Rossmann store sales quantity prediction

cs229 course project

Vladimir Sazontyev, Stanford University

wolod@stanford.edu

Abstract

My project aimed to solve machine learning problem for Rossmann inc. I use Rossmann dataset that contains data since 2013 year till 2015. Rossmann is challenges to predict 6 weeks of daily sales for 1,115 stores located across Germany. End-to-end workflow of data science was applied in the project, from data retrieval, data processing, statistical modelling, machine learning, and data visualization and data analysis. In my project I implement various machine learning algorithms, with focus on showing the evolution of machine learning algorithms, that I use to solve the problem.

Introduction

Motivation

Rossmann store managers are tasked with predicting their daily sales for up to six weeks in advance. Store sales are influenced by many factors, including promotions, competition, school and state holidays, seasonality, and locality. With thousands of individual managers predicting sales based on their unique circumstances, the accuracy of results can be quite varied.

Problem definition

We are given dataset from Rossman inc. They ask us to write an algorithm that will predict quantity of sales in each of next 48 days. The evaluation metric for this problem is RSMPE. The RSMPE calculated as follows:

$$\mathrm{RMSPE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{y_i}\right)^2}$$

,where y_i - denotes the sales of a single store on a single day and $\hat{y}_i$ - denotes the corresponding prediction. Any day and store with 0 sales is ignored in scoring. We challenged to predict continuous variable. To be explicit, we should predict field Sales.

Related Work

As a related work I found useful article by prof D'yakonov [1]. In his article he suggested to use different weighting schemes and probability of coming to store in specific day of week, this is a very clever approach, but unfortunately this approach can't be applied directly to our problem, but I made a try in my project, - unfortunately it doesn't improve my algorithm at all.
Also I was impressed by Havard Rue and Oyvind Salvesen [2] who used a Bayesian linear model to predict soccer results. They used a time-dependent model using relative strength of attack and defense of each team.
I found interesting work by Pierre Geurts [3] his approach helped me how to move to dicision trees, but unfortunately his techniques are hard to implement in short time, and they highly depends of timeseries that you really have, and as I see it won't improve more than I can extract from dates analytically.
One more notable book by Ian H. Witten, Eibe Frank [4] especially interesting chapters was about decision trees, this helped me to get into xgboost approach, as well as paper by Jerome H. Friedman [5] related to this library, it is very deep and good explanation how approximation accuracy of gradient boosting can be substantially improved by incorporating randomization into the procedure; and also explains based on what xgboost library works so well.

Dataset

Dataset is provided by Rossman. List of available columns presented in Table 1.

Train	Store	Test
Store	Store	Id
DayOfWeek	StoreType	Store
Sales	Assortment	DayOfWeek
Customers	CompetitionDistance	Date
Open	CompetitionOpenSinceMonth	Open
Promo	Promo2	Promo
StateHoliday	Promo2SinceWeek	StateHoliday
SchoolHoliday	Promo2SinceYear	SchoolHoliday
	PromoInternal	

Table 1. Columns by files of given dataset

Data visualization and analysis

I start my analysis with Table 2 that shows amount of unique values, nans, and unique values if there less than 10 of them.

	Amount of unique values		Unique values		NaNs	
Field name	**Training set**	**Test set**	**Training set**	**Test set**	**Training**	**Test**
Store	1115	856			0	0
DayOfWeek	7	7	5 4 3 2 1 7 6	4 3 2 1 7 6 5	0	0
Date	942	48			0	0
Sales	21734	-			0	0
Customers	4086	-			0	0
Open	2	2	1 0	1 nan 0	0	11
Promo	2	2	1 0	1 0	0	0
StateHoliday	5	2	'0' 'a' 'b' 'c' 0L	'0' 'a'	0	0
SchoolHoliday	2	2	1 0	1 0	0	0

Table 2. Simple data information

From this table we see,that there 11 NaNs in "Open" on test; also in test there only 0,'a' value in "StateHoliday" field(it's different from what we see in training set). On the next figure 1 we see distribution among all sales (left part of figure 1). I took logarithm of it and it become even more bell-shaped (right part of figure 1).

