# Using trained (AR-XYZ) models with new data #2788

Open **Acertainuser** opened this issue on Jan 26, 2016 · 30 comments

**Labels** comp-tsa FAQ type-enh

Acertainuser commented on Jan 26, 2016

Hello I do timeseries forecasting with TSA, AR, ARIMA, ARMA. What I would like to be doing is, training the models with a subset of the data I have to get the weights and then try the models on new unseen data. I could not figure out a way to do this in a clean way, since the training data + fitting results seem to be calculated in the fit() function and directly used in the predict() function.

Could you give me a hint how to do this or where to find this in the documentation? Maybe there is a way replacing the internal endogen values and reacalculate using the old weights?

Thanks
David

6

josef-pkt commented on Jan 27, 2016

I don't think this is documented and there is a direct user friendly version.
There is a helper function for the prediction in the models that can be used for prediction given some initial conditions. Getting also other features for example prediction standard errors might be more difficult.

Prediction for an ARMA process with known coefficient is just a filtering problem. scipy.signal.lfilter works well but has a bit complicated initial conditions.

If you need more than the predicted or filtered values of the results instance, then it might be easier to create a model with the new data, but insert the old parameters.
(I'm planning to allow maxiter=0 in the fit methods to use the given start_params, but that's not implemented yet.)
Inserting and overwriting endog and resid might be possible in the simple univariate case (no trend, and no exog).

josef-pkt added comp-tsa type-enh FAQ labels on Jan 27, 2016

ChadFulton commented on Jan 27, 2016

You can do this using state space models; as you suggest, it requires creating two models and inserting the fitted parameters from the training sample into the full model. You could do something like the following:

```
import numpy as np
import pandas as pd
import statsmodels.api as sm

dta = sm.datasets.macrodata.load_pandas().data
dta.index = pd.date_range(start='1959-01-01', end='2009-07-01', freq='QS')

training_mod = sm.tsa.SARIMAX(dta.realgdp.ix[:'1999-10-01'], order=(4,1,1))
training_res = training_mod.fit()

mod = sm.tsa.SARIMAX(dta.realgdp, order=(4,1,1))
res = mod.filter(training_res.params)

insample = res.predict()
T = len(dta.realgdp.ix['2000-01-01':])
forecast_error = dta.realgdp.ix['2000-01-01':] - insample.ix['2000-01-01':]
print(np.sqrt(np.sum(forecast_error**2) / T))
```

12 5

Acertainuser commented on Jan 27, 2016

Hey thanks for the reply. I will try ChadFultons solution until this is changed.

@josef-pkt I also thought that just having maxiter=0 and using the model.params as initial parameters would be a straight forward and intuitive solution to that challenge.

josef-pkt commented on Jan 27, 2016

the estimation sample. One idea would be to add also a pre-existing cov_params for this use case.
(My main use case for maxiter=0 was to get results for constrained parameters on the original data.)

I think we need a more specific solution for prediction for a different sample in tsa. This is built-in in the predict for models outside of tsa.
For tsa we started with ARMAProcess and VARProcess to have a representation of an already parameterized process, that can be used for filtering and for properties of the process. However, that doesn't know anything about parameter uncertainty and cannot generate prediction intervals or confidence intervals.

josef-pkt mentioned this issue on Mar 29, 2016

save calibration parameter (mle_model_results.MLEResults) #2867

Open

ChadFulton mentioned this issue on Oct 11, 2016

Problem in ARIMA endog change - using new data for prediction #3233

Closed

pedromorfeu commented on Oct 28, 2016 • edited

@ChadFulton , does .filter() exist for ARIMA models?

ChadFulton commented on Oct 28, 2016

Only `sm.tsa.SARIMAX` models have the `filter` method, but SARIMAX includes ARIMA models. For example, for an ARIMA(1, 2, 3) model, you could do:

```
mod = sm.tsa.SARIMAX(endog, order=(1, 2, 3))
```

4

josef-pkt commented on Oct 28, 2016 • edited

it is ARMAProcess. (But at the time there were some problems with the initial conditions in scipy.signal.lfilter and test coverage of ARMAProcess is not great. I'm not sure what the status is.) lfilter does the standard 1-step ahead prediction if the ARMA polynomials are specified.

