Rossmann Store Sales Prediction

Tian Yang, Zhuyuan liu

Abstract

In this project, we applied machine learning techniques to a real-world problem of predicting stores sales. This kind of prediction enables store managers to create effective staff schedules that increase productivity and motivation. We used popular open source statistical programming language R. We used feature selection, model selection to improve our prediction result. In view of nature of our problem, Root Mean Square Error (RMSE) is used to measure the prediction accuracy.

Introduction

Rossmann is a chain drug store that operates in 7 European countries. We obtained Rossmann 1115 Germany stores' sales data from Kaggle.com. The goal of this project is to have reliable sales prediction for each store for up to six weeks in advance. The topic is chosen, because the problem is intuitive to understand. We have a well understanding of the problem from our daily life, which makes us more focused on training methodology.

The input to our algorithm includes many factors impacting sales, such as store type, date, promotion etc. The result is to predict 1115 stores' daily sale numbers. Generalized linear model (GLM) and Supporting vector machine (SVM) regression were used to train model and predict sales.

Dataset and Features

Training data is comprised of two parts. One part is historical daily sales data of each store from 01/01/2013 to 07/31/2015. This part of data has about 1 million entries. Data included multiple features that could impact sales. Table 1 describes all the fields in this training data.

Field Name	Description
Store	a unique Id for each store: integer number
DayofWeek	the date in a week: 1-7
Date	in format YYYY-MM-DD
Sales	the turnover for any given day: integer number (This is what to be predict)
Customers*	the number of customers on a given day: integer number (this is not a feature. Based on the test data from Kaggle, this feature is not included in test data)
Open	an indicator for whether the store was open: 0 = closed, 1 = open
Promo	indicates whether a store is running a promo on that day: 0 = no promo, 1 = promo
StateHoliday	indicates a state holiday. Normally all stores, with few exceptions, are closed on state holidays. Note that all schools are closed on public holidays and weekends. a = public holiday, b = Easter holiday, c = Christmas, 0 = None
SchoolHoliday	indicates if the (Store, Date) was affected by the closure of public schools: 1 = school holiday, 0 = not school holiday

Table 1: Historical sales data table features

The second part of training data is supplement store information. It has 1115 store info entries, which listed the store type, competitor and a different kind promotion info. Table 2 below describes all the field in this file.

Field Name	Description
Store	a unique Id for each store: integer number
StoreType	differentiates between 4 different store models: a, b, c, d
Assortment	describes an assortment level: a = basic, b = extra, c = extended
CompetitionDistance	distance in meters to the nearest competitor store
CompetitionOpenSinceMonth	gives the approximate year and month of the time the nearest competitor was opened
CompetitionOpenSinceYear	
Promo2	Promo2 is a continuing and consecutive promotion for some stores: 0 = store is not participating, 1 = store is participating
Promo2SinceWeek	describes the year and calendar week when the store started participating in Promo2
Promo2SinceYear	
Promointerval	describes the consecutive intervals Promo2 is started, naming the months the promotion is started anew. E.g. "Feb,May,Aug,Nov" means each round starts in February, May, August, November of any given year for that store

Table 2: Store Information data table features

We did several things to combine features and create features directly related to sales number. The work we did is:

1. The supplement store information can't be used directly. We merged store information and historical sales data. Story type and Assortment is merged into each entry of historical sales data
2. Combine Promo2, Promo2SinceWeek, Promo2SinceYear and Promointerval to a promotion 2 indicator in historical sales data. The indicator indicates on a certain day whether a certain store is on promotion 2.
3. Similarly, we combined CompetitionDistance, CompetitionOpenSinceMonth, CompetitionOpenSinceYear to a competitor indicator. The indicator indicates on a certain day whether a certain store has competitor.
4. Since CompetitionDistance is provided, we used CompetitionDistance to train model, instead of competitior indicator. For any date and any store which doesn't has competitor (competitor==1), we assign CompetitionDistance as a large number 100000. This method enable us to use only one CompetitionDistance feature. It also models the no competitor case by weakening CompetitionDistance impact.
5. Historical sales dataset has Date feature. We created Month and Year feature based on Date feature. Month and Year is used as feature, since they correlated with sales data.

The final training/test dataset used includes the following features.