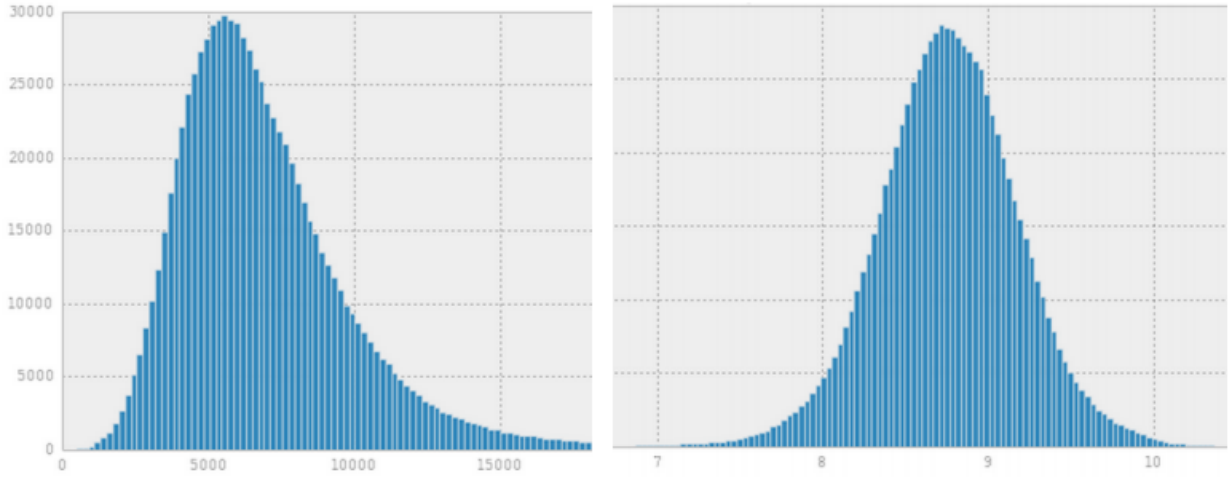

Figure 1. Distribution among all sales (left side) ; logarithm of it (right side)

Also from Table 1, we clearly see, that we have field Sales and Customers, that are presented in Train.csv, but not presented in Test.csv . Sales are not presented since we should predict the values of this field for whole this Test.csv. Customers are not presented since they highly correlate with Sales field. I plotted graph date days in training set by week and over all stores and years.

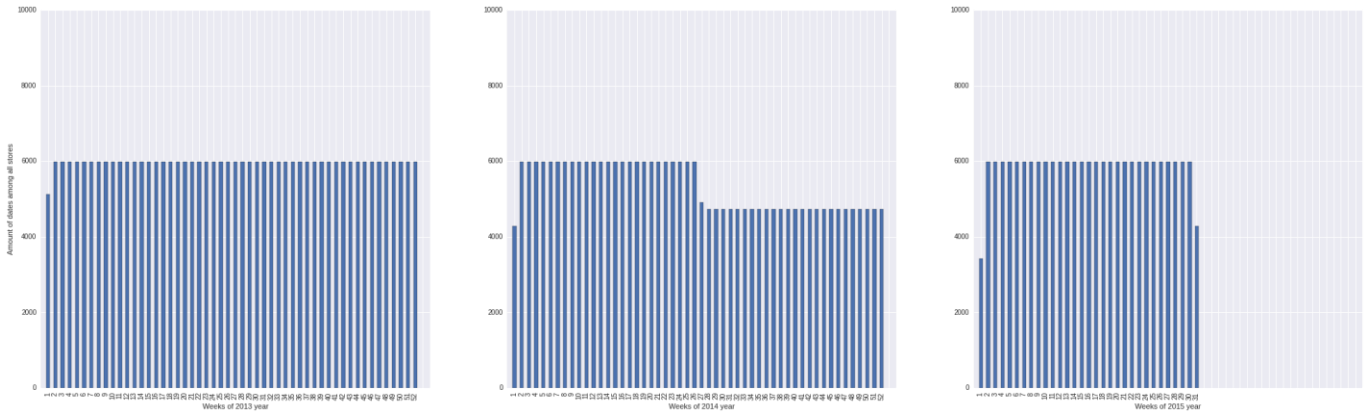

Figure 2. Amount of days by weeks and by years among all stores.

From figure 2 we clearly see we see, that in the second year (2014) there is a clearly less days registered since 27-th week and amount of days restored since new year(2015). It happens, because statistics for 180 store ids are missing since 27-th week up to 52 week. But they present in test set (we should predict for them). One more point, that I found, that last 5 days of month greatly changes the behavior of sales curve.

