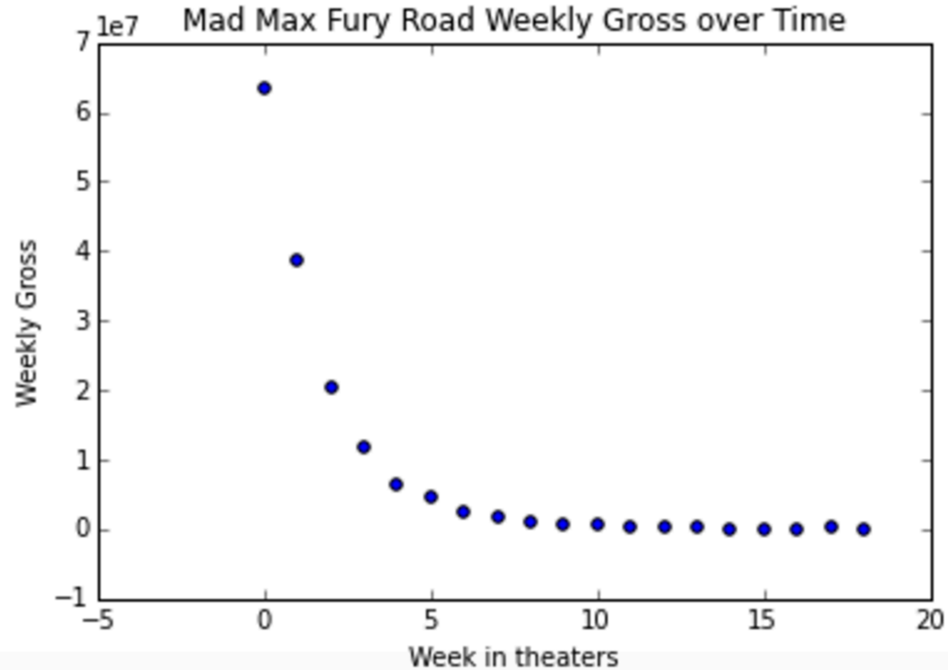
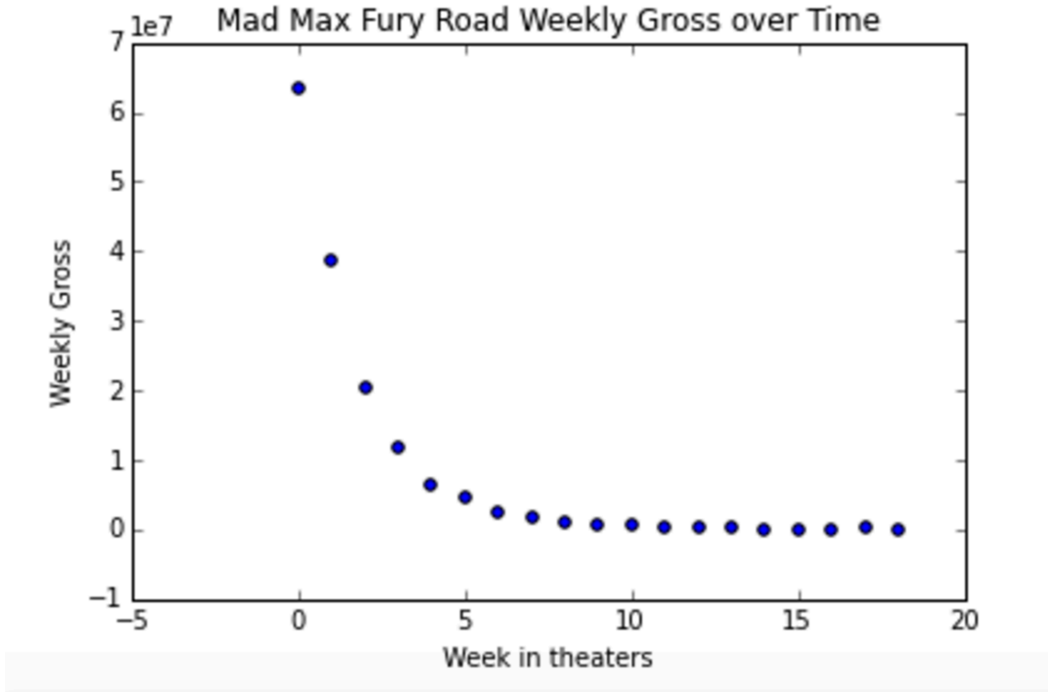
Time Series
REVISITED

DATA SCIENCE BOOTCAMP

Previously, on Time Series…



Oh no!
A time series!
What do we do?



Fear not!

Auto**R**egressive models

will save the day!

AR-2

auto-regressive model of order 2

What we know
(features)

What we predict
(target)