- StoreID
- DayOfWeek
- Open
- StateHoliday
- Promo2 indicator
- StoreType

- Month
- Year

SchoolHoliday
Promo

Assortment
CompetitionDistance

Methodology

We start with GLM with poisson function to train data. Poisson function is known to be a good distribution model to model customer number and sales [6]. Before training poisson GLM model, we first did analysis on all the features available to choose the significant features and try to make sure all the features are relatively IID.

Feature Selection

The input features have many category features. Category feature should be treated as factor in training. Too many factors increase runtime a lot. We tried to combine similar category features as much as possible. We looked at feature distribution vs sales and combine features with similar distribution. Two examples are listed below.

DayofWeek feature:

Removing store close data and plotting Mon through Sun's sales data distribution. From the plot, we can tell that Tue through Fri's sales distributions are very close. Mon, Sat and Sun's sales distributions are unique. In database, DayofWeek is represented as numeric number 1-7. From intuitive, we know that there is no linear relationship from 1-7 number to sales data. We treat DayofWeek as four factors, Mon, Weekday(Tue-Fri), Sat, Sun.

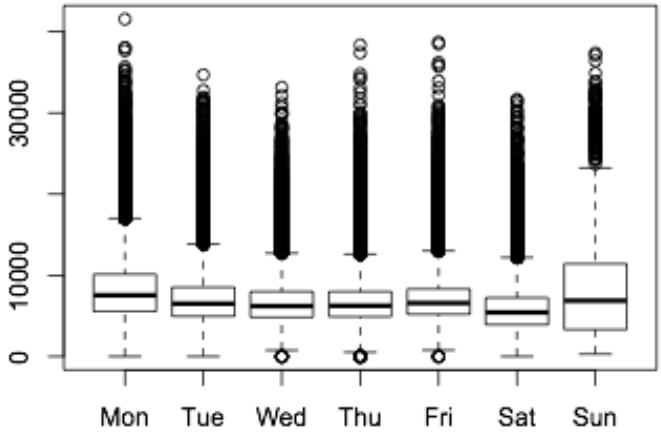


Figure 1: Sales distribution of each weekday

Holiday Features:

StateHoliday, SchoolHoliday and Open are highly correlated features. They don't meet IID assumption. Removing store close data and plotting state holiday sales distribution. State holiday a, b, c's sales distribution is not similar. However, state holiday == b only has 145 data points. state holiday == c only has 71 data points. Since the training data points are not large, we combined state holiday b and c as one category.

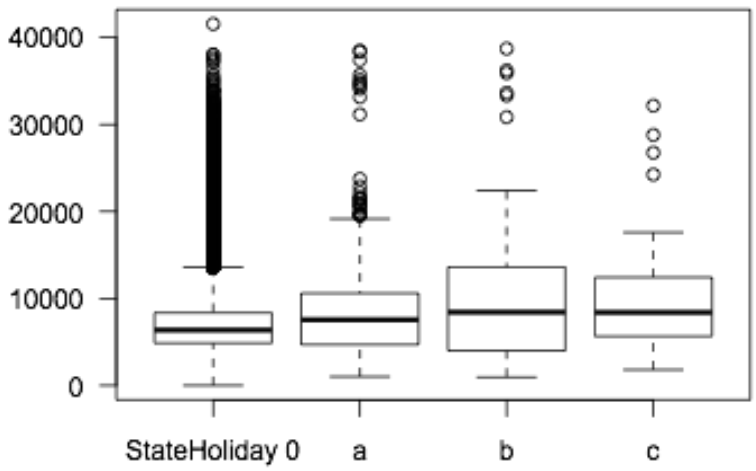


Figure 2: Sales distribution of each state holiday

CompetitionDistance Feature:

We did correlation calculation between sales data and different math transfers of CompetitionDistance. We found that 1/sqrt(CompetitionDistance) has the best correlation value.

Result evaluation metrics:

RSME (root mean square error) method is used to evaluate the prediction quality. We took the percentage error of predicted sale data to real sale data and then calculated the standard deviation. Equation is shown below.

$$\epsilon = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(\frac{|PredSales^{(i)} - Sales^{(i)}|}{Sales^{(i)}}\right)^2}$$

Result and Analysis

GLM regression:

Feature exploration:

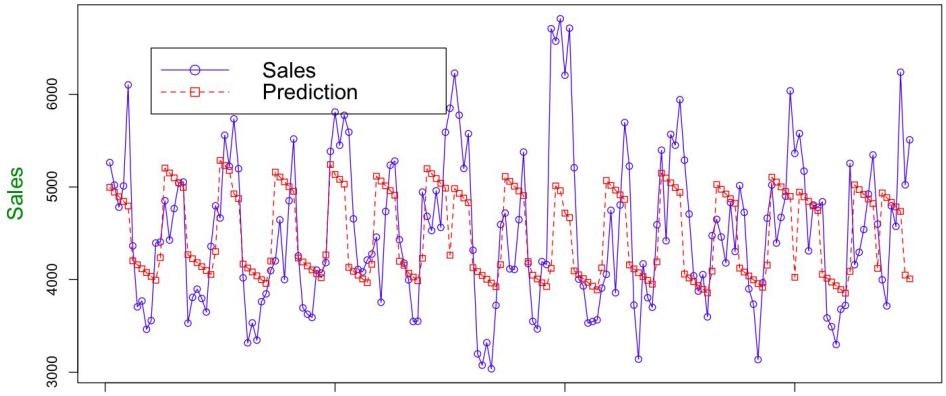
We started with taking the features as is. Using “DayOfWeek”, “Promo”, “SchoolHoliday”, “StateHoliday” as feature vector feed into GLM regression. Realizing that in Christmas season, the sales are much higher than what the model predicted, we added “WeekOfYear” feature to the model. By treating these as factors instead of numeric numbers, we can achieve better training and testing error. We also noticed that although most of the stores close on state holidays, the sales near state holidays are higher than normal. Therefore, we added flag on whether a date is before or after a holiday. With that, we lower the testing error further by 0.009.

	Training Error	Testing Error
a. Basic features	0.17891	0.18315

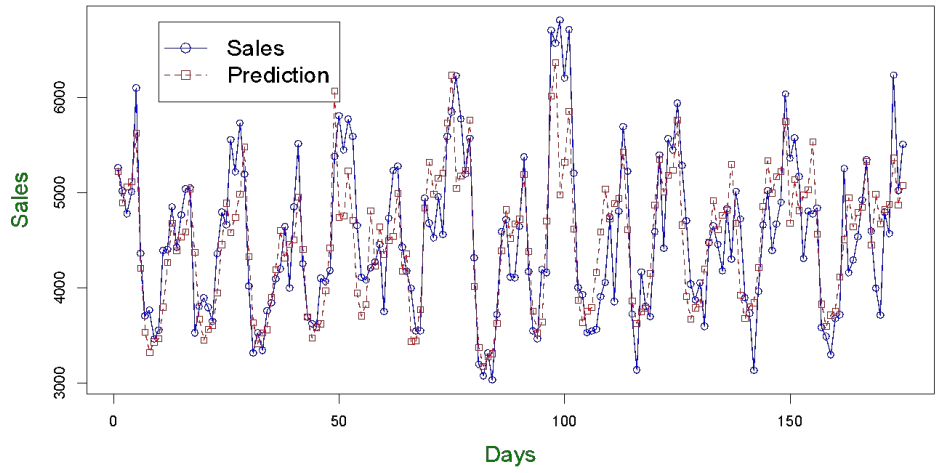
b. Treat “DayOfWeek” and “WeekOfYear” as factor	0.17747	0.14393
c. Adding “Year”, “WeekOfYear” as factor	0.12670	0.11033
d. Flag if before and after holidays	0.11433	0.10136

Table 3: training and testing error with different features added to GLM regression

Table 3 shows the results with using different sets of features. Results show that adding more related features gives better fitting and smaller testing error.



(a)



(b)

Figure 3: GLM testing result with and with out feature engineering (one store):
(a) Test result with basic features used as is; (b) Test result after adding more features that are related to realistic problem.

Figure 3 is an example fitting result from one store. Figure 4(a) is without feature engineering which has big difference between real sales number versus prediction, especially on peaks and dips. Figure 4(b) is the results after feature engineering plotted on the same test set, which shows much better fit between the real and prediction numbers.

Different GLM functions:

We explored GLM with different functions. Poisson function shows the best result on both training and testing. However, the difference is very small. We continued using Poisson function for the rest of our GLM experiments. Table 4 shows the training and testing error numbers with respect to different function used.

Function	Training Error	Testing Error
Poisson	0.11433	0.10136
Gaussian	0.12164	0.10569
Gamma	0.11996	0.10312
Quasi	0.12918	0.11377

Table 4: training and testing error with different functions used in GLM regression

SVM regression:

Kernels selection:

We explored SVM regression using e1071 package in R with different kernels. Results are shown in Table 5.

