How To Backtest Machine Learning Models for Time Series Forecasting

by **Jason Brownlee** on December 19, 2016 in **Time Series**

k-fold Cross Validation Does Not Work For Time Series Data and Techniques That You Can Use Instead.

The goal of time series forecasting is to make accurate predictions about the future.

The fast and powerful methods tha ain-
test splits and k-fold cross validatic is
is because they ignore the tempora

In this tutorial, you will discover hov
series data with Python. In the field ng
or hindcasting.

After completing this tutorial, you w

- The limitations of traditional m
and why evaluating models on

- How to create train-test splits and multiple train-test splits of time series data for model evaluation in Python.
- How walk-forward validation provides the most realistic evaluation of machine learning models on time series data.

Let's get started.

- **Updated Apr/2019**: Updated the link to dataset.

How To Backtest Mach
Photo

Model Evaluation

How do we know how good a giver

We could evaluate it on the data us de
insight into how the selected mode ny
estimate of performance on this da on
this performance would be biased.

Why?

It is helpful to take it to an extreme:

A model that remembered the timestamps and value for each observation would achieve perfect performance.

All real models we prepare will report a pale version of this result.

When evaluating a model for time series forecasting, we are interested in the performance of the model on data that was not used to train it. In machine learning, we call this unseen or out of sample data.

We can do this by splitting up the data that we do have available. We use some to prepare the model and we hold back some data and ask the model to make predictions for that period. The evaluation of these predictions will provide a good proxy for how the model will perform when we use it operationally.

In applied machine learning, we often split our data into a train and a test set: the training set used to prepare the model and the test set used to evaluate it. We may even use k-fold cross validation that repeats this process by systematically splitting the data into k groups, each given a chance to be a held out model.

These methods cannot be directly used with time series data.

This is because they assume that there is no relationship between the observations, that each observation is independe

This is not true of time series data, that we cannot randomly split them respect the temporal order in which

In time series forecasting, this evalu backtesting. In some time series do hindcasting, as opposed to forecas

We will look at three different metho learning models on time series prob

1. **Train-Test split** that respect temporal order of observations.
2. **Multiple Train-Test splits** that respect temporal order of observations.
3. **Walk-Forward Validation** where a model may be updated each time step new data is received.

First, let's take a look at a small, univariate time series data we will use as context to understand these three backtesting methods: the Sunspot dataset.

Monthly Sunspot Dataset

This dataset describes a monthly count of the number of observed sunspots for just over 230 years (1749-1983).

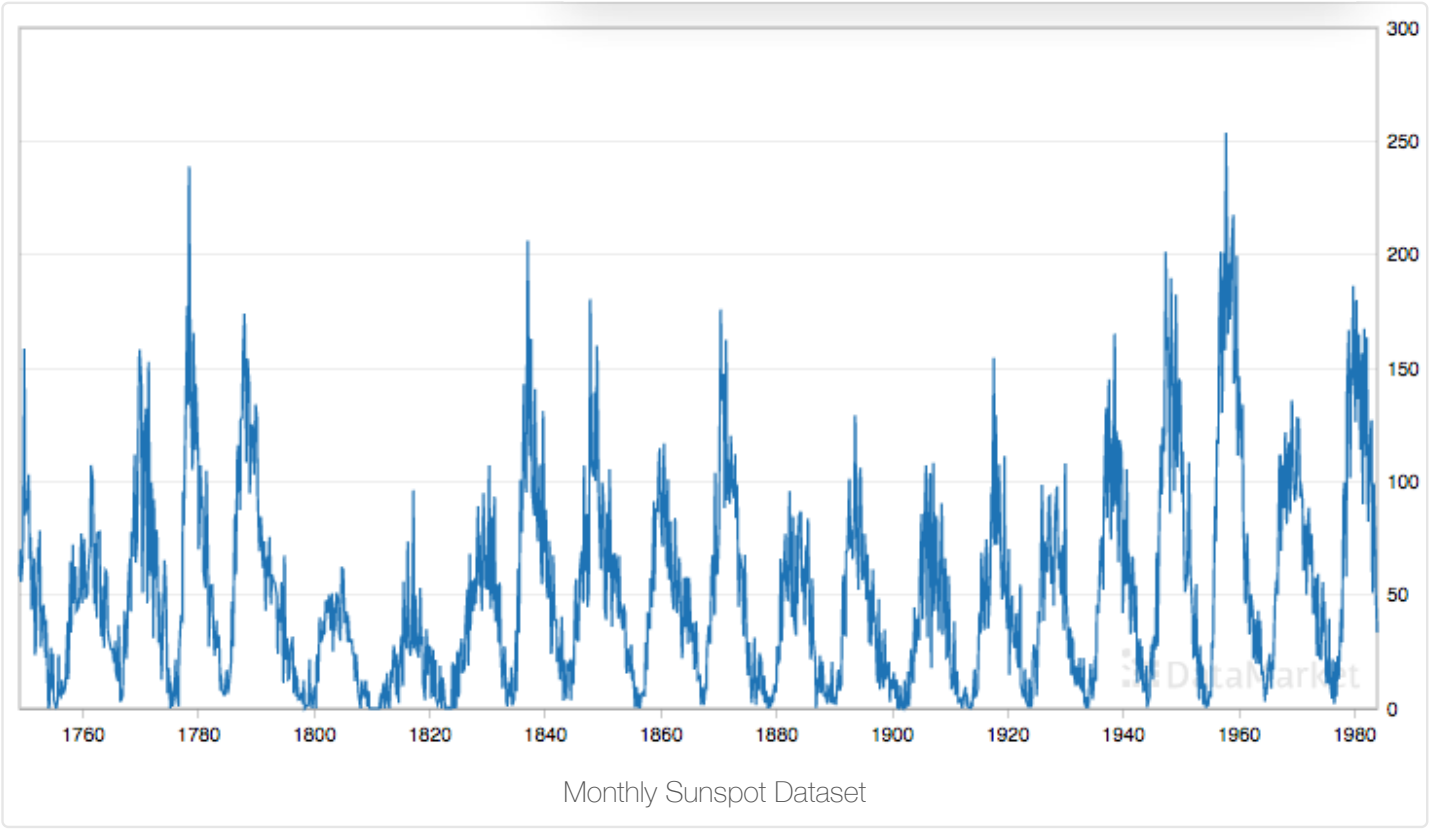
The units are a count and there are
credited as Andrews & Herzberg (1

- Download the dataset.

Below is a sample of the first 5 row

```
"Month","Sunspots"
"1749-01",58.0
"1749-02",62.6
"1749-03",70.0
"1749-04",55.7
"1749-05",85.0
```

Below is a plot of the entire dataset



Monthly Sunspot Dataset

The dataset shows seasonality with large differences between seasons.

Download the dataset and save it into your current working directory with the filename "*sunspots.csv*".

- Download the dataset.

Load Sunspot Dataset

We can load the Sunspot dataset u

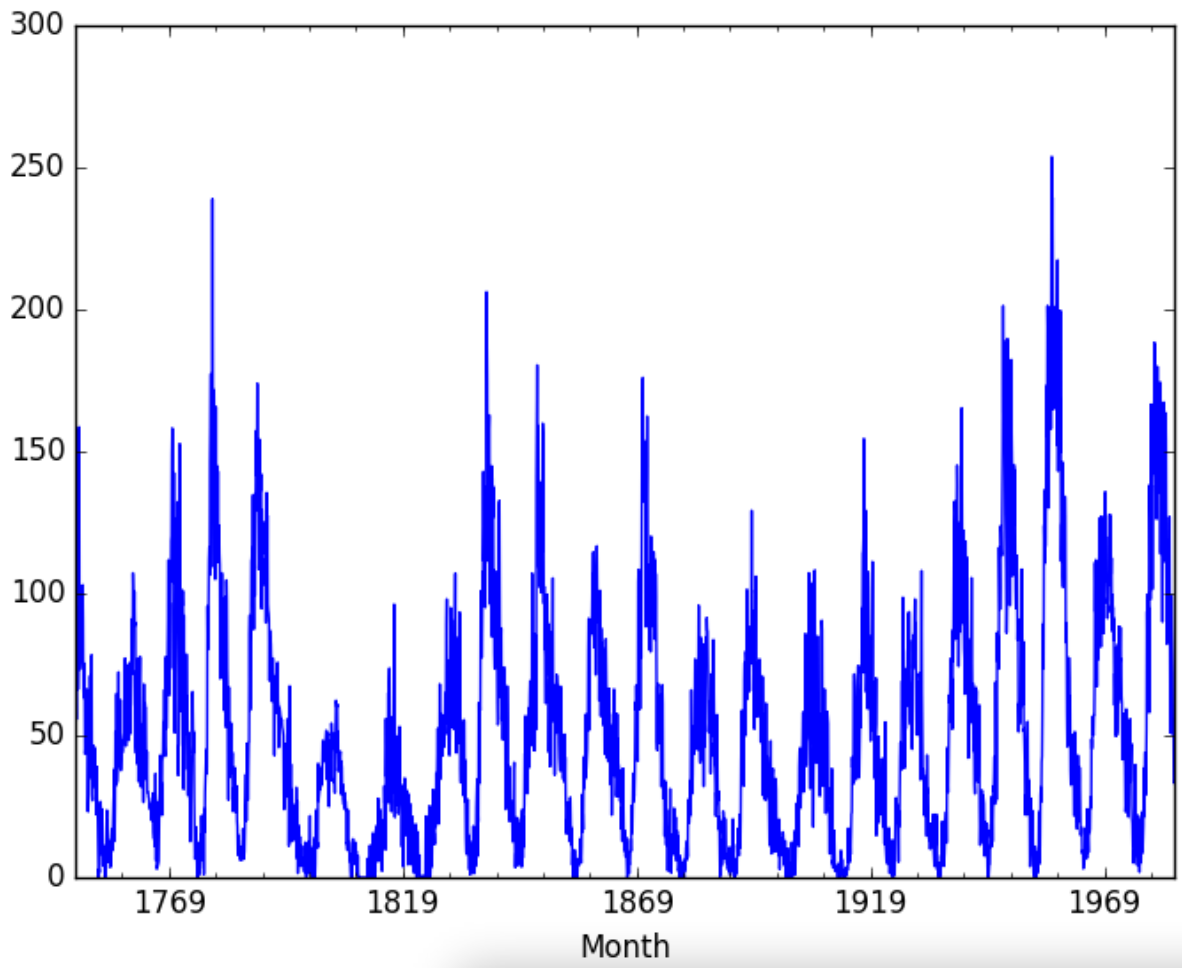
```
# Load sunspot data
from pandas import Series
from matplotlib import pyplot
series = Series.from_csv('sun
print(series.head())
series.plot()
pyplot.show()
```