Gross of previous week
Gross of two weeks ago

Model

Gross of this week

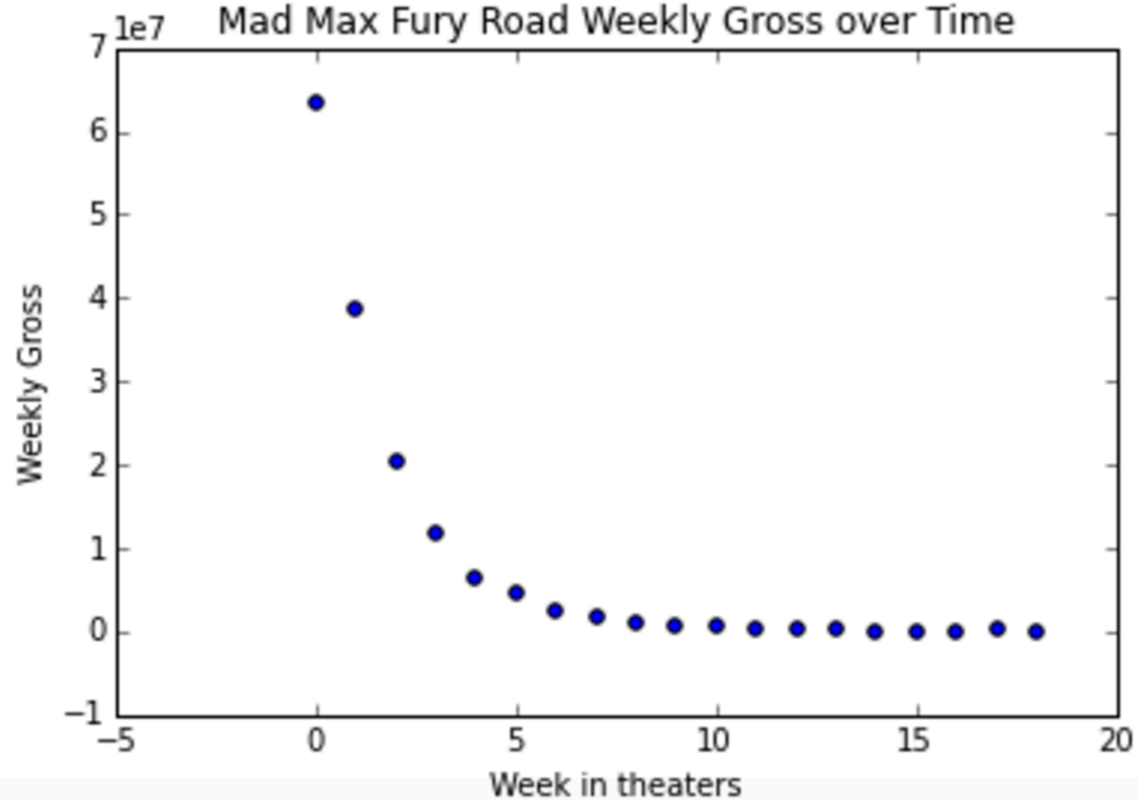
Target | Feature 1 | Feature 2

	weekly_gross	one_prev_weeks_gross	two_prev_weeks_gross
0	63440279	NaN	NaN
1	38849255	63440279	NaN
2	20544731	38849255	63440279
3	11643562	20544731	38849255
4	6309002	11643562	20544731
5	4555993	6309002	11643562
6	2648047	4555993	6309002
7	1645168	2648047	4555993
8	966275	1645168	2648047
9	601794	966275	1645168
10	663222	601794	966275

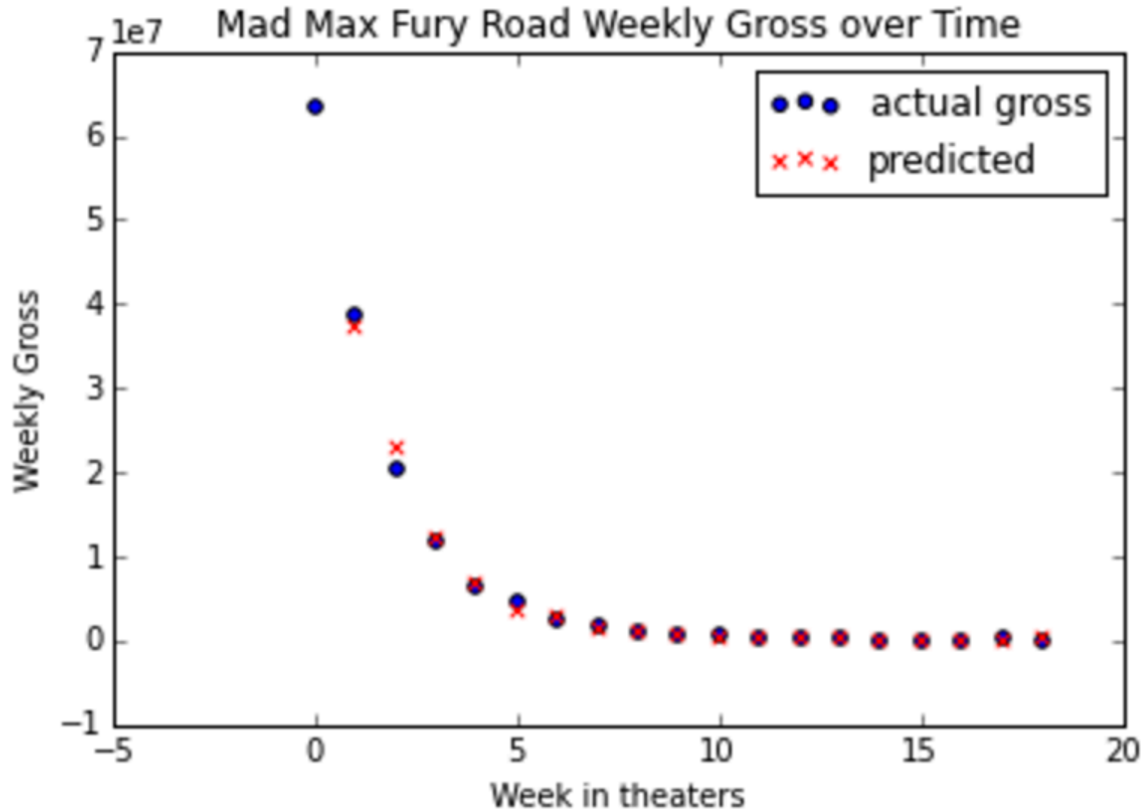
AR-2

__A__uto__R__egressive model with lag __2__

Two previous points as features

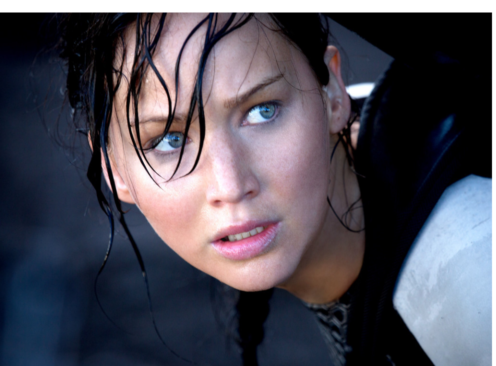

Mad Max Fury Road Weekly Gross over Time
1e7
Weekly Gross
Week in theaters
7
6
5
4
3
2
1
0
−1
−5
0
5
10
15
20

