Rossmann Data Exploration Report

Liqi Gu, Shuai Jiang, Shengtao Zhong, Runying Jiang
lglu20@soton.ac.uk, sj4n20@soton.ac.uk, sz3g20@soton.ac.uk, rj1u20@soton.ac.uk

Abstract: In this paper, we tried to dig deeper into the data exploratory phase of a real-world problem — drug store sales forecasting. **Rossmann Store Sales Prediction** is a problem based on a time series dataset. A time series is a sequence of observations taken sequentially in time. Our findings were documented from three aspects: problem analysis, data exploration and project reflection. Python data wrangling tools (Pandas, Numpy, Seaborn, Matplotlib etc.) were used to analyze and visualize the dataset.

1. Problem Analysis

1.1. Problem Description

Dirk Rossmann GmbH (usual: Rossmann) is one of the largest drug store chains in Europe with around 56,200 employees and more than 4000 stores across Europe. In the Kaggle competition, competitors were challenged to predict daily sales of six weeks for 1,115 stores located across Germany. The model performance was evaluated by Root Mean Square Percentage Error: $\text{RMSPE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i-\hat{y}_i}{y_i}\right)^2}$.
From time to time, the store will hold short-term promotional activities and continuous promotional activities to increase sales. Moreover, store sales are also affected by many factors, including promotions, competition, school and national holidays, seasonality and periodicity.

1.2. Dataset Description

The dataset can be found online. To start off, a simple preview and statistics of our dataset was generated.

Name	Type	Description
Store	Int	a unique Id for each store
Sales	**Int**	**the turnover for any given day (target)**
Customers	Int	the number of customers on a given day
Open	Categorical	whether the store was open
DayOfWeek	Categorical	indicates the day of week
StateHoliday	Categorical	indicates a state holiday
SchoolHoliday	Categorical	indicates if the data was affected by the closure of public schools
Promo	Categorical	indicates whether a store is running a promo on that day

Table I: Description of train dataset

Table.1 shows the description of the train dataset, which contains the historical sales data from 2013-01-01 to 2015-07-31. There are 1017209 rows and has no missing data. Our target attribute is the sales, which is the turnover for any given day. And in test dataset, the same attributes except for Customers and Sales are given for the period from 2015-08-01 to 2015-09-17.

Store	DayOfWeek	Date	Sales	Customers	Open	Promo	StateHoliday	SchoolHoliday
1	5	2015/07/31	5263	555	1	1	0	1
2	5	2015/07/31	6064	625	1	1	0	1
3	5	2015/07/31	8314	821	1	1	0	1
4	5	2015/07/31	13995	1498	1	1	0	1
5	5	2015/07/31	4822	559	1	1	0	1

Table II: Preview of train dataset

Store	DayOfWeek	Date	Open	Promo	StateHoliday	SchoolHoliday
1	4	2015/09/17	1	1	0	0
3	4	2015/09/17	1	1	0	0
7	4	2015/09/17	1	1	0	0
8	4	2015/09/17	1	1	0	0
9	4	2015/09/17	1	1	0	0

Table III: Preview of test dataset

Apart from the train and test dataset, the competition also provides us with the store dataset, which provides any identity store's attribute.

Name	Type	Description
StoreType	Categorical	indicates the store type
Assortment	Categorical	describes an assortment
CompetitionDistance	Int	distance to the nearest competitor store
CompetitionOpenSince	Date	the approximate year and month of the time the nearest competitor was opened
Promo2	Categorical	a continuing promotion for some stores
Promo2Since	Date	describes the year and calendar week when the store started participating in Promo2
PromoInterval	Categorical	describes the intervals Promo2 is started

Table IV: Description of store dataset

Table.2 shows the supplemental information about the stores, which has 1115 rows with some missing data in features related to Competition and Promo.

