Time Series Forecasting with Python 7-Day Mini-Course

by **Jason Brownlee** on March 22, 2017 in **Time Series**

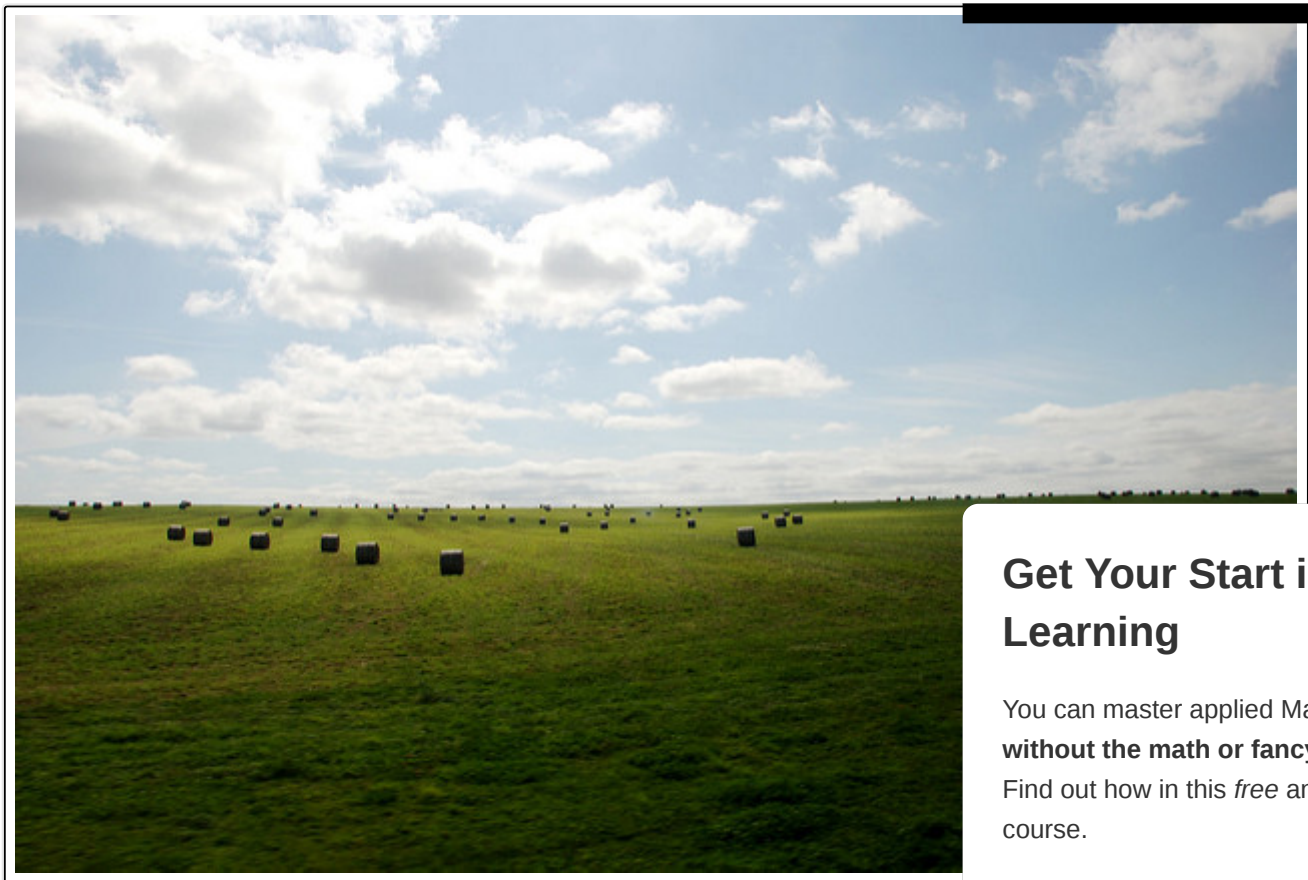
From Developer to Time Series Forecaster in 7 Days.

Python is one of the fastest-growing platforms for applied machine learning.

In this mini-course, you will discover how you can get started, build accurate models and confidently complete predictive modeling time series forecasting projects using Python in 7 days.

This is a big and important post. You might want to bookmark it.

Let's get started.



Time Series Forecasting with Python 7-Day Mini-Course
Photo by Raquel M, some rights reserved.

Who Is This Mini-Course For?

Before we get started, let's make sure you are in the right place.

The list below provides some general guidelines as to who this course was designed for.

Don't panic if you don't match these points exactly, you might just need to brush up in one area or another to keep up.

- **You're a Developer**: This is a course for developers. You are a developer of some sort. You know how to read and write code. You know how to develop and debug a program.