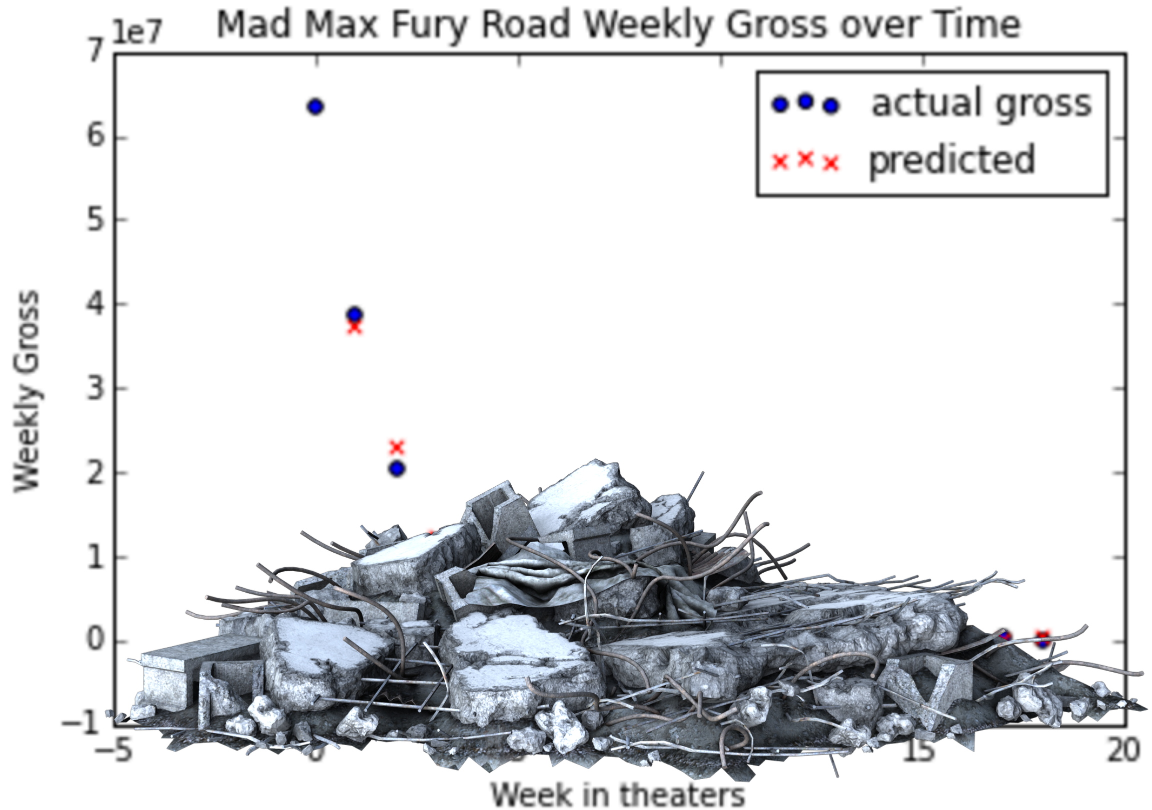
Here we go!


Mad Max Fury Road Weekly Gross over Time
1e7
Weekly Gross
Week in theaters
actual gross
predicted
7
6
5
4
3
2
1
0
−1
−5
0
5
10
15
20

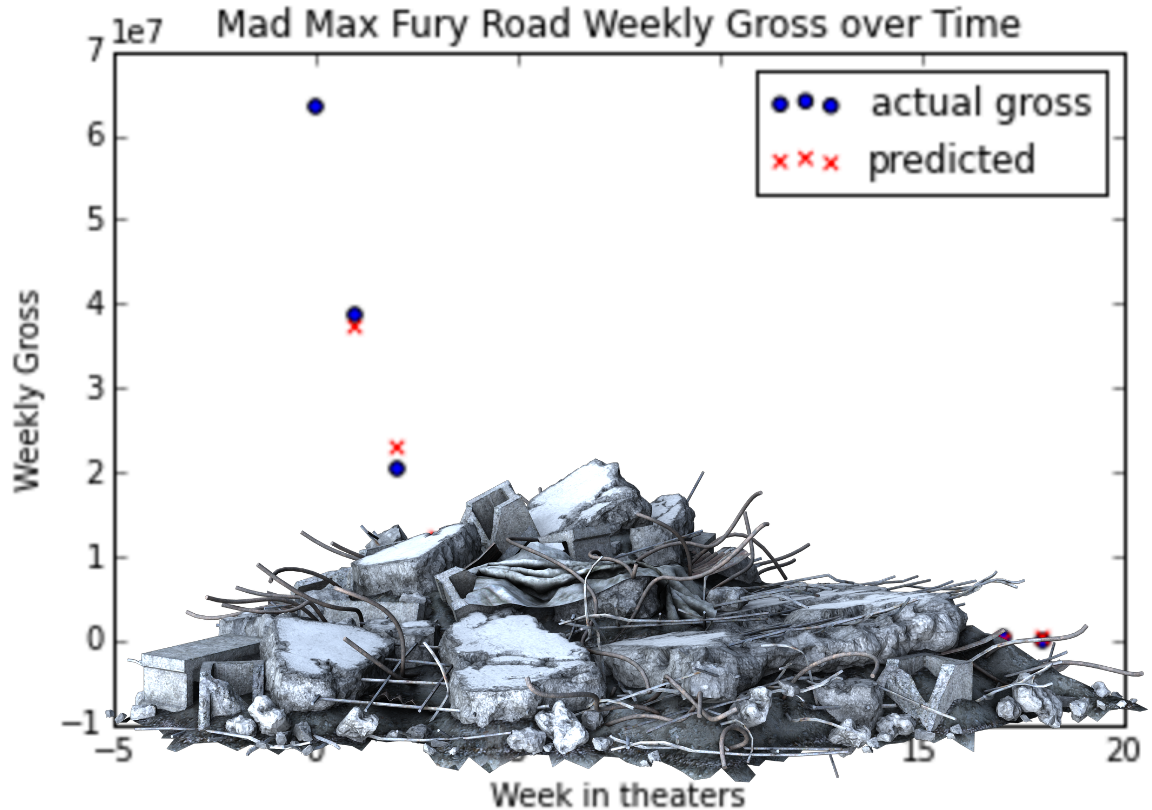
Here we go!