Function	Training Error	Testing Error
Radial	0.117639	0.1179943
Linear	0.1424787	0.1127987
Sigmoid	0.3016894	0.2089361
Polynomial	0.1300421	0.1207464

Table 5: training and testing error with different kernels used in SVM regression

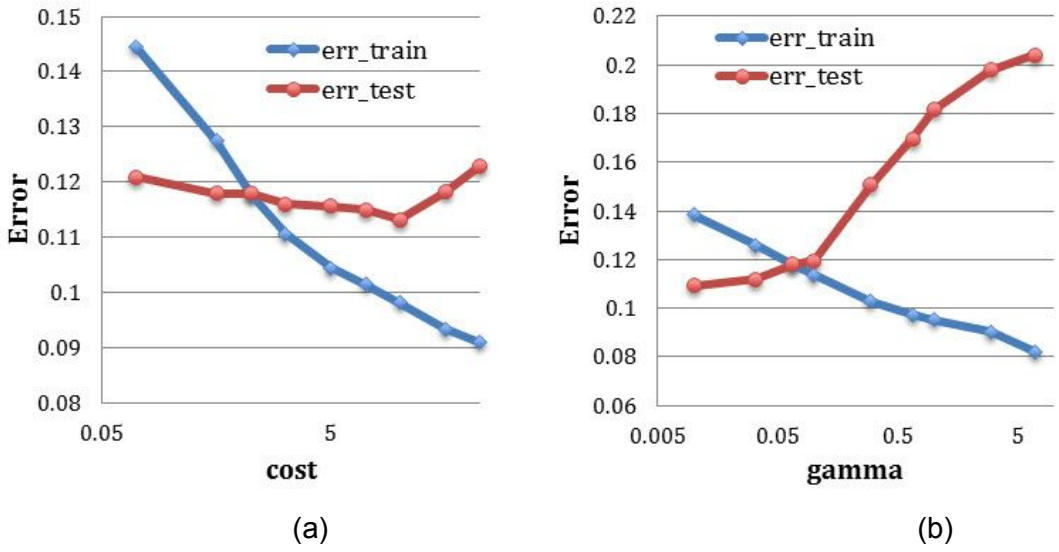
Radial kernel gives the best result in this problem. Sigmoid is not fit in this problem. Linear or Polynomial kernel is difficult to fit the training data very well without selecting a very big cost.

Cost and gamma:

The cost parameter rules the error of the cutting plane. Higher cost forces the creation of a more accurate model [5]. However, the creation of a more accurate model may not generalize well. A model that fits better the training data is done by adding more SV until achieving a convex hull of the data. This reduced the "soft margin" of the model and could result in very bad testing error.

Figure 4(a) shows the training error and test error with respect to different cost values. We can clearly see that the higher the cost, the lower the training error. Testing error reduces when model has better fitting, but increases again if the cost it too large, resulting in over-fitting.

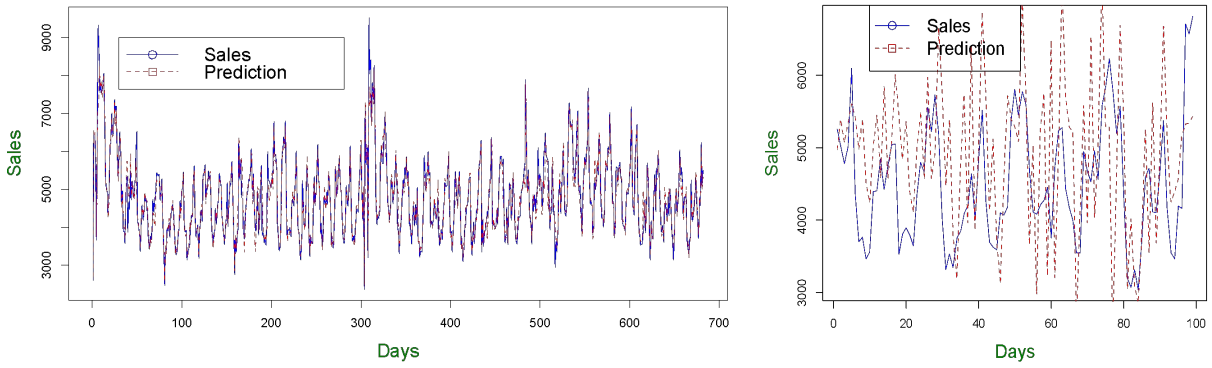
The gamma parameter defines how far the influence of a single training example reaches. It can be seen as the inverse of the radius of influence of samples selected by the model as support vectors. The behavior of the model is very sensitive to the gamma parameter. If gamma is too large, the radius of the area of influence of the support vectors is small, and it only influence the support vector itself. In that case, little regularization with cost will be able to prevent overfitting. On the other hand, if gamma is very small, the model is too constrained and fails to generalize. The region of influence of any selected support vector would include the whole training set.



