**OBJECTIVES**

Using R language, train the model and predict new products sales volume.

**PREPARATORY STAGE**

Initially, we have 2 datasets, one of which we used for the prediction model training and the second one used for prediction purposes.

Most of the operations were conducted with “caret” library of R. Also, used Python for the feature selection tests and the final visualization.

The complete R project is added to the archive with this report.

The data preparation steps I performed:

1. Data Loading
2. Structure Observation
3. Data types Adjusting.

Here I translated factor variables to binary mode. Each variable was assigned “0” or “1”.

1. Feature selection.

At this stage, I tried to avoid overfitting of the Multi Linear Regression model combining x5 – x1 StarReviews into a one column:

xCombined = x5StarReviews \* 2 + x4StarReviews \* 1 + x3StarReviews \* 0 + x2StarReviews \* (-1) + x1StarReviews \* (-2)

Unfortunately, though the Pearson correlation of Volume and xCombined became 0.87, the final result left unchanged – R2 was around 1.

Also, I dropped the following columns:

BestSellerRank (too many NAs), ProductNum (indexing), ProfitMargin (lowest correlation).

1. Normalization.

For the purpose of data scaling I used Min/Max method.

1. 5 CPU cores activation (to boost the speed of the algorithms)

The results of this step were:

* Training dataset consisted of sampled 75% rows of the initial dataset.
* Testing dataset consisted of the rest 25% rows.

**MODEL BUILDING STAGE**

During this step I built total 5 models: LR, SVM (with two different kernels), Random Forest and Gradient Boosting.

**IMPORTANT NOTE:** Due to the lack of a sufficient amount of observations, **set.seed** state played a crucial role in the models’ statistics. I worked with seed values equal 123, 888, 107

**Multi-Linear Regression**

As I wrote in the Preparatory stage, this model appeared overfitted due to star reviews. The attempts to fix it, using feature filtering and feature construction were useless.

|  |
| --- |
| **SVM (linear kernel)** |

**Training**

RMSE Rsquared MAE

1122.503 0.8498755 557.6157

**Testing**

RMSE Rsquared MAE

332.1336280 0.9205558 211.1483167

|  |
| --- |
| **SVM (RBF kernel)** |

**Training**

Resampling results across tuning parameters:

C RMSE Rsquared MAE

0.25 1034.2658 0.6367332 585.6108

0.50 1004.2243 0.6857448 571.0125

1.00 986.9742 0.6854058 563.8956

|  |
| --- |
| **VARIABLES IMPORTACE CHART FOR “SVM : RBF” LOOKS IDENTICALLY TO “SVM : LINEAR”** |

**Testing**

RMSE Rsquared MAE

372.2687195 0.7703832 245.2214731

|  |
| --- |
| **Random Forest** |

**Training**

mtry RMSE Rsquared MAE

2 802.2662 0.8687749 440.7870

3 777.2767 0.8964338 400.4031

5 737.1184 0.9211168 359.6128

6 727.2399 0.9260371 349.4357

8 693.3807 0.9425182 326.0895

10 674.5851 0.9503966 313.1383

11 664.8600 0.9528144 306.7383

13 650.7984 0.9588471 297.4902

15 636.2524 0.9621867 288.5368

**Testing**

RMSE Rsquared MAE

471.8273967 0.8364649 164.0470386

|  |
| --- |
| **Gradient Boosting** |

**Training**

interaction.depth n.trees RMSE Rsquared MAE

1 50 1002.7577 0.7762999 595.3035

1 100 1053.9730 0.7419680 641.8267

1 150 1085.8330 0.7165156 666.2534

1 200 1131.8139 0.6874144 697.0682

1 250 1152.8781 0.6726426 709.5959

2 50 994.3429 0.8130762 586.8190

**Testing**

RMSE Rsquared MAE

416.0315404 0.8318738 261.1407002

**CONCLUSION**

According to the experiments conducted, I have chosen Gradient Boosting model as one the most suitable for this task.

After applying the model to the “new\_products” dataset, the following outcomes were obtained: