Rossmann Sales Prediction

# Abstract

In this project, machine learning algorithms are applied to a real-world problem of predicting stores sales. This prediction can help store managers to make effective staff schedules increasing the productivity, to make decisions regarding supply chain management, to make new investments. Open source programming language R is used all along the project. Feature and model selection are used to improve prediction results. Root Mean Square Error (RMSE) of the models is used in the analysis to measure the prediction accuracy.

*Keywords: Random Forest, Regression, Binning, Handling, RMSE*

# Rossmann Sales Prediction

This paper is based on the Rossman stores Kaggle competition. Rossmann is an European drug store with over 3000 locations. The company provides the sales and store data for 1,115 stores of nearly 3 years including holiday promotions, competitors etc. The goal is to predict daily sales of each Rossmann store in advance. These sale predictions can be used to arrange supply chain management, investment and workforce for stores.

Firstly, we analyzed and understood the data and relationships between the attributes, then the pre-processing using R Code is done to remove the unnecessary data, also the data is prepared to be given as input to build a prediction model. Lastly, we compared the prediction models built and with the best prediction model, sales are predicted for the test data provided.

# Dataset

**Description and Purpose**

We are provided with historical sales data for 1,115 Rossmann stores. The task is to forecast the "Sales" column for the test set. **Source** [**https://www.kaggle.com/anshumanyp/rossman/data**](https://www.kaggle.com/anshumanyp/rossman/data)

# Business Research/Understanding

**Project Objectives**

**Problem Domain.**

Predicting sales is one of the key challenges. It is important for firms to predict customer demands to offer the right product at the right time and at the right place. In our case, sales forecasts help store managers to make effective staff schedules increasing the productivity, to make decisions regarding supply chain management, to make new investments.

**Restrictions.** The variables in the train and test datasets are not the same. This may result in the false prediction if not taken care of. Since, store dataset has many missing values the prediction will be biased.

**Data Mining Problem Definition.**

Understanding the relationship between the attributes, mainly relationship of the sales attribute with other variables and building the prediction model with the train data and applying the model to test data to predict future sales values.

**Strategy**. The merged dataset with train and store data values is considered, the data is partitioned into train and test data. The models are built on the partitioned train data and test values are predicted. Thus, accuracy of the models is calculated and compared, from which we can decide the best model and apply it to the actual test dataset to predict sales.

# Data Understanding

**Exploratory Data Analysis Description of the Data.**

Files Used:

* train.csv - historical data including Sales
* test.csv - historical data excluding Sales
* store.csv - Information about the stores

Data fields:

Most of the fields are self-explanatory. The following are descriptions for those that

aren't.

* Id - an Id that represents a (Store, Date) duple within the test set
* Store - a unique Id for each store
* Sales - the turnover for any given day (this is what you are predicting)
* Customers - the number of customers on a given day
* Open - an indicator for whether the store was open: 0 = closed, 1 = open
* 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
* 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
* CompetitionOpenSince[Month/Year] - gives the approximate year and month of the time the nearest competitor was opened
* Promo - indicates whether a store is running a promo on that day
* Promo2 - Promo2 is a continuing and consecutive promotion for some stores: 0 = store is not participating, 1 = store is participating
* Promo2Since[Year/Week] - describes the year and calendar week when the store started participating in Promo2
* 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

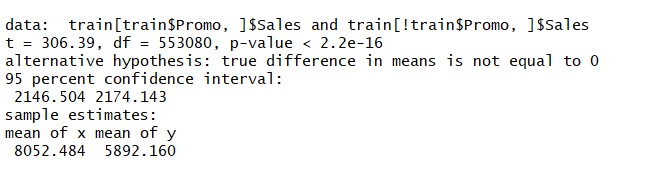
**Estimation.**

Few observations are made as described below:

I. We can observe that there is no considerable variation on the number of customers when analyzed with and without promo, but there is a significant change in Sales when promos are applied, so we can understand from the above analysis that promos are not attracting more customers instead make customers spend more.