Store	StoreType	Assortment	CompetitionDistance	CompetitionOpenSinceMonth	Promo2
1	c	a	1270.0	9.0	0
2	a	a	570.0	11.0	1
3	a	a	14130.0	12.0	1
4	c	c	620.0	9.0	0
5	a	a	29910.0	4.0	0
6	a	a	310.0	12.0	0

CompetitionOpenSinceYear	Promo2SinceWeek	Promo2SinceYear	PromoInterval
2008.0	NaN	NaN	NaN
2007.0	13.0	2010.0	Jan,Apr,Jul,Oct
2006.0	14.0	2011.0	Jan,Apr,Jul,Oct
2009.0	NaN	NaN	NaN
2015.0	NaN	NaN	NaN
2013.0	NaN	NaN	NaN

Table V: Preview of store dataset

2. Data Exploration

2.1. Time Series Exploration Data Analysis

In this section, we analyse the trend and seasonality of the data. The trend is the linear increasing or decreasing behavior of the series over time while seasonality is the repeating patterns or cycles of behavior over time.

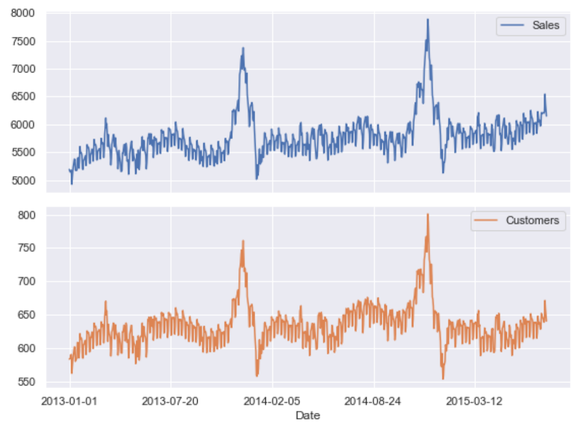

Figure 1: The 30-day Moving Average Sales and Customers

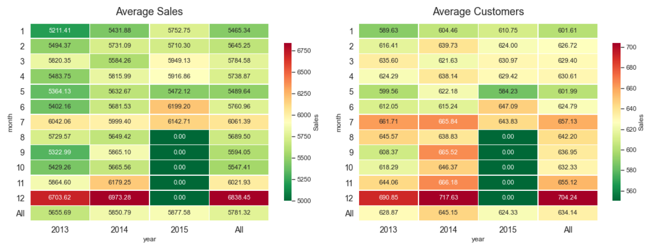

Figure 2: The Heatmap of Monthly Average Sales and Customers

Figure 1 and **Figure 2** show the average sales and customers of each day and month. What can be clearly seen in figures above is the general pattern of monthly average sales and customers that sales and customers usually peak in December and reach a low point in January.

2.2. Categorical Data Analysis

The objective for categorical data analysis is to know the unique values and their corresponding count. As shown in **Figure 3**, most of stores closed on State Holidays and Sundays while School Holidays have some impact on store opening.

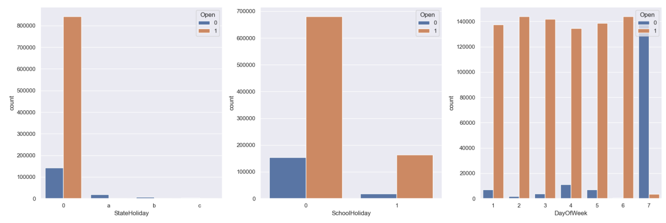

Figure 3: Counts of the openings on holidays and Day of Week: Note that all schools are closed on public holidays and weekends. a = public holiday, b = Easter holiday, c = Christmas, 0 = None.

As can be seen in **Figure 4**, the market is mainly occupied by Type **a** and Type **d** stores. Moreover, most varieties are basic ones or extended ones. Almost no extra variety.

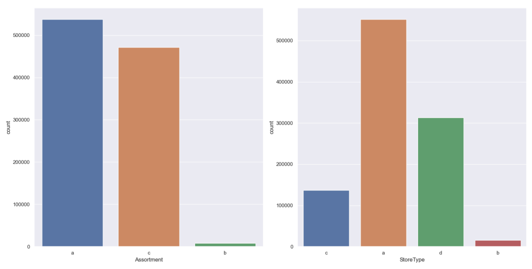

Figure 4: Counts of Assortment and StoreType: Assortment describes an assortment level: a = basic, b = extra, c = extended; StoreType differentiates between 4 different store models: a, b, c, d.

We can learn from **Figure 5** that about half of the stores participate in discount promotions, while the other half do not. In the stores with discounts, the months of their promotions are mainly January, April, July, October.