Running the example prints the firs

```
Month
1749-01-01 00:00:00 58.0
1749-02-01 00:00:00 62.6
1749-03-01 00:00:00 70.0
```

```
5 1749-04-01 00:00:00 55.7
6 1749-05-01 00:00:00 85.0
7 Name: Sunspots, dtype: float64
```

The dataset is also plotted.



Train-Test Split

You can split your dataset into train

Your model can be prepared on the d evaluated for the test dataset.

This can be done by selecting an a ns and creating two new datasets. De e and the amount of data required, y

It is straightforward to split data in Python.

After loading the dataset as a Pandas Series, we can extract the NumPy array of data values. The split point can be calculated as a specific index in the array. All records up to the split point are taken as the training dataset and all records from the split point to the end of the list of observations are taken as the test set.

Below is an example of this in Python using a split of 66-34.

```
from pandas import Series
series = Series.from_csv('sunspots.csv', header=0)
X = series.values
train_size = int(len(X) * 0.66)
train, test = X[0:train_size], X[train_size:len(X)]
print('Observations: %d' % (len(X)))
print('Training Observations: %d' % (len(train)))
print('Testing Observations: %d' % (len(test)))
```

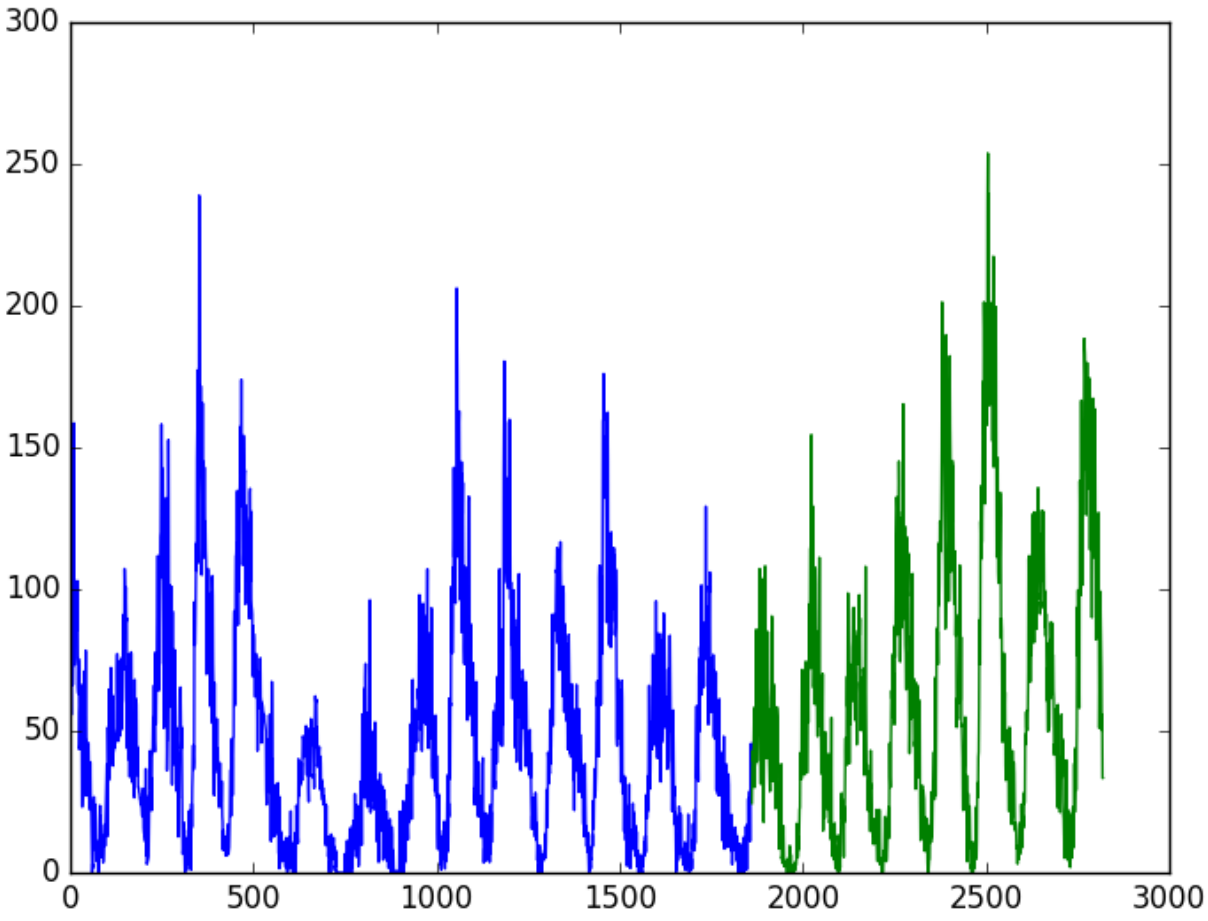
Running the example prints the size of the loaded dataset and the size of the train and test sets created from the split.

```
Observations: 2820
Training Observations: 1861
Testing Observations: 959
```

We can make this visually by plotting the training and test sets using different colors.

```
from pandas import Series
from matplotlib import pyplot
series = Series.from_csv('sunspots.csv', header=0)
X = series.values
train_size = int(len(X) * 0.
train, test = X[0:train_size
print('Observations: %d' % (
print('Training Observations
print('Testing Observations:
pyplot.plot(train)
pyplot.plot([None for i in t
pyplot.show()
```

Running the example plots the trai



Sunspot Dataset Train-Test Split

Using a train-test split method to evaluate machine learning models is fast. Preparing the data is simple and intuitive and only one model is created and evaluated.

It is useful when you have a large a are representative of the original proble

Next, we will look at repeating this

Multiple Train-Test S

We can repeat the process of splitt times.

This will require multiple models to computational expense will provide

performance of the chosen method and configuration on unseen data.

We could do this manually by repeating the process described in the previous section with different split points.

Alternately, the scikit-learn library provides this capability for us in the *TimeSeriesSplit* object.

You must specify the number of splits to create and the *TimeSeriesSplit* to return the indexes of the train and test observations for each requested split.

The total number of training and test observations are calculated each split iteration (*i*) as follows:

```
1 training_size = i * n_samples / (n_splits + 1) + n_samples % (n_splits + 1)
2 test_size = n_samples / (n_splits + 1)
```

Where *n_samples* is the total number of observations and *n_splits* is the total number of splits.

Let's make this concrete with an example. Assume we have 100 observations and we want to create 2 splits.

For the first split, the train and test sizes would be calculated as:

```
1 train = i * n_samples / (n_splits + 1) + n_samples % (n_splits + 1)
2 train = 1 * 100 / (2 + 1) + 100 % (2 + 1)
3 train = 33.3 or 33
4
5 test = n_samples / (n_splits
6 test = 100 / (2 + 1)
7 test = 33.3 or 33
```

Or the first 33 records are used for ng.

