



Rolling Window Regression: A Simple Approach for Time Series Next Value Predictions

by Srinath Perera MVB · Jul. 15, 16 · Big Data Zone · Opinion

Hortonworks Sandbox for HDP and HDF is your chance to get started on learning, developing, testing and trying out new features. Each download comes preconfigured with interactive tutorials, sample data and developments from the Apache community.

Given a time series, predicting the next value is a problem that fascinated programmers for a long time. Obviously, a key reason for this attention is stock markets, which promised untold riches if you can crack it. However, except for few (see A rare interview with the mathematician who cracked Wall Street), those riches have proved elusive.

Thanks to IoT (Internet of Things), time series analysis is poised to come back into the limelight. IoT let us place ubiquitous sensors everywhere, collect data, and act on that data. IoT devices collect data through time and resulting data are almost always time series data.

The following are few use cases for time series prediction:

1. Power load prediction
2. Demand prediction for Retail Stores
3. Services (e.g. airline check-in counters, government offices) client prediction
4. Revenue forecasts
5. ICU care vital monitoring
6. Yield and crop prediction

Let's explore the techniques available for time series forecasts.

The first question is that "isn't it regression?". It is close, but not the same as regression. In a time series, each value is affected by the values just preceding this value. For example, if there is a lot of traffic at 4.55 in a junction, chances are that there will be some traffic at 4.56 as well. This is called autocorrelation. If you are doing regression, you will only consider $x(t)$ while due to auto correlation, $x(t-1)$, $x(t-2)$, ... will also affect the outcome. So we can think about time series forecasts as regression that factor in autocorrelation as well.

For this discussion, let's consider "Individual household electric power consumption Data Set", which is data collected from one household over four years in one-minute intervals. Let's only consider three fields, and the data set will look like the following:

	Date	Time	Global_active_power
1	16/12/2006	17:24:00	4.216
2	16/12/2006	17:25:00	5.360
3	16/12/2006	17:26:00	5.374
4	16/12/2006	17:27:00	5.388
5	16/12/2006	17:28:00	3.666
6	16/12/2006	17:29:00	3.520
7	16/12/2006	17:30:00	3.702
8	16/12/2006	17:31:00	3.700
9	16/12/2006	17:32:00	3.668
10	16/12/2006	17:33:00	3.662
11	16/12/2006	17:34:00	4.448
12	16/12/2006	17:35:00	5.412

The first question to ask is how do we measure success? We do this via a loss function, where we try to minimize the loss function. There are several loss functions, and they are different pros and cons.

1. MAE (Mean absolute error) — here all errors, big and small, are treated equally.
2. Root Mean Square Error (RMSE) — this penalizes large errors due to the squared term. For example, with errors [0.5, 0.5] and [0.1, 0.9], MSE for both will be 0.5 while RMSE is 0.5 and. 0.45.
3. MAPE (Mean Absolute Percentage Error) — Since #1 and #2 depend on the value range of the target variable, they cannot be compared across data sets. In contrast, MAPE is a percentage, hence relative. It is like accuracy in a classification problem, where everyone knows 99% accuracy is pretty good.
4. RMSEP (Root Mean Square Percentage Error) — This is a hybrid between #2 and #3.
5. Almost correct Predictions Error rate (AC_errorRate)—percentage of predictions that is within %p percentage of the true value

If we are trying to forecast the next value, we have several choices.

ARIMA Model

The gold standard for this kind of problems is ARIMA model. The core idea behind ARIMA is to break the time series into different components such as trend component, seasonality component etc and carefully estimate a model for each component. See Using R for Time Series Analysis for a good overview.

However, ARIMA has an unfortunate problem. It needs an expert (a good statistics degree or a grad student) to calibrate the model parameters. If you want to do multivariate ARIMA, that is to factor in multiple fields, then things get even harder.