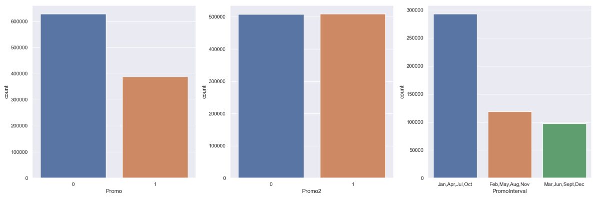

Figure 5: Counts of categories related to Promo : Promo2 is a continuing and consecutive promotion for some stores: 0 = store is not participating, 1 = store is participating; PromoInterval describes the consecutive intervals Promo2 is started, naming the months the promotion is started anew.

2.3. Numeric Data Analysis

2.3.1. Univariate Distribution of Numeric data

The basic features of numeric data (Sales, Customers, Competition Distance) are provided in the following **Table VI**. To better understand the variates, we go into deeper with the distributions. As shown in **Figure 6**, all the three variates have positively skewed distributions, moreover, they are approximately log-normally distributed.

	count	mean	std	min	25%	50%	75%	max
Sales	83897.0	6071.416999	3834.150546	0.0	4120.0	6048.0	8143.0	41551.0
Customers	83897.0	636.519721	451.684626	0.0	428.0	613.0	824.0	4989.0
CompetitionDistance	83673.0	5404.869193	7659.567708	20.0	720.0	2320.0	6880.0	75860.0

Table VI: Univariate Descriptive statistics of numeric data

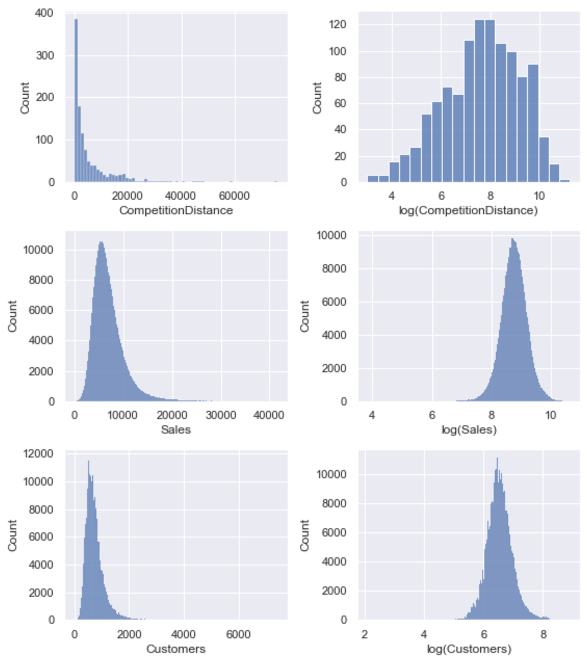

Figure 6: Distributions of CompetitionDistance, Sales and Customers (left side) as well as their log distributions (right side).

2.3.2. Bivariate Distribution of Numeric Data

This section analyzes the distribution of sales across several levels of categorical variables (StoreType, DayofWeek and etc). In the violin plot **Figure 7**, we select Store one as a sample to check the distribution of sales. We can infer that stores closed on state holiday and Sunday, which is the same result we get from the count plots above **Figure 3**. At the same time, it implicates that school holiday has limited influence on sales. The school festival does not increase the number of sales.

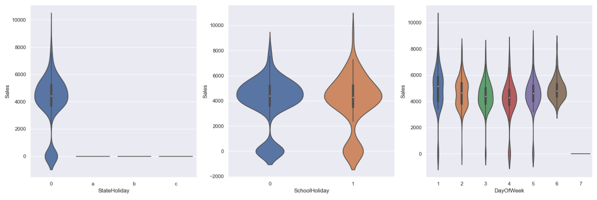

Figure 7: Distribution of Store one sales on holidays and Day of Week

Figure 8 shows that there is almost no difference of sales distribution between Store Type **a** and Store Type **c**, however, stores of Type **b** seems to be open all the year and have better performance in sales. This situation is very similar to extra assortment, although extra assortment has a small proportion.