1

**pedromorfeu** commented on Oct 28, 2016

Thanks to both for your immediate response. I'm now able to implement the proposed solution.

My use case is that I need to perform some changes over the predicted values and then use that new data in new predictions, over and over again. So, as per your suggestion, in every iteration of my loop, I should create the second SARIMAX, filter and predict. With your solution I save the time of fitting a new model with the new data. Is your approach still valid?

This is something like what can be done in R using `forecast` package's `Arima` method, where you can use a previously estimated model (https://cran.r-project.org/web/packages/forecast/forecast.pdf):

> parameter model: Output from a previous call to Arima. If model is passed, this same model is fitted to y without re-estimating any parameters

**makmanalp** commented on Dec 7, 2016

Just linking issues for posterity: #3212

**pedromorfeu** commented on Jan 3, 2017 • edited

Guys,
I've just installed statsmodels 0.8.0rc1 in my MacOS, but can't find `SARIMAX` in package `statsmodels.tsa.statespace.sarimax` as stated in the documentation in http://www.statsmodels.org/dev/generated/statsmodels.tsa.statespace.sarimax.SARIMAX.html.

It's strange because it was working in my Windows PC (but under package `statsmodels.tsa.api` ).

**ChadFulton** commented on Jan 3, 2017

pedromorfeu commented on Jan 3, 2017

In the end the version 0.8.0rc1 wasn't correctly installed (as PyCharm said).

```
> import statsmodels as sm
> print(sm.__version__)
0.6.1
```

Is version 0.8.0rc1 installable from *pip*?

pedromorfeu commented on Jan 3, 2017

Hi **@ChadFulton**, it's fixed now. I had to uninstall the previous statsmodels version (0.6.1).

ChadFulton mentioned this issue on Apr 25, 2017

**How can ARIMA do one step predict with new test data?** #3623

Open

ChadFulton mentioned this issue on Oct 6, 2017

**How to get prediction within the dates provided in sample but with different exog variable values** #3969

Closed

jengelman commented on Jan 2, 2018

**@ChadFulton** This is a kinda old issue, but I wanted to check if creating a new model is still the only way to perform online state estimation, and if you have any plans to change this.

As far as I can tell, the way to do online updates is something like this, for a (simple example) AR(1) model:

```
params = model.fit().params
while True:
    last_obs = model.data.endog[-1]
    new_obs= GetNewObservation()
    observations = [last_obs, new_obs]
    model = SARIMAX(observations, **kwargs)
    res = model.filter(params)
    prediction = res.predict()[-1]
    DoSomethingWithPrediction(prediction)
    params = res.params
```

Given that pykalman is no longer maintained, statsmodels has the only (reasonably performant) open source Python Kalman filter implementation. If there was a way to append or substitute new data to a model without creating a new one, one could use MLEModel.update() instead of copying the params every time.

2

ChadFulton mentioned this issue on Jan 3, 2018

ENH: Add support for on-line filtering in state space models. #4188

Open

ChadFulton commented on Jan 3, 2018

(I opened up an issue for this topic, but I wanted to mention something here).

The way you have it set up is actually not quite right, because for the models within the `while` loop, you would want to initialize the state mean vector and covariance matrix with the last ones from the previous model. See this example:

http://nbviewer.jupyter.org/gist/ChadFulton/d744368336ef4bd02eadcea8606905b5

jengelman commented on Jan 3, 2018

@ChadFulton right, forgot about initialize_known, thanks! Excited for PR #4042 as well.

colonder commented on Jun 4, 2018

ChadFulton commented on Jun 4, 2018

> is your jupyter example still valid? I'm getting error 'SARIMAX' object has no attribute 'predicted_state'

It still works for me if when I downloaded it and ran all the cells. Have you modified it somehow?

colonder commented on Jun 4, 2018

All I have different from the example is that my initial model is ARMA, not SARIMAX. I fit it, and that's my initial `res`. Then I used a loop like this:

```
    for t in range(data_len):

        last_obs = model.data.endog[-1]
        new_obs= data.iloc[t]
        observations = [last_obs, new_obs]
        model = sm.tsa.SARIMAX(endog=observations, order=(4,4,0))
        model.initialize_known(model.predicted_state[:, -2], model.predicted_state_cov[:, :,
-2])
        res = model.smooth(params)
```

colonder commented on Jun 4, 2018

Oh and initial `params` are from ARMA

ChadFulton commented on Jun 4, 2018

> `model.initialize_known(model.predicted_state[:, -2], model.predicted_state_cov[:, :, -2])`

This line is the problem. You probably mean `res.predicted_state` or something here?