Mad Max Fury Ro
eekly
oss over Time
1e7
ctual gross
icted
Weekly Gross
Week in theaters
7
6
5
4
3
2
1
−1
−5
0
5
10
15
20




Mad Max Fury Road Weekly Gross over Time
1e7
actual gross
predicted
Weekly Gross
Week in theaters
7
6
5
4
3
2
1
0
−1
−5
0
15
20

We’re saved!



DIRECTED BY
Michael Bay

We are modeling a stochastic process

The value of each point in time is a linear combination of features plus luck factor

(or “noise”, the part our model doesn’t predict)

We are modeling a stochastic process

The value of each point in time is a linear combination of features plus luck factor

(or "noise", the part our model doesn't predict)

AR-2

(second order autoregressive)

$$x_t = \beta_0 + \beta_1 x_{t-1} + \beta_2 x_{t-2} + \varepsilon_t$$

We are modeling a stochastic process

The value of each point in time is a linear combination of features plus luck factor

(or “noise”, the part our model doesn’t predict)

$$x_t = \beta_0 + \beta_1 x_{t-1} + \beta_2 x_{t-2} + \varepsilon_t$$

Value at this point

Intercept (constant)

Coefficient 1

Value at previous point

Coefficient 2

Value at two-previous point

Random noise at this point

something like…

We expect this week's gross for our movie to be
$20 Thousand
+ 45% of last week's gross
+ 15% of the gross two weeks ago
+ luck factor
(random noise: roll the dice for +- $90 Thousand)

$$x_t = \beta_0 + \beta_1 x_{t-1} + \beta_2 x_{t-2} + \varepsilon_t$$

We are modeling a stochastic process

The value of each point in time is a linear combination of features plus luck factor

(or “noise”, the part our model doesn’t predict)

$$x_t = \beta_0 + \beta_1 x_{t-1} + \beta_2 x_{t-2} + \varepsilon_t$$

Value at this point

Intercept (constant)

Coefficient 1

Value at previous point

Coefficient 2

Value at two-previous point

Random noise at this point

Moving Average Models

A different approach to modeling time series as a stochastic process

MA-2

(second order moving average model)

$$x_t = \mu + \beta_0 + \beta_1 \varepsilon_{t-1} + \beta_2 \varepsilon_{t-2} + \varepsilon_t$$

Moving Average Models

A different approach to modeling time series as a stochastic process

$$x_t = \mu + \beta_0 + \beta_1 \varepsilon_{t-1} + \beta_2 \varepsilon_{t-2} + \varepsilon_t$$

Value at this point

Mean value over all time

Intercept (constant)

Coefficient 1

Random noise at previous point

Coefficient 2

Random noise at two-previous point

Random noise at this point

something like…

We expect this week's gross for our movie to be
average weekly gross
- $10 Thousand
+ 10% of the luck factor we got last week
- 5% of the luck factor we got two weeks ago
+ luck factor
(random noise: roll the dice for +- $90 Thousand)

$$x_t = \mu + \beta_0 + \beta_1 \varepsilon_{t-1} + \beta_2 \varepsilon_{t-2} + \varepsilon_t$$

Moving Average Models

A different approach to modeling time series as a stochastic process

$$x_t = \mu + \beta_0 + \beta_1 \varepsilon_{t-1} + \beta_2 \varepsilon_{t-2} + \varepsilon_t$$

Value at this point

Mean value over all time

Intercept (constant)

Coefficient 1

Random noise at previous point

Coefficient 2

Random noise at two-previous point

Random noise at this point

Autoregressive (AR)

What we know (features)

What we predict (target)

Previously observed values → Model → Value at this time point

Moving Average
(**MA**)

What we know
(features)

Random noise contributions to the previously observed values → Model → **Value at this time point**

What we predict
(target)

Moving Average
(**MA**)

What we know (features)

What we predict (target)

Random noise contributions to the previously observed values → Model → **Value at this time point**

A different set of lag features. Separate the past values into their noise and non-noise contributions (luck factor vs everything else). Use noise contributions as the features.

Wait…what?

Why would past noise be the specific thing to contribute to the current point's value, rather than the whole past value (like the sensible AR)?

Wait…what?

We are using **stochastic processes**.
The noise is what our model doesn’t know about.

We call it “luck factor”, because the reasons that determine those changes are not known to the model, it cannot be predicted, only revealed, so it is not different to us than rolling some dice.

Not because it really is a random process in nature, it is a collection of deterministic processes. But when we don’t know these deterministic processes and cannot model them directly, we model their collective contribution as a noise term (stochastic process).

for example…

We expect a week’s gross for a movie to be
$10 Thousand
+ 60% of what it made last week
+ luck factor
(roll the dice for +- $110 Thousand)

for example…

We expect a week's gross for a movie to be
$10 Thousand
+ 60% of what it made last week
+ luck factor
(roll the dice for +- $110 Thousand)

This is actually determined by tons of stuff, such as
did it rain all week this week,
did another big blockbuster in the same genre open this week, …etc.
But we do not have data on these, so we cannot train a model with them.
All such contributions are modeled by the luck factor (what we can't predict).