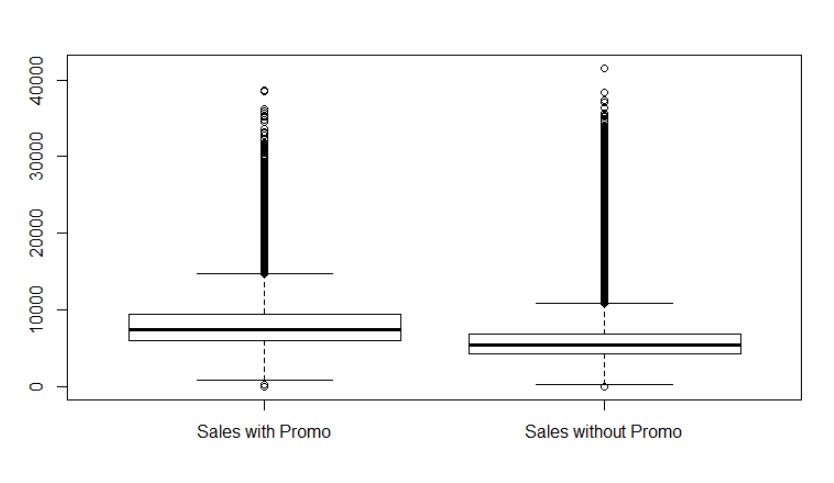
* Performing ttest for Sales with Promo and without promo

t.test(train[train$Promo,]$Sales,train[!train$Promo,]$Sales)



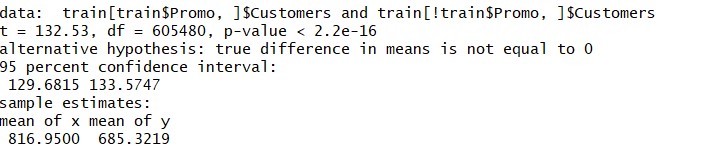
* Boxplot between Sales and Promo:

boxplot(train[train$Promo,]$Sales, train[!train$Promo,]$Sales, names= c("Sales with Promo", "Sales without Promo"))



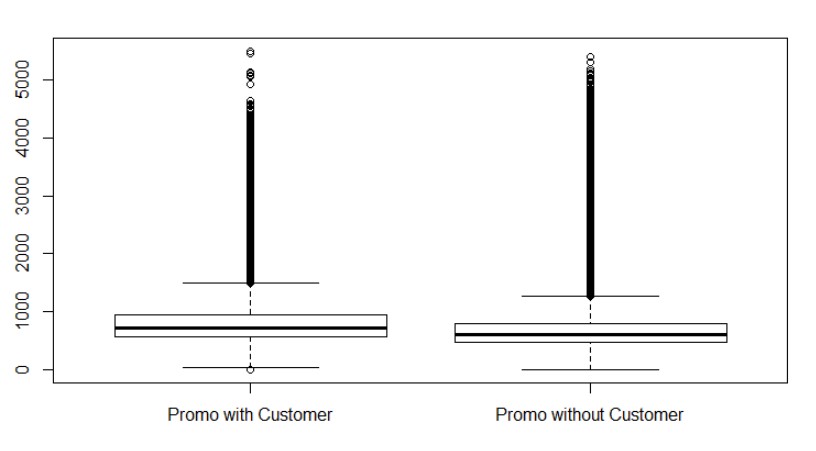
* Performing ttest for with Promo and without promo w.r.t. Customers

t.test(train[train$Promo,]$Customers,train[!train$Promo,]$Customers)



* Boxplot between Promo and Customers:

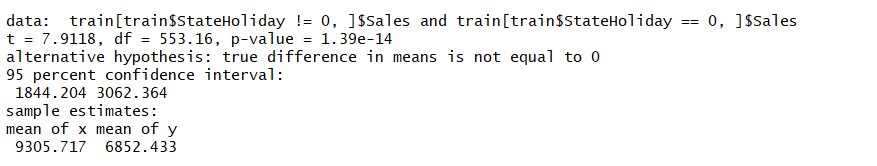
boxplot(train[train$Promo,]$Customers, train[!train$Promo,]$Customers, names= c("Sales with Customer", "Sales without Customer"))



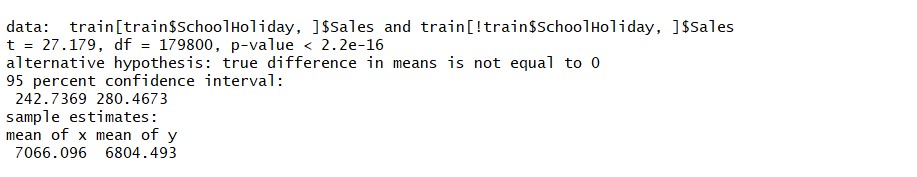
1. State holidays of all types have a very large effect on sales as well, whereas the impact of school holidays is relatively negligible, although statistically still highly significant.