I corrected it, but the problem is still the same. Nevertheless, I figured another way to do predictions

 xxxyyy13 commented on Aug 21, 2018 • edited

@ChadFulton I trained a SARIMAX model with 11-month(from January to November) data, and then I constructed a new SARIMA model with the same order and seasonal order as the previous trained model and only December data. And then I applied the Kalman filter on the new model with the parameters from the previous trained model. But the results looks really bad. However, if I use the whole year data to construct the model instead of only December data and then apply filter on it, the result looks as similar as the previous trained model. I am quite confused about the different predictions I got because I thought the model parameters are the same right?

  franchesoni mentioned this issue on Oct 2, 2018

PERF: Increase in training size makes prediction slower. #5287

Closed

  nicholasg97 mentioned this issue on Mar 27, 2019

Generating One Generalized Model for Multiple Datasets? #5549

Open

  dinhtrang24 mentioned this issue on Jul 8, 2020

VAR - How to get one-step ahead static forecast? #6863

Closed

 hrudhai98 commented on Sep 22, 2020

@ChadFulton I am training an ARIMA model that requires me to append data everyday. There are two issues that I'm facing with this.
When I try to append data to the model, it says that ARIMAResults object has to attribute append().
If I try and change that to a SARIMAX model with no seasonality, the forecasted values are completely different to what I'm getting with the ARIMA model.

I am using statsmodels 0.12.0
Any solution to either of these issues would be helpful

**ChadFulton** commented on Sep 22, 2020

- The old `sm.tsa.ARIMA` model has been deprecated and will not be getting new features, including the `append` method.
- `sm.tsa.SARIMAX` estimates parameters slightly differently, but unless the estimates are pretty unstable, you shouldn't be getting very different forecasts. However, `SARIMAX` does not include a constant term by default. If you were using a trend in `sm.tsa.ARIMA`, did you make sure to add it to the `SARIMAX` specification?
- You can alternatively use the new `sm.tsa.arima.ARIMA` model, which includes a constant term by default.

**hari-packetai** commented on Jul 23

> Only `sm.tsa.SARIMAX` models have the `filter` method, but SARIMAX includes ARIMA models. For example, for an ARIMA(1, 2, 3) model, you could do:
>
> ```
> mod = sm.tsa.SARIMAX(endog, order=(1, 2, 3))
> ```

I suppose the situation is still the same or is `filter` method now available in other models like Holt-Winters?

**ChadFulton** commented on Jul 23

If you use the `sm.tsa.ETSModel` class (which includes Holt-Winters models), then there is a `smooth` method, which acts like the `filter` method from `SARIMAX`.

1

**hari-packetai** commented on Jul 29 • edited

> If you use the `sm.tsa.ETSModel` class (which includes Holt-Winters models), then there is a `smooth` method, which acts like the `filter` method from `SARIMAX`.

For me the fitting time for the same Holt-Winters model using `statsmodels.tsa.exponential_smoothing.ets.ETSModel`, is much slower compared to `statsmodels.tsa.holtwinters.ExponentialSmoothing`.

train_mod = ETSModel(df_train['value'], seasonal_periods=seasonal_period, error='add', trend='add', seasonal='add')

train_mod = HWES(df_train, seasonal_periods=seasonal_period, trend='add', seasonal='add')

df_train is a data frame with two columns: datetime and value. Its index is based on datetime. For some reason, ETSModel expects a one dimensional object, while the HWES is happy with the dataframe.

 **rericroberts** commented on Sep 12

Hi Chad: How do we forecast values using MarkovAutoregression class if there is not a forecast() or get_forecast method?

 **ChadFulton** commented on Sep 13

Hi **@rericroberts**, unfortunately forecasting for Markov switching models has not been implemented. See e.g. #5537. Contributions adding this feature would be very welcome.

**Assignees**

No one assigned

**Labels**

comp-tsa FAQ type-enh

**Projects**

None yet

**Milestone**

No milestone

**Linked pull requests**

Successfully merging a pull request may close this issue.

None yet

11 participants