TIME SERIES FORECASTING – TAKING KAGGLE ROSSMANN CHALLENGE AS EXAMPLE

January 9, 2016 (http://blog.bikashagrawal.com.np/2016/01/09/time-series-forecasting-taking-kaggle-rossmann-challenge-example/) admin (http://blog.bikashagrawal.com.np/author/admin/) Installation (http://blog.bikashagrawal.com.np/category/installation/)

A time series is a sequence of data points, typically consisting of successive measurements made over a time interval. Forecasting is the use of a model to predict future based on past informations. This problem is a bit different to what most known as the pure cross-sectional problem.

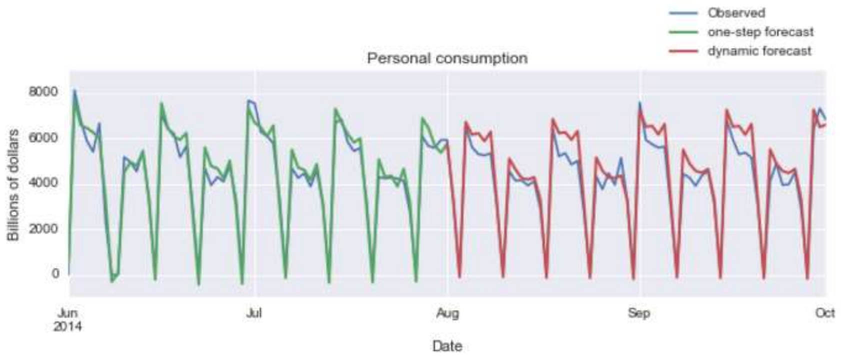
In this post, I take the recent Kaggle challenge as example, sharing the finding and tricks I used. The competition – Rossmann Store Sales – attracted 3,738 data scientists, making it the second most popular competition by participants ever. Rossmann is a drug store giant operates over 3,000 stores in European, who challenged Kagglers to forecast 6 weeks of daily sales for 1,115 stores. The data (https://www.kaggle.com/c/rossmann-store-sales/data) is mainly comprised of store index, store type, competitor store information, holiday event, promotion event, whether store open, customers, and the sales which is what we're tasked to predict.

Doing time series forecasting, a few things specific to time-series you need to know about are

1. time-dependent features.
2. validating by time.

I will walk thorough them latter.

First I recommend you learn ARIMA, that would help you learn traditional ways tackling time series and those knowledges also useful to modern models. The classic ARIMA is a combination of Autoregression, difference of lag values, and Moving Average Model. They respectively accounts for response dynamics, non-stationary of the series, the noise dynamics. And what making it distinct from simple regression model is that they are capable to learn dynamic pattern of values along the time.
I use the great Python package `statsmodle` , a package for statistics, which get Seasonal ARIMA upgrade at version 0.7 that make Python on par with R at time series. ARIMA has many parameters to tune, the way I am used to is first looking at the time series, partial autocorrelation and difference of time series, from those visualization I get a prior belief of the range the parameters should likely to be and put them into model selection to automate parameters selection. For more thorough introduction to ARIMA, I recommend Rob J. Hyndman's book – *Forecasting: principles and practice*[1] (http://www.elasticmining.com/post/2016-01-02/time-series-forecasting-kaggle.html#fn:1).



The prediction of ARIMA to store 6.

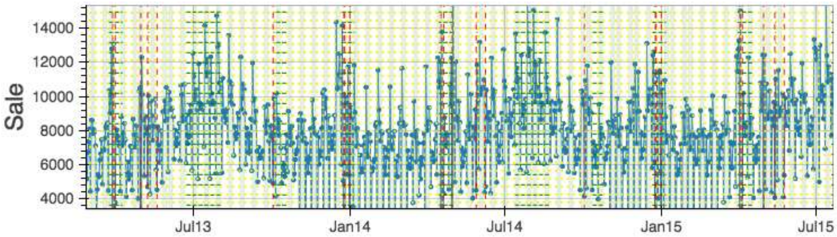
Though the ARIMA has captured the seasonal effect, for reason of a linear model it is not good at accuracy. That is not to say it is useless, it still useful as a base model when estimating prediction interval or ensembling. So I put it aside and turn the attention to Gradient Boosting Tree (GBT). In the GBT model I do a lot of feature engineering, including scraping the external dataset. The scraped and derived features is so many that I must choose a good validation set to avoid overfitting. Finally I do a simple aggregation to improve the leaderboard one step further. I list each steps as follows:

FEATURES

What most important features to time series are calendar effect, weather and past values. In the sales data the important calendar effect is holiday information, that is so much important that I have to scrape a more detailed holiday data instead of using the default provided by the host of competition. Because this competition permits the use of publicly available external data , the only limit is your creativity. The following is the external data I used : state of store (https://www.kaggle.com/c/rossmann-store-sales/forums/t/17048/putting-stores-on-the-map), weather information by state (https://www.kaggle.com/c/rossmann-store-sales/forums/t/17058/weather-at-berlin-us-airport/97075#post97075), and a more detailed holiday information scraped from internet. After feature engineering I ended with a lot of features, around 400. Many of which are the same with winner's features[2] (http://www.elasticmining.com/post/2016-01-02/time-series-forecasting-kaggle.html#fn:3). I do feature selection to eliminate 200 non-relevant features that spares my model from contamination by those noises.