• Testing the same for School Holiday and State Holiday

t.test(train[train$StateHoliday != 0,]$Sales,train[train$StateHoliday == 0,]$Sales)



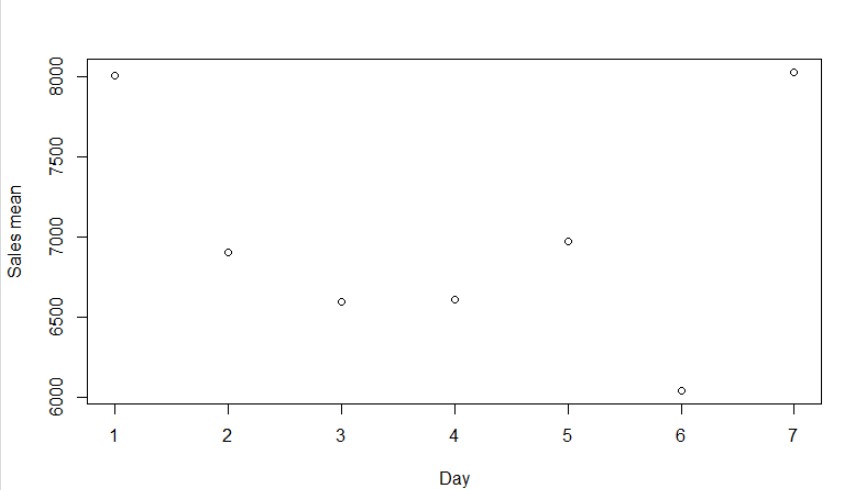
t.test(train[train$SchoolHoliday,]$Sales,train[!train$SchoolHoliday,]$Sales)



1. Sales are highest on Mondays and Sundays and at a very even level from Tuesday to Friday. Sales are lowest on Saturdays, which might have to do with the fact that some stores close earlier on Saturday. At times when very good sales can be expected, which probably explains the high sales seen on Sundays. tapply(train$Sales,train$DayOfWeek,mean)

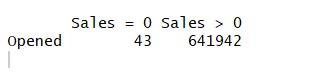


plot(tapply(train$Sales,train$DayOfWeek,mean),xlab="Day",ylab="Sales mean")



1. There are Some stores which made no sales although they were opened even if they had some customers. These observations *may* be errors in the data / outliers:

table(ifelse(train$Open == 1, "Opened"), ifelse(train$Sales > 0, "Sales > 0", "Sales = 0"))



**Data Quality.**

In the above boxplots, the outliers are detected. Promo variable has more outliers compared to other variables. From this we can conclude that the variable is not distributed evenly.

# Data Preparation

**Cleansing.**

Sales of Closed Shops:

There is no need to predict the sales of closed shops since the sales of closed shops is zero. So, the data of the closed shops can be removed from the dataset which reduces the size of the dataset by a significant value.

Shops absent from the test data:

Not all shops in the test data are used. There are 48 days for all stores in the test set, but only 40800 rows, which means that we only need to generate models for 850 stores instead of the total of 1115.

So, we filter out the data for unused stores. By this we can reduce the size of the data that needs to be preprocessed.

Customers:

The customers variable in the data set serves no purpose, as it is not given in the input variables in the test set, fitting a model to this variable would be useless from the perspective of the task. On the other hand, the task is not to predict the number of customers, so fitting a model based on the given variables in the test set to predict the number of customers surely makes no sense. Therefore, we can drop this variable from the train dataset.

**Binning and Discretization.**

Binning is done for the sales output variables as low and high to implement the logistic regression model.

**Missing Values.**

Missing values for Open variable in test data:

In the test data there are some NA values in the Open variable. These values should be filled to get proper predictions of the data. We can observe that all the NA values are to a single store 622, and they are for all other days except for Sundays in this period. Since the intervening Sundays are explicitly marked as 0. Therefore, we can assume that these days are all Open days and can set the A values to 1.

