Feature Selection for Time Series Forecasting with Python

by **Jason Brownlee** on March 29, 2017 in **Time Series**

The use of machine learning methods on time series data requires feature engineering.

A univariate time series dataset is only comprised of a sequence of observations. These must be transformed into input and output features in order to use supervised learning algorithms.

The problem is that there is little limit to the type and number of features you can engineer for a time series problem. Classical time series analysis tools like the correlogram can help with evaluating lag variables, but do not directly help when selecting other types of features, such as those derived from the timestamps (year, month or day) and moving statistics, like a moving average.

In this tutorial, you will discover how you can use the machine learning tools of feature importance and feature selection when working with time series data.

Get Your Start in Machine Learning

After completing this tutorial, you will know:

- How to create and interpret a correlogram of lagged observations.
- How to calculate and interpret feature importance scores for time series features.
- How to perform feature selection on time series input variables.

Let’s get started.

Tutorial Overview

This tutorial is broken down into the following 5 steps:

1. **Monthly Car Sales Dataset**: That describes the dataset we will be working with.
2. **Make Stationary**: That describes how to make the dataset stationary for analysis and forecastir
3. **Autocorrelation Plot**: That describes how to create a correlogram of the time series data.
4. **Feature Importance of Lag Variables**: That describes how to calculate and review feature imp
5. **Feature Selection of Lag Variables**: That describes how to calculate and review feature select

Let’s start off by looking at a standard time series dataset.

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Stop learning Time Series Forecasting the

Take my free 7-day email course and discover data prep, modeling and

Click to sign-up and also get a free PDF Ebook version of the course.

Start Your FREE Mini-Course Now!

Monthly Car Sales Dataset

Get Your Start in Machine Learning

In this tutorial, we will use the Monthly Car Sales dataset.

This dataset describes the number of car sales in Quebec, Canada between 1960 and 1968.

The units are a count of the number of sales and there are 108 observations. The source data is credited to Abraham and Ledolter (1983).

You can download the dataset from DataMarket.

Download the dataset and save it into your current working directory with the filename "*car-sales.csv*". Note, you may need to delete the footer information from the file.

The code below loads the dataset as a Pandas *Series* object.

```
# line plot of time series
from pandas import Series
from matplotlib import pyplot
# load dataset
series = Series.from_csv('car-sales.csv', header=0)
# display first few rows
print(series.head(5))
# line plot of dataset
series.plot()
pyplot.show()
```