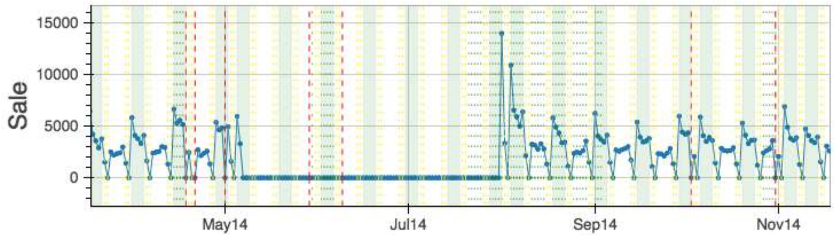
Some features that I feel important but others didn't mentioned or used are:

Sinusoid features[3] (http://www.elasticmining.com/post/2016-01-02/time-series-forecasting-kaggle.html#fn:2): $\sin(2\pi kt/m)\sin(2\pi kt/m)$ and $\cos(2\pi kt/m)\cos(2\pi kt/m)$ for multiple frequency kk and period mm.
After adding it we can make periodic https://www.viagrasansordonnancefr.com/viagra-cialis/ (https://www.viagrasansordonnancefr.com/viagra-cialis/) pattern more notable to the model. That is important to some of stores, for example:



this store has salient periodic pattern.

Past event features: statistics about last weeks.
this one is inspired from a visualization of time series, where I spot the refurbishment effect – before or after refurbishment it has abnormal sales, seems like a closing-down sales. Given this I add a feature about whether refurbishment happens or not and how long the time has passed since that event.



The long state of zero sales is in fact a refurbishment. The surge of sale after refurbishment seems like a special sale event for reopening.

VALIDATION

A good validation strategy keep us from overfitting and let us know how much we have improved. The key difference I adopt is splitting validate set by time – a split is of a 2 month interval. Totally I use a selected weighted average of 3 sets for validation. Because that simulate how the test set is generated, it reflect of true score and let us push the limit without overfitting. I use it everywhere, including early stopping methods when training model. If you don't do this you would scoring bad at private leaderboard. Taking those highest-public-score-scrips as example, they shack down to the extent of 200 ranks at private board.

AGGREGATION

I do a simple ensemble learning – aggregation of 2 GBT by different seeds, and a attribute bagging[4] (http://www.elasticmining.com/post/2016-01-02/time-series-forecasting-kaggle.html#fn:4) of a GBT where the features is selected from the top half important features of a Random Forest. Aggregation of 3 models help me advance near 40 ladders at private board.

I got the 55th/3303, around 1.6% at this competition, thanks to ElasticMining colleague's suggestions and inspirations. We deliver tailored data solution, experience and knowledge helped me a lot in this competition. This type of competition is practical meaningful. Currently Rossmann's managers are tasked with predicting their daily sales that are not of constant quality. Some companies use a standard tool that is not flexible enough to suit their needs. The specific solution and reliable automatic sales forecasts can enable store managers to create effective staff schedules that increase productivity and motivation. If your company wants the tailored data solution, contact us now.

1. R. J. Hyndman and G. Athanasopoulos, Forecasting: principles and practice, 2014 (https://www.otexts.org/book/fpp) ↩ (http://www.elasticmining.com/post/2016-01-02/time-series-forecasting-kaggle.html#fnref:1)
2. winner's post (https://www.kaggle.com/c/rossmann-store-sales/forums/t/17896/share-your-solution/101318#post101318) ↩ (http://www.elasticmining.com/post/2016-01-02/time-series-forecasting-kaggle.html#fnref:3)
3. R. J. Hyndman, Forecasting with long seasonal periods (http://robjhyndman.com/hyndsight/longseasonality/) ↩ (http://www.elasticmining.com/post/2016-01-02/time-series-forecasting-kaggle.html#fnref:2)
4. wiki (https://en.wikipedia.org/wiki/Random_subspace_method) ↩ (http://www.elasticmining.com/post/2016-01-02/time-series-forecasting-kaggle.html#fnref:4)

kaggle (http://blog.bikashagrawal.com.np/tag/kaggle/) prediction (http://blog.bikashagrawal.com.np/tag/prediction/)

rossman (http://blog.bikashagrawal.com.np/tag/rossman/) time-series (http://blog.bikashagrawal.com.np/tag/time-series/)

LEAVE A REPLY

Your email address will not be published. Required fields are marked *

Name *

Email *

Website

Comment

POST COMMENT

‹ Getting Started with Markov Chains (http://blog.bikashagrawal.com.np/2016/01/07/getting-started-with-markov-chains/)

Accurately Measuring Model Prediction Error › (http://blog.bikashagrawal.com.np/2016/01/24/accurately-measuring-model-prediction-error/)

Search...

Bikash Agrawal