**New Variables.**

Handling date attribute:

Since the algorithms cannot handle dates as an input variable directly, they should either be converted into character or numerical variables. Here the date is converted to three different numerical variables: year, month, day.

**Other transformations.**

Assuming the data in the train.csv as entire data, it is divided into two sets as train and test data with train data being 60%, test data being 40% respectively.

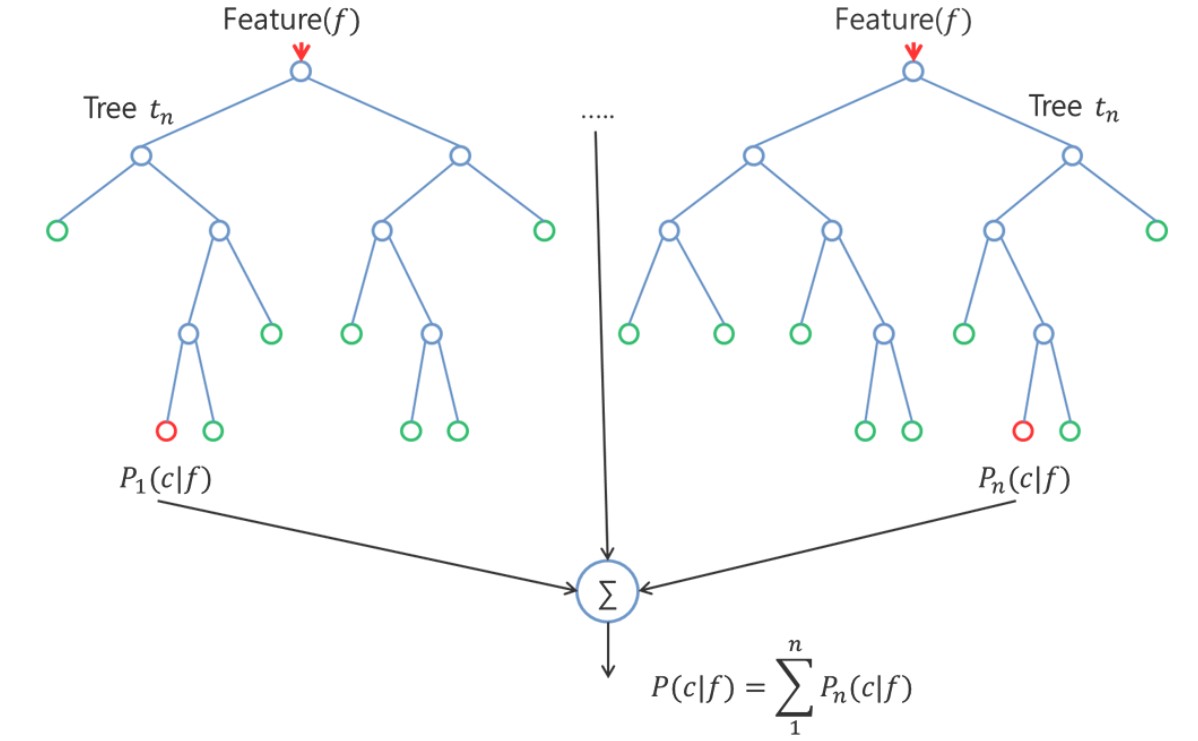
Using the model built on the train data the test data sales are predicted and the model accuracy is noted. Thus, among these models built, best model is taken and the sales values for the test.csv dataset are predicted.

# Modeling

Prediction using Random Forests:

Random Forests are used for the prediction as this algorithm helps to improve the performance of decision trees.

The algorithm like a normal algorithm starts by building out trees by making a split with small random subset of features instead of the full set of features. Multiple trees are built using this process and average of these trees is considered as final model.



## Figure 1

R-Code:

library(randomForest) x <- sample(1:nrow(traindata))

# RandomForest set.seed(32) rf <- randomForest(Sales ~ ., data = traindata, mtry = 2, importance = TRUE, ntree = 20) print(rf)

#Rf <- randomForest(Sales ~ ., traindata , ntree= 100, mtry= 4 ,importance=TRUE, do.trace=50 ) pred <- predict(rf, testdata) summary(pred)

str(pred) sqrt(mean((testdata$Sales - pred) ^ 2))

Classification using Logistic Regression:

To classify the sales output variable the logistic regression approach is used, this approach categorizes the output to two classes. The output variable is binned into two bins: sales <18k, sales>18k.