The second split is calculated as fo

```
1 train = i * n_samples / (n_sp
2 train = 2 * 100 / (2 + 1) + 1
3 train = 66.6 or 67
4
5 test = n_samples / (n_splits
6 test = 100 / (2 + 1)
7 test = 33.3 or 33
```

Or, the first 67 records are used for training and the remaining 33 records are used for testing.

You can see that the test size stays consistent. This means that performance statistics calculated on the predictions of each trained model will be consistent and can be combined and compared. It provides an apples-to-apples comparison.

What differs is the number of records used to train the model each split, offering a larger and larger history to work with. This may make an interesting aspect of the analysis of results. Alternately, this too could be controlled by holding the number of observations used to train the model consistent and only using the same number of the most recent (last) observations in the training dataset each split to train the model, 33 in this contrived example.

Let's look at how we can apply the TimeSeriesSplit on our sunspot data.

The dataset has 2,820 observations. Let's create 3 splits for the dataset. Using the same arithmetic above, we would expect the following train and test splits to be created:

- **Split 1**: 705 train, 705 test
- **Split 2**: 1,410 train, 705 test
- **Split 3**: 2,115 train, 705 test

As in the previous example, we will plot the train and test observations using separate colors. In this case, we will have 3 splits, so that will be 3 separate plots of the data.

```
1  from pandas import Series
2  from sklearn.model_selection
3  from matplotlib import pyplo
4  series = Series.from_csv('su
5  X = series.values
6  splits = TimeSeriesSplit(n_s
7  pyplot.figure(1)
8  index = 1
9  for train_index, test_index
10     train = X[train_index]
11     test = X[test_index]
12     print('Observations: %d'
13     print('Training Observat
14     print('Testing Observati
15     pyplot.subplot(310 + ind
16     pyplot.plot(train)
```

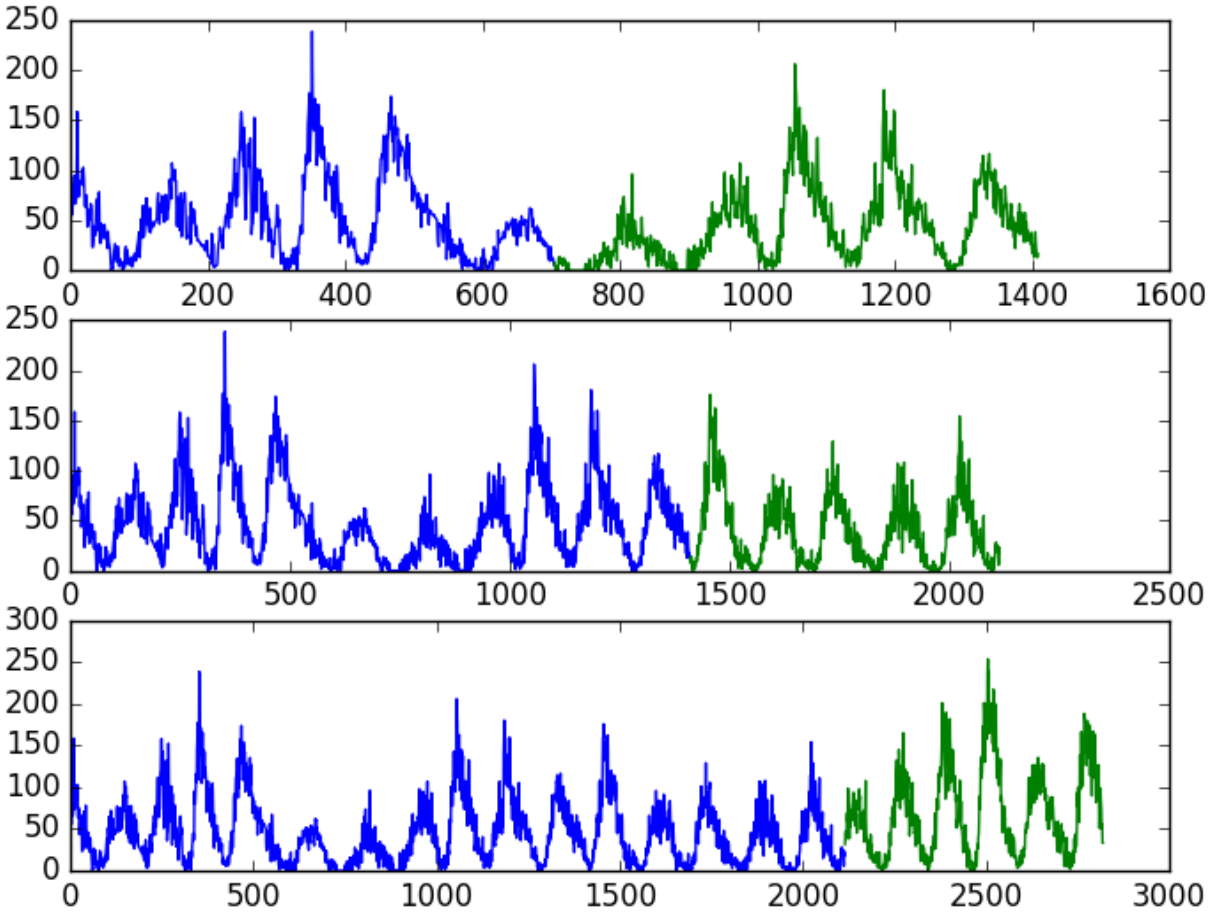
```
    pyplot.plot([None for i in train] + [x for x in test])
    index += 1
pyplot.show()
```

Running the example prints the number and size of the train and test sets for each split.

We can see the number of observations in each of the train and test sets for each split match the expectations calculated using the simple arithmetic above.

```
Observations: 1410
Training Observations: 705
Testing Observations: 705
Observations: 2115
Training Observations: 1410
Testing Observations: 705
Observations: 2820
Training Observations: 2115
Testing Observations: 705
```

The plot also shows the 3 splits and the growing number of total observations in each subsequent plot.



Sunspot Dataset Multiple Train-Test Split

Using multiple train-test splits will result in more models being trained, and in turn, a more accurate estimate of the performance of the models on unseen data.

A limitation of the train-test split ap
they are evaluated on each evaluat

This may not be realistic as models
observations are made available. T

Walk Forward Valida

In practice, we very likely will retrai

This would give the model the best
step. We can evaluate our machine

There are few decisions to make:

1. **Minimum Number of Observations**. First, we must select the minimum number of observations required to train the model. This may be thought of as the window width if a sliding window is used (see next point).
2. **Sliding or Expanding Window**. Next, we need to decide whether the model will be trained on all data it has available or only on the most recent observations. This determines whether a sliding or expanding window will be used.

After a sensible configuration is chosen for your test-setup, models can be trained and evaluated.

1. Starting at the beginning of the time series, the minimum number of samples in the window is used to train a model.
2. The model makes a prediction for the next time step.
3. The prediction is stored or evaluated against the known value.
4. The window is expanded to include the known value and the process is repeated (go to step 1.)

Because this methodology involves moving along the time series one-time step at a time, it is often called Walk Forward Testing or Walk Forward Validation. Additionally, because a sliding or expanding window is used to train a model, this method is also referred to as Rolling Window Analysis or a Rolling Forecast.

This capability is currently not available in scikit-learn, although you could contrive the same effect with a carefully configured TimeSeriesSplit.

Below is an example of how to spli ard Validation method.

```
1 from pandas import Series
2 from matplotlib import pyplot
3 series = Series.from_csv('sun
4 X = series.values
5 n_train = 500
6 n_records = len(X)
7 for i in range(n_train, n_rec
8     train, test = X[0:i], X[i
9     print('train=%d, test=%d'
```

Running the example simply prints the size of the training and test sets created. We can see the train set expanding teach time step and the test set fixed at one time step ahead.

Within the loop is where you would train and evaluate your model.

```
train=500, test=1
train=501, test=1
train=502, test=1
train=503, test=1
train=504, test=1
...
train=2815, test=1
train=2816, test=1
train=2817, test=1
train=2818, test=1
train=2819, test=1
```

You can see that many more models are created.

This has the benefit again of providing a much more robust estimation of how the chosen modeling method and parameters will perform in practice. This improved estimate comes at the computational cost of creating so many models.

This is not expensive if the modeling method is simple or dataset is small (as in this example), but could be an issue at scale. In the above case, 2,820 models would be created and evaluated.

As such, careful attention needs to be paid to the window width and window type. These could be adjusted to contriv ntly less computationally expensive.

