Predicting Customer Preferences (C2)

Lessons Learned

#### John T. Leonard, Krishna Venkatraman, Ross Tomlin

#### March 13, 2019



Introduction

In this report, we review the key insights from our studies on predicting customer preferences for Blackwell Electronics, including lessons learned for three related tasks.

**Task 2: Predicting Brand Preference**

In this task, various classification models were tested to predict customers’ brand preference between Acer and Sony. Preprocessing the data and distinguishing features that were continuous versus categorical helped ensure proper model interpretation of the data. The continuous data was scaled using the standard deviation to ensure each feature of interest had a similar range of units, thereby preventing the models from over-emphasizing the importance of one feature due to its scale. Two models were tested and trained: C5.0 decision tree and random forest (fold = 10). The C5.0 model employed an automatic tuning grid while for the random forest used a manual tuning grid (mtry = 1, 2, 3, 4, 5), where mtry represents the number of variables randomly sampled as candidates at each split. For the C5.0 decision tree, we also defined the tune length hyperparameter to be equal to 2. Both models demonstrated a high degree of accuracy and kappa scores on the testing set while additional analysis was performed on the relative importance of each variable in making predictions (see Appendix). Taking all this into account, the random forest proved superior, illustrating the shortcomings of relying purely on the train-test metrics and the need to understand a given model’s weighting of each feature.

Task 2 lessons learned:

1. Understanding data type is critical to building effective ML models
2. Scaling continuous data is important to avoid biasing one feature over others
3. Analyzing variable importance is an important step in evaluating how models work and assessing their likely success in generalizing.

**Task 3: Predicting New Product Sales Volume**

Multiple regression in R was used to predict the volume of sales for new products. One-hot encoding of the relevant features generated a correlation matrix (see Appendix), which revealed those features that are linearly correlated with volume predicted and also those features that are correlated with each other. (Visualization was greatly aided by hierarchical clustering (hclust).) One important takeaway is that 5-, 4-, and 3-star reviews are heavily cross-correlated and also correlated with volume. Further, accessories is strongly negatively correlated with profit margin (i.e. accessory sales reduce profit margin). Additionally, the 5-star reviews feature is perfectly correlated to volume (factor = 1), which suggests there is likely some error in the raw data. This feature, along with any features with cross correlations >95%, were removed from the training process.

Three models—support vector machine (SVM), random forest (RF), and gradient boosted tree (gbTree)—were trained to predict log10(Volume), as transforming the Volume to a log scale allowed us to prevent any of the models from predicting a negative volume (see Appendix for performance metrics). RF proved to be the best model, since it did not appear to be overfitting (unlike the gbTree) while also showing a low RMSE and high R-squared on the test set. The trained RF model enabled predictions on a validation set of new products. The resulting breakdown in total volume vs. product type revealed tablets to be the highest volume products, while printer supplies, displays, extended warranty insurance, accessories, and software would generate poor sales (see Appendix).

Task 3 lessons learned:

1. One-hot encoding/dummification is an important tool to allow models to leverage character based categorical data.
2. Correlation matrices are simple and efficient analytical elements that give strong insight into datasets with large numbers of features.
3. Transforming the label you are trying to predict into logarithmic units is an effective way to prevent ML models from predicting non-physical/real negative values for that label.

**Task 4: Discover Associations Between Products**

The apriori algorithm was used to obtain insights over Electronidex’s transactional datasets and ultimately determined whether acquiring the company makes business sense. The apriori algorithm assesses association rules using support measurement (itemsets/rules frequency within transactional data) and confidence measurement (accuracy of the rules). A strong rule rates high in both support and confidence.

* Discoveries from market basket analysis using Apriori algorithm identifies rules that are insightful and irrelevant.
* Usage of apriori() function to leverage Apriori algorithm to find association rules using Support & Confidence measures.
* Lift measures the importance of a rule. A high value for lift strongly indicates that the rule is important.
* inspect() function is used to summarize all relevant options, plots and statistics that should be usually considered.
* plot() function can be utilized to visualize a subset of your related rules.

Insights obtained indicate that Blackwell can increase its sales by cross-selling Electronidex products. Therefore, acquiring the company seems prudent.

Task 4 lessons learned:

1. …
2. …

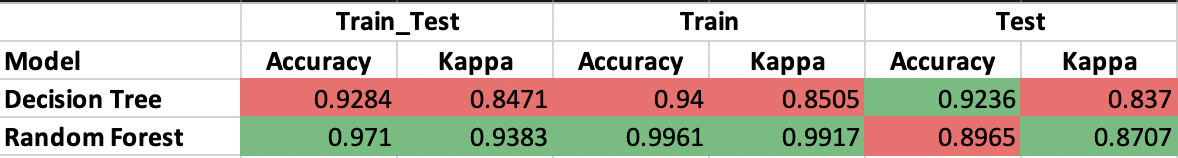
**Conclusion and Recommendations**

These three tasks offered a number of important insights into building machine learning models and performing data analytics generally. Furthermore, they offered very useful practical experience in getting comfortable with code-based data analytics pipeline building in R. In each task, understanding the data was a critical first step. Because of this, one of our recommendations to other specialists is to get in the habit of building data analytics pipelines using functions. This allows you to recycle code very easily and speed up your analytical procedure. This is particularly useful for performing common data visualization algorithms. Another important recommendation is related to analyzing the train-test error from the ML algorithms. Specifically, in a number of cases, choosing the best model was significantly aided by plotting the prediction vs. the true values from the train-test sets. Our final recommendation is something often overlooking in building pipelines: make your code clean and comment it frequently. To make your pipelines have the most value, you need to make them legible to others, and thus it is always important to make it very obvious what is going on in your code and the reason for.

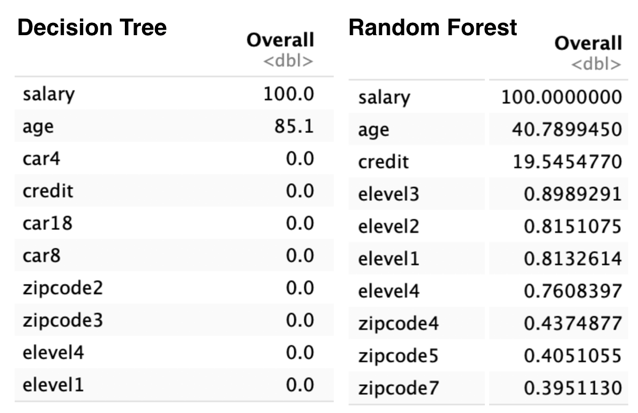
In conclusion, through these tasks, we believe we have not only strengthened our own skills in data analytics, but have also brought critical insights to Blackwell Electronics. These insights are undoubtedly going to make a power impact on Blackwell’s sales and profits, paving the way for deeper and more powerful machine learning models at Blackwell Electronics in the future.