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

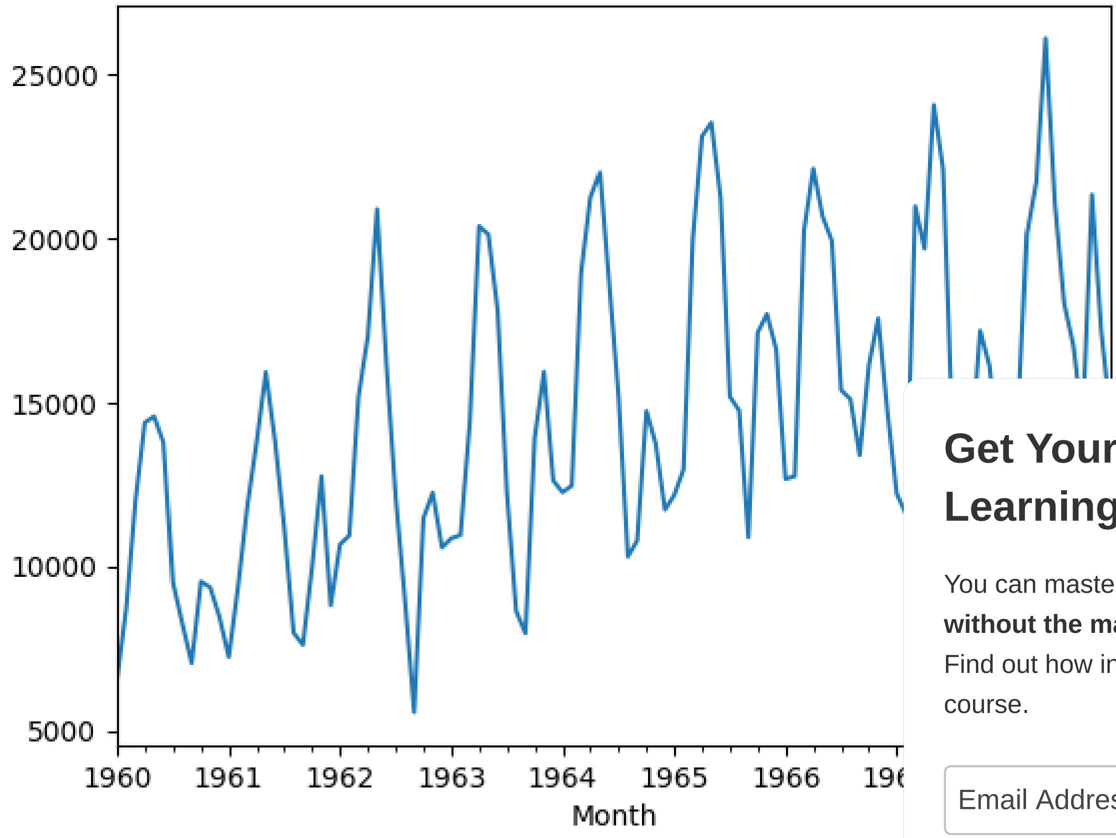
START MY EMAIL COURSE

Running the example prints the first 5 rows of data.

```
Month
1960-01-01 6550
1960-02-01 8728
1960-03-01 12026
1960-04-01 14395
1960-05-01 14587
Name: Sales, dtype: int64
```

A line plot of the data is also provided.

Get Your Start in Machine Learning



Monthly Car Sales Dataset Line Plot

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Make Stationary

We can see a clear seasonality and increasing trend in the data.

The trend and seasonality are fixed components that can be added to any prediction we make. They are useful, but need to be removed in order to explore any other systematic signals that can help make predictions.

A time series with seasonality and trend removed is called stationary.

Get Your Start in Machine Learning

To remove the seasonality, we can take the seasonal difference, resulting in a so-called seasonally [redacted]

The period of the seasonality appears to be one year (12 months). The code below calculates the seasonally adjusted time series and saves it to the file "*seasonally-adjusted.csv*".

```
# seasonally adjust the time series
from pandas import Series
from matplotlib import pyplot
# load dataset
series = Series.from_csv('car-sales.csv', header=0)
# seasonal difference
differenced = series.diff(12)
# trim off the first year of empty data
differenced = differenced[12:]
# save differenced dataset to file
differenced.to_csv('seasonally_adjusted.csv')
# plot differenced dataset
differenced.plot()
pyplot.show()
```

Because the first 12 months of data have no prior data to be differenced against, they must be disca

The stationary data is stored in "*seasonally-adjusted.csv*". A line plot of the differenced data is create

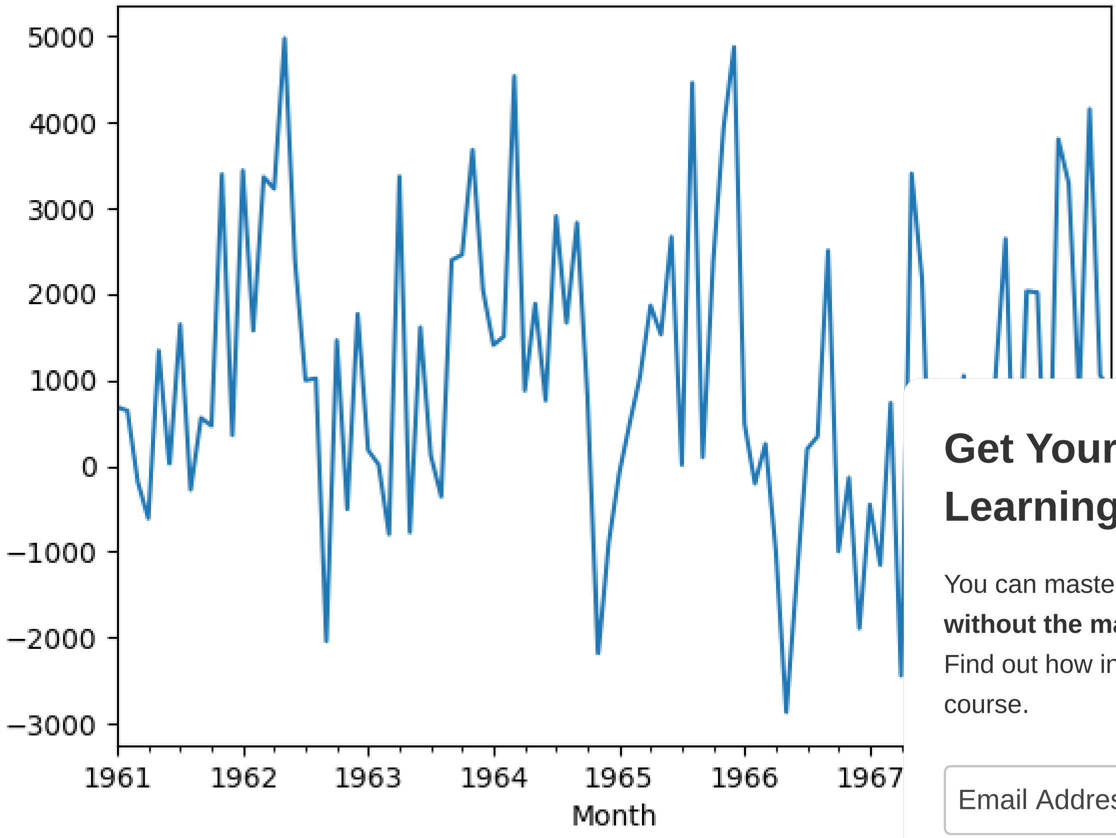
Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Get Your Start in Machine Learning



