Rossmann Store Sales Challenge

Kandarp Vyas, NamanBhargava
Department of COE, Northeastern University

Abstract

This paper is about the analysis of Rossmann Store Sales dataset and builds a model to predict next six-week store sales values. The dataset comes from Kaggle competition supported by Rossmann Store Sales. We have followed these steps: Data Wrangling, EDA, Timeseries Analysis, Feature Engineering and Machine Learning. We started with comprehensive analysis on the dataset, explored most features and collected all features we thought were useful. Before implementing the model, we changed some columns into categorial variables. We also removed unnecessary columns from the prediction model. During the model building process, we used Linear Regressor, Random forest classifier algorithm and Bagging regressor. Lastly, we used Keras Sequential Neural Network with TensorFlow in the backend.

Introduction

Our task here is to predict six weeks store sales value so that store manger can effectively schedule staffs a way ahead and assign staff accordingly. First, we have removed N/A values with the median value of the column. Converting data type of the columns into appropriate data type for the faster execution. Our task here is to predict the six weeks sales values by applying different machine learning algorithm. We have adopted tweaking the existing algorithm by changing its configuration parameters. For that We have carried out case studies of configuration of machine learning algorithm parameters, PCA, more data exploratory analysis and have spent time on try and error of analysis of the algorithm.

Background and Motivation

Sales forecasting is critical for inventory management in the retail industries. Ideally, store managers can use accurate predictions to meet demand while minimizing inventory footprint and therefore operational costs. Further, discrete factors such as holidays, the opening of competitors and promotions all have a significant level of demand on any given day. We seek to analyze the impact of these factors with the aid of time series analysis and machine learning techniques.

Problem Statement

Predict 6 weeks of daily sales for 1115 drug stores operated by Rossmann located across Germany to enable store managers to create effective staff schedules that increase productivity and motivation.

Approach taking/Solution

First, we have analyzed different features in the datasets by visualizing the distribution of data for those parameters and their statistics. By defining some reasoning based on how parameters affect the sales, we have picked out the features of the prediction model. We analyzed two linear regression models one with single feature and other with a set of features, random forest and bragging regression. We also created a neural network using Keras. By taking a close look to different classifiers, we could find that Bagging regressor has the best accuracy of all the models.

Code with Documentation

https://github.com/kandarpvyas/Rossmann-Store-Sales

https://github.com/namanbhargava/Data-Science

Understanding the dataset

Datasets Information

There are two training datasets provided to us (train.csv & store.csv) and one test dataset (test.csv) whose sales we must predict.

Train.csv

This is the main training dataset which contains data about the sales figures for a store on a date. Data Fields:

1. Store: a unique numerical store identifier (1 - 1,115)

2. DayOfWeek: the day of week (1 - 7)

3. Date: the date, ranging from 2013-01-01 to 2015-07-31

4. Sales: the turnover of the specified store on the specified date

5. Customers: the number of customers of the specified store on the specified date

6. Open: indicates whether the store was open (0 = Closed, 1 = Open)

7. Promo: indicates whether the store was running a promotion (0 = No, 1 = Yes)

8. StateHoliday: indicates if it was a state holiday (a = Public Holiday, b = Easter holiday, c = Christmas, 0 = None)

9. SchoolHoliday: indicates if it was a school holiday (0 = No, 1 = Yes)

Store.csv

This dataset contains supplementary information about each of the 1,115 stores and helps identify unique features which may (or may not) affect sales.

Data Fields:

1. Store: a unique numerical store identifier (1 - 1,115)

2. StoreType: differentiates between the 4 different types of stores (a, b, c, d)

3. Assortment: describes the assortment of goods carried by the store (a = Basic, b = Extra, c = Extended)

4. CompetitionDistance : the distance (in metres) to the nearest competitor's store

5. CompetitionOpenSinceMonth : the month in which the competition opened

6. CompetitionOpenSinceYear : the year in which the competition opened

7. Promo2: indicates if a store is participating in a continuing and consecutive promotion (0 = No, 1 = Yes)

8. Promo2SinceWeek: the week of the year in which the store began participating in Promo2 (from 1 - 52, presumably, but some weeks are unrepresented in the data)

9. Promo2SinceYear: the year in which the store began participating in Promo2 (from 2009 - 2015)

10. PromoInterval: describes the consecutive intervals in which Promo2 is activated, giving the months the promotion is renewed (either "Jan,Apr,Jul,Oct", "Feb,May,Aug,Nov" or "Mar,Jun,Sept,Dec")

Test.csv

This dataset is to be used for testing and evaluating the model.

Data Fields: Same as train.csv, with the exclusion of Customers & Sales (Sales to be predicted by the model) and an additional field Id which represents a (Store, Date) tuple that is used to label predictions for submission to Kaggle.

Preprocessing the data

Replacing Missing Values & Fixing Corrupted Data

1) Replacing NaN values for Open: Missing values in the Open data field were replaced with '1' if the DayOfWeek was not Sunday.

2) Replacing NaN values for CompetitionDistance: Since no record exists where CompetitionDistance is NaN and CompetitionOpenSince[X] is not NaN, the Competition- Distance was set to 0 if it was NaN.

3) Replacing NaN values for CompetitionSince[X]: If CompetitionDistance is not 0 but the CompetitionSince[X] columns are missing, the earliest possible values were inserted i.e. 1900 and 01 for CompetitionSinceYear & CompetitionSinceMonth respectively.

4) Converting all 0 (number) values in StateHoliday to "0" (string) values: Data in the StateHoliday is a mix of numerical and string values. Hence, for the sake of consistency, all values were converted to a string.