Machine learning algorithms and approaches

In my work I tried to follow prof. Andrew Ng workflow [6]. Unfortunately, his approach to analyze learning curve, is not clearly useful, because we deal with timeseries date, and older data not always train our model better, and we can say that old data is equally useful as recent one.

I started with linear regression. Because it is a really simplest and fasted (in terms of implementation time) model. I used this model as a baseline. Just to comment algorithm - this is an approach for modeling the relationship between a scalar dependent variable y and one or more explanatory variables (or independent variables).

Next model - is a locally constant solution, this was not taught during this course, but I think this is a good model compared to our baseline, because it is obviously separate our decision space better than our baseline. By separating I mean, that you separate space and assign some fixed value. As a fixed value I choose simple mean. I made experiments with median, but it worked worse. Also I tried to multiply fixed value by constant, but since this is not a final model, I left that idea.

Third model, is based on previous one, I just substituted fixed value - mean by group, with linear regression by group.

One more step that I tried to do, is to incorporate knowledge of related works, especially a lot of effort was done to implement weighting schemes that was proposed by prof. D'yakonov's paper [1]. Those schemes are:

$$w_i^N = \left(\frac{d-i+1}{d}\right)^{\delta} \quad \delta \in [0,+\infty) \qquad w_i^N = \lambda^i \quad \lambda \in (0,1] \qquad w_i^N = \frac{1}{i^{\gamma}}, \quad \gamma \in [0,+\infty)$$

Unfortunately, no one of them even barely decreased my RMSPE, as well as trying to use month/year as feature. (To make sure that I implemented it correctly, I tried this on toy problem, and it worked, but here it doesn't).

And the final model is based on random forests - xgboost [7] - this is a library that is designed, and optimized for boosted (tree) algorithms. It simply choose features and build decision trees on that features, and after that it averages the result. Also - I must mention that we can increase number of trees to as much as we want and make it as large as we want, only computational time really matters in this case. Figure 3 perfectly describes the decision tree in case of regression problem.

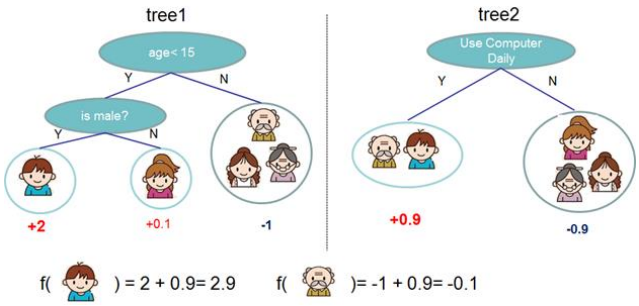

Figure 3. Decision tree of xgboost library for regression problem

And final thing that I used - I tried ensemble different algorithms. Even if we train only models based on xgboost, it turns out that if we ensemble different number of trees (iterations) it used to give better results. One more approach that I used is based on bell-shaped distribution of number of sales - is to take logarithm of number of sales, then use exponent of predictions as solutions.

I tried to justify based only on learning curve, but best justification to move in that direction is that each generation of models that I used is simply separate better our space, and from this point of view turns our better for us.

Machine learning algorithms and approaches; and findings

The first step was to implement very simple model, just to have a baseline.I split my training set on two parts. First part for training - it contains all dates from very first date 2013-01-01 till 2015-06-12. And second part for test training - since 2015-06-13 till 2015-07-31. The intuition behind these dates, that for local evaluation I used same exact amount of days, as we asked to

predict in problem. I can't truncate 70% of training set and set this as training set, and 30% as test set, since this problem involves timeseries. And old data might produce non-relevant prediction.

I tried very simple model, as a baseline - sklearn.linear_model.LinearRegression() [8]. Without any regularization, since we clearly do not overfit our data. As features I used: Store and DayOfWeek. I got 0.4761 on train set and 0.4170 on test set (based on RMSPE) on local cv evaluation.

From figure 4, we see that obviously, such simple model can't represent good our data, since it lack of features and what is really meaningful that it separates space badly. (see prev. paragraph).

