Final Report: Rossmann Store Sales Prediction

Hao Qian(hq43), Yuning Yang(yy693)

1 Project Description

Our project aims at modeling the sales of these Rossmann stores and predicting their 6 weeks of sales to support the stores daily operation. The prediction would help the specific stores to schedule their staff more efficiently. It would also help the company to find out both the most and the least profitable stores and adjust their business strategies accordingly. From our analysis, we can learn in what degree various factors contribute to the sales.

2 Data Description

Kaggle provides us with two dataset for training and we join them together.
One dataset has over 1 million records with 9 columns of store, sales, customer, competitors and promos, which are all closely related to sales results. The other dataset has one record per store, describing a store's type, recurring promos and competitors with 10 columns. We joined the two datasets together by StoreID and Table 1 lists definitions of each column in details. In Section 4, we will describe how we deal with various fields for feature transformation.
For testing, we use the test dataset with 40k records provided by Kaggle and submit our predication results to evaluate our models. We find that some test records hold NA values on StoreOpen attribute, which indicates there may be some issues preparing the testset. We fix them by updating StoreOpen=1 because the sales of a closed store is always zero and won't be counted into error evaluations on Kaggle.

Table 1: Definition of Training Dataset

Store	a unique Id for each store
DayOfWeek	dayofweek of this sale record
Open	0 = store closed; 1 = store open
Promo	whether a store is running a promo
StateHoliday	indicates a state holiday. a = public holiday, b = Easter holiday, c = Christmas, 0 = None
SchoolHoliday	indicates if the (Store, Date) was affected by the closure of public schools
StoreType	4 different store models: a, b, c, d
Assortment	assortment level: a = basic, b = extra, c = extended
Competition Distance	distance to the nearest competitor store
Competition OpenSince [Month/Year]	the year when the nearest competitor was opened
Promo2	Promo2 is a continuing and consecutive promotion for some stores: 0 = store is not participating, 1 = store is participating
Promo2Since [Week/Year]	the year and calendar week when the store started participating in Promo2
PromoInterval	the consecutive intervals Promo2 is started, E.g. "Feb,May,Aug,Nov", "Jan,April,June,Oct"

3 Exploratory Data Analysis

Figure 1 shows how Promo influences sales on different days of a week. We use sales data of Store No.1000 in 2015, we can learn that Promo can increase sales significantly. We can also tell from this figure that the store is closed on Sunday, it doesn't have any promos on Saturday and it influences sales differently on Monday and Friday.

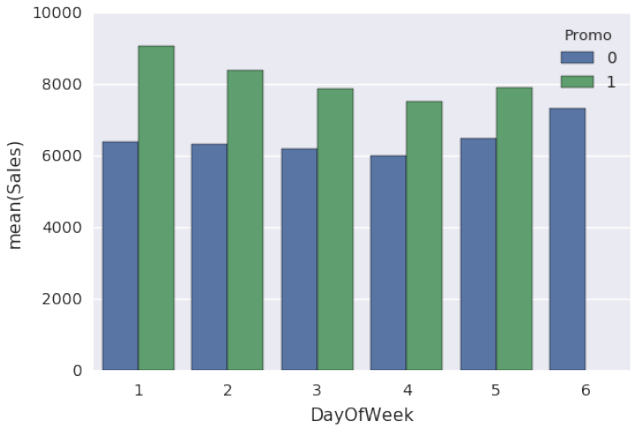

Figure 1: Mean sales of store No.1000 by DayofWeek, Promo

Figure 2 shows sales distribution by day of week. We can learn from this figure that stores sell more on Monday than other weekdays. Since many stores are closed on Sunday, the open stores can have more sales than weekdays.

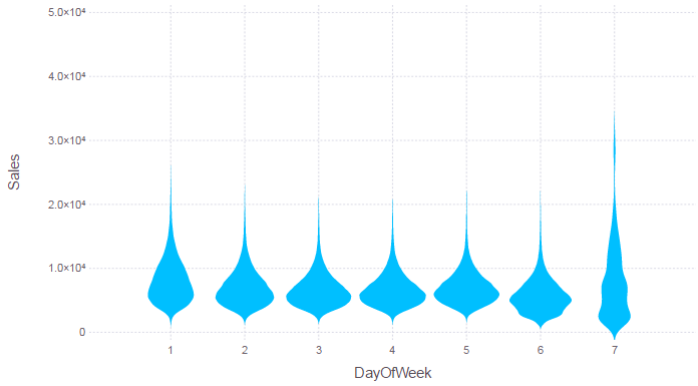

Figure 2: Sales distribution on different days of a week

From Figure 3, we can learn that different types of stores have different sales. Specifically, stores of type b often sell more goods than a,c,d.

Figure 4 shows how sales changes after competitors came. We learn from store.csv that competition started from 2013.8, where we can see a dramatic decrease of sales on the figure. This figure tells us that sometimes competition can influence sales significantly.

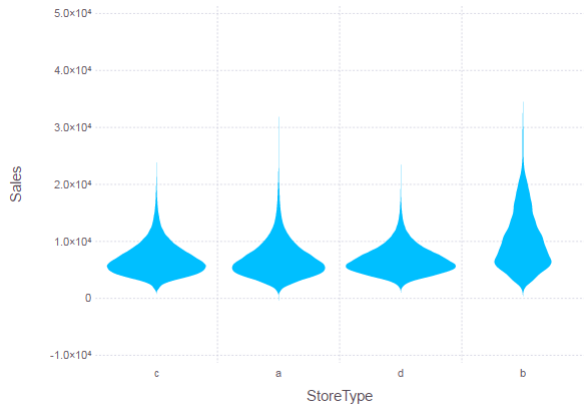

Figure 3: Sales distribution on different StoreType

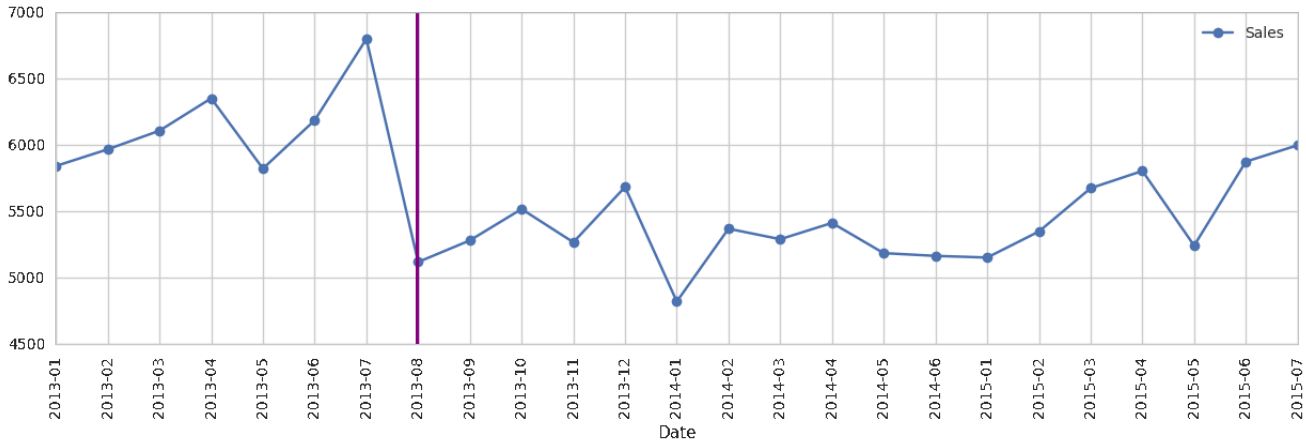

Figure 4: Sales trend of store No.191, competition since 2013.8

4 Feature Transformation

As we see above, some fields are closely related to the sales of stores and we can use them as features to train our model.

Some fields are categorical values, so we use binary vectors to encode them. E.g. DayOfWeek has nominal values (1,2,3,4,5,6,7) and wil be transformed into 7 columns isMonday{0,1}, isSunday{0,1}, etc. Similarly, we transform nominal fields like StateHoliday{a,b,c,0}, StoreType{a,b,c,d}, Assortment{a,b,c} to dummy variables.

As we mentioned in Section 2, we joined two provided datasets and we transformed some columns into new features:

For fields Promo2{0,1},Promo2Since[Week/Year] and PromoInterval, we use these three columns and the attribute "Date" to check if a store is participating a Promo on a specific date. Then we are able to add four binary features *hasPromo2*, *Promo2IntervalJan*, *Promo2IntervalFeb*, *Promo2IntervalMarch* indicating if a store has Promo2 running and when this round of Promo2 starts.

For fields (CompetitionDistance, CompetitionOpenSince[Month/Year]), we will check if a store has competitors at specified date, if not, we'll assign a very big value (80,000) of CompetitonDistance which indicates it doesn't influence the store. In this way we can eliminate NA values of CompetitionDistance and only reserve one feature *CompetitionDistance*.

Having joined the two datasets, we end up having 24 binary features and 1 numerical feature (*CompetitionDistance*).

5 Prediction

We tried to predict the sales with four kinds of models. The performance of the models are evaluated using root mean square percentage error (RMSPE). It is computed using the equation below. We used the mean of public and private score on Kaggle as the test error. (Kaggle splits the test set into two parts and computes the error on this two sets respectively) Detailed descriptions and errors are as follows.

$$RMSPE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\frac{y_i - x_i^T * w}{y_i})^2}$$

5.1 Model 1: Global Linear Model

To start with, we first used 4 basic features (DayOfWeek, StateHoliday, SchoolHoliday, Promotion) and 844392 examples (shops that are open) from our first dataset and trained some simple linear models. Multinominal features like DayOfWeek has been transferred as explained in previous part and 10 features in total are used. Constant offset is also added when doing computation.

We first applied linear regression with quadratic loss, but both the train error and test error are relatively large, indicating that this model is still underfitting. This might be caused by the existence of outliers. As huber regression tends to be more robust to outliers, we further applied linear regression with huber loss ($\sigma = 1$). The errors become smaller but still appear to be large. As the stores themselves might be largely different from each other, a global model with only one set of fixed parameters can't fit most stores. Therefore, we decided to try some other models.

Table 2: Error of Model #1

Model		**Train Error**	**Test Error**
Global Linear Model	Linear Regression	0.55255	0.45762
	Huber Regression	0.49168	0.39953

5.2 Model 2: Local Linear Model

The second model we used is local linear model. Considering that each store has over 700 records, we could train linear model for each store respectively. Since there are 1115 stores in total, this local model consists of 1115 sub-models. Again, Quadratic loss and huber loss were used. Besides training using the basic features, we also tried to add the additional features (CompetitionDistance, Promo2, Assortment, StoreType) in this model. The errors for this local model is shown below.

Table 3: Error of Model #2

Model		**Train Error**	**Test Error**
Local Linear Model with basic features	Linear Regression	0.05496	0.15894
	Huber Regression	0.05216	0.15085
Local Linear Model with additional features	Linear Regression	0.05442	0.15224
	Huber Regression	0.05931	0.14985

The test errors decrease in comparison with the previous model. In this regard, this model performs better. After adding the additional features, the errors didn't change too much. This

might indicate that the new features added are not of great contribution to the sales. Besides, train errors are relatively small than test errors, which indicating that this model might be overfitting to some degree.

5.3 Model 3: Local Linear Model with Regularizers

Regularizers might help to prevent overfitting. Therefore, we further applied regularizers to our models. Because there is no other constraint on our model, we simply added quadratic regularizer on the coefficients. However, how much weight the regularizer should be given is worth considering. We experimented a few λ (weight of the regularizer) and the results are as follows. (Only huber loss was used in this model.)

Table 4: Error of Model #3

Model		**Train Error**	**Test Error**
Local Linear Model with quadratic regularizer	$\lambda = 0.1$	0.07048	0.17908
	$\lambda = 0.05$	0.06434	0.16460
	$\lambda = 0.02$	0.06053	0.15748
	$\lambda = 0.01$	0.05931	0.15579

Even though adding regularizers increase the train error, it doesn't decrease test error in comparison with the previous model. It seems that regularizers does not help too much with this specific problem.

5.4 Model 4: Global Non-linear Model - Random Forest

Non-linear models can usually find some more complicated structures of data and therefore have better performances. After trying out some linear models, we began to seek some other methods. Since almost all the features in our prediction are categorical features, tree-based models might be used to solve the problem.

The fourth model we used is random forest, which is developed based on decision tree. Besides splitting the input space into some subspaces, random forest would generate a forest with different subspaces and form each tree with a sub-train set using bootstrapping. It could both split the training samples and features to avoid overfitting. The final result is based on majority votes.

Both basic features and additional features were used in this model. And only competition distance was used as numeric feature. We tried a few parameters (NumberOfTrees , BootstrappingRate, ... etc) and interestingly the performance of random forest model is quite similar to, and even a little bit worse than local linear models. Some of our results are shown in the table below.

Table 5: Error of Model #4

Model	**NumOfTrees**	**MaxDepth**	**Boostrapping Rate**	**Train Error**	**Test Error**
Random Forest	50	20	0.1	0.05509	0.16853
	100	20	0.1	0.05352	0.16693
	100	50	0.1	0.05037	0.16202

6 Interpretation

In this part we would analyze some problems based on previous models. And the analysis is based on huber regression from model #2 and random forest model, which have similar acceptable performance. As for the local model, each store has a different set of parameters in this model, but we

could analyze the problem using the distribution of the parameters. And using the variable importance from random forest, we could also see the relative importance of features in sales prediction.

Table 6: Variable Importance From Random Forest Model

Feature	**Scaled Importance**	**Percentage**
StoreID	1.000000	0.655216
Promo	0.303030	0.198550
DayOfWeek	0.101535	0.066527
CompetitionDistance	0.044861	0.029394
StoreType	0.041100	0.026930
Promo2Type	0.023011	0.015077

"If the variable is not important, then rearranging the values of that variable will not degrade prediction accuracy." (From: https://www.r-bloggers.com/random-forest-variable-importance/) Variable importance could reveal what are the important predictors. From the table above, we could see that storeID, if there is promotion and DayOfWeek are the three most important features. We further looked at these 3 features in our linear model for a much clear interpretation.

We first plotted out the distribution of the constant in our huber regression model in model #2 (with additional features) to see the difference between stores. Most stores have quite similar constant sales but some stores have really large constant sales. It might because these stores are located in places with large flow of people. However, since we don't have much information of the details of these stores, we could only make assumptions about the stores.

As for the distribution of the parameters of Promo, we could see that a short-term promotion would boost sales greatly. It would increase the sales by about 2000 turnovers on average, which is also the feature that has the largest contribution to sales.

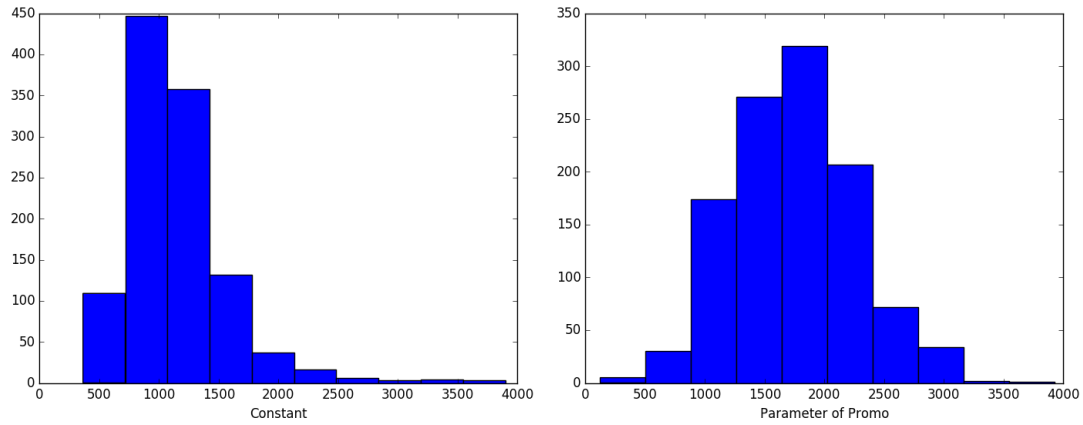

Figure 5: Constant and Parameters of Promo

As for the parameters of different DayOfWeek, the first thing to notice is that most parameters for Sunday is 0. This is related to the fact that most stores are closed on Sunday. Besides of that, the parameters for each day seem to vary, from negative to positive. On average, being Mondays would increase the most sales and on Saturdays the sales vary the most greatly. This might indicate that on Mondays the stores should arrange more staff in general. While on Saturdays, we should look at specific stores and decide how to schedule the staff.

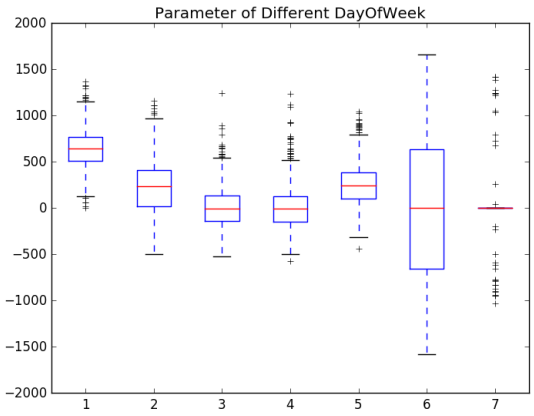

Figure 6: Parameters of different DayOfWeek

7 Summary

In this semester-long project, we started by exploring our dataset then added more features and tried different models. Our best model has a test error of 0.15 which performs pretty well on Kaggle leaderboard. We also analyzed the parameters of our model and the importance of different parameters, it turns out they can be interpreted in a reasonable way. With a well fitting model and a reasonable set of parameters, we're very confident that our model and interpretations make sense.

If we were the managers of Rossmann Store, we can make some changes on store operations based on this analysis. For example, we can schedule more staff on Monday and Sunday (if a store is open that day) because sales tend to be high on these two days. We're also able to set reasonable but different sales goals by store based on the prediction results of each store.

In our analysis, we tried several methods introduced in class, such as linear regression model, huber loss, quadratic regularizer, feature engineering for categorical values, and proximal gradient method to optimize our prediction results. We also used Random Forest to learn the importance of different variables.