One-Hot Encoding

Using the pandas.get_dummies() function available in the Pandas library for Python, we performed one-hot encoding on the categorical features present in our dataset, including DayOfWeek & StateHoliday in train.csv & test.csv and StoreType & Assortment in store.csv.

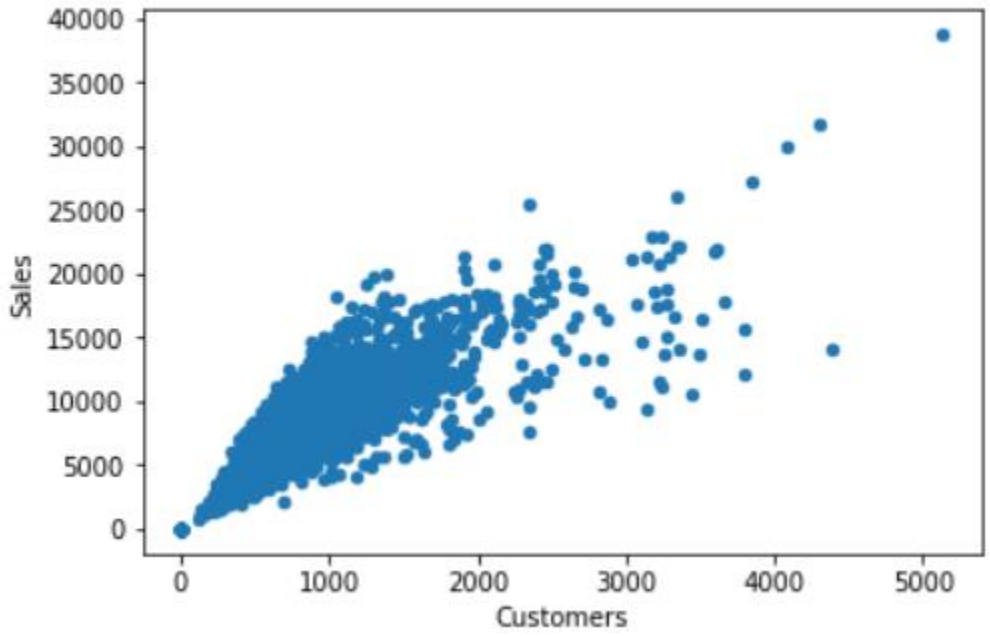
Methods

Random Forest

Random Forest Tree tries is to construct a multitude of decision trees. Then it classifies the data into the decision tree node and for each node it calculate the mean value and use this value for prediction. Random forest tree uses random amount of data for training. With this randomized data, it is hard for random forest tree to overfit. Therefore, it is much easier to tune the data compared to Gradient Boosting Tree.

Linear Regression

This model is a rudimentary first look into the large-scale trends. We did not expect it to capture the granular movements of the sales numbers and indeed it didn't, reporting an average RMSPE of 6.8% Shown in Figure below is the trend observations for Customers v/s Sales.



Bagging Regressor

Bootstrap aggregating also called bagging, is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It also reduces variance and helps to avoid overfitting. Although it is usually applied to decision tree methods, it can be used with any type of method. Bagging is a special case of the model averaging approach.

Neural Network

We have used Keras sequential model to build a neural network to predict the sales for the Rossmann stores. The sequential model is a linear stack of layers. In our model we have developed a multilayer perceptron model with 4 layers and relu as the activation function.

Timeseries Analysis using Prophet

Prophet is an open source forecasting tool built by Facebook. It can be used for time series modeling and forecasting trends into the future. Unlike typical time-series methods like ARIMA (which are considered generative models), Prophet uses something called an additive regression model. This is essentially a sophisticated curve-fitting model. Also, Prophet builds separate components for the trend, yearly seasonality, and weekly seasonality in the time series (with holidays as an optional fourth component).

Results

We have used three different models: Linear Regression, Random Forest Regressor and Bagging Regressor. Also, we have implemented Keras sequential recurrent neural network model with backend as a TensorFlow. While comparing the performance of Keras Sequential Model, we have spilt merged dataset train.csv and store.csv into 3:7 (train: test)

Model	RMSPE	Accuracy
Linear Regressor with single feature	0.279	67.88%
Linear Regressor with multiple feature	0.061	90.56%
Random forest regressor	0.068	91.24%
Bagging regressor	0.104	60.36%
Bagging regressor with PCA	0.052	95.56%
Keras Sequential Model	N/A	Training Set:54.19% and Test Set:53.19%

Conclusion

The Rossmann store challenge was very interesting data science project. We analyzed the data for trends to make feature selection. We also worked with several models using different features and assumptions. We concluded that ensemble models with PCA (Principal component analysis) improves the prediction accuracy considerably. Our highest performing method was the bagging regressor which displayed the lowest amount of testing error. For the bagging regressor, we achieved the best results with the linear kernel, which achieved the best balance between overfitting and underfitting.

References

Notebook references

https://github.com/cutd/Rossmann-Store-Sales

https://www.kaggle.com/novikovanastya/submission-novikova

https://www.kaggle.com/evgenyvasilyev/evgeny-vasilyev

https://www.kaggle.com/dynamic22/rossman-store-sales-basic

Bagging Regressor

https://machinelearningmastery.com/ensemble-machine-learning-algorithms-python-scikit- learn/

http://scikitlearn.org/stable/modules/generated/sklearn.ensemble.BaggingRegressor.html

Time Series Analysis

https://facebook.github.io/prophet/docs/installation.html

https://www.youtube.com/watch?v=95-HMzxsghY

https://blog.exploratory.io/an-introduction-to-time-series-forecasting-with-prophet-package-in-exploratory-129ed0c12112

https://facebook.github.io/prophet/docs/quick_start.html