Get Your Start in Machine Learning

- **You know Python**: This is a course for Python people. You know the Python programming language, or you're a skilled enough developer that you can pick it up as you go along.
- **You know some Machine Learning**: This is a course for novice machine learning practitioners. You know some basic practical machine learning, or you can figure it out quickly.

This mini-course is neither a textbook on Python or a textbook on time series forecasting.

It will take you from a developer that knows a little machine learning to a developer who can get time series forecasting results using the Python ecosystem, the rising platform for professional machine learning.

Note: This mini-course assumes you have a working Python 2 or 3 SciPy environment with at least NumPy, Pandas, scikit-learn and statsmodels installed.

Mini-Course Overview

This mini-course is broken down into 7 lessons.

You could complete one lesson per day (*recommended*) or complete all of the lessons in one day (*h*
available and your level of enthusiasm.

Below are 7 lessons that will get you started and productive with machine learning in Python:

- **Lesson 01**: Time Series as Supervised Learning.
- **Lesson 02**: Load Time Series Data.
- **Lesson 03**: Data Visualization.
- **Lesson 04**: Persistence Forecast Model.
- **Lesson 05**: Autoregressive Forecast Model.
- **Lesson 06**: ARIMA Forecast Model.
- **Lesson 07**: Hello World End-to-End Project.

Each lesson could take you 60 seconds or up to 30 minutes. Take your time and complete the lessons at your own pace. Ask questions and even post results in the comments below.

The lessons expect you to go off and find out how to do things. I will give you hints, but part of the point of each lesson is to force you to learn where to go to look for help on and about the Python platform for time series (hint, I have all of the answers directly on this blog, use the search feature).

I do provide more help in the early lessons because I want you to build up some confidence and inertia.

Post your results in the comments, I'll cheer you on!

Hang in there, don't give up.

Lesson 01: Time Series as Supervised Learning

Time series problems are different to traditional prediction problems.

The addition of time adds an order to observations that both must be preserved and can provide adc

A time series dataset may look like the following:

```
1 Time, Observation
2 day1, obs1
3 day2, obs2
4 day3, obs3
```

We can reframe this data as a supervised learning problem with inputs and outputs to be predicted.

```
Input,  Output
?,      obs1
obs1,   obs2
obs2,   obs3
obs3,   ?
```

You can see that the reframing means we have to discard some rows with missing data.

Once it is reframed, we can then apply all of our favorite learning algorithms like k-Nearest Neighbors and Random Forest.

For more help, see the post:

- Time Series Forecasting as Supervised Learning

Lesson 02: Load Time Series Data

Before you can develop forecast models, you must load and work with your time series data.

Pandas provides tools to load data in CSV format.

In this lesson, you will download a standard time series dataset, load it in Pandas and explore it.

Download the daily female births dataset from DataMarket in CSV format and save it with the filenan

You can load a time series dataset as a Pandas Series and specify the header row at line zero, as fo

```
from pandas import Series
series = Series.from_csv('daily-births.csv', header=0)
```

Get used to exploring loaded time series data in Python:

- Print the first few rows using the *head()* function.
- Print the dimensions of the dataset using the *size* attribute.
- Query the dataset using a date-time string.
- Print summary statistics of the observations.

For more help, see the post:

Get Your Start in Machine Learning

- How to Load and Explore Time Series Data in Python

Lesson 03: Data Visualization

Data visualization is a big part of time series forecasting.

Line plots of observations over time are popular, but there is a suite of other plots that you can use to learn more about your problem.

In this lesson, you must download a standard time series dataset and create 6 different types of plots.

Download the monthly shampoo sales dataset from DataMarket in CSV format and save it with the filename "*shampoo-sales.csv*".

Now create the following 6 types of plots:

1. Line Plots.
2. Histograms and Density Plots.
3. Box and Whisker Plots by year or quarter.
4. Heat Maps.
5. Lag Plots or Scatter Plots.
6. Autocorrelation Plots.

Below is an example of a simple line plot to get you started:

```
1 from pandas import Series
2 from matplotlib import pyplot
3 series = Series.from_csv('shampoo-sales.csv', header=0)
4 series.plot()
5 pyplot.show()
```


For more help, see the post:

- Time Series Data Visualization with Python

Lesson 04: Persistence Forecast Model

It is important to establish a baseline forecast.

Get Your Start in Machine Learning

The simplest forecast you can make is to use the current observation (t) to predict the observation at the next time step (t+1).

This is called the naive forecast or the persistence forecast and may be the best possible model on some time series forecast problems.

In this lesson, you will make a persistence forecast for a standard time series forecast problem.

Download the daily female births dataset from DataMarket in CSV format and save it with the filename "*daily-births.csv*".

You can implement the persistence forecast as a single line function, as follows:

```
# persistence model
def model_persistence(x):
    return x
```

Write code to load the dataset and use the persistence forecast to make a prediction for each time s[illegible] make a forecast for the first time step in the dataset as there is no previous observation to use.