R-Code:

#Logistic regression library(dplyr)

#Converting Sales variable from continuous to categorical traindata$Category <- cut(traindata$Sales, breaks = c(0,16000,Inf), labels = c("low","high"))

#fitting logistic regression model <- glm (Category ~ ., data = traindata, family = binomial) summary(model)

#predicting the model predict <- predict(model,type = 'response') predict

View(traindata) str(traindata)

#evaluating the accuracy using confusion matrix traindata <- na.omit(traindata) table(traindata$Category, predict > 0.5) #evaluating Logistic Regression library(pscl) pR2(model)

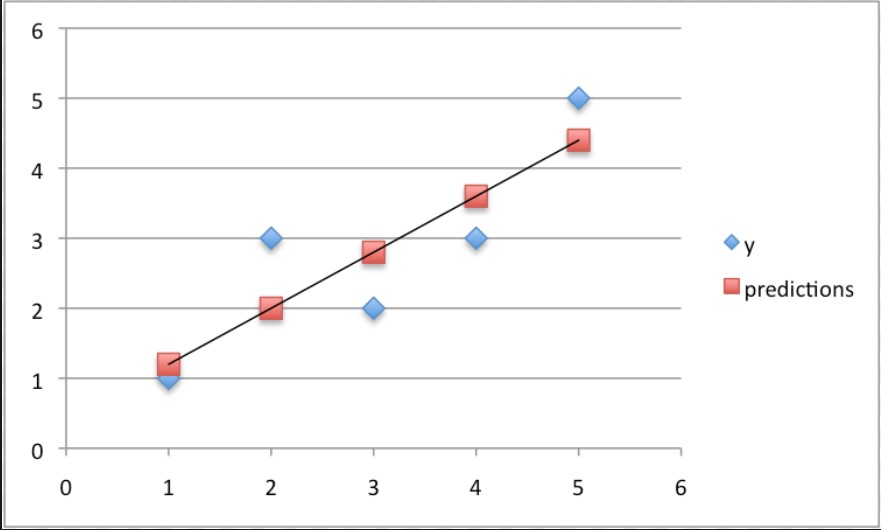
#Consider the McFadden value, as it is close to 1, the predictive power of this model is high.

#ROCR Curve library(ROCR)

ROCRpred <- prediction(predict, traindata$Category) ROCRperf <- performance(ROCRpred, 'tpr','fpr') plot(ROCRperf, colorize = TRUE, text.adj = c(-0.2,1.7))

#plot glm library(ggplot2) ggplot(traindata, aes(x=Sales, y=Category)) + geom\_point() + stat\_smooth(method="glm", family="binomial", se=FALSE) Prediction using Linear Regression:

In general, Linear regression is used to predict continuous variables. In this case, linear regression is applied as sales is the output variable to be predicted which is continuous.



## Figure 2

R-Code:

trainmodel <- lm(Sales ~ ., data = traindata) options(warn = -1) predictmodel <- predict(trainmodel, testdata) options(warn = 1) sqrt(mean((testdata$Sales - predictmodel) ^ 2)) **Evaluation**