Walk-forward validation is the gold s validation of the time series world a

Further Reading

- sklearn.model_selection.TimeS
- Rolling-Window Analysis of Tir
- Backtesting on Wikipedia to le

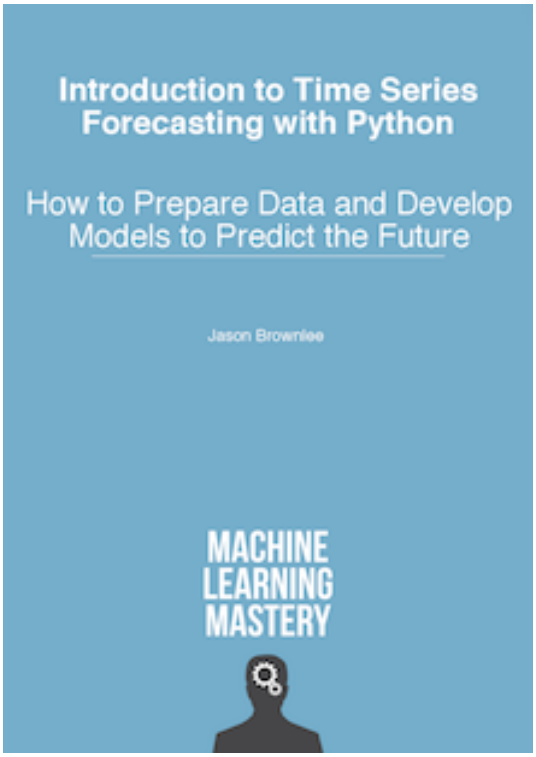
Summary

In this tutorial, you discovered how to backtest machine learning models on time series data with Python.

Specifically, you learned:

- About the importance of evaluating the performance of models on unseen or out-of-sample data.
- How to create train-test splits of time series data, and how to create multiple such splits automatically.
- How to use walk-forward validation to provide the most realistic test harness for evaluating your models.

Do you have any questions about evaluating your time series model or about this tutorial?
Ask your questions in the comments below and I will do my best to answer.



Tweet Share Share

About Jason Brownlee

Jason Brownlee, PhD is a machine learning specialist who teaches developers how to get results with modern machine learning methods via hands-on tutorials.

View all posts by Jason Brownlee →

‹ How To Resample and Interpolate Your T

How to Use ar

hon ›

156 Responses to *How To* *for Time Series Forecasting*

Michael December 19, 2016 at 7:57 pm #

REPLY

Jason,

second link from “Further Reading” should probably point to mathworks.com instead of amathworks.com, which is not found

Jason Brownlee December 20, 2016 at 7:31 am #

REPLY

Thanks Michael, fixed!

SalemAmeen December 19, 2016 at 9:32 pm #

REPLY

Many thanks, it is short and full of information.

Jason Brownlee December 20, 2016 at 7:30 am #

REPLY

I’m glad to hear you found it useful.

Shreyak Tiwari December

For walking forward valida
each single interation and even res
Better way would be to increase h
in that manner. Train data could be
steps for each iteration rather than
point of view. No hard rules here.

Jason Brownlee De

Hi Shreyak,

Yes, that would be a sampled version of walk-forward validation, a subset.

This is pretty much what the multiple train-test splits provides in the sklearn TimeSeriesSplit object – if I understood you correctly.

Kingsley Udeh January 27, 2019 at 4:28 am #

REPLY

Hi Dr. Jason,

Based on Shreyak's idea of simplifying WFV, it requires you to have a common h-steps factor for total observations and minimum train samples , respectively, evenly dividing both quantities. Therefore expanding the window width by the common factor, and at same time, keeping h steps for prediction consistent at each iteration.

However, the limitation with this approach is if the common factor is 1 or even 2, or 3, and you have a large train samples, it would still default back to the traditional WFV, which is time consuming.This is where TimeSeriesSplit object comes in – that is, having a reasonable set of splits that is relevant even with large train samples.

In conclusion, it's possible to combine multiple train-test splits method and WFV technique by expanding each train split and retraining the model, while maintaining a consistent test split.

Dr. Jason, what's your take

Jason Brow

My take is tha

stick to simple walk fo

Kingsley

Sounds good, as long as whatever approach one uses worked for their problems.

Thank you!

Saurabh Bhagvatula December 27, 2016 at 5:18 am #

REPLY

My query is related to walk forward validation:

Suppose a time series forecasting model is trained with a set of data and gives a good evaluation with test-set in time_range-1 and model produces a function F1. For time_range-2 and another set of training and testing data model generates function F2. Similarly for time_range-N the model generate Function FN. How the different models when combined and implemented forecast the result based on forecasting function based of local model and not the combined model of all time range model, which may possibly be producing error in forecasting.

Jason Brownlee December 27, 2016 at 5:26 am #

REPLY

Hi Saurabh,

Sorry, I don't quite understand the last part of your question. Are you able to restate it?

Ram January 20, 2017 at 12:1

I am just going through yo r resource as a reference material fo

Jason Brownlee Jar

A shelf of textbooks r

Ian February 11, 2017 at 10:01 am #

REPLY

Hi Jason

Thanks so much for this in-depth post. My question is:
Which performance measure should we use in selecting the model?
For example, if I add one test subset at a time in a binary(1, 0) classification problem, the accuracy would be either 1 or 0.
In this case, how should I select a model? Should I use other measures instead?

I am building my model as stock price classification where 1 represents up, and 0 means down. I use TimeSeriesSplit and divide into T (sample size) – m (rolling window) + 1.

Thanks a lot and I look forward to listening your insights!

Jason Brownlee February 12, 2017 at 5:33 am #

REPLY

Hi Ian,

This is a problem specific question.

Perhaps classification accuracy on the out of sample dataset would be a good way to pick a model in your case?

Ian February 12, 201

Jason,

Thanks so much for answe
If we walk one step forwar
Forward Validation, doesn'
sample?

Hope this is not too proble

Jason Brownlee February 13, 2017 at 9:11 am #

REPLY

Hi Ian,

Walk forward validation is a method for estimating the skill of the model on out of sample data. We contrive out of sample and each time step one out of sample observation becomes in-sample.

We can use the same model in ops, as long as the walk-forward is performed each time a new observation is received.

Does that make sense?

Magnus March 4, 2017 at 2:39 am #

REPLY

Thanks Jason for an informative post!
If the time series is very long, e.g. minute values for 10 years, it will take a very long time to train. As I understand you, another way to do this is to fix the length of the training set, e.g. 2 years, but just move it, like this:
Split 1: year 1+2 train, year 3 test
Split 2: year 2+3 train, year 4 test
…
Split 8: year 8+9 train, year 10 test
Is this correct and valid?

Jason Brownlee Ma

Sounds good to me.

Also consider how valuable th
10 years ago is not predictive

jessie August 9, 20

well how is this sp

jessie August 9, 2018 at 6:35 pm #

REPLY

well ? how will it do ? what is the it's code?

Jason Brownlee August 10, 2018 at 6:11 am #

REPLY

Splitting a dataset is really specific to the dataset.

If you need help getting started with numpy arrays, perhaps start here: https://machinelearningmastery.com/index-slice-reshape-numpy-arrays-machine-learning-python/

jessiee August 10, 2018 at 8:28 pm #

REPLY

well which code should we use for this splits?

Jason Brownlee August 11, 2018 at 6:09 am #

REPLY

Sorry, I cannot write code for you, what problem/error are you having exactly?

marwa May 11, 2017 at 12:14

Thank you for your post J

I would like to ask you which mode
In fact, for example if the time seri
applied in this way:

Split 1: year 1 train, year 2 test and
Split 2: year 1+2 train, year 3 test a

which model should we then choose ?

Jason Brownlee May 11, 2017 at 8:32 am #

REPLY

Great question.

Pick the model that best represents the performance/capability required for your application.

Elie Kawerk June 24, 2017 at 8:14 pm #

REPLY

Jason,

I think that when Marwa mentioned ‘models’, she meant applying the same model (such as ARMA) on different data (corresponding to the expanding window).
I think that the walk-forward method, just like k-fold CV, gives an array of metrics whose mean somehow corresponds to the true skill of the model.

I think that when this mean is evaluated, the model should be trained on the entire dataset (check Practical Time Series Forecasting with R- Shmueli) just like with K-fold CV.

Please correct me if I am wrong.

Regards

Jason Brow

Walk forward
model.

Walk forward validatio
model and some to ev
be made available to t

entire training dataset, if we did and made a prediction, what would we compare the prediction to in order to estimate the skill of the model?

Shifen June 29, 2017 at 5:46 pm #

REPLY

Dear Jason,
Thanks so much for this in-depth post. My question is:
If my time series are discontinuous(such as two weeks in March and two weeks in September), How should I divide the data set?
If I use time series as supervised learning, it could lead to a sample containing data for March and September.
This question has puzzled me for a long time and I look forward to hearing from you.

Jason Brownlee June 30, 2017 at 8:10 am #

REPLY

I don't have a good answer.

Perhaps try to fill in the missing time with 0s or nans.
Perhaps try to ignore the missing blocks.
Perhaps focus on building a model at a lower scale (month-wise).

Nick July 7, 2017 at 12:57 pm

Hey Jason, can you comm
fact, be used for time-series data (

Nick July 7, 2017 at 1:05

As a follow-up, would
http://machinelearningmastery
python/

Thank you in advance for your

Jason Brownlee July 9, 2017 at 10:35 am #

REPLY

Try it and see.