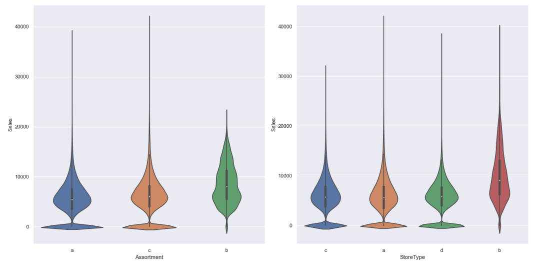

Figure 8: Distribution of sales on Assortment and StoreType

From **Figure 9**, we can find that discounts have a little boost to sales. From the overall situation, whether there are discount activities in shops has almost no impact on sales. Similarly, the discount range has little effect on sales.

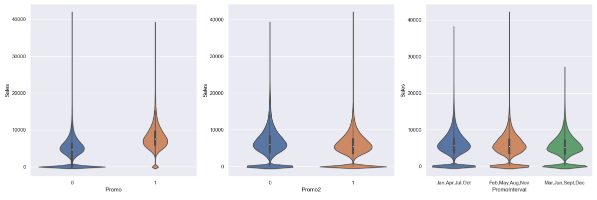

Figure 9: Distribution of sales on Promo, Promo2 and Promo Interval

2.3.3. Pairwise Distribution of Numeric Data

What can be clearly seen in the pairwise joint distribution plot of sales volume and number of customers is that they are positively correlated. There is also a certain degree of correlation between sales volume and competition Distance, Competition Since Year and Promo Since Year.

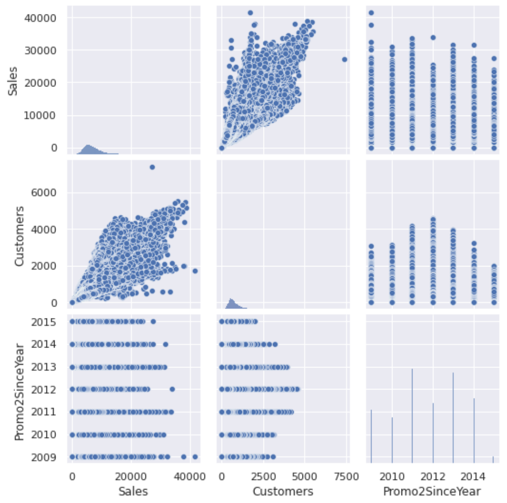

Figure 10: Pairwise Distribution of Sales, Customers, Promo Since Year

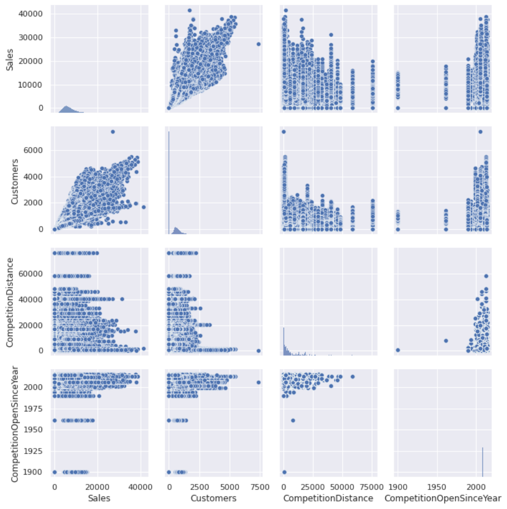

Figure 11: Pairwise Distribution of Sales, Customers, Competition Distance and Competition Since Year

3. Project Reflection

3.1. External Data and Information

Although, in general, the use of external data is prohibited in Kaggle competitions, some public data sets are very reasonable and are indeed helpful for accurate prediction of the model. The followings are the interesting data sets we found to better predict sales:

- **Data on mapping of stores to states**: The data set succeeded in grouping stores in the different state based on the Holidays and Observances in Germany in 2013, since the public holidays in Germany differ state by state. Moreover, this data set can be further used to predict the weather, which is proved to be very effective in predicting sales.
- **Weather data**: Previous research [1] has established that the weather has generally a complex effect on daily sales while the magnitude and the direction of the weather effect depend on the store location and the sales theme. It is shown that weather forecast information improves sales forecast accuracy up to seven days ahead, however, the improvement of the forecast accuracy diminishes with a higher forecast horizon.
- **Google trends data**: Combining standard time series models, the researchers has found evidence that using Google Trends data can enhance the prediction performance of conventional models [2]. The data set can be downloaded as a csv file within a google account.
- **World cup dates**: The dataset of 2014 FIFA world cup dates is the most interesting part. In the final, Germany defeated Argentina 1–0 to win the tournament and secure the country's fourth world title, the first after the German reunification in 1990. Though so far there is evidence to prove that there is a clear relationship between the World Cup and drug store sales, we believe that the influence of significant historical time on drug store sales cannot be ignored.