However, R has a function called `auto.arima`, which estimates model parameters for you. I tried that out.

```
1 library("forecast")
2 ....
3 x_train <- train data set
```

```
4 X-test <- test data set
5 ..
6 powerTs <- ts(x_train, frequency=525600, start=c(2006,503604))
7 arimaModel <- auto.arima(powerTs)
8 powerforecast <- forecast.Arima(arimaModel, h=length(x_test))
9 accuracy(powerforecast)
```

You can find detail discussion on how to do ARIMA from the links given above. I only used 200k from the data set as our focus is mid-size data sets. It gave a MAPE of 19.5.

Temporal Features

The second approach is to come up with a list of features that captures the temporal aspects so that the auto correlation information is not lost. For example, Stock market technical analysis uses features built using moving averages. In the simple case, an analyst will track 7 days and 21 days moving averages and take decisions based on cross-over points between those values.

Following are some feature ideas

1. collection of moving averages/ medians(e.g. 7, 14, 30, 90 day)
2. Time since certain event
3. Time between two events
4. Mathematical measures such as Entropy, Z-scores etc.
5. $X(t)$ raised to functions such as $\text{power}(X(t),n)$, $\cos((X(t)/k))$ etc

Common trick people use is to apply those features with techniques like Random Forest and Gradient Boosting, that can provide the relative feature importance. We can use that data to keep good features and drop ineffective features.

I will not dwell too much time on this topic. However, with some hard work, this method have shown to give very good results. For example, most competitions are won using this method (e.g. <http://blog.kaggle.com/2016/02/03/rossmann-store-sales-winners-interview-2nd-place-nima-shahbazi/>).

The down side, however, is crafting features is a black art. It takes lots of work and experience to craft the features.

Rolling Windows-based Regression

Now we got to the interesting part. It seems there is an another method that gives pretty good results without lots of hand holding.

Idea is to to predict $X(t+1)$, next value in a time series, we feed not only $X(t)$, but $X(t-1)$, $X(t-2)$ etc to the model. A similar idea has being discussed in Rolling Analysis of Time Series although it is used to solve a different problem.

Let's look at an example. Let's say that we need to predict $X(t+1)$ given $X(t)$. Then the source and target variables will look like the following:

$X(t)$	$X(t+1)$
1	2
2	3
3	4
4	5

Data set would look like the following after transformed with rolling window of three:

$X(t-2)$	$X(t-1)$	$X(t)$	$X(t+1)$
1	2	3	4
2	3	4	5
3	4	5	

Then, we will use above transformed data set with a well-known regression algorithm such as linear regression and Random Forest Regression. The expectation is that the regression algorithm will figure out the autocorrelation coefficients from $X(t-2)$ to $X(t)$.

For example, with the above data set, applying Linear regression on the transformed data set using a rolling window of 14 data points provided following results. Here AC_errorRate considers forecast to be correct if it is within 10% of the actual value.

1 LR AC_errorRate=44.0 RMSEP=29.4632 MAPE=13.3814 RMSE=0.261307

This is pretty interesting as this beats the auto ARIMA right way (MAPE 0.19 vs 0.13 with rolling windows).

So we only tried Linear regression so far. Then I tried out several other methods, and results are given below.

Data set Household power, 200k			
Algorithm	AC_errorRate	RMSEP	MAPE
Linear Regression	44	29	13
Random Forest Regression	26	37	13
Gradient Boosting Regression	33	37	15
Deep Learning	19	28	9

Linear regression still does pretty well, however, it is weak on keeping the error rate within 10%. Deep learning is better on that aspect, however, took some serious tuning. Please note that tests are done with 200k data points as my main focus is on small data sets.

I got the best results from a Neural network with 2 hidden layers of size 20 units in each layer with zero dropout or regularization, activation function "relu", and optimizer Adam(lr=0.001) running for 500 epochs. The network is implemented with Keras. While tuning, I found articles [1] and [2] pretty useful.

Then I tried out the same idea with few more datasets.

1. Milk production Data set (small < 200 data points)

2. Bike sharing Data set (about 18,000 data points)
3. USD to Euro Exchange rate (about 6500 data points)
4. Apple Stocks Prices (about 13000 data points)

Forecasts are done as univariate time series. That is we only consider time stamps and the value we are forecasting. Any missing value is imputed using padding (using most recent value). For all tests, we used a window of size 14 for as the rolling window.