Jason Brownlee July 9, 2017 at 10:35 am #

REPLY

I hope to when I have a pocket of time.

Daniel July 14, 2017 at 7:54 am #

REPLY

Is there a way to store the model fit values in such a way that we can update the model after every iteration instead of recreate an entirely new one?
My dataset has 55,000 samples and I want to run a test set of 5,000, but recreating 5,000 models would take roughly 80 hours. Thanks.

Jason Brownlee July 14, 2017 at 8:38 am #

REPLY

Yes, here's how to save the model:
http://machinelearningmastery
python/

Daniel July 14, 201

Thanks for respor
on a test sample x1 and th
load the old model, how w
making it perform better?
That way I am always prec
always creating a new mo

Jason Brownlee July 15, 2017 at 9:34 am #

REPLY

You would have to fit the model on just the new data or on a combination of the new and old data.

This can be done with a new model or by updating the existing model.

I do not have an example with ARIMA, but I do have examples with LSTMs here:
http://machinelearningmastery.com/update-lstm-networks-training-time-series-forecasting/

Huzefa Barwaniwala October 4, 2017 at 3:51 am #

REPLY

Hi, Jason

Thanks a lot for this post, I have recently gone through many for your blog post on time series forecasting and found it quite informative; especially the post on feature engineering for time series so it can be tackled with supervised learning algorithms.

Now, if I have a time series data for demand forecasting, and I have used a lot of feature engineering on the 'date' variable to extract all the seasonality, for example, day, month, the day of the week, if that day was a holiday, quarter, season, etc. I have also used some FE on the target variable to create lag features, min, max, range, average, etc.

My question to you is: Do I still nee
now use a simple k-fold cross vali
important?

Thanks a lot. Keep doing this awes

Huzefa

Jason Brownlee Oct

It really depends on y

In practice, I do recommend walk-forward validation when working with time series data. It is a great way to make sure you are not tricking yourself.

Huzefa Barwaniwala October 5, 2017 at 3:22 am #

REPLY

Jason,

Thank you for getting back. Yes, I agree with you. One more thing I realized is, I have made lags as a feature and if in any of the fold of CV a future data is used to predict past then it will act as a target leakage!

Best,
Huzefa

Jason Brownlee October 5, 2017 at 5:28 am #

REPLY

Absolutely!

Danilo November 4, 2017 at 6:14 am #

REPLY

Hi Jason

Your posts are really amazing. I ha
appreciate if you can help me with
series to supervised learning.

May I used backtest, to identify th
learning ?

Jason Brownlee No

Sure.

annesoj November 4, 2017 at 7:49 pm #

REPLY

Hi Jason,

Thank you so much for this post.
However I will have a question that might seems stupid but…

This give me a graphical version of the reality (on the train) and of my predictions (on the test). But it is not an evaluation of my model….

How do I know using those methods, if my models is great or bad?

Imagine I want to try an ARIMA (5,2) and an ARIMA (6,3). How do I do to pick the best one? How do I evaluate each one using “Walk Forward Validation”????

To evaluate the first model, I can do the mean of the error, for each split, between the prediction and the real value?

To pick the best model I can compare those mean between the 2 models?

Would it be a good evaluation methods?

Thank you again!

Jason Brownlee November 5, 2017 at 5:15 am #

REPLY

You can compare the predictions to the expected values and calculate an error score.

Here are some examples:
https://machinelearningmaster
measures-with-python/

Urfa January 22, 2018 at 10:38

Hi Jason,

I have a set of monthly panel data
values. In detail, I want to predict
regression with a fixed window siz

panel data like regression with fixed effects, but in my case, with pooled OLS I'm getting accurate predictions.) Regression model looks like this: y_{i,t+1}= b0+ b1*x_{i,t} + b2*x2_{i,t} +… + x10_{i,t} where t is the current month and i is the id.

Furthermore, I select a new model in every step by using a dynamic model selection. In detail:

1. Take a fixed windows size of five years and split it into a training and validation set. The first 58 months as training and the month 59 as validation set.

2. Choose Explanatory Variables or rather a regression model by running a stepwise regression for model selection with the training and validation set and the Average Square Error of the validation set as a criterion.

3. Take the data from month 60 and the regression model from step 2, to make a forecast for month 61.

4. Go to step 1 and roll the window one month forward.

I couldn't find any literature where you select a new regression model or new explanatory variables at every step of the rolling regression. Do you know if there is any literature on that?

Thank you!

Jason Brownlee January 23, 2018 at 7:48 am #

REPLY

Nice!

Good question. Can't think of
sniffing around applied stats b

Urfa January 23, 20

Thank you!

Until now I can't find anyth
and books on that topic. I

By the way, does this appr

Jason Brownlee January 24, 2018 at 9:47 am #

REPLY

Hang in there.

Generally, there is no one size fits all approach. Often you need to dive in and try stuff and see what suits the problem/data/project.

Mohammed Helal February 6, 2018 at 5:57 am #

REPLY

Correct me if I'm wrong, but it seems to me that TimeSeriesSplit is very similar to the Forward Validation technique, with the exceptions that (1) there is no option for minimum sample size (or a sliding window necessarily), and (2) the predictions are done for a larger horizon.

PS. Thanks a lot for your posts!

Jason Brownlee February 6, 2018 at 9:23 am #

REPLY

It is a one-time split, where as walk-forward validation splits on each time step from one point until the end of the dataset.

Does that help?

Alexis March 2, 2018 at 5:14 p

Hi Jason, I don't see why
to create a test set of constant size
window at the first iteration, and th
be correct ?

Jason Brownlee Ma

Yes, nice one. You have essentially described a variation on walk forward validation.

Tarun March 15, 2018 at 10:31 pm #

REPLY

Hi Jason,

I have a query regarding Walk forward validation of TS. Let's say I need to forecast for next 3 months (Jan-Mar 18) using last 5 years of data (Jan13-Dec 17).
In principle I would want to use Walk forward as I would like to see how well the model generalizes to unseen data. I'd use your approach which is:

1) Set Min no of observations : Jan12-Dec 16
2) Expanding TEST window : Entire 2017, which means I would forecast next 3 points (Jan-Mar 17) in iteration 1 and in next iteration, Jan 17 becomes part of train and I predict for Feb-mar-April 17.I do it for entire 2017.

My question is why do I need to RETRAIN model everytime I add 1 data point? Why can't I just score the next 3 TEST points assuming the fact that model that I have trained before ITR1 is the best one?

Can't I select (let's say) top 5 models from step 1,calculate their Average for all TEST samples (3 months window) and select the one with least RMSE?.

Eagerly awaiting your reply!

Jason Brownlee Ma

You can, but the furth
skill will be come.

This post will give you some a
without using obs as inputs:
https://machinelearningmaster

Tarun March 16, 20

Hi Jason,

Thanks for the reply.

“the further you get from real obs” by that do you mean to say that I am not retraining my model using real data?

Jason Brownlee March 17, 2018 at 8:37 am #

REPLY

I mean that the longer the lead time – further you forecast into the future, the less stable/skillful the results.

Tarun March 19, 2018 at 4:44 pm #

Thanks Jason. You are indeed doing a great job.

Jason Brownlee March 20, 2018 at 6:11 am #

Thanks.

Ha Pham April 17, 2018 at 1

Hi Jason,

Thanks a lot for your post. I am wo
thousands of products, and I only
be days, but for now I aggregate ir
zero and some bursty sales weeks
years. I tried two approaches:
– Using sales data of previous 4 w
– Using sales data of year 1 to pre
to the model

My questions:
– Is there any theoretical error in these approaches? I can clarify a few things more if you need
– In this post you only talk about one time series. Can this be applied to my case where I have thousands of time series needed to be forecast at the same time?
– For this kind of problem, which algorithm tend to give best result? Can an out-of-the-box algo like XGBoost do the job? I have browsed through some papers and they introduced different methods like Neural Networks or Bayesian methods, which haven't touched yet.

Thanks.

Jason Brownlee April 17, 2018 at 2:54 pm #

REPLY

That sounds like a great problem.

I'm eager to dive in and offer some advice, but it would be a big time investment for me, I just don't have the capacity. I hope to have more posts on this topic soon.

In general, I recommend testing a suite of modeling methods and framings of a problem to help you discover what works best for your specific dataset.

I'm eager to hear how you go.

Dicky Chou April 23, 2018 a

REPLY

Hi Jason
I am a meteorologist currently wor
My colleagues make forecasts eve
I find that there are some time shif
example, we think it will be raining
6. If we use normal verification me
false alarm. However, I think this e
we the weather condition at 4 and
attribution of these data. Can you
kind of times series data?

Jason Brownlee April 24, 2018 at 6:25 am #

REPLY

Great question.