Seasonally Differenced Monthly Car Sales Dataset Line Plo

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

The plot suggests that the seasonality and trend information was removed by differencing.

Autocorrelation Plot

Traditionally, time series features are selected based on their correlation with the output variable.

This is called autocorrelation and involves plotting autocorrelation plots, also called a correlogram. These show the correlation of each lagged observation and whether or not the correlation is statistically significant.

Get Your Start in Machine Learning

For example, the code below plots the correlogram for all lag variables in the Monthly Car Sales dat[redacted]

```
from pandas import Series
from statsmodels.graphics.tsaplots import plot_acf
from matplotlib import pyplot
series = Series.from_csv('seasonally_adjusted.csv', header=None)
plot_acf(series)
pyplot.show()
```

Running the example creates a correlogram, or Autocorrelation Function (ACF) plot, of the data.

The plot shows lag values along the x-axis and correlation on the y-axis between -1 and 1 for negatively and positively correlated lags respectively.

The dots above the blue area indicate statistical significance. The correlation of 1 for the lag value of 0 indicates 100% positive correlation of an observation with itself.

The plot shows significant lag values at 1, 2, 12, and 17 months.

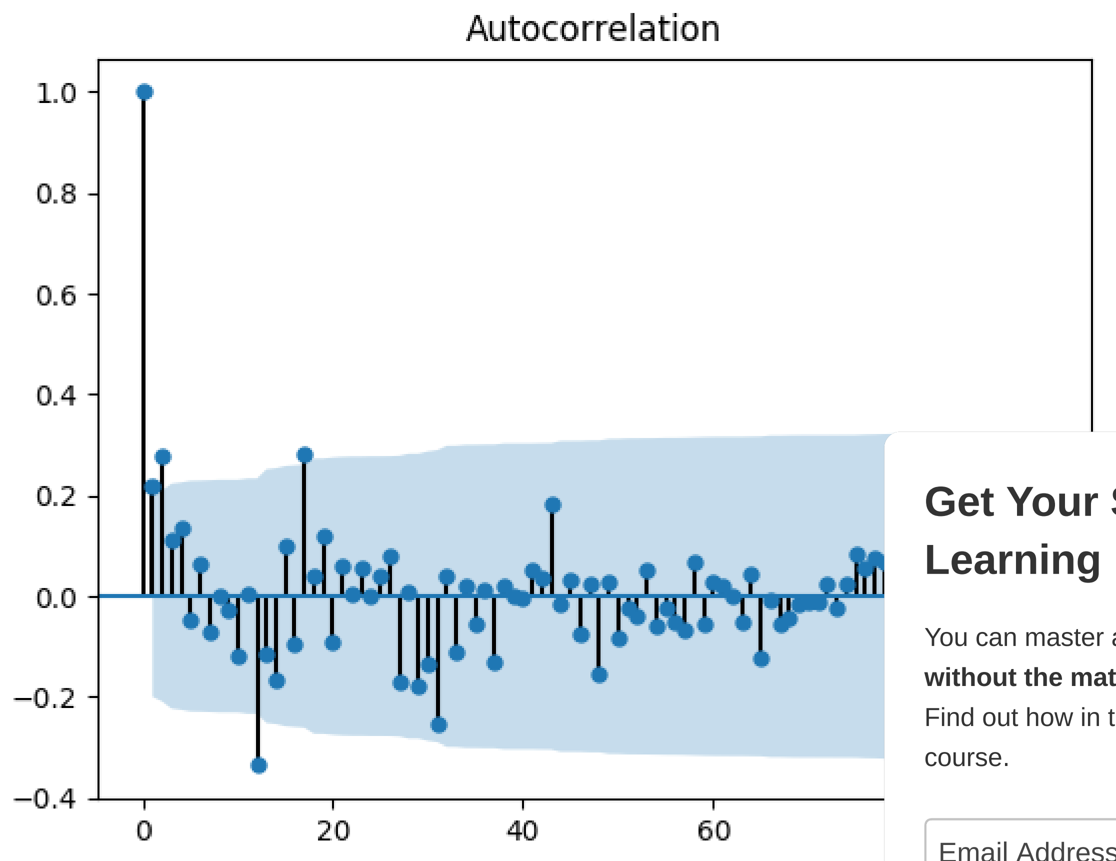
Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Get Your Start in Machine Learning



Correlogram of the Monthly Car Sales Dataset

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

This analysis provides a good baseline for comparison.