Moving Average
(MA)

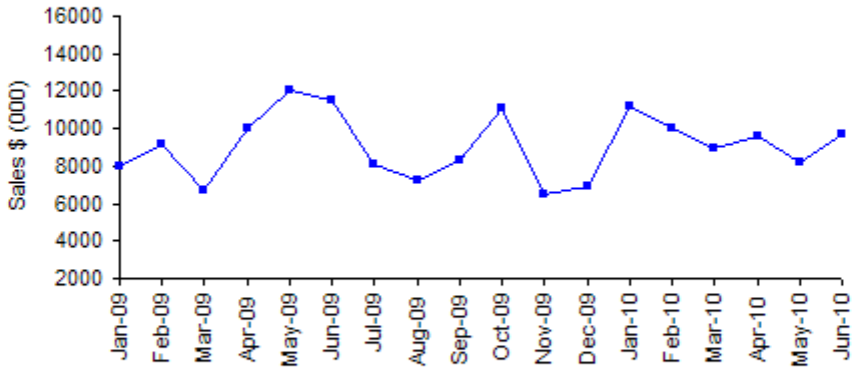
What we know (features)

What we predict (target)

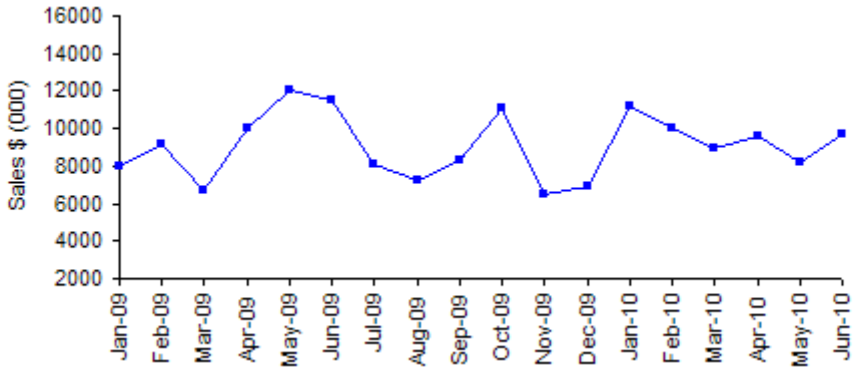
Random noise contributions to the previously observed values → Model → **Value at this time point**

If we think that these stochastic parts (the processes we model with random noise) have a direct effect on future time points (rather than the full sum of deterministic + stochastic parts), MA is the appropriate model

Example: Effect of ads on sales



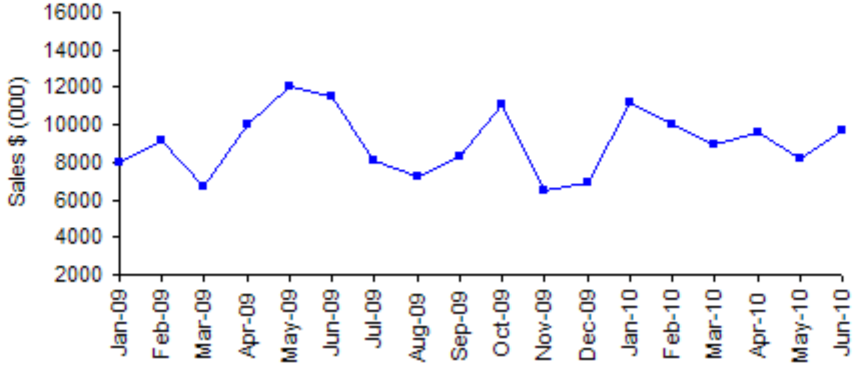
Example: Effect of ads on sales



We are modeling the monthly sales of an item.

Imagine that we come up with a new ad every month, and advertise the same volume (but the content changes)

Example: Effect of ads on sales

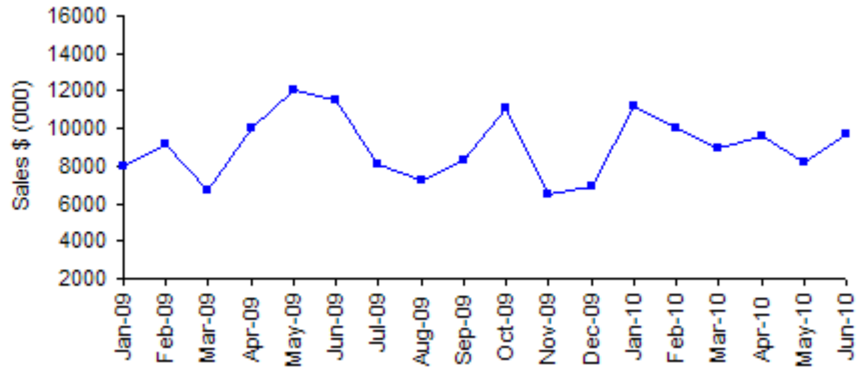


We are modeling the monthly sales of an item.

Imagine that we come up with a new ad every month, and advertise the same volume (but the content changes)

We cannot quantify the "quality" of the ad outside of the final sales, so we cannot use it as an input feature.
It goes into the luck factor.

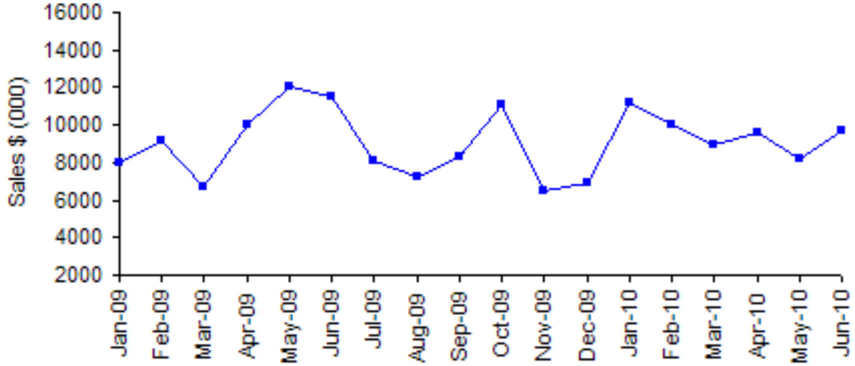
Example: Effect of ads on sales



If one month's ad is good, we get a positive bump to sales,
If it is bad, we sell less. We cannot

But a good ad's impression left in minds does not disappear immediately. The overall effect for today is last month's ad effect to a small degree, and this months effect. Perhaps even the ad from two months ago also has a tiny portion of the effect.