I believe there is a huge body of literature on weather forecasting verification (I used to work in this area).

Here is a great place to start:
http://www.cawcr.gov.au/projects/verification/

Deniz May 16, 2018 at 10:09 am #

REPLY

Hi Jason,

Is using AIC for forecasting a good method? Or should I use cross-validation while building forecasting models?

Jason Brownlee May 17, 2018 at 6:21 am #

REPLY

It really depends on your project goals and on your specific data.

Mustafa Qamar-ud-Din

REPLY

Thank you for informative
you could please correct me whetl
thing. In the sense that you move t

Jason Brownlee Jun

Sliding window refers
learning problem.

Backtesting is a general term f

Anthony The Koala June 20, 2018 at 10:38 am #

REPLY

Dear Dr Jason,
For those having difficulty plotting the data sourced from the site https://datamarket.com/data/set/22ti/zuerich-monthly-sunspot-numbers-1749-1983 ,
The following may be helpful before even using Python.

Even if you imported the file from the website as a CSV file, the trouble is that there are NaN values and extraneous information at the bottom of the spreadsheet. It requires cleaning the file. Otherwise if the file is not cleaned, Python will produce error messages.
.
(1) Open the sunspot.csv file into a spreadsheet program eg MS Excel
(2) Leave the header at the top of the file alone.
(3) Scroll down to the very end of the data file (2821 rows down). Delete the rows containing Nan and text “Zuerich monthly sunspot numbers 1749-1983”.
(4) Save the file as sunspot.csv in CSV format
(3) In Python import the data as usual

```
1  Series.read_csv('sunspot.csv', header=0)
```

Everything should be OK from that point.

Thank you,
Anthony of Sydney

Jason Brownlee Jur

I also provode all data
https://github.com/jbrownlee/D

Brian March 27, 20

I wonder why the
http://www.sidc.be/silso/d

vary so much from
https://github.com/jbrownlee/Datasets/blob/master/monthly-sunspots.csv

Jason Brownlee March 27, 2019 at 2:06 pm #

REPLY

Good question, I'm not sure exactly.

My source was the CSV file on "data market", linked in the post.

Gautam July 2, 2018 at 1:22 am #

REPLY

Hello Jason,
You have become a one stop website for machine learning. Thank you for all the efforts!
I am little stuck and validate my approach here, if you can:
I am trying to predict a stock market index using multiple time series: ex say many commodity indexes besides the targeted index itself. Is this approach terribly wrong? If not can you please possible point to good start point. I am really stuck here badly.
Appreciate your thoughts

Jason Brownlee July 2, 2018 at 6:25 am #

REPLY

Thanks.

This is a common question tha
https://machinelearningmaster
machine-learning-for-finance-

Gautam July 2, 2018 at 1:25

Just additional comment
a multi time series problem using s
Elastic Net

Kingsley Udeh July 12, 2018 at 7:39 pm #

REPLY

Hi Jason,

Thanks as always.

Please how do I train and evaluate my model within the loop of a Walk Forward Validation approach?

Within the Walk Forward Validation, after choosing my min training size, I created, say for, eg.

range of train to len(records):
train, test = X_obs[0:i], X_obs[i:i+1]
# Fit Model
history = model.fit(train_X, train_y, epochs=1000, batch_size=4192, validation_data=(test_X, test_y), verbose=0, shuffle=False)
# Evaluate Model
loss = model.evaluate(test_X, test_y, verbose=0)
print(loss)
model.save('lstm_model.h5')

At the end, I have 10 different loss or validation scores. Is the last saved model the average of all the 10 models? How do I make predictions and calculate the RMSE for the average model?

I'm still learning the Walk Forward Validation method and will appreciate your help in guiding me on the right thing to do

I look forward to hearing from you

Jason Brownlee Jul

I recommend not usin
score is calcualted based on t
validation loop.

Kingsley Udeh August 3, 2018 at 7:15 am #

REPLY

I used validation set because I wanted to monitor the validation loss value with modelcheckpoint. Thus, I would pick the best model and see how it would perform on a new or independent test set.

In addition, I would use the method or approach for the the hyperparamenter tunings to fit a final model and compare the final model with the model from modelcheckpoint.

Abdessamad July 30, 2018 at 7:42 pm #

REPLY

Hi Jason,

Thanks a lot for your post. You said in the Walk Forward Validation section that “”In the above case, 2,820 models would be created and evaluated.”” Is it not 2,320 since we use the 500 first observations as the minimum ?

Thanks.

Jason Brownlee July 31, 2018 at 6:00 am #

REPLY

Yes. Nice catch.

Prince Grover August 10,

Hi Jason,

Thanks for the article. I like the wa
the same approach in one of the p
discuss with you.

Q: How can we make train, validat
approach? We generally split data
evaluation. If we are keeping a win

use to either tune hyperparameters or final validation score. What about test score and generalizability of our model?

Thanks in advance!

Jason Brownlee August 10, 2018 at 2:16 pm #

REPLY

Good question.

Perhaps choose a period over which the model will make predictions, not be updated with true values and the holdout set can be used as validation for tuning the model?

Philip P August 18, 2018 at 12:55 am #

REPLY

Jason,

So, I'm wondering how these folds from Walk Forward Validation would be passed into a python pipeline or as a CV object into a sklearn model like xgboost. I've used GridSearchCV to create the cross-validation folds before. My project at work has sales data for a number of stores each week. I'm creating a model that will predict sales 4 weeks out by each store. Right now, I have a max of 80 weeks of data. I figured to start with a minimum train size of 52 weeks and test on the next 4 weeks. Each fold would jump 4 weeks ahead. Here, n_train = 52 and max_week = 80. My code and output are below. Thanks so much!

Code:

```
for i in range(n_train, max_week):
if i % 4 == 0:
train, test = df[(df.WeekCount_ID >
4)]
print('train=%d, test=%d' % (len(tr
```

Output:

```
train=3155, test=260
train=3415, test=260
train=3675, test=260
train=3935, test=260
```

train=4195, test=272
train=4467, test=282
train=4749, test=287

Jason Brownlee August 18, 2018 at 5:40 am #

REPLY

Good question.

I write my own validation and grid search procedures for time series, it's also my general recommendation in order to give more control. The sklearn tools are not suited to time series data.

Philip P. August 21, 2018 at 12:56 am #

REPLY

Jason, thanks for the quick reply. So, for someone who is learning all of this concurrently (machine learning, time series, python, sql, etc) and not sure how to write my own python procedures, is this custom code of yours something that you cover in any of your books? If not, is this something that you would share or that I could find posted on another forum? Thanks again.

Jason Brownlee August 21, 2018 at 6:18 am #

REPLY

I give an exan
time series example I p

I have a new book con

Dimos September 20, 2018 a

Hi Jason,

Amazing tutorial!!!!!!!!

Let's assume that i have training data for periods 1-100 and i want to make predictions for periods 101-120. Should i predict the target variable for period 101 and then as an input dataset predict the period 102 etc?

Many thanks

Jason Brownlee September 21, 2018 at 6:24 am #

REPLY

Yes, you can use the real observation is available or use the prediction as the input. The latter is called a recursive model:
https://machinelearningmastery.com/multi-step-time-series-forecasting/

Tianyu September 24, 2018 at 8:51 pm #

REPLY

Hi Jason,
May I ask two questions?
1. How to apply early stopping in walk forward validation to select the model in each walk forward step?
2. I think for time series data, we can Convert a Time Series to a Supervised Learning Problem. As the result, each sample is consist of past time step data as input and one target output. Every sample is now independent and there is no time order existed when using stateless LSTM for training. We can now shuffle all the samples and split the data as training and validation set as normal. Correct me if I am wrong.

Jason Brownlee Sep

You can do early stop
prediction in each step of the v

Perhaps. It might depend on th

Tianyu September

Thanks for your reply.
If the model is to predict classification problem. The accuracy for each step will only be 0 or 1, which cannot be used for validation based early stopping.

Jason Brownlee September 25, 2018 at 2:45 pm #

REPLY

Why not?

Tianyu September 26, 2018 at 6:18 am #

Do you mean we can make it like if for 10 epochs' accuracy is 1 then stop training? But in this situation how to compare two models in two epochs with same accuracy=1? I mean if there are many samples for validation, I can save the best model with highest val_acc by check point function from Keras.

Jason Brownlee September 26, 2018 at 2:21 pm #

Not sure I follow.

Early stopping with time series is hard, but I think it is possible (happy to be proven wron
what samples are

I don't have a worl

Venkata phanikrishna

Hi Jason,
I am new to the ML. I understood I
case, really it is very hard for me. T
work. Thanks for your help.

Coming to my question,
how to use ML binary classification concepts in case of nonstationary data (Example: EEG data)?