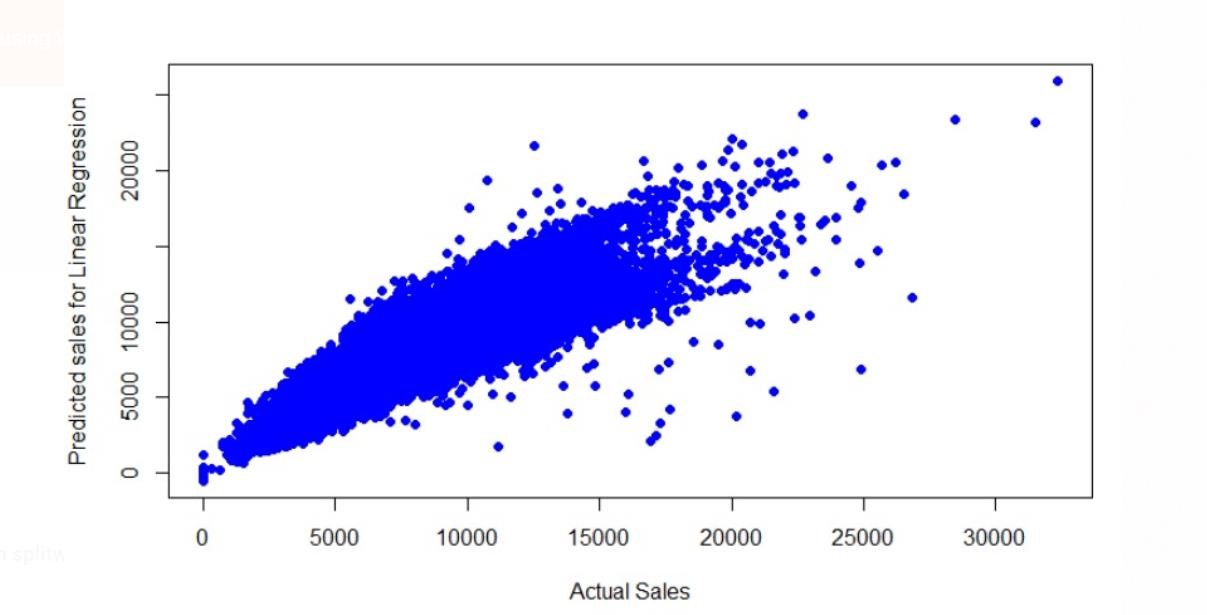
**Results.** (discuss the results of the modeling that has been performed)

Linear Regression:

The root-mean square error for the model is calculated as below:

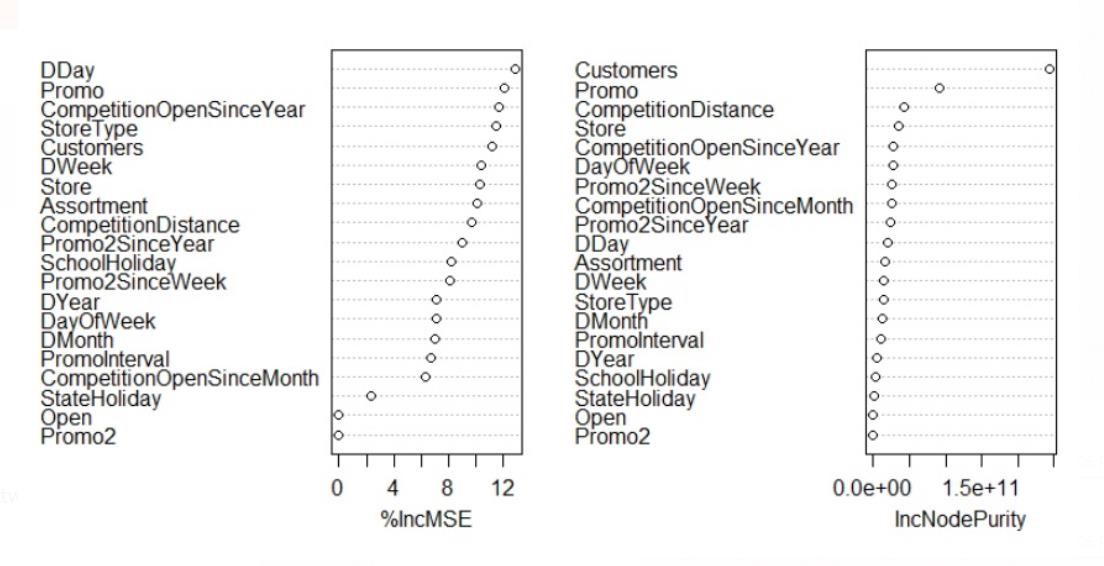


## Scatterplot of the predicted sales vs actual sales

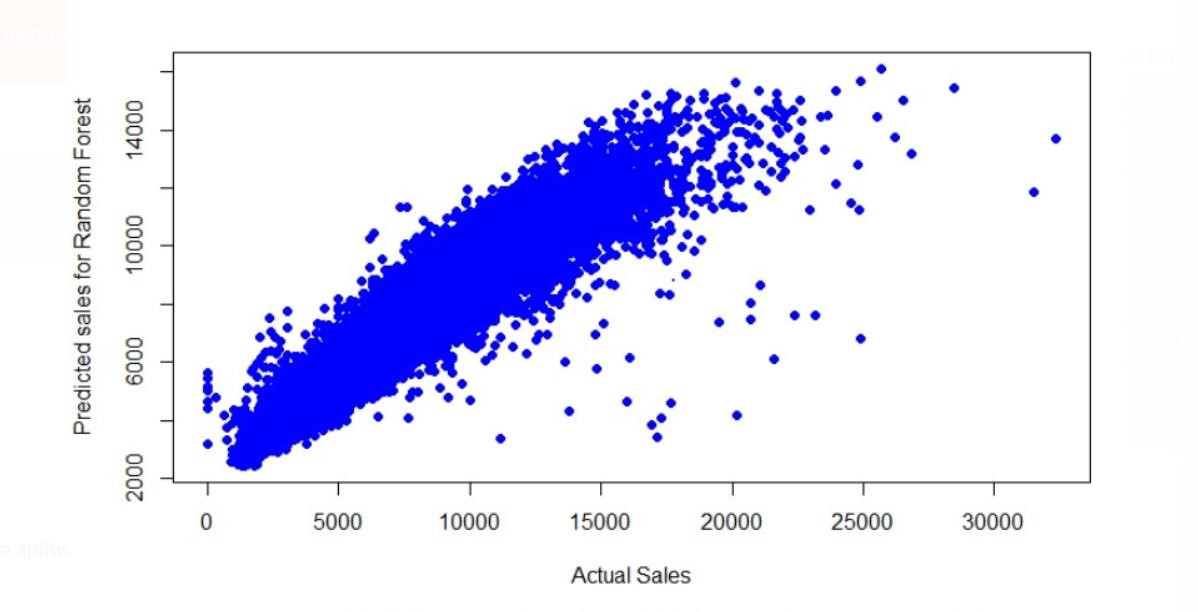


Random Forest algorithm:

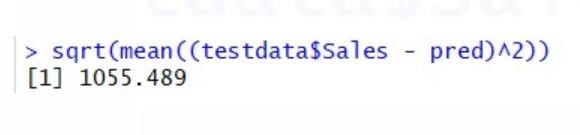
The promo, DDay and Competition Distance are main factors effecting the output variable Sales as shown in the below Variable importance plot.



## Scatterplot of Actual sales vs prediction sales using Random forest



The root-mean square error for the model is calculated as below:



# Report of Results

**Knowledge discovered.**

When compared the model results and the root mean square values of the respective models, the Random forest algorithm stands out because it has low RMS value i.e.1055.489

**Predictive Capabilities.**

Hence the random forest model is used to predict the sales variable in the main test dataset.

**Limitations**.

Logistic regression done in the analysis just classifies the output variable and is not of much use in this case.