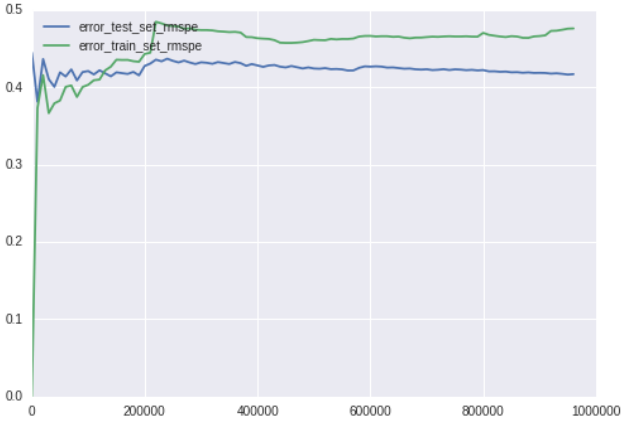

Figure 4.Learning curve for the first step model. Training and test set error. x-axis - amount of training examples; y-axis RMSPE error.

The second step that I made was to take mean of sales grouped by Store and DayOfWeek, and report that mean as a prediction that gave me on my local cv evaluation - 0.2199 (RMSPE). The intuition behind it, that we made more complex model, with same features. The learning curve is on figure 5(left side).

When I added Promo ; SchoolHoliday and Open feature in this model my model showed me on my local cv evaluation 0.14221 and on public leader board 0.13693. Also I tried to incorporate my finding, that last 5 days significantly changes curve of sales amount. On figure 5 (right side) - you can see learning curve. Where 1 means - use 1 month as training (2015-05-12 -2015-06-12), 2 - use 2 month for training (2015-04-12 - 2015-06-12). The greater value means we use more older data. All other learning curves are given based on same scheme.

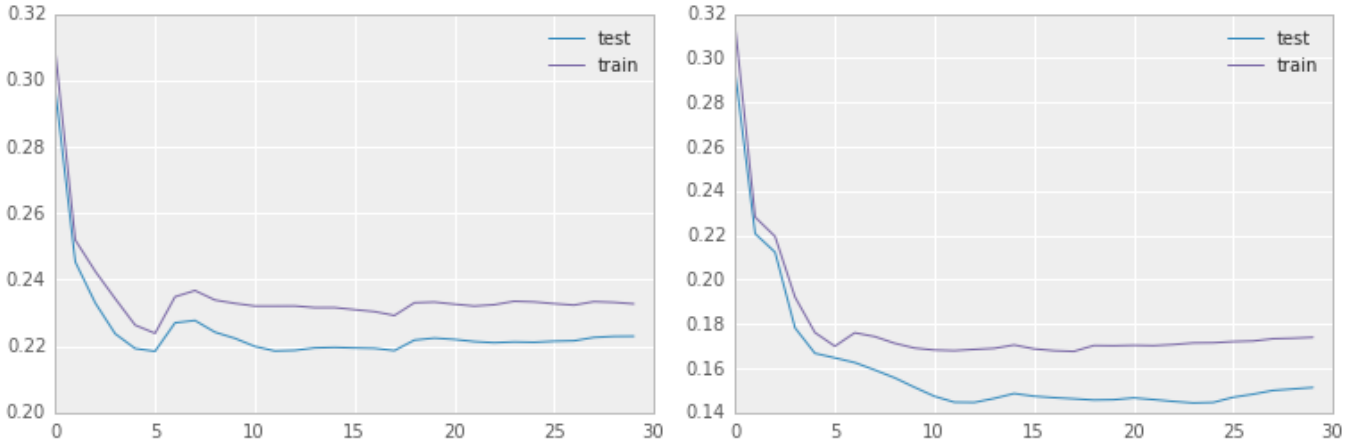

Figure 6.(left side) Learning curve for second step. Right side - same step with 3 new features. Training and test set error. x-axis - month that used on training; y-axis RMSPE error

The third step - I substituted simple mean, with linear regression just like in first step, but now I have a lot of such linear regressions for each group separated by Store; DayOfWeek; Promo; SchoolHoliday; Open. This gave me on my local control, 0.125 , and I used last 5 month for training(2015-01-12 - 2015-06-12). Learning curve is provided on figure 7.