3.2. Project Outcome

Traditionally, we can use auto-regression to predict time series data. That is only relying on past sales to predict sales for the next six weeks. However, the pattern of sales in the Rossmann data sets is not that clear, and many factors are contributing to sales volume. Therefore, only applying the auto-regression model manually on Rossmann data sets is not wise.

To conclude, the purpose of time series analysis is to understand and observe the random pattern of the sequence, and predict the future value of the sequence based on the observed pattern.

References

[1] F. Badorf and K. Hoberg, "The impact of daily weather on retail sales: An empirical study in brick-and-mortar stores," *Journal of Retailing and Consumer Services*, vol. 52, p. 101921, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0969698919303236

[2] B. Fritzsch, K. Wenger, P. Sibbertsen, and G. Ullmann, "Can google trends improve sales forecasts on a product level?" *Applied Economics Letters*, vol. 27, no. 17, pp. 1409–1414, 2020. [Online]. Available: https://doi.org/10.1080/13504851.2019.1686110

Model Report on Rossmann Dataset

Liqi Gu, Shuai Jiang, Shengtao Zhong, Runying Jiang
lglu20@soton.ac.uk, sj4n20@soton.ac.uk, sz3g20@soton.ac.uk, rj1u20@soton.ac.uk

Abstract: This report discusses a selection of models to solve the **Rossmann Store Sales Prediction** problem, which is based on a time series dataset. First, we analysed the problem and defined the objective. In order to achieve this goal, we went through a pipeline of data preprocessing, including data cleaning, feature engineering/selection and feature scaling, then several dirty models were made to be the baseline. Afterward, more complicated models (XGBoost, LSTM, Informer) were implemented and fine-tuned to make more precise predictions. Finally, we simply averaged the results of different models and achieved a good performance.

1. Problem Analysis

The Rossmann Store Sales training dataset comprises some attributes of 1115 different stores in Germany, along with totally 1017209 sales records from 2013-01-01 to 2015-07-31, while the objective of this problem is to predict the sales of these stores during the period from 2015-08-01 to 2015-09-17. A rational assupmtion of sales is that they are affected by the seasonality and periodicity, thus, we define the problem as a time-series regression task. The performance is evaluated by Root Mean Square Percentage Error according to kaggle:

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

In the next few sections, we will do data preprocessing and train several models to minimise the error on the test dataset.

2. Date Preparation

2.1. Data cleaning

In the aspect of data cleaning, we deal with the missing values, and then drop the outliers according to the distribution structure of features.

2.1.1. Missing Value

The missing data is statistically analysed and shows in **Table I** and the list below

- train.csv - historical data (2013-01-01 to 2015-07-31) including Sales.
  The training dataset has 1017209 rows and 9 colomns with no missing data.
- test.csv - historical data (2015-08-01 to 2015-09-17) excluding Sales
  The testing dataset has 41088 rows and 8 columns with only 11 missing data in *Open* column.
- sample_submission.csv - a sample submission file in the correct format
- store.csv - supplemental information about the stores
  The store dataset has 1115 rows with 10 columns and there are some missing values

Name	Dataset	Type	Missing Number
Open	test	float64	11
CompetitionDistance	store	Int	3
CompetitionOpenSinceMonth	store	float64	354
CompetitionOpenSinceYear	store	float64	354
Promo2SinceWeek	store	float64	544
Promo2SinceYear	store	float64	544
PromoInterval	store	object	544

Table I: The Number Of Missing Data

We notice that the missing value in *CompetitionOpenSinceMonth*, *CompetitionOpenSinceYear* indicates the store faced no competition with other stores. In the **Section 2.2.2**, we will compute the number of months that the competitor had opened, and fill the missing number as zero. We do the same processing to the missing value in *Promo2SinceWeek* and *Promo2SinceYear*. Then we use the median value to impute *CompetitionDistance* since the distribution is skewed. As for *PromoInterval*, we fill the missing value with a new category, indicating the store had no promotion. Also, in the testing dataset, we assume the store opened if the opening status is not given.