Store all of the predictions in a list. You can calculate a Root Mean Squared Error (RMSE) for the pr[illegible] follows:

```
from sklearn.metrics import mean_squared_error
from math import sqrt
predictions = []
actual = series.values[1:]
rmse = sqrt(mean_squared_error(actual, predictions))
```

For more help, see the post:

- How to Make Baseline Predictions for Time Series Forecasting with Python

Lesson 05: Autoregressive Forecast Model

Autoregression means developing a linear model that uses observations at previous time steps to predict observations at future time step ("auto" means self in ancient Greek).

Autoregression is a quick and powerful time series forecasting method.

Get Your Start in Machine Learning

The statsmodels Python library provides the autoregression model in the AR class.

In this lesson, you will develop an autoregressive forecast model for a standard time series dataset.

Download the monthly shampoo sales dataset from DataMarket in CSV format and save it with the filename "*shampoo-sales.csv*".

You can fit an AR model as follows:

```
model = AR(dataset)
model_fit = model.fit()
```

You can predict the next out of sample observation with a fit AR model as follows:

```
prediction = model_fit.predict(start=len(dataset), end=len(dataset))
```

You may want to experiment by fitting the model on half of the dataset and predicting one or more of predictions to the actual observations.

For more help, see the post:

- Autoregression Models for Time Series Forecasting With Python

Lesson 06: ARIMA Forecast Model

The ARIMA is a classical linear model for time series forecasting.

It combines the autoregressive model (AR), differencing to remove trends and seasonality, called int which is an old name given to a model that forecasts the error, used to correct predictions.

The statsmodels Python library provides the ARIMA class.

In this lesson, you will develop an ARIMA model for a standard time series dataset.

Download the monthly shampoo sales dataset from DataMarket in CSV format and save it with the filename "*shampoo-sales.csv*".

The ARIMA class requires an order(p,d,q) that is comprised of three arguments p, d and q for the AR

Get Your Start in Machine Learning

You can fit an ARIMA model as follows:

```
model = ARIMA(dataset, order=(0,1,0))
model_fit = model.fit()
```

You can make a one-step out-of-sample forecast for a fit ARIMA model as follows:

```
outcome = model_fit.forecast()[0]
```

The shampoo dataset has a trend so I'd recommend a d value of 1. Experiment with different p and q values and evaluate the predictions from resulting models.

For more help, see the post:

- How to Create an ARIMA Model for Time Series Forecasting with Python

Lesson 07: Hello World End-to-End Project

You now have the tools to work through a time series problem and develop a simple forecast model.

In this lesson, you will use the skills learned from all of the prior lessons to work through a new time

Download the quarterly S&P 500 index, 1900-1996 dataset from DataMarket in CSV format and sav

Split the data, perhaps extract the last 4 or 8 quarters to a separate file. Work through the problem a ng:

1. Load and explore the dataset.
2. Visualize the dataset.
3. Develop a persistence model.
4. Develop an autoregressive model.
5. Develop an ARIMA model.
6. Visualize forecasts and summarize forecast error.

For an example of working through a project, see the post:

- Time Series Forecast Study with Python: Monthly Sales of French Champagne

The End!
(Look How Far You Have Come)

You made it. Well done!

Take a moment and look back at how far you have come.

You discovered:

- How to frame a time series forecasting problem as supervised learning.
- How to load and explore time series data with Pandas.
- How to plot and visualize time series data a number of different ways.
- How to develop a naive forecast called the persistence model as a baseline.
- How to develop an autoregressive forecast model using lagged observations.
- How to develop an ARIMA model including autoregression, integration and moving average ele
- How to pull all of these elements together into an end-to-end project.

Don't make light of this, you have come a long way in a short amount of time.

This is just the beginning of your time series forecasting journey with Python. Keep practicing and de

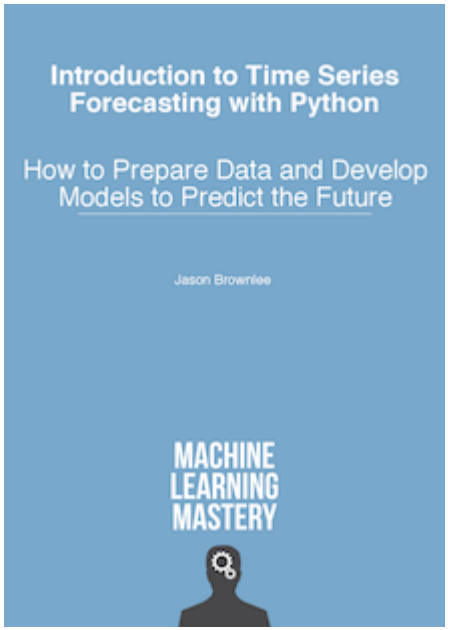
Summary

How Did You Go With The Mini-Course?
Did you enjoy this mini-course?