Time Series to Supervised Learning

We can convert the univariate Monthly Car Sales dataset into a supervised learning problem by taking the lag observation (e.g. t-1) as inputs and using the current observation (t) as the output variable.

We can do this in Pandas using the shift function to create new columns of shifted observations.

Get Your Start in Machine Learning

The example below creates a new time series with 12 months of lag values to predict the current ob[redacted]

The shift of 12 months means that the first 12 rows of data are unusable as they contain *NaN* values.

```
from pandas import Series
from pandas import DataFrame
# load dataset
series = Series.from_csv('seasonally_adjusted.csv', header=None)
# reframe as supervised learning
dataframe = DataFrame()
for i in range(12,0,-1):
dataframe['t-'+str(i)] = series.shift(i)
dataframe['t'] = series.values
print(dataframe.head(13))
dataframe = dataframe[13:]
# save to new file
dataframe.to_csv('lags_12months_features.csv', index=False)
```

Running the example prints the first 13 rows of data showing the unusable first 12 rows and the usa

```
              t-12   t-11   t-10    t-9     t-8     t-7     t-6     t-5  \
1961-01-01     NaN    NaN    NaN    NaN     NaN     NaN     NaN     NaN
1961-02-01     NaN    NaN    NaN    NaN     NaN     NaN     NaN     NaN
1961-03-01     NaN    NaN    NaN    NaN     NaN     NaN     NaN     NaN
1961-04-01     NaN    NaN    NaN    NaN     NaN     NaN     NaN     NaN
1961-05-01     NaN    NaN    NaN    NaN     NaN     NaN     NaN     NaN
1961-06-01     NaN    NaN    NaN    NaN     NaN     NaN     NaN   687.0
1961-07-01     NaN    NaN    NaN    NaN     NaN     NaN   687.0   646.0
1961-08-01     NaN    NaN    NaN    NaN     NaN   687.0   646.0  -189.0
1961-09-01     NaN    NaN    NaN    NaN   687.0   646.0  -189.0  -611.0
1961-10-01     NaN    NaN    NaN  687.0   646.0  -189.0  -611.0  1339.0
1961-11-01     NaN    NaN  687.0  646.0  -189.0  -611.0  1339.0    30.0
1961-12-01     NaN  687.0  646.0 -189.0  -611.0  1339.0    30.0  1645.0
1962-01-01   687.0  646.0 -189.0 -611.0  1339.0    30.0  1645.0  -276.0

                t-4     t-3     t-2     t-1       t
1961-01-01      NaN     NaN     NaN     NaN   687.0
1961-02-01      NaN     NaN     NaN   687.0   646.0
1961-03-01      NaN     NaN   687.0   646.0  -189.0
1961-04-01      NaN   687.0   646.0  -189.0  -611.0
1961-05-01    687.0   646.0  -189.0  -611.0  1339.0
1961-06-01    646.0  -189.0  -611.0  1339.0    30.0
1961-07-01   -189.0  -611.0  1339.0    30.0  1645.0
1961-08-01   -611.0  1339.0    30.0  1645.0  -276.0
1961-09-01   1339.0    30.0  1645.0  -276.0   561.0
```

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Get Your Start in Machine Learning

```
26 1961-10-01    30.0  1645.0  -276.0   561.0   470.0
27 1961-11-01  1645.0  -276.0   561.0   470.0  3395.0
28 1961-12-01  -276.0   561.0   470.0  3395.0   360.0
29 1962-01-01   561.0   470.0  3395.0   360.0  3440.0
```

The first 12 rows are removed from the new dataset and results are saved in the file "*lags_12months_features.csv*".

This process can be repeated with an arbitrary number of time steps, such as 6 months or 24 months, and I would recommend experimenting.

Feature Importance of Lag Variables

Ensembles of decision trees, like bagged trees, random forest, and extra trees, can be used to calculate a feature importance score.

This is common in machine learning to estimate the relative usefulness of input features when devel

We can use feature importance to help to estimate the relative importance of contrived input feature

This is important because we can contrive not only the lag observation features above, but also feat ng statistics, and much more. Feature importance is one method to help sort out what might be more us

The example below loads the supervised learning view of the dataset created in the previous sectio (RandomForestRegressor), and summarizes the relative feature importance scores for each of the 1

A large-ish number of trees is used to ensure the scores are somewhat stable. Additionally, the rand ame result is achieved each time the code is run.