2.1.2. Outliers

An outlier is an observation that lies an abnormal distance from other values in a random sample from a population. To avoid the influence of those extreme points, we set up a Quantile filter to remove them. We only keep the data between upper and lower quartiles, which is the data have a difference between 75th and 25th percentiles.

2.2. Feature Engineering

Feature engineering is the process of transforming raw data into features that better represent the underlying problem to the predictive models, resulting in improved model accuracy on unseen data.

2.2.1. Decompose features

- Split String value: Original *PromoInterval* feature 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. We split the string and leave it for further processing.
- Mapping String value : Since the model cannot handle strings, we digitally encode the]*Storetype*, *Assortment*, and *Stateholiday* features which is mapping the string value from a∼d to 0∼4. Besides, We map the processed *PromoInterval* to numbers from 1 to 12 and add a new feature *IsPromoMonth* which shows that whether the store participate in the promotion in the current month.

2.2.2. Transform features

- Transform datetime value: The date format cannot be processed in the model so we split the datetime into three features: *year*, *month* and *day*. As we have already stated in the data exploration report, there are repeated patterns of the week of the year and day of the week, we also add them as a feature.
- Transform competition and promo start data: In the original data set, *CompetitionOpenSinceYear* and *CompetitionOpenSinceMonth* features gives the approximate year and month of the time the nearest competitor was opened, which can not better express its internal meaning and the specific relationship between sales. Therefore We build the feature *CompetitionOpen*, which describes the number of months the store's current date is away from the competition opening. In the same way, *Promo2SinceYear* and *Promo2SinceYear* features show that the year and calendar week when the store started participating in *Promo2*. According to these two features, we calculate the number of months from the store's current date to participating in promo2 as *PromoOpen*.

2.2.3. Create new Features

We also join the store dataset and train dataset to create more features:*SalesPerDay*, *CustomersPerDay*, *SalesPerCustomersPerDay*, *StoreGroup'* and from 1, it can be seen that they work really well.

- Add *StoreGroup*:Based on common sense that store sales are usually relatively stable, we divide the stores into four groups by Quantile filter based on sales to help with the prediction. And the counts and Sales mean of each group are listed in the following table.
- Add *SalesPerDay*,*CustomersPerDay*: We add the average customer and average sales of the store as the new features*SalesPerDay*,*CustomersPerDay* .
- Add *SalesPerCustomersPerDay*: Since we have *Customers* feature which indicates the customer flow in train set but we don't have it in test set, to fully use all the features given, we take the average sales of user purchases as a new feature *SalesPerCustomersPerDay*.

Group	count	Sales mean
1	210160	4516
2	207646	5934
3	212232	7210
4	213187	10052

2.3. Feature selection

Feature selection refers to the process of reducing the features for processing and analysis. the **Figure1**, we can find even the least relative feature *StateHoliday*, *IsPromoMonth*(not showing in the figure),is still useful to a certain extent. Besides, after doing some experiments of choosing only the top importance features and comparing the performance, we decide to keep all the related feature s. In addition, Some features such as *CompetitionOpenSinceYear*, *CompetitionOpenSinceMonth*,*Promo2SinceYear* and *Promo2SinceYear* has been transformed in the Feature Engineering section, so we delete them. MoreOver, the feature *Customer* does not show in test dataset, we delete it as well.

2.4. Feature Scaling

Traditionally, all of the numerical data should be normalized to ensure a better model performance.

- *CompetitionDistance*: We apply the standard scaler to this numerical feature though it has alomost no effect on the performance of the models.
- *Customers*: it has been deleted while training.
- *Sales*: The target of our prediction. It is not reasonable to apply the standard scaler on target value, instead the log transformation is used to make it less skewed.

3. Model Training

3.1. Train many quick and dirty models

Table II shows the performance of the quick models with dirty data. We use the Nested Cross-Validation instead of the traditional one to split the time series data without causing data leakage. We divide the training set into different subsets by month. The first subset is data from January to March, and we add April to the second one and so on , add month one by one to prevent the dependency in time series data. The training of Random forest regression is relatively slow and it behaves a little bit worse than the other two models. The nonlinear characteristics of random forest regression have more advantages than linear algorithm but random forest regression has its own limitations in the time series problem, which is that the prediction range of random forest is constrained by the highest and lowest label values in the training data. Because SVR model is sensitive to missing data, the performance of the model is not as good as random forest.

