the data has a relationship with its own historical data

E.g.

- The amount of bananas that were sold last week has an effect on the current week's banana sales

What is Autoregressive model (AR)?

AR model: uses previous data to forecast future data (as there is an autocorrelation)

Lag: previous data point that is used to predict future data point

AR(p):

- p is a parameter to represent how many lagged terms are used
- E.g:
  - We saw **correlation** between **previous** weeks banana sales and **current** weeks banana sales, therefore use **previous** weeks sales to **forecast future** banana sales
  - Can be the previous week (p = 1), previous 2 weeks (p = 2), etc...

What should p equal? (AR(p))

PACF (Partial Autocorrelation Function): Direct correlation between a lag and its series. This test **excludes** the contributions from the intermediate lags.

- We can use this to see which lags are significant and which are not in an AR(p) model

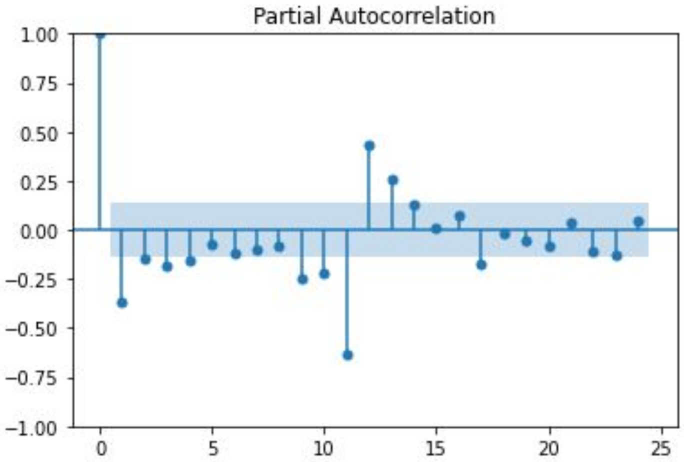
- p = Number of significant lags found in the PACF test

PACF:

We can use the PACF to see how many AR terms to use (number of lags)

```
plot_pacf(df.value.diff().dropna(), method='ywm')
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



As we can see, lag 1 is very sigificant as it is way outside the bounds. Therefore we can use 1 lag (p = 1) for our model

What is Moving Average (MA)?

MA model: uses previous lags errors to forecast future data

MA(q):

- q is a parameter to represent how many lagged forecast errors terms are used
- E.g:
  - Model that predicts expected banana sales based on the **error** of banana sales in the previous week
  - Can be the error from previous week (q = 1), previous 2 weeks (q = 2), etc ...

What should q equal? (MA(q))

ACF (Autocorrelation Function): Non-direct correlation between a lag and its series. This test **includes** the contributions from the intermediate lags.

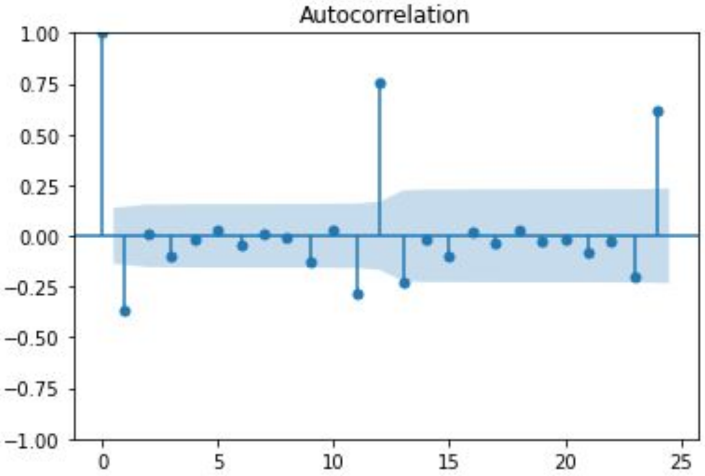
- We can use this to see which lags are significant and which are not in an MA(q) model
- q = Number of significant lags found in the PACF test

ACF:

We can use the ACF to see how many MA terms to use (number of lagged forecast errors)

```
plot_acf(df.value.diff().dropna())
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



As we can see, using 1 is very sigificant as it is way outside the bounds. Therefore we can use q = 1 for our model

What is ARIMA?

ARIMA (Autoregressive integrated moving average): Combining an AR with a MA model, and have it be stationary (this is what integrated means).

ARIMA(p, d, q):

- p = number of lagged terms used
- d = order of differencing
- q = number of lagged forecast errors terms used

We can use this model to forecast our time series data!

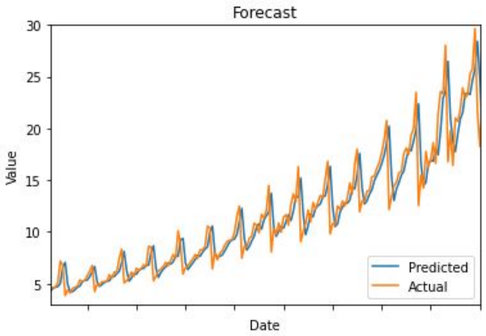
ARIMA Model:

```
arima_model = ARIMA(df.value, order=(1,2,1))
model = arima_model.fit()
results = model.predict()

plt.plot(results)
plt.plot(df.value)
plt.title('Forecast')
plt.ylabel('Value')
plt.xlabel('Date')
ax = plt.gca()
ax.axes.xaxis.set_ticklabels([])
plt.xlim([25, 200])
plt.ylim([3, 30])

plt.legend(["Predicted", "Actual"], loc ="lower right")
plt.show
```

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



As we can see, the forecast is pretty accurate in comparison to the actual data. This is using an ARIMA model of p=1, d=2, q=1

Thanks for listening!

Any Questions?

Please leave feedback at this link:

https://forms.gle/H87MdhT1h3sDruQQ7

I appreciate all the feedback I can get! :)