At present, with the help of available samples, I train the model using KV fold cross-validation.

clf=ML-Classifcationmodel();
y_pred = cross_val_predict(clf,MyX,MyY,cv=10)

every time I am getting the same results.

but if I shuffle the samples before training using below syntax, every time I am getting different results.
from sklearn.utils import shuffle
mydataset = shuffle(df1)

how to find the best model in such cases.

Jason Brownlee October 1, 2018 at 6:25 am #

REPLY

It's not valid to use cross validation for time series data, regression or classification.

The train/test data must be split in such a way as to respect the temporal ordering and the model is never trained on data from the future and only tested on data from the future.

Will November 13, 2

There has been a
that if your problem is a pu
framing of an ML problem
valid to use K-Fold cross v
produced by the model are

The paper can be found he
https://pdfs.semanticscho
c480.pdf

Jason Brownlee November 13, 2018 at 7:24 am #

REPLY

Nice find, thanks. I'll have to give it a read.

Yue Lee October 25, 2018 at 2:15 am #

REPLY

In this post, it is explained that a Time Series problem could be reframed as a machine learning one with inputs and outputs. Could we consider in this case that each row is an independent observation and use Cross Validation , Nested Cross validation or any method for hyperparameters tuning and validation?

Jason Brownlee October 25, 2018 at 8:03 am #

REPLY

Nearly. The problem is that rows that contain information about the future in the training set will bias the model.

andre December 3, 2018 at 2:35 am #

REPLY

Dear Jason, thanks for yo
always happy to see machinelearn

One question:

I want to know what window-size i

Imagine your dataset has 2000 rov
model. That means i get 1500 RM

Later in that loop my windows size
fitting and i only get 800 RMSE res

Is it fair to compare the performan

Or would it be better to ensure a s

I have much more data available – but i want to use as less as possible to get high performance.

Regards from berlin 😉

Jason Brownlee December 3, 2018 at 6:53 am #

REPLY

Excellent question.

I recommend testing a suite of window sizes in order to discover the effect it has on your model for your specific dataset.

Bigger is not always better.

Andre December 3, 2018 at 12:13 pm #

REPLY

Sorry for my reply but i think i didn't get the point ;).

A better example: A data set with 10000 rows, we want to know which window size performs best. For fast execution we only use the last N Values to run some tests…

TEST A) First we use the last 2000 data points to test different window sizes (200,300,400). We start our first run with win-size 200, we train on 1:200 and check performance on 201:201+horizon. We collect RMSE values for each "fold & horizon" and go further, step by step (+1) until we reach index >20004000 The smaller the window size (200), the more data i
have less data to test.

Do i have to use a "fixed" test area
thumb here?

That's really confusing me, many t

Jason Brownlee Dec

The model evaluation
want to vary) should be perfor

Andre December 4, 2018 at 12:20 am #

REPLY

I really love you jason 😉

A solution would be to start my test area always at i.e. index 1000 and to expand the fitting window into the past.

1….[800:999]:wns200 >> test from 1000 to 2000
1….[600:999]:wns400 >> test from 1000 to 2000

What do you think about it ?

You're the best, many thanks again…

Jason Brownlee December 4, 2018 at 6:06 am #

REPLY

Sure, sounds good.

Eriz December 16, 2018 at 9:29 am #

REPLY

Hi Jason,

I'm very thankful for all your posts; I learn a lot with all of them.

Regarding the last option (i.e. the V
the window. I have come up with t

sliceListX = [arr[i: i + window] for i

Being "window" the integer value.
I'm unable to insert another param
to move the window not only one s
question on SO because I think tha

https://stackoverflow.com/questio
adding-window-rolling-quantity-as

Any idea on how to achieve it? (no

Thanks in advance,

Jason Brownlee December 17, 2018 at 6:19 am #

REPLY

Sorry, I cannot give good off the cuff advice.

Eriz Zárate December 18, 2018 at 1:37 pm #

REPLY

Finally got a working example. You can check it in the above link both the anchored version and the non anchored version. Hope it helps!

Jason Brownlee December 18, 2018 at 2:35 pm #

REPLY

Well done!

Jamie December 18, 2018 at 5:09 am #

REPLY

Amazing article! Helped me a lot!

I would like to ask one question, though. If I use the shuffled splitting function from sklearn, my model is strongly biase

Can you explain how this is prever
test set after that point in time? I d

Many thanks!

Jason Brownlee De

I recommend using w

mk123qwe March 3, 2019 at 11:31 pm #

REPLY

Train-Test split that respect temporal order of observations.
and then Train shuffle only in Train dataset，eg：np.random.shuffle(Train)

Test shuffle only in Test dataset，eg：np.random.shuffle(Test)
Does it work?

Jason Brownlee March 4, 2019 at 7:00 am #

REPLY

Sounds reasonable.

Ros March 4, 2019 at 8:32 am #

REPLY

Hello Jason, thank you for great article.

I am not sure I understand the concept of walk forward validation entirely 🙂

For example, I have a time series dataset with 3000 rows. I want to do walk forward validation. I start from the row # 500 and go ahead. Finally, I will have 2500 models created with correspondent errors.

Which one model form these 2500 should I use than for future forecasting?

Thank you!

Jason Brownlee Ma

You may or may not c

You are evaluating the “model
specific model.

Once you pick a strategy (e.g.
data and start making predicti
https://machinelearningmaster

Jurij Simoncic March 19, 2019 at 7:21 am #

REPLY

Hi Jason,
I liked the explanation and the alternatives that are offered, but I'm curious about one thing. How would you implement a cross-validation approach in time-series data where the previous periods' data are used to predict the future (for instance stock market prices)?

Jason Brownlee March 19, 2019 at 9:05 am #

REPLY

Walk forward validation will support whatever framing of the problem you require.

Simran March 20, 2019 at 9:58 pm #

REPLY

Hi, great post. I just have one doubt. Should we split our time-series data into train and test samples and then do the required normalization. Or should we normalize our series first and then do the split?

Jason Brownlee Ma

It really depends on th

I have to design a test framew
the model.

Just like using all data to fit a r
prepare any scaling, this appli

Carlos G April 16, 2019 at 6:

Thank you Jason for your work.

When you work with a neural network with a sliding window, you make a new training at each step. Would it make sense to start the new step with the neural networks weights obtained in the previous one as big part of the training samples can be the same?
If so, how would you do it in Python?
Best Regards.

Jason Brownlee April 17, 2019 at 6:57 am #

REPLY

You can choose to refit for each window or not. I typically do not.

I have many examples, you can start here:
https://machinelearningmastery.com/start-here/#deep_learning_time_series

Christian Pappenberger April 18, 2019 at 11:29 pm #

REPLY

Hej Jason,

thank you for this well-written & well-explained tutorial. I would be very glad if you could answer one remaining question related to Multiple Train-Test Splits.

So for my understanding Multiple Train-Test Splits is a good choice to find the optimal window size for the walk forward modeling later. Do you agree on that?

Since we can check the performar
had…

Thanks! 😊

Regards from Germany
Chris

Jason Brownlee Apr

Yes, it can be a good
what works well for your speci

Chris Pa April 19, 2019 at 6:49 pm #

REPLY

Cool! Thanks for you reply! 🙂

Scarlette April 26, 2019 at 5:30 pm #

REPLY

Hi Jason,
Thank you for your article which is very help for me as a beginner. I'm just quiet curious that the formula of multiple train split ,splitting train_sets and test_sets, why is formed like that. it will be so grateful if you can tell me where to get more info of it.

thanks!

best regards.

Jason Brownlee April 27, 2019 at 6:28 am #

REPLY

What do you mean exactly?

Scarlette April 27,

REPLY

I mean why is the

training_size = i * n_sampl
test_size = n_samples / (n_

can i get derivation proces

thank you for your reply.

Jason Brow

Perhaps check the original code library?

Scarlette April 28, 2019 at 12:42 pm #

get it .thank you very much.
best regards.

Anat May 2, 2019 at 7:19 am #

REPLY

Hi Jason,
Thank you for this great tutorial.
What do you think about doing the multiple train-test splits in a different way:
You split the data at a certain point to a trainset and a testset.
Then you use bootstrap or a “semi” k-fold cross validation where you randomly split both the train and test sets into k folds and then train the model k times, each time on k-1 folds of the trainset and evaluate the model on k-1 folds of the testset.

What are the advantages of using the multiple train-test splits that you proposed above over this approach?

Thanks!!
Anat

Jason Brownlee Ma

Generally, you cannot
results in data leakage.

Anat May 2, 2019 at

but I am not sugg
suggest separately splittin
the test set (of later obser

This just allows to repeat the training / evaluation process k times for significance of the results.

Anat May 2, 2019 at 10:35 pm #

REPLY

I think I understand the advantage of the multiple train-test splits: it accounts for the model performance at different windows in time. The solution I proposed did not… it only evaluates the model on one period of time

Adhithya May 8, 2019 at 7:27 am #

REPLY

Hey Jason,

Thanks for the post. I'm planning to use a supervised learning approach for time series data of 1 year (I have a retailer's store, week level sales information). How do you suggest I go about it?