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**. Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

```
from pandas import read_csv
from sklearn.ensemble import RandomForestRegressor
from matplotlib import pyplot
# load data
dataframe = read_csv('lags_12months_features.csv', header=0)
array = dataframe.values
# split into input and output
X = array[:,0:-1]
y = array[:,-1]
# fit random forest model
model = RandomForestRegressor(n_estimators=500, random_state=1)
model.fit(X, y)
# show importance scores
```

Get Your Start in Machine Learning

```
print(model.feature_importances_)
# plot importance scores
names = dataframe.columns.values[0:-1]
ticks = [i for i in range(len(names))]
pyplot.bar(ticks, model.feature_importances_)
pyplot.xticks(ticks, names)
pyplot.show()
```

Running the example first prints the importance scores of the lagged observations.

```
[ 0.21642244  0.06271259  0.05662302  0.05543768  0.07155573  0.08478599
  0.07699371  0.05366735  0.1033234   0.04897883  0.1066669   0.06283236]
```

The scores are then plotted as a bar graph.

The plot shows the high relative importance of the observation at t-12 and, to a lesser degree, the im

It is interesting to note a difference with the outcome from the correlogram above.

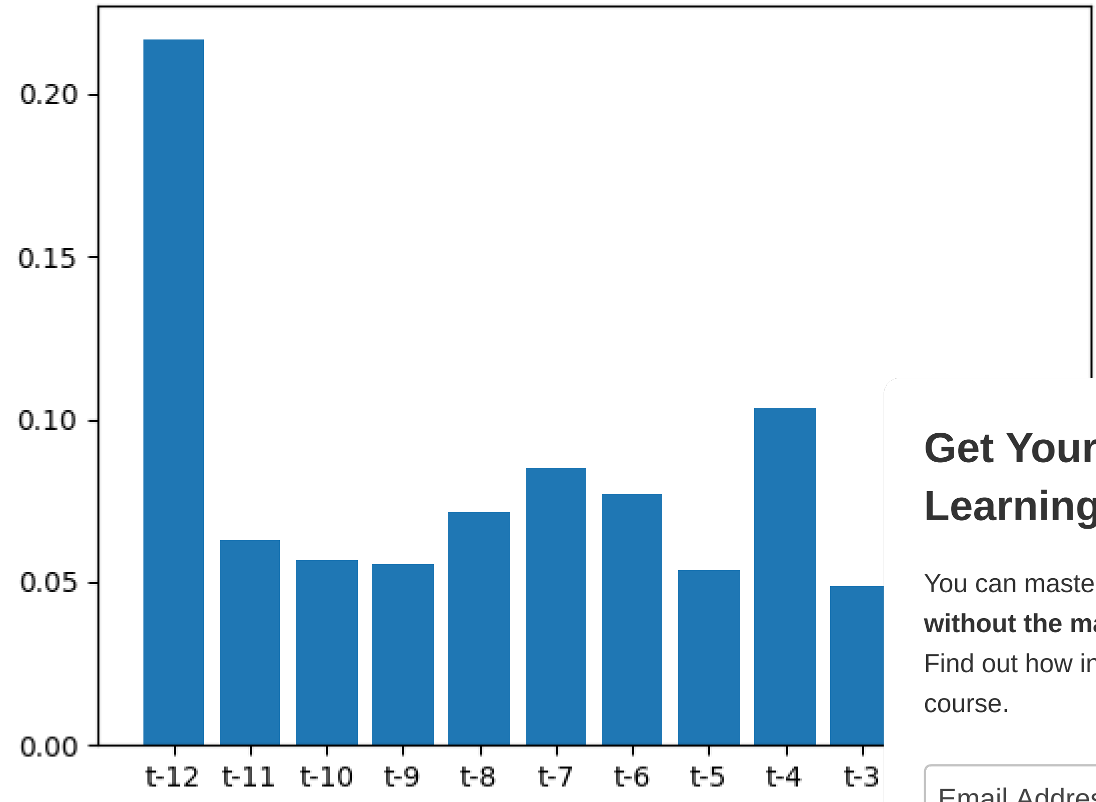
Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**. Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Get Your Start in Machine Learning



Bar Graph of Feature Importance Scores on the Monthly Car Sales

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

This process can be repeated with different methods that can calculate importance scores, such as gradient boosting, extra trees, and bagged decision trees.

Feature Selection of Lag Variables

We can also use feature selection to automatically identify and select those input features that are most predictive.

A popular method for feature selection is called Recursive Feature Selection (RFE).