The following tables shows the results. Here except for Auto.Arima, other methods using a rolling window based data set:

Dataset	MAPE				
	Linear Regression	Random Forest Regression	Gradient Boosting Regression	Auto Arima	Neural Networks
Dataset milk_production (180)	1.01	2.00	1.74	0.69	1.47
Dataset bikesharing (18k)	175.68	33.27	60.33	65.71	37.35
exchangeRate Data Set (6.5k)	0.34	0.43	0.41	0.34	0.42
appleStocks (13k)	1.65	2.05	2.49	1.63	2.44

There is no clear winner. However, rolling window method we discussed coupled with a regression algorithm seems to work pretty well.

Conclusion

We discussed three methods: ARIMA, Using Features to represent time effects, and Rolling windows to do time series next value forecasts with medium size data sets.

Among the three, the third method provides good results comparable with auto ARIMA model although it needs minimal hand holding by the end user.

Hence we believe that “Rolling Window based Regression” is a useful addition to the forecaster’s bag of tricks!

However, this does not discredit ARIMA, as with expert tuning, it will do much better. At the same time, with hand-crafted features methods two and three will also do better.

One crucial consideration is picking the size of the window for rolling window method. Often we can get a good idea from the domain. Users can also do a parameter search on the window size.

The following are few things that need further exploration:

- **Can we use RNN and CNN? I tried RNN, but could not get good results so far.**
- **It might be useful to feed other features such as time of day, day of the week, and also moving averages of different time windows.**

Hope this was useful. If you enjoyed this post you might also like Stream Processing 101: From SQL to Streaming SQL and Patterns for Streaming Realtime Analytics.

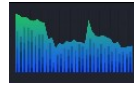
I write at <https://medium.com/@srinathperera>.

Hortonworks Community Connection (HCC) is an online collaboration destination for developers, DevOps, customers and partners to get answers to questions, collaborate on technical articles and share code examples from GitHub. Join the discussion.

Like This Article? Read More From DZone



Creating Your First Time Series Model With the Dashboard



Behind the Scenes of BigML's Time Series Forecasting




Time Series Analysis: Developing the Intuition



Free DZone Refcard Data Warehousing

Topics: TIME SERIES , ROLLING WINDOW REGRESSION , EXAMPLE , APPROACH , TESTING , PREDICTIONS

Published at DZone with permission of Srinath Perera , DZone MVB. [See the original article here.](#) 
Opinions expressed by DZone contributors are their own.

Big Data Partner Resources

White Paper: 6 Simple Steps for Replatforming in the Age of the Data Lake

StreamSets

|

Book: Simplify Apache Kafka

StreamSets

|

Whitepaper: Scaling Capabilities to Build a Homogeneous Big Data Ecosystem

HPCC Systems


|

White Paper: 12 Best Practices for Modern Data Ingestion

StreamSets

|

What Is Data Quality?

by Garrett Alley  MVB · Nov 01, 18 · Big Data Zone · Analysis

The open source HPCC Systems platform is a proven, easy to use solution for managing data at scale. Visit our Easy Guide to learn more about this completely free platform, test drive some code in the online Playground, and get started today.

Data is everywhere. As the volume, sources, and velocity of data creation increase, businesses are grappling with the reality of figuring out what to do with it all and how to do it. And if your business hasn't determined the most effective way to use its own data, then you're missing out on critical opportunities to transform your business and gain a decisive advantage.

Of course, without good data, it's a heck of a lot harder to do what you want to do. Whether you're launching a new product or service, or simply responding to the moves of your biggest competitor, making smart, timely business decisions depends almost entirely on the quality of data you have at hand.

People try to describe data quality using terms like *complete*, *accurate*, *accessible*, and *de-duped*. And while each of these words describes a specific element of data quality, the larger concept of data quality is really about whether or not that data fulfills the purpose or purposes you want to use it for.