Thanks,
Adhithya

Jason Brownlee May

REPLY

I recommend followin
https://machinelearningmaster
forecasting-model/

These tutorials will help to get
https://machinelearningmaster

BobIsYourUncle May 10,

If my model does not use features that incorporate information about prior samples, then how does a k-fold “cheat”? It seems like if the data in any fold is likely to repeat in the future, then why is the test result from fold 1 (thus training occurs on folds 2 thru 10) in a 10 fold CV invalid? I can see how re-sampling before k-fold CV is problematic (i.e. removing the data’s temporality), but have a hard time understanding the assertion that k-fold cannot be applied to time series data – full stop. Is your assertion always true, or is it model/feature dependent? I apologise for inquiring about something that should probably be obvious. Your website is awesome and your pedagogy’s on point. Thanks for all the help.

Jason Brownlee May 10, 2019 at 1:39 pm #

REPLY

The ordering of observations by time can cause a problem, as the model may have access to a future it may be asked to predict.

Does that help?

BobIsYourUncle May 11, 2019 at 4:49 am #

REPLY

I can wrap my head around it at a high level and never consider using k-fold for my time-series dataset/feature/model configurations. That said, as I fall further down the ML rabbit hole, I find certain configurations that make me question the assertion. Questioning it will undoubtedly come at a cost if I don’t make an effort to rigorousl

Can you recommend any b
temporal/nontemporal dat

Jason Brow

Not really, for
information about the

Instead, perhaps put t
evaluate skill under dif

data leakage gives optimistic evaluation of model performance.

More on data leakage here:
https://machinelearningmastery.com/data-leakage-machine-learning/

BobIsYourUncle May 11, 2019 at 10:29 am #

REPLY

I think I have a very fundamental misunderstanding of data types. While some data I have is sampled temporally, previous samples do not inform the outcome of future examples. That is, I currently consider data to be “time series” data even though no autocorrelation exist. If a set of time series data exhibits no autocorrelation, is it still considered flawed in a k-fold CV. I wish we had a whiteboard between us so you could explain this to me in 10 secs.

Jason Brownlee May 12, 2019 at 6:37 am #

REPLY

If the data is sequential (e.g. ordered by time), k-fold cross validation is probably a bad idea.

KabaL May 20, 2019 at 7:28 am #

REPLY

Hello, Jason, wonderful a
Let me ask you something, I’m doi
good time to buy or sell a stock (-1
values.

I did what you said and implement
first 500 rows for training and next
In each iteration on the for loop, I c
finally I saved the model on each it
model has the right weights for the
Should I use the last saved model
One last thing, in each iteration, I s
the mean of it, and surprisily the m

this model is able to predict new incoming data ?
Thanks in advance, cheers from Brasil !

Jason Brownlee May 20, 2019 at 2:35 pm #

REPLY

Generally, I don't believe the stock market is predictable:
https://machinelearningmastery.com/faq/single-faq/can-you-help-me-with-machine-learning-for-finance-or-the-stock-market

The evaluation procedure sounds good. I recommend fitting a final model on all available for making predictions after evaluation.

That is a nice accuracy, perhaps confirm that is reliable across many different time periods (decades).

KabaL May 21, 2019 at 11:42 am #

REPLY

Thank you for your reply, I'll read this article you sent..

Saying that I've already trained the model and it's "good" to go, so I start to make predictions in new data. e.g: when the market closes at 5 p.m, I'll get the latest data and feed into the network, which will predict either -1 or 1.

1) Should I train and save the model everyday with this newest data ? (I mean, if this helps keeping performance and im

2) Suppose that the model already
my point of view some predictions
model it again ? What's the smarte

Thanks for providing this quality co

Jason Brownlee Ma

Try re-training and usi
whichever strategy offers bette

Perhaps use a naive method, e.g. persistance as a fallback method when you don't trust the model in real time? Or perhaps fall back to the model from the prior day/week/month?

KabaL May 22, 2019 at 8:37 am #

REPLY

In fact, I won't use it real time.. I'll wait the market close, download the today's prices, predict with the model and decide (based on the prediction) if I'm going to long or short..

"Perhaps use a naive method, e.g. persistance as a fallback method when you don't trust the model in real time? Or perhaps fall back to the model from the prior day/week/month? " I didn't understand this part, is this the answer for the second question?

In a nutshell, the model is predicting what should I do next day. Later on I'll try to put some prints of the predictions here. Ty again

KabaL May 22, 2019 at 11:31 am #

REPLY

After running the WFV, here's a graph of the predictions: https://imgur.com/a/SUyOTzJ

As you can see, the model made some wrong predictions, could you help me improving this ? W
targets and training th
disapear..

Jason B

If you are
performance here:
https://machinelea

I strongly recommend this process:
https://machinelearningmastery.com/how-to-develop-a-skilful-time-series-forecasting-model/

Jason Brownlee May 22, 2019 at 2:31 pm #

REPLY

I see, if you're not predicting real time, you can ignore my comment about falling back to another model.

KabaL May 23, 2019 at 3:16 am #

Yes, I'm using that MLPClassifier() from sklearn.. I'l learn about RNN as soon as possible too.. Thank you for the links !

Lee July 23, 2019 at 2:09 pm #

REPLY

Hello Jason,

It amazes me after reading dozens of your blogs about time series.It still remains some confusions.

In my case, I try to use LSTM for univariate forecast.

1. I have read your post "How to C
Problem in Python" before, and tra
learning. If I want to apply backtes
transformed sequence?

2. In "Multiple Train-Test Splits" se
case(with as much data as possibl

3. It seems ambiguous when comb
forward validation". According to r
supervised-learning data first, ther
training data.

Best regard,
Lee

Jason Brownlee July 23, 2019 at 2:46 pm #

REPLY

The shortest path to understanding is here:
https://machinelearningmastery.com/start-here/#deep_learning_time_series

Or my book.

Or a complete tutorial, like this one:
https://machinelearningmastery.com/how-to-develop-deep-learning-models-for-univariate-time-series-forecasting/

Nevertheless:

1. Backtesting is performed on the transformed data, e.g. transform the data to a supervised learning problem after scaling/differencing/etc.

2. The idea is to estimate the performance of the model when making predictions on new data and determine if it has skill by comparing performance to a baseline model. Your chosen algorithm may or may not perform better with more history.

3. You can choose to update the model after every step forward or not. I often do not as its to computationally expensive..

Finally, LSTMs are terrible at univariate time series forecasting:
https://machinelearningmastery.com/findings-comparing-classical-and-machine-learning-methods-for-time-ser

Use an SARIMA or ETS instea

Lee July 23, 2019 at

Hello Jason,

Thank you for your reply s

I got a really long time seri
perform well in my two-lay

performs well operationally).

I would try other methods later.

Best regard,
Lee

Jason Brownlee July 24, 2019 at 7:50 am #

REPLY

Great!

Lee July 23, 2019 at 2:11 pm #

REPLY

Typo:

In question 2: “with as much data as possible” -> “with as much training data as possible”

Leave a Reply

Website

SUBMIT COMMENT

Welcome to Machine Learning Mastery!

Hi, I'm **Jason Brownlee**, PhD.
I write tutorials to help developers (*like you*) get results with machine learning.

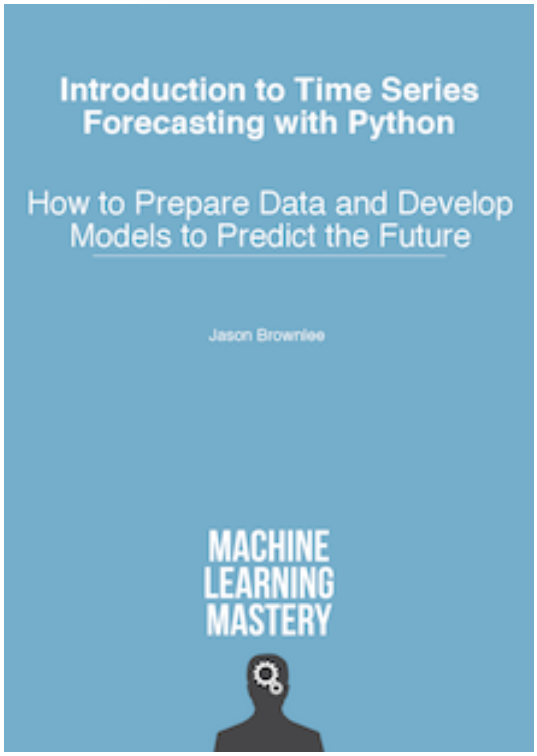
Read More

More Tutorials?

Start Here
(over 780 tutorials!)

Time Series Forecasting with Python
Explore time series data and predict the future.

Click to Get Started Now!



Join Newsletter

Subscribe Now
(join over 150,000 readers!)