Example: Effect of ads on sales



In this case, we know that the change is coming from the luck factor (noise), and the noise of the previous points have an effect on this point's value.

This is a Moving Average Model.

Remember!

Moving Average Smoothing

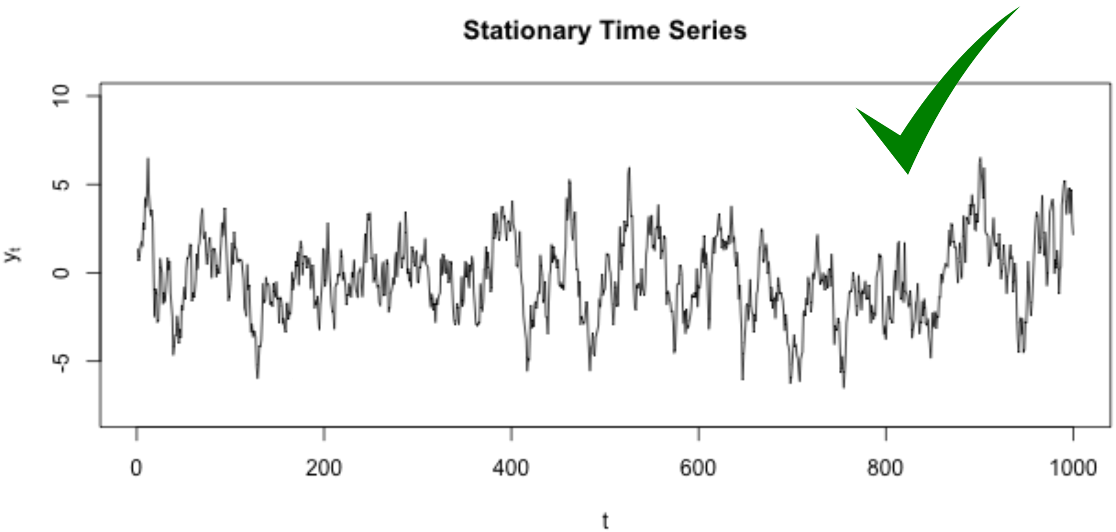
is <u>**not**</u>

Moving Average Models (MA)

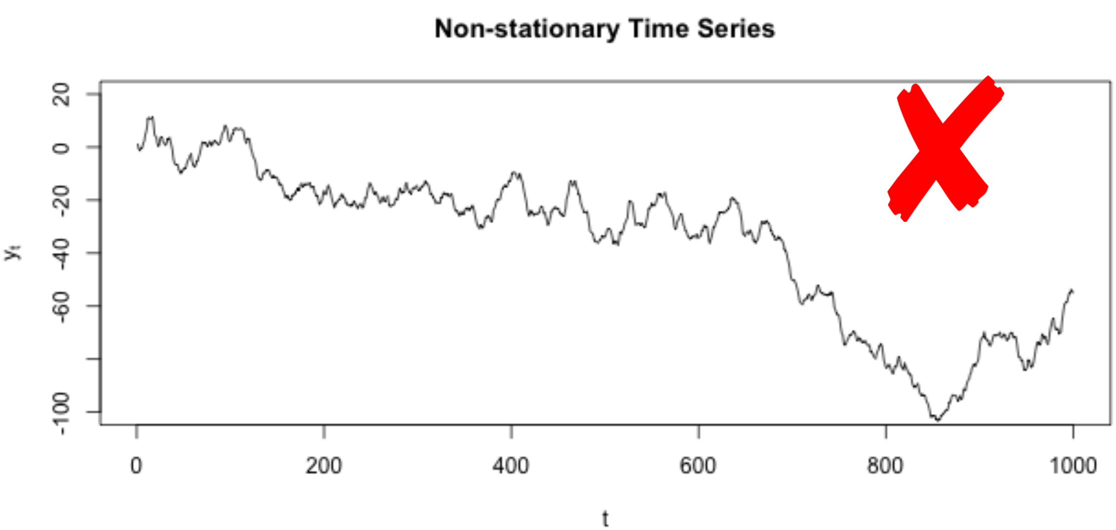
One is preprocessing to remove some of the noise

The other is a way of modeling time series

MA is for stationary time series



For MA to work, all of the change to the value should be coming from the stochastic part (luck factor, noise).



This only works for stationary time series (mean, std etc of the series stays the same)

Nonlinear Fitting

$$x_t = \mu + \beta_0 + \beta_1 \varepsilon_{t-1} + \beta_2 \varepsilon_{t-2} + \varepsilon_t$$

MA uses noise from previous points (ε_{t-i}) as input features.

But at any time *t* we do not observe the deterministic and stochastic parts separately. We only see x_t, not the different contributions from luck factor and others to the final value.

This means we cannot just do regular least squares fitting (like we did in AR). We need to use complicated non-linear algorithms such as **Box-Jenkins** (we won't get into the mathy details of that here)

ARMA

In the TV sales example, using MA features is good to capture the ad effect, but there may be info on the non-luck-factor side as well, which you don't want to miss. In some cases you want to use both types of features. Since these are linear models, just add them all up!

What we know (features) | What we predict (target)

Previously observed values → **AR** → **Value at this time point**

ARMA

In the TV sales example, using MA features is good to capture the ad effect, but there may be info on the non-luck-factor side as well, which you don't want to miss. In some cases you want to use both types of features. Since these are linear models, just add them all up!

What we know (features)

Random noise contributions to the previously observed values → MA → **Value at this time point**

What we predict (target)

ARMA

In the TV sales example, using MA features is good to capture the ad effect, but there may be info on the non-luck-factor side as well, which you don't want to miss. In some cases you want to use both types of features. Since these are linear models, just add them all up!