Get Your Start in Machine Learning

RFE works by creating predictive models, weighting features, and pruning those with the smallest [redacted] number of features are left.

The example below uses RFE with a random forest predictive model and sets the desired number of input features to 4.

```
from pandas import read_csv
from sklearn.feature_selection import RFE
from sklearn.ensemble import RandomForestRegressor
from matplotlib import pyplot
# load dataset
dataframe = read_csv('lags_12months_features.csv', header=0)
# separate into input and output variables
array = dataframe.values
X = array[:,0:-1]
y = array[:,-1]
# perform feature selection
rfe = RFE(RandomForestRegressor(n_estimators=500, random_state=1), 4)
fit = rfe.fit(X, y)
# report selected features
print('Selected Features:')
names = dataframe.columns.values[0:-1]
for i in range(len(fit.support_)):
    if fit.support_[i]:
        print(names[i])
# plot feature rank
names = dataframe.columns.values[0:-1]
ticks = [i for i in range(len(names))]
pyplot.bar(ticks, fit.ranking_)
pyplot.xticks(ticks, names)
pyplot.show()
```

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**. Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

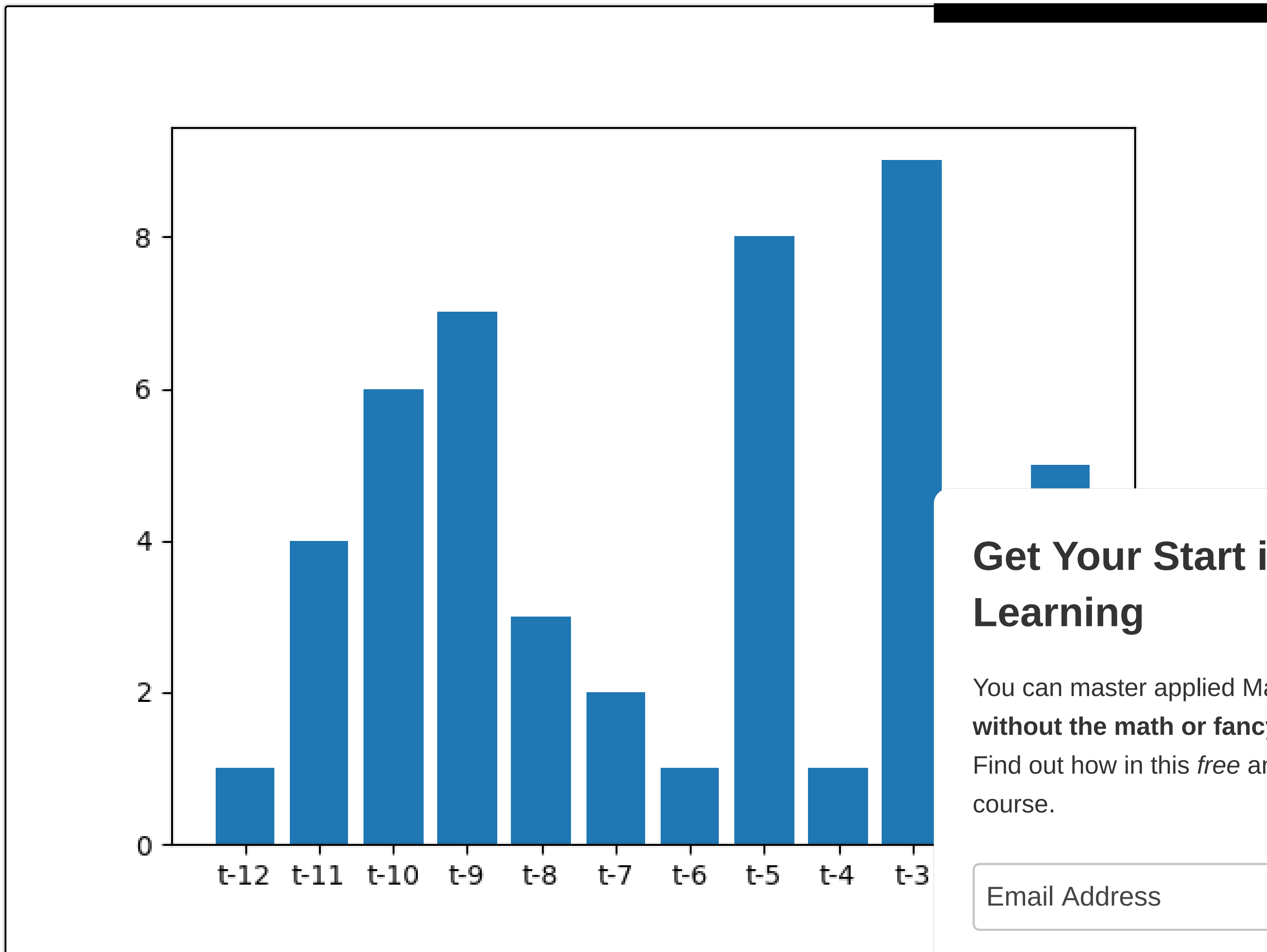
Running the example prints the names of the 4 selected features.