**Future Work/Conclusion**.

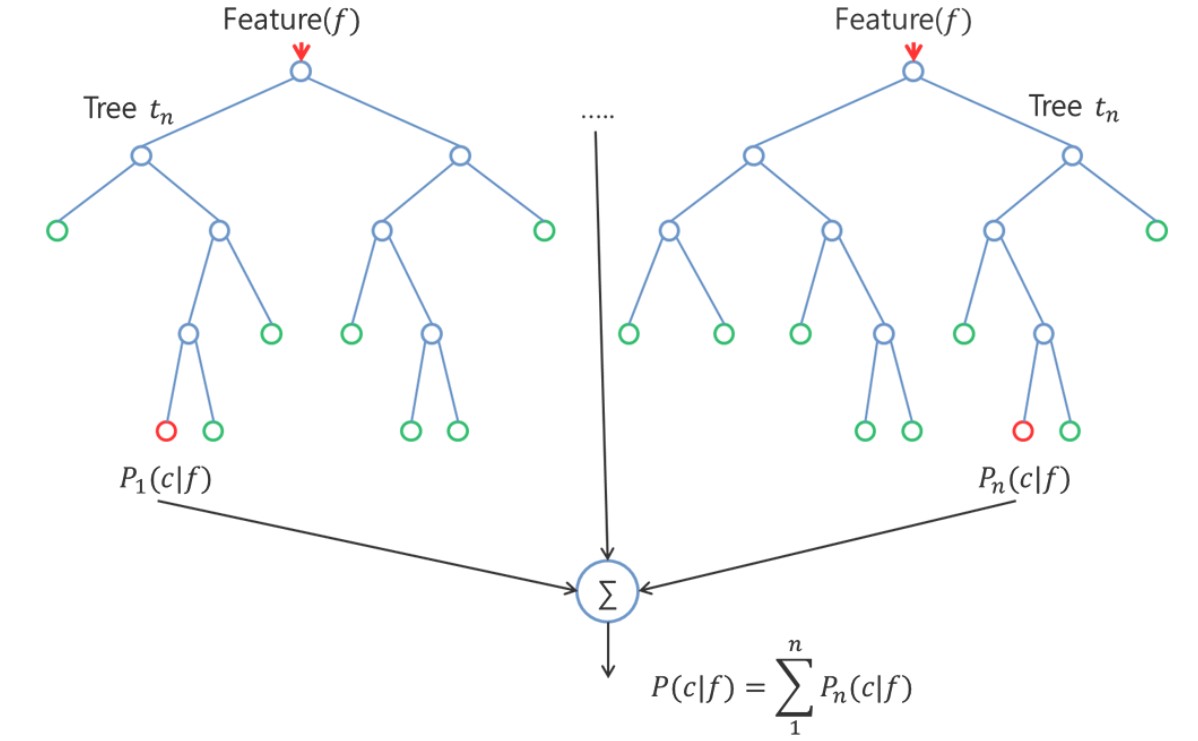
Promotions play a vital role in the sales prediction. Sales number of a day is also related to the sales number before that day. Model built with Random forest algorithm gives valid result values. Adding time series to the feature vector of the model can improve accuracy.

## Références

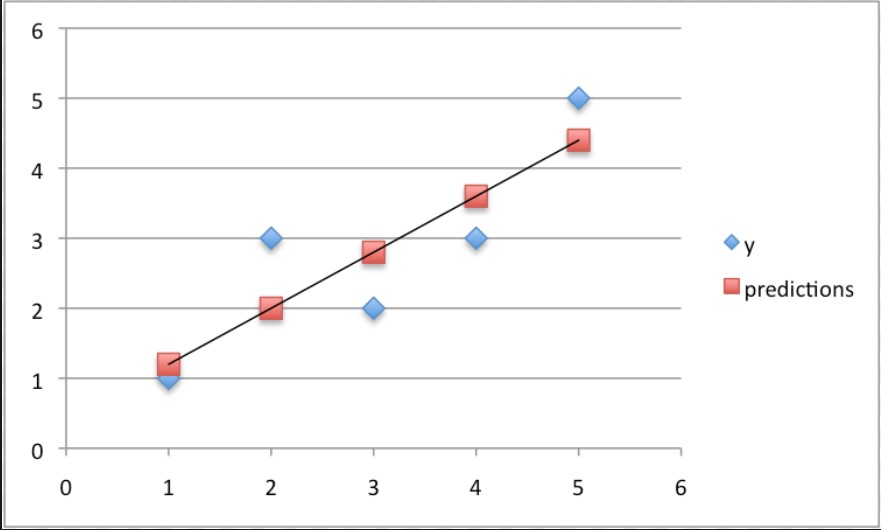
[1][http://rstudiopubstatic.s3.amazonaws.com/142772\_218508e3f94e4419944e9104505b6f17.ht ml](http://rstudiopubstatic.s3.amazonaws.com/142772_218508e3f94e4419944e9104505b6f17.html)

1. <http://r-statistics.co/Linear-Regression.html>
2. <http://rstatistics.net/linear-regression-with-r-a-numeric-example/>

[4][http://yilinwei.com/project/Prediction\_of\_Rossmann\_Store\_Sales/Prediction\_of\_Rossmann\_S tore\_Sales\_Report.pdf](http://yilinwei.com/project/Prediction_of_Rossmann_Store_Sales/Prediction_of_Rossmann_Store_Sales_Report.pdf)



*Figure 1.* Random forest algorithm for prediction.



## *Figure 2*: Linear regression for prediction

ROSSMANN SALES PREDICTION 24