Do you have any questions? Were there any sticking points?
Let me know. Leave a comment below.

Want to Develop Time Series Forecasts

Get Your Start in Machine Learning



About Jason Brownlee

Dr. Jason Brownlee is a husband, proud father, academic researcher, author, professional devel
to helping developers get started and get good at applied machine learning. Learn more.
View all posts by Jason Brownlee →

‹ How to Handle Missing Data with Python

How to Make Out-of-Sample Forecasts with ARIMA in Python ›

Get Your Start in Machine Learning

6 Responses to *Time Series Forecasting with Python 7-Day Mini-Course*

Luca May 5, 2017 at 2:29 am #

REPLY

Hi

Thanks for so many articles in your blog. Really appreciate.

I have a question that I see sometimes we use a fixed-parameter model (e.g. parameters in ARIMA model is always fixed), while other times use an iterative way to determine the model parameters in each iteration of a test data sample. Are there any differences or reasons behind that? and when fixed model is useful and when to use an iterative way?

my understanding from the examples are: iterative way of modeling ARIMA seems more appropriate to s

Thanks a lot

Jason Brownlee May 5, 2017 at 7:32 am #

In general, I would suggest evaluating a suite of different models for a problem and see wh

Gururaj August 13, 2017 at 12:21 pm #

Thanks Jason for these helpful articles. I have a general question. When we have a very high number of sensors, are there simpler methods to model them over building a model for each sensor using its own time series data?

If we have some intuition that we may find groups of sensors which may exhibit similar behaviour, is there a method to cluster them and validate, given the individual time series data?

Jason Brownlee August 14, 2017 at 6:23 am #

REPLY

Good question Gururaj, sorry I have not worked on consolidating sensor data, I can't give you expert advice.

joseph September 10, 2017 at 4:02 pm #

REPLY

Thanks for the course.
I intend on doing the course
I would like to know:
do you have anomaly detection course?
are hidden markov models and recurrent nn fit this area(time series)?
thanks
joseph

Jason Brownlee September 11, 2017 at 12:05 pm #

Not at this stage, perhaps in the future.

Leave a Reply

Get Your Start in Machine Learning

Name (required)

Email (will not be published) (required)

Website

SUBMIT COMMENT

Welcome to Machine Learning Mastery

Hi, I'm Dr. Jason Brownlee.
My goal is to make practitioners like YOU awesome at applied machine learning.

Read More

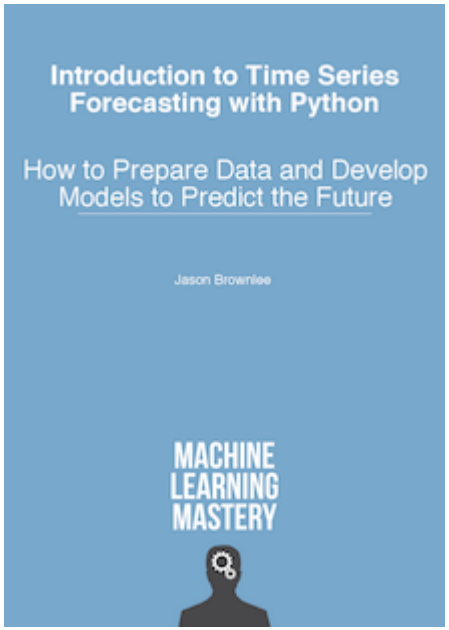
Get Good at Time Series Forecasting

Need visualizations and forecast models?
Looking for step-by-step tutorials?

Get Your Start in Machine Learning

Want end-to-end projects?

Get Started with Time Series Forecasting in Python!



POPULAR

Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras
JULY 21, 2016

Your First Machine Learning Project in Python Step-By-Step
JUNE 10, 2016

Develop Your First Neural Network in Python With Keras Step-By-Step
MAY 24, 2016

Sequence Classification with LSTM Recurrent Neural Networks in Python with Keras
JULY 26, 2016

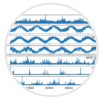
Get Your Start in Machine Learning

How to Setup a Python Environment for Machine Learning and Deep Learning with Anaconda
MARCH 13, 2017

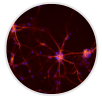
Time Series Forecasting with the Long Short-Term Memory Network in Python
APRIL 7, 2017

Multi-Class Classification Tutorial with the Keras Deep Learning Library
JUNE 2, 2016

Regression Tutorial with the Keras Deep Learning Library in Python
JUNE 9, 2016



Multivariate Time Series Forecasting with LSTMs in Keras
AUGUST 14, 2017



How to Implement the Backpropagation Algorithm From Scratch In Python
NOVEMBER 7, 2016

Get Your Start in Machine Learning