Model	mean of validation	std	RMSPE of test
Random forest Regressor	0.45740	4.7485e-05	0.48810
Linear regressor	0.46951	5.8273e-05	0.45215
SVR	0.45740	4.7485e-05	0.45215

Table II: Quick models

3.2. XGBoost

The common regression model has a high bias (too simple, the fitting ability of the algorithm itself is poor) or large variance (too complex, the stability and robustness are not strong). Ensemble learning can combine multiple weak regression models to get a better one. XGBoost is a model of ensemble learning, which is widely used in data science competitions and industry. XGBoosting uses CPU multithreading, introduces regularization term, and adds pruning to control the complexity of the model based on the traditional boosting. It has the characteristics of parallel processing, high flexibility and built-in cross-validation. It also has a good performance in solving time series problem in the past few years.

3.2.1. Important features

Figure 1 shows that *Day* is the most important which is what we expected, since it is a time series dataset. Moreover, all of the features related to time such as *CompetitionOpen WeekofYear* and *Day of Week* ranks high. As the sales of each store vary greatly the *store* is also highly related to the sales. We also found that whether it is the promotion month and whether the store has a continuous sales policy seems to have little effect on sales.

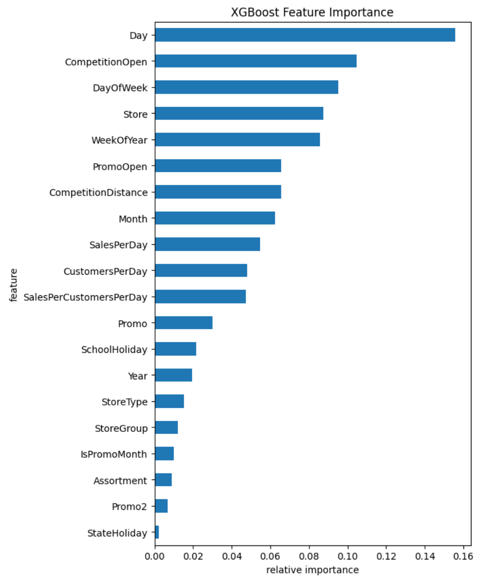

Figure 1: Important features by XGBoost

3.3. LSTM

A popular sequence modeling method in deep learning is the LSTM (Long short-term memory) neural network. Unlike the machine learning method, the neural network architecture is able to extract features automatically. Specifically, the LSTM or other RNN-based model can learn the dependencies within a sequence of data. In this sales prediction problem, the sales can be affected by some factors in the previous days or in the next few days. For example, if the store will start a promotion tomorrow, then it is more likely to have low sales today. However, there are much more dependencies that we cannot perceive intuitively, so we come to the neural network.

Figure 2 shows the structure of our proposed neural network. The model contains a forward LSTM cell and a backward LSTM cell to extract the time series features, as well as a fully connected layer to deal with the non time series features. The time series features, which keep changing to a store and are not deterministic, include *Open, Promo, StateHoliday, SchoolHoliday, Events* The non time series features are fixed attributes to a store, or the changing is deterministic and have no time dependencies, including *Store, DayOfWeek, StoreType, Assortment, CompetitionDistance, Promo2, etc.* The features are either categorical or numerical while the neural network can only accept numerical inputs, thus, we turn the categorical features into learnable embeddings[1].