Appendix

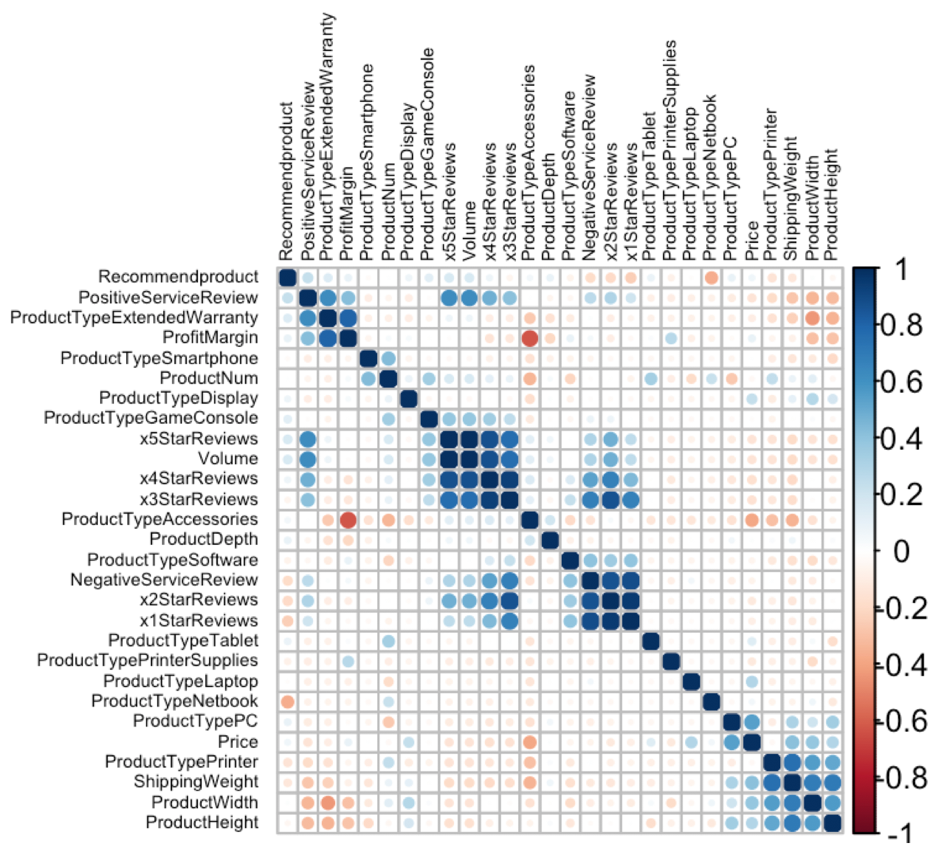
**Metrics summary table (Task 2)**



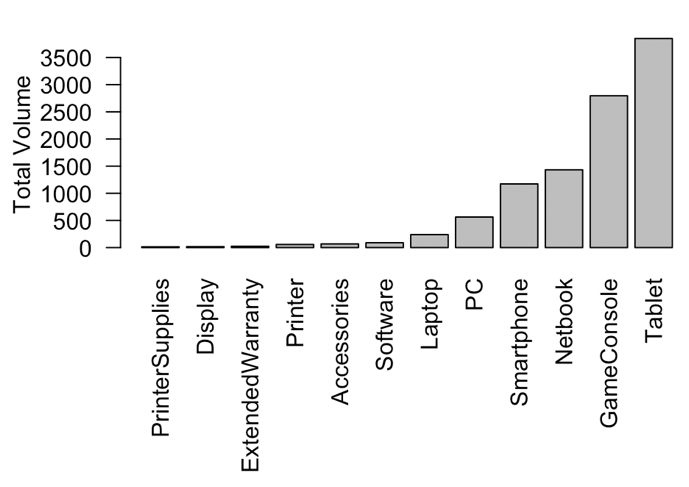
**Variable/feature importance (Task 2)**



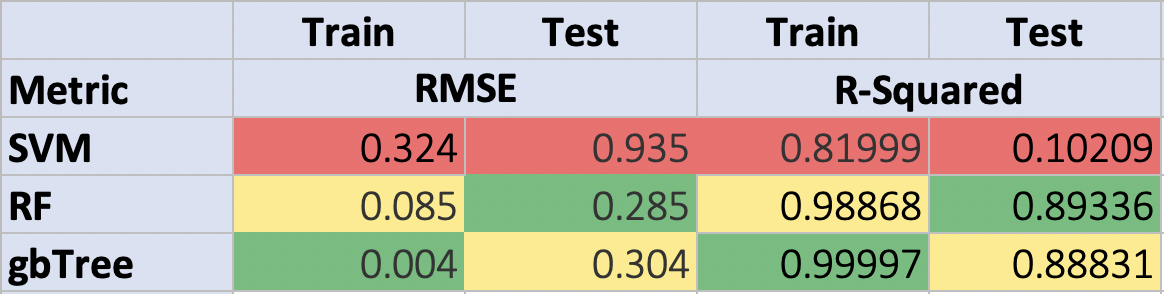
**Correlation matrix (Task 3)**



**Volume vs. product type (Task 3)**



**Metrics vs. models type (Task 3)**



**Relative item frequency (Task 4)**