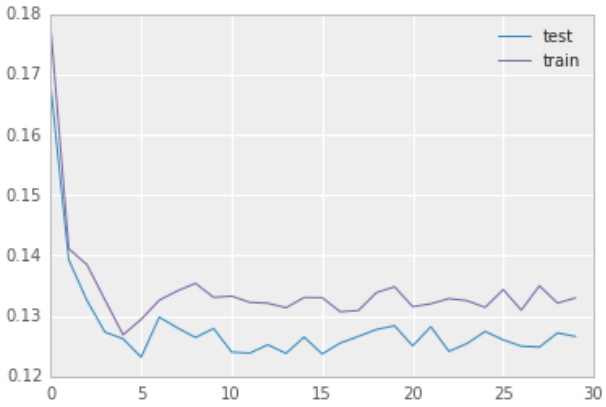

Figure 7.Learning curve of third step. Training and test set error. x-axis - month that used on training; y-axis RMSPE error

The forth step - I used random forests eXtreme gradient boosting - xgboost. This is a library that is designed, and optimized for boosted (tree) algorithms. The good explanation how it works is given on their page [7] - based on this explanation I see that it does pretty similar work as I did in the second and third steps, but it splits space even more. I used these features Store,CompetitionDistance,CompetitionOpenSinceMonth,CompetitionOpenSinceYear,Promo,Promo2,Promo2SinceWeek,Promo2SinceYear,SchoolHoliday,DayOfWeek,month,year,day,year,StoreType,Assortment that gave me 0.11520 on public leaderboard (number of iterations: 300).

As a **fifth experiment** I tried to mix the solutions, - ensemble - simply add them with some coefficient (obviously, coefficients summed up to one), I mixed models with different numbers of iterations (200,300,450,500) of trees and one model, where solutions where I took logarithm of solutions (and took exponent of prediction). But this approach worked only for this case, it did not worked with previous models.Also I added feature isInLast5Days (I made it to show it to help algorithm, to understand that it must react differently if data is in range of last 5 days of month). On my local control, this approach achieved 0.091, but on public leaderboard it might show greater value. Since this method take a lot of time , and I did not planed to improve after this method, I did not made a plot, because it will took days of computation.

Results:

Model	Local control	Public Leaderboard
Linear regression	0.4170	
Locally constant solution	0.2199	
Locally constant solution (with 3 more features)	0.14221	0.13693
Linear regressions by groups	0.125	
Xgboost random forests with 300 iterations		0.11520
Ensemble of xgboosts models	0.091	

Table 3. Results of algorithms

For the final mode l used same configuration for all xgboost models in ensemble (except number of trees):

"objective": "reg:linear", "eta": 0.29 ,"max_depth": 8, "subsample": 0.87, "colsample_bytree": 0.71 . I used random search to find these parameters. In simple grid search we go over fixed points in space and simply try our model. In grid search we only specify range in which parameters can vary and then chose points with random "coordinates" (by point I mean set of parameters; by coordinate I mean parameter that we try to find). This page Efficiently searching for optimal tuning parameters [9] explains the advantages of this approach.

Further works

As further work I suggest to use external weather data, and actually I tried to use it, and find that difference between temperature of current day and previous one significantly influential on curve of number of sales. If you add features like that to your model you probably we get even better score.

Contributions based on course

Since this course was a homework for other course. I must mention, that data visualization (to be more specific - the second part, where I discuss about lost data for 180 stores and that 5 last days matters) was done for that course and cs229; also xgboost (only forth step) was done for both courses. All other findings was done after homework submission - specifically for cs229.

References

[1] A. D'yakonov, Supermarkes clients behavior forecasting by weighted methods of probability and density estimations, Business Informatics№1(27)-2014, MSU , 2014

[2] H. Rue and O. Salvesen, Prediction and retrospective analysis of soccer matches in a league. Journal of the Royal Statistical Society: Series D (The Statistician) 49.3 (2000): 399-418

[3] P.Geurts, Pattern extraction for time series classification,Principles of Data Mining and Knowledge Discovery, Springer Berlin Heidelberg, 2001

[4] Ian H. Witten, Eibe Frank, Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems),Morgan Kaufmann Publishers Inc. San Francisco, CA, USA, 2005

[5] Jerome H. Friedman, "Stochastic gradient boosting." *Computational Statistics & Data Analysis* 38.4 (2002): 367-378.

[6] Andrew Y. Ng, Advice for applying Machine Learning, Stanford University, http://cs229.stanford.edu/materials/ML-advice.pdf

[7]Tianqi Chen, http://xgboost.readthedocs.org/en/latest/model.html, 2014

[8] Pedregosa et al.,Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830, 2011.

[9] Kevin Markham,http://blog.kaggle.com/2015/07/16/scikit-learn-video-8-efficiently-searching-for-optimal-tuning-parameters, 2015