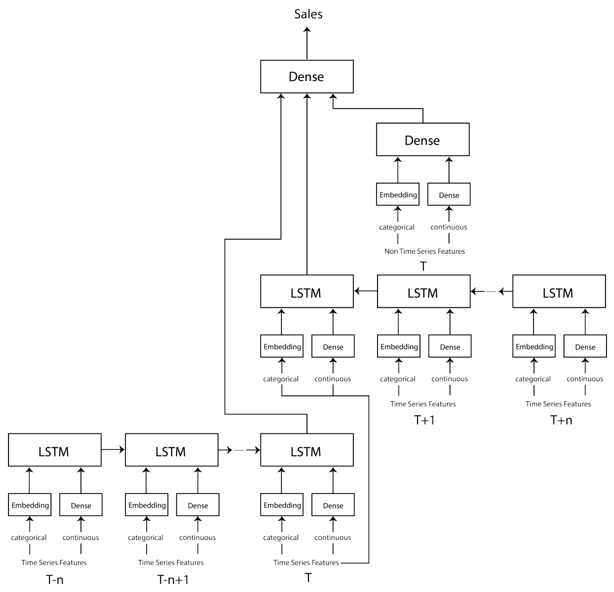

Figure 2: The structure of Rossmann LSTM neural network

3.4. Informer

To further improve the prediction accuracy, we observe the recently introduced state-of-the-art method on time series forecasting named Informer.

Informer is a new transformer based model for long sequence time series forecasting[2]. It leverages the self-attention Transformer architecture to extend predictions to much farther temporal horizons compare to the other time series forecasting method.

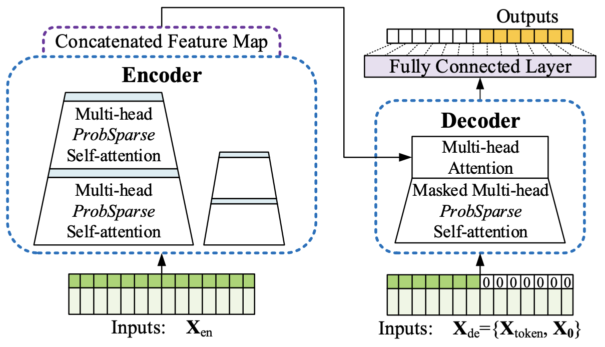

Figure 3: Architecture of Informer

Informer aims to improve the self attention mechanism, reduce memory usage, and accelerate inference speed. It utilizes both a transformer encoder and transformer decoder layers. The decoder can effectively forecast long sequences in a single forward pass. This feature helps accelerate inference speed when forecasting long sequences. The Informer model employs a probabilistic attention mechanism to forecast long sequence. It also includes learned embedding of relevant temporal features. This allows the model to generate an effective task based representation of time. Finally, the Informer likewise can stack n levels of encoder and decoders based on the complexity of the task.

3.5. Measure and compare their performance

After we each train the model, we submit it to Kaggle and get the result of the model **Figure : III** . Then we ensemble the outcome of each base model and results in the final prediction. Our final prediction with RMSPE of **0.11073** ranked 373 on the Kaggle LeaderBoard.

Model	RMSPE of test
XGBoost	0.12953
LSTM	0.12645
Informer	0.11981
Ensemble	0.11073

Table III: Model performance on Kaggle test dataset

3.6. Conclusion

The most interesting part is the XGBoost importance. It reveals that in the long run, the promotion has no effect on sales quality, which is contrary to the common knowledge of the economics of most people. But this can be explained in some respects. The service and quality of the store are always the most important determinants of sales. Another point worth mentioning is that we plan to add external features like *State* and *Weather* at the beginning. Then due to too much data and too noisy, the LSTM training speed is very slow and does not help the improvement of the model. We abandon this method.
Time series forecasting is one of the most widely used data science techniques in business. However, processing and training time-series data requires more skills to ensure time dependence. Generally speaking, the model we trained has a relatively good performance in predicting time series data.

References

[1] C. Guo and F. Berkhahn, "Entity embeddings of categorical variables," *arXiv preprint arXiv:1604.06737*, 2016.

[2] H. Zhou, S. Zhang, J. Peng, S. Zhang, J. Li, H. Xiong, and W. Zhang, "Informer: Beyond efficient transformer for long sequence time-series forecasting," *arXiv preprint arXiv:2012.07436*, 2020.

4. Contribution Statement

Due to some unreasonable parts in the division of labor, we redistributed the contribution ratio. The contribution assigned are as follow and we finally reach an agreement on it.

- Shuai Jiang (sj4n20@soton.ac.uk) : 10%
- Shengtao Zhong (sz3g20@soton.ac.uk) : 30%
- RunYing Jiang (rj1u20@soton.ac.uk) : 30%
- Liqi Gu (lglu20@soton.ac.uk) : 30%

This is the contribution ratio for all of the work (including the last report). In the first report, we did not mention the issue of task assignment because the tasks has not all be completed.