What we know (features)

What we predict (target)

Previously observed values
Random noise contributions to the previously observed values

→ **ARMA** →

Value at this time point

ARMA

```
sm.tsa.ARMA(timeseries, (3,0)).fit()
```
AR3

```
sm.tsa.ARMA(timeseries, (0,1)).fit()
```
MA1

```
sm.tsa.ARMA(timeseries, (2,1)).fit()
```
AR2-MA1

What we know (features)

What we predict (target)

Previously observed values
Random noise contributions to the previously observed values

→ **ARMA** →

Value at this time point

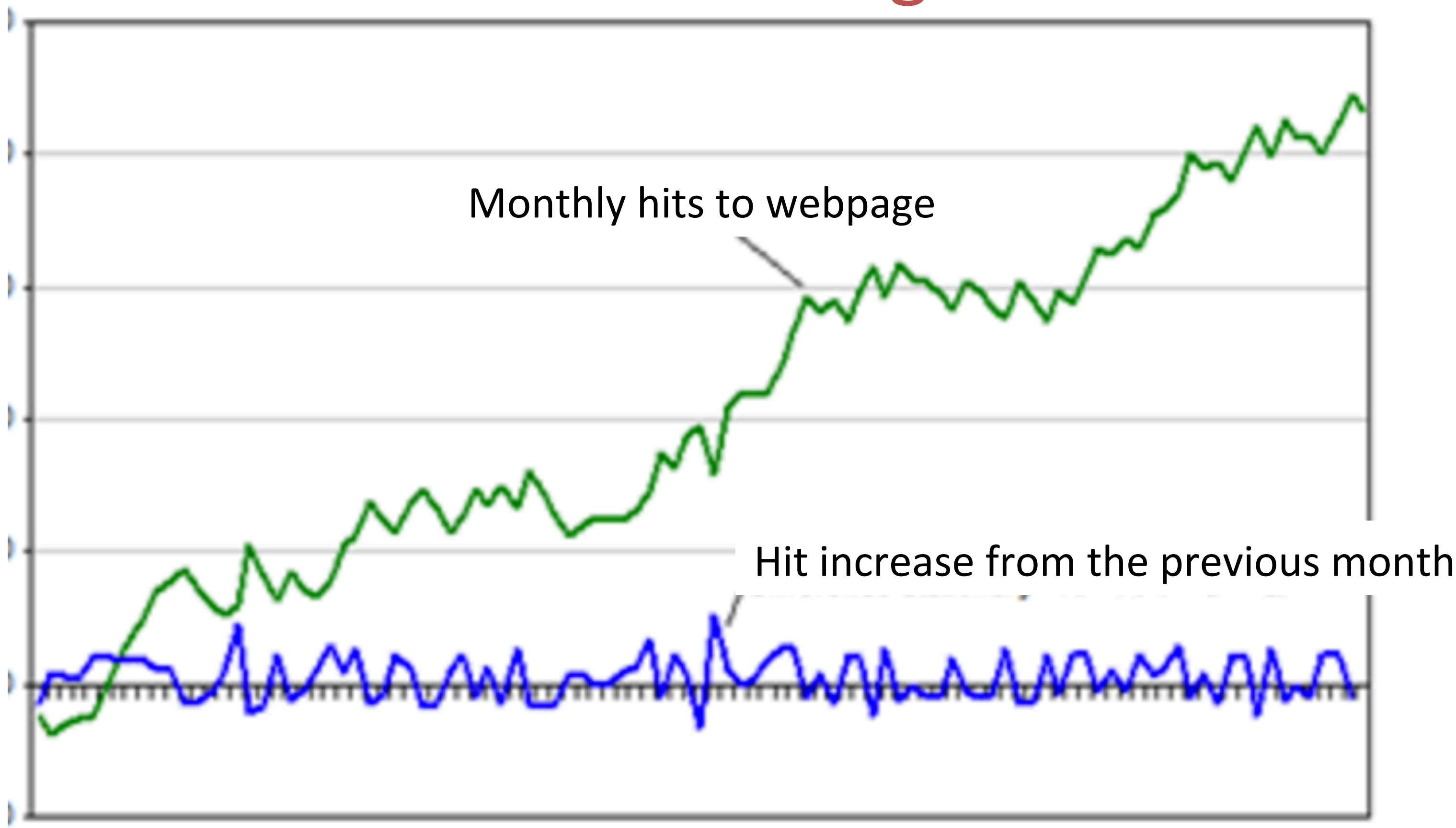
Dealing with Non-stationary Time Series

MA only works for stationary time series, so how can we process the data to use ARMA models to maximum efficiency?

Dealing with Non-stationary Time Series

MA only works for stationary time series, so how can we process the data to use ARMA models to maximum efficiency?

Differencing



Differencing

First order differencing (value increase velocity)

Subtract the previous point's value (can be positive/negative)

Second order differencing (value increase acceleration)

Do first order differencing, then do it again: subtract the previous point's 'increase' value from this points

Third order differencing (value increase jerk)

Basically you can do differencing repeatedly over and over again – This may be necessary since sometimes lower order differencing still leads to nnon-stationary data.

Dealing with Non-stationary Time Series

Also…

Seasonal differencing

Instead of subtracting previous point's value, subtract the value from the previous *season* corresponding to this point

For example, for monthly sales data over years 2005-2015, to represent sales of July '15, you can use
[July '15 sales] – [July '14 sales]
or, if July sales did not have a consistent upwards or downwards trends in all the previous years:
[July '15 sales] – [average July sales from previous years]

ARIMA

If we do differencing first, and then fit an AR model, or an MA model, or include both types of features and fit an ARMA model, we are doing ARIMA

What we know (features)

What we predict (target)

Previously observed values
Random noise contributions to the previously observed values

→ **ARMA** →

Value at this time point

ARIMA

If we do differencing first, and then fit an AR model, or an MA model, or include both types of features and fit an ARMA model, we are doing ARIMA

Difference the time series before fitting

What we know (features)

What we predict (target)

Previously observed values
Random noise contributions to the previously observed values

ARIMA

Value at this time point

"I" for differencing?

“I” for differencing?