Why Data Quality Is So Tough

Nearly 85% of CEOs say they're worried about the quality of data they're using to make decisions. Part of that fear comes from the fact that poor data has proven to cost companies up to 25% of their annual revenue in lost sales, lost productivity, or bad decisions.

Clearly, achieving data quality is still a challenge for many organizations, but the solutions are not as illusory as they seem. Most businesses experience some or all of these problems that directly impact the quality of their data:

- **Isolated data.** Otherwise known as "data silos," these separate data groups are either owned by a particular business unit or contained within a particular piece of software. The problem with siloed data is that it's inaccessible to the rest of the organization because the software may not be compatible with anything else or the business unit tightly controls user permissions. And while the data may offer useful, even extraordinarily valuable insight, since it can't be easily accessed, the business can't form a complete picture of it, let alone benefit from it.
- **Outdated data.** Enterprise structures are large and complex with multiple teams and departments. Thus, gathering data across the organization is often a slow and laborious process. By the time all the data is collected, some — if not most — of it has already fallen behind in relevance, therefore greatly reducing its value to the organization.
- **Complex data.** Data comes from many different sources and in many different forms. Data is generated from smartphones, laptops, websites, customer service interactions, sales and marketing, databases, and more. And it can be structured or unstructured. Making sense of the volume and variety of data coming in and standardizing it for everyone to use is a resource-intensive process many organizations don't have the bandwidth or expertise to keep up with.

How to Achieve Quality Data

Like any worthwhile business endeavor, improving the quality and utility of your data is a multi-step, multi-method process. Here's how:

1 Method 1: Big data scripting takes a large volume of data and uses a scripting language that can

1. **Method 1:** Big data scripting takes a large volume of data and uses a scripting language that can communicate and combine with other existing languages to clean and process the data for analysis. While engineers appreciate the agility of scripting, it does require a significant understanding of the types of data that need to be synthesized and the specific contexts in which the data exists to know which scripting language to use. Errors in judgment and execution can trip up the whole process.
2. **Method 2:** Traditional ETL (extract, load, transform) tools integrate data from various sources and load it into a data warehouse where it's then prepped for analysis. But it usually requires a team of skilled, in-house data scientists to manually scrub the data first in order to address any incompatibilities with schemas and formats that exist between the sources and the destination. Even less convenient is that these tools often process in batches instead of in real-time. Traditional ETL requires the type of infrastructure, on-site expertise, and time commitment that few organizations want to invest in.
3. **Method 3:** Open source tools offer data quality services like de-duping, standardizing, enrichment, and real-time cleansing along with quick signup and a lower cost than other solutions. However, most open source tools still require some level of customization before any real benefit is realized. Support may be limited for getting the services up and running, which means organizations once again have to fall back on their existing IT team to make it work.
4. **Method 4:** Modern data integration removes the manual work of traditional ETL tools by automatically integrating, cleaning, and transforming data before storing it in a data warehouse or data lake. The organization defines the data types and destinations and can enrich the data stream as needed with, for example, updated customer details, IP geolocation data, or other information. The transformation process standardizes data from all sources and in all formats to make it usable to anyone in the organization. And because it processes data in real time, users can check the data stream and correct any errors as they're happening.

Managing data at scale doesn't have to be hard. Find out how the completely free, open source HPCC Systems platform makes it easier to update, easier to program, easier to integrate data, and easier to manage clusters. Download and get started today.

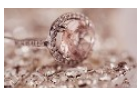
Like This Article? Read More From DZone



Why Data Quality Should Be the Red Thread of your Data Strategy [Interview]



When We Talk Data Quality, We Need to Talk About Waste



It's Time to End Bad Data. Here's How Data Quality Can Help.




Free DZone Refcard Data Warehousing

Topics: BIG DATA, DATA QUALITY, BIG DATA TOOLS, BIG DATA SCRIPTING

11/7/2018

Rolling Window Regression: A Simple Approach for Time Series Next Value Predictions - DZone Big Data

Published at DZone with permission of Garrett Riley, DZone MVP. [See the original article here.](#) 

Opinions expressed by DZone contributors are their own.