(a) (b)

Figure 4: Experiment on svm parameters
((a) Training and testing error by changing cost value. (b) Training and testing error by changing gamma)

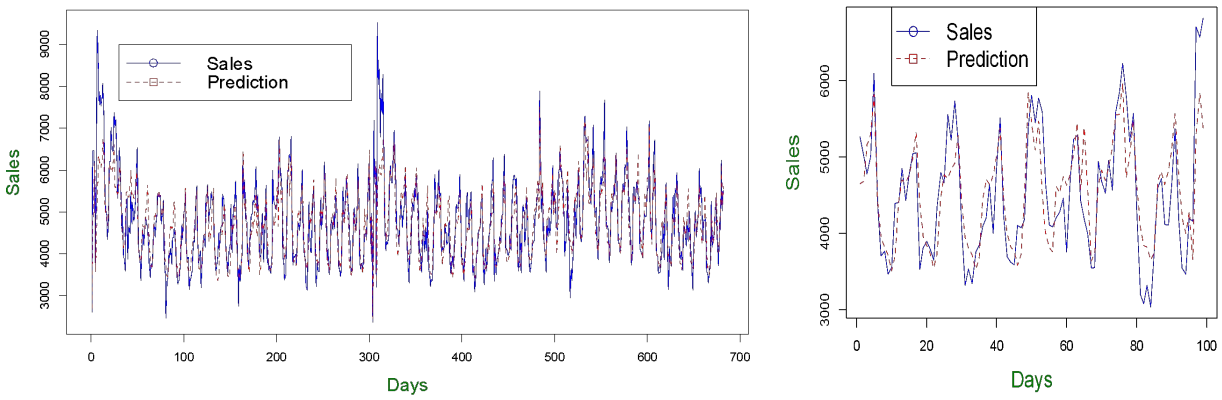
By selecting relative bigger gamma and cost, we can achieve very low training error, but bad testing result due to the overfitting problem. As shown in Figure 4, 4(a) is the beautifully fitted training result, and 4(b) is the testing result, which doesn't predict the numbers very well.



(a) Training results (b) Testing results

Figure 4: Over-fitting training and testing results (one store)

Carefully select cost and gamma gives much better testing result. (training error: 0.1006815, testing error: 0.1062105) As shown in Figure 5.



(a) Training results (b) Testing results

Figure 5: A better svm fitting results on training and testing data (one store)

Cross Validation:

10-fold cross validation was used to tune our model. We took every 3 months' historical Sales record as a folder, and separated it from training set and learn the model from rest of the training data, then test on the folder. Looping through all the folders, we tuned the parameters whenever seeing an abnormally big error. Our final model ran on the test data was an average of all 10 models that we learned from the training data.

Future work

We believe the sales number of a particular day is also related to the sales number before that day. Adding time series to the model can improve accuracy [4]. We will try adding time series to the feature vector to see what we can achieve. Different machine learning algorithm such as random forest or RNN can also be interesting to explore.

Conclusion

We did another training with our best model on kaggle.com provided test data. The test error feedbacked from kaggle.com is 0.12742. From our study, the feature treatment has a large impact on training model quality. Once features are chosen and formatted correctly, the prediction error improved dramatically. Model wise, GLM poisson model has the best prediction accuracy. Main reason is that the sales is directly related to number of customers. Poisson distribution is a good model for count data (number of customer visits). Time factor also played a major role to improve prediction accuracy. By adding before and after holiday feature, the prediction error reduced by >0.01.

Reference

1. Data source: www.kaggle.com
2. CS229 Machine learning lecture note
3. R-Programming site: https://cran.r-project.org/
4. J. Fan, Q. Yao, Nonlinear Time Series. Nonparametric and Parametric Methods, Springer Series in Statistics, Springer, New York (2003)

5. http://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html
6. Poisson distribution https://en.wikipedia.org/wiki/Poisson_distribution