Unsurprisingly, the results match features that showed a high importance in the previous section.

```
Selected Features:
t-12
t-6
t-4
t-2
```

A bar graph is also created showing the feature selection rank (smaller is better) for each input feature.

Get Your Start in Machine Learning



Bar Graph of Feature Selection Rank on the Monthly Car Sales D

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

This process can be repeated with different numbers of features to select more than 4 and different models other than random forest.

Summary

In this tutorial, you discovered how to use the tools of applied machine learning to help select features from time series data when forecasting.

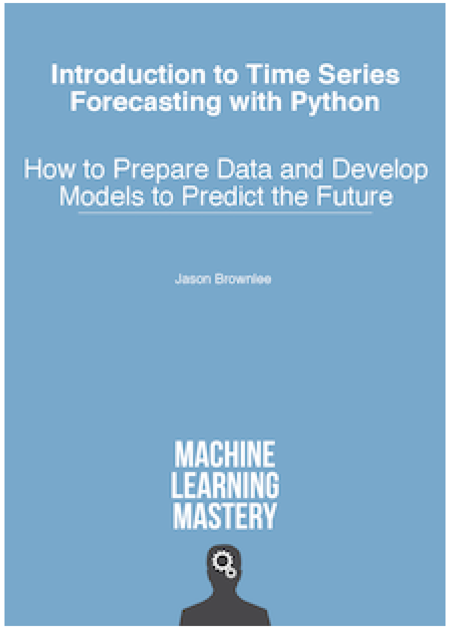
Specifically, you learned:

Get Your Start in Machine Learning

- How to interpret a correlogram for highly correlated lagged observations.
- How to calculate and review feature importance scores in time series data.
- How to use feature selection to identify the most relevant input variables in time series data.

Do you have any questions about feature selection with time series data?
Ask your questions in the comments and I will do my best to answer.

Want to Develop Time Series Forecasts with Python?



Develop Your Own Forecasts in Minutes

...with just a few lines of p

Discover how in my ne
Introduction to Time Series Forec

It covers **self-study tutorials** and **end-to-**
Loading data, visualization, modeling, algori

Finally Bring Time Series Your Own Proj

Skip the Academics. Jus

Click to learn mo

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

About Jason Brownlee

Get Your Start in Machine Learning

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

< Sensitivity Analysis of History Size to Forecast Skill with ARIMA in Python

Simple Time Series Forecasting Models to Test So That You Don't Fool Yourself >

20 Responses to *Feature Selection for Time Series Forecasting with Pytho*

Andrewcz March 29, 2017 at 5:33 pm #

Hi Jason big fan! I was wondering if you are going to a series on multivariate array time series f
Many thanks,
Best,
Andrew

Jason Brownlee March 30, 2017 at 8:48 am #

Yes, I hope to cover this soon Andrew.

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Benson Dube April 2, 2017 at 6:13 am #

REPLY

Hello Jason,

Many thanks for this blog. I will be so Interested to see how the multivariate Time Series Forecast is dea

Get Your Start in Machine Learning

Keep up the good works,

Best Regards

Ben

Jason Brownlee April 2, 2017 at 6:33 am # REPLY

Thanks Ben, I hope to cover multivariate time series soon.

Kélian April 13, 2017 at 2:05 am #

Hello Jason,

I wondered about your choice to keep only the last 12 lags for the feature importance and feature selecti

Because i understand the correlogram showed you should push the study until the 17 lag (correlogram s state)

I m I right?

Thanks for your work!

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Jason Brownlee April 13, 2017 at 10:07 am #

Yes, I kept it short for brevity.

Mehrdad May 26, 2017 at 5:18 am # REPLY

Get Your Start in Machine Learning

The output of this lines
'plot_acf(series)'
'pyplot.show()'
is not like yours. It just shows an straight line.
May you please check it.
Thanks

Merlin June 1, 2017 at 8:58 pm # REPLY

Yeah, the plot_acf thing is not working properly.

Jason Brownlee June 2, 2017 at 12:57 pm #

What problem do you see exactly?

What version of statsmodels are you using?

Jason Brownlee June 2, 2017 at 11:50 am #

I can confirm the example, please check that you have all of the code and the same source

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Ralph Li June 30, 2017 at 6:09 pm # REPLY

Hello Jason!

Can you recommend some references about recursive feature selection and random forest on feature selection for time series?

Thanks!

Get Your Start in Machine Learning

Jason Brownlee July 1, 2017 at 6:29 am # REPLY

No. My best advice: try it, get results and use them in developing better models.

Saurav Sharma July 27, 2017 at 2:38 am # REPLY

Hi Jason!

I am still unable to understand the importance of lag variable?

Is lag applied to a feature variable to find correlation with the target variable?

Thanks!

Jason Brownlee July 27, 2017 at 8:11 am #

A lag is a past observation, an observation at a prior time step.

We can use these as input features to learning models. So abstractly we can predict today based o

Yesterday's ob is a lag variable.

Does that help?

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**. Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Mert August 26, 2017 at 6:43 pm # REPLY

Dear Jason,
I am trying to run your code above with X size of (358,168) and test y (358,24), and having error "ValueError: bad input shape (358, 24)". I would like to find the most relevant 12 features from 168 features in X(358,168) depending on 24 output of y(358,24)

Get Your Start in Machine Learning

My y matrix has 24 output instead of 1. What might be the reason for the error?

X = array[:,0:168]
y = array[:,168:192]
rfe = RFE(RandomForestRegressor(n_estimators=500, random_state=1), 12)
fit = rfe.fit(X, y)

Jason Brownlee August 27, 2017 at 5:48 am #

REPLY

That might be too many output variables, most algorithms expect a single output variable in sklearn.

I can't think of any that support multiple, but I could be wrong.

You might like to explore a neural network model instead?

Mert August 28, 2017 at 10:49 am #

Thanks for your comment Jason.
Actually, what I would like to do is determining the most relevant feature with RFE, then training
think it is a reasonable approach?
For the multiple output error, I will run RFE for each output instead of 24 one by one.

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Jason Brownlee August 29, 2017 at 5:00 pm #

REPLY

You could try it and it would make sense if there is one highly predictive feature, but I would encourage you to test many configurations.

Orry October 9, 2017 at 9:59 pm #

REPLY

Get Your Start in Machine Learning

Thanks for the great tutorial.

I was wondering if you could explain the logic of why ACF might show some lags as statistically significant, while feature selection might show totally different lags as having predictive power.

Jason Brownlee October 10, 2017 at 7:44 am # REPLY

Different operate under different assumptions and in turn, produce differing results. This is to be expected.

Leave a Reply

Name (required)

Email (will not be published) (required)

Website

SUBMIT COMMENT

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Get Your Start in Machine Learning

Welcome to Machine Learning Mastery

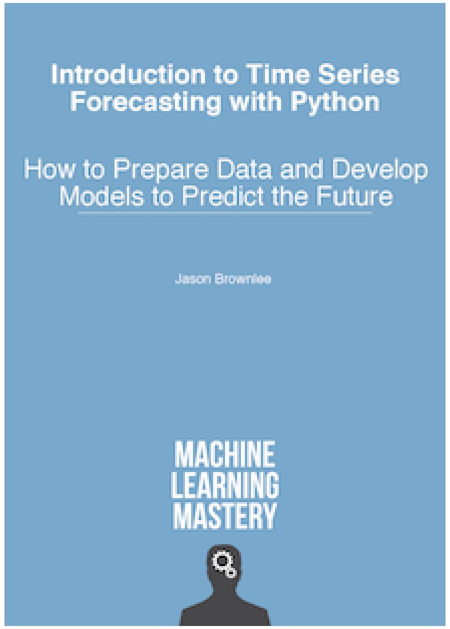
Hi, I'm Dr. Jason Brownlee.
My goal is to make practitioners like YOU awesome at applied machine learning.

Read More

Get Good at Time Series Forecasting

Need visualizations and forecast models?
Looking for step-by-step tutorials?
Want end-to-end projects?

Get Started with Time Series Forecasting in Python!



Get Your Start in Machine Learning

×

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

POPULAR

Get Your Start in Machine Learning


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

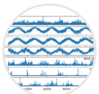

Sequence Classification with LSTM Recurrent Neural Networks in Python with Keras
JULY 26, 2016

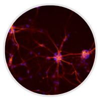
How to Setup a Python Environment for Machine Learning and Deep Learning with Anaconda
MARCH 13, 2017


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

×

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Get Your Start in Machine Learning

© 2017 Machine Learning Mastery. All Rights Reserved.

Privacy | Contact | About

Get Your Start in Machine Learning

You can master applied Machine Learning **without the math or fancy degree**.
Find out how in this *free* and *practical* email course.

Email Address

START MY EMAIL COURSE

Get Your Start in Machine Learning