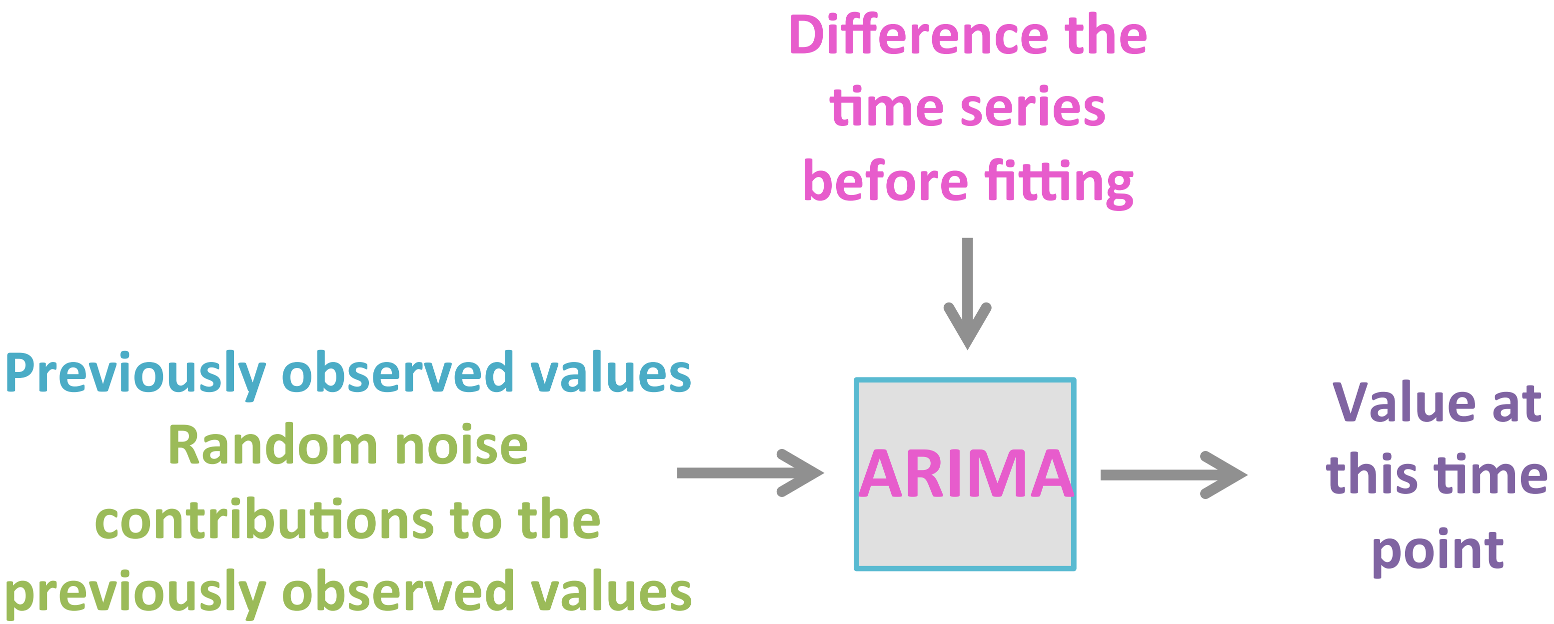
Auto**r**egressive

Integrated .....ah, right. differencing.

Moving **A**verage

ARIMA

ARIMA of orders (3, 2, 0) means
Do 2nd order differencing, then fit a 3rd order AR
(since MA is 0th order,
we are not using any MA features)



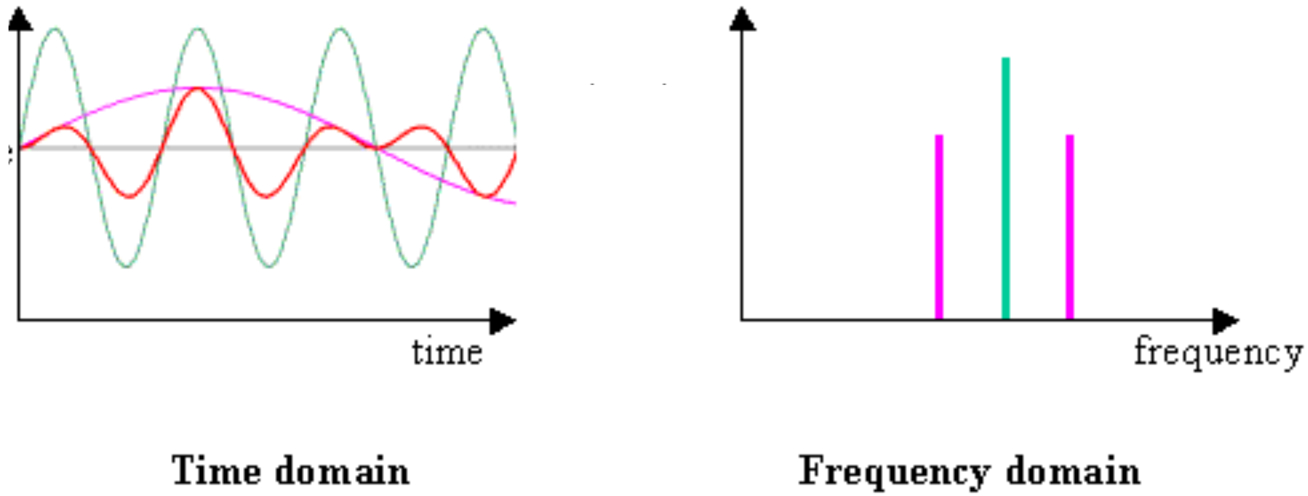
Frequency Domain Analysis

Our efforts were all in the time domain.

It is also possible to look at a time series as a combination of repeated patterns over different scales.

Like, sales data being the combination of the
daily sales cycle +
Weekly sales cycle +
Monthly sales cycle +
Yearly sales cycle … etc.

Frequency Domain Analysis



Once converted to frequency domain (spectrum),
You can gain insights over the periodic changes
And can do other analyses that fall under *spectral analysis*
(which we won't cover here)