Kaggle Report - Regression

Statistics 101C - Introduction to Statistical Models and Data Mining

Professor: Miles Chen

Team: Daniel Do, Khang Thai, Andy Ho, Jessie Yu, Christopher Le

July 28, 2024

Introduction:

In this project, regression analysis was performed based on datasets that contained information about Amazon purchases from the specific demographics of customary, by training the model through the train and test datasets. The primary objective of this project was to predict the response variable log_total, given the explanatory variables from the provided dataset.

From the initial investigation of the datasets, the explanatory variables of count_hh4 and count_G were narrowed, which could have contributed to a strong association of the response variable log_total. These predictions are based on the fact with more orders, there would be an expected positive correlation with the response variable as well. Furthermore, with a larger household and income, it is expected there would be an increase in order volume and total. To understand specific variables affecting the explanatory variable, model evaluation was performed through various data pre-processing, model recipes, and hyperparameters.

Exploratory Data Analysis

To further understand the explanatory variables in the dataset, data visualizations were produced to understand the multicollinearity through the relationship of explanatory variables.

Figure 1 shows a scatter plot of Count vs. Count_hh4 to see the relationship between these two explanatory variables. It is shown that there is a positive correlation of R = 0.9450997, with most of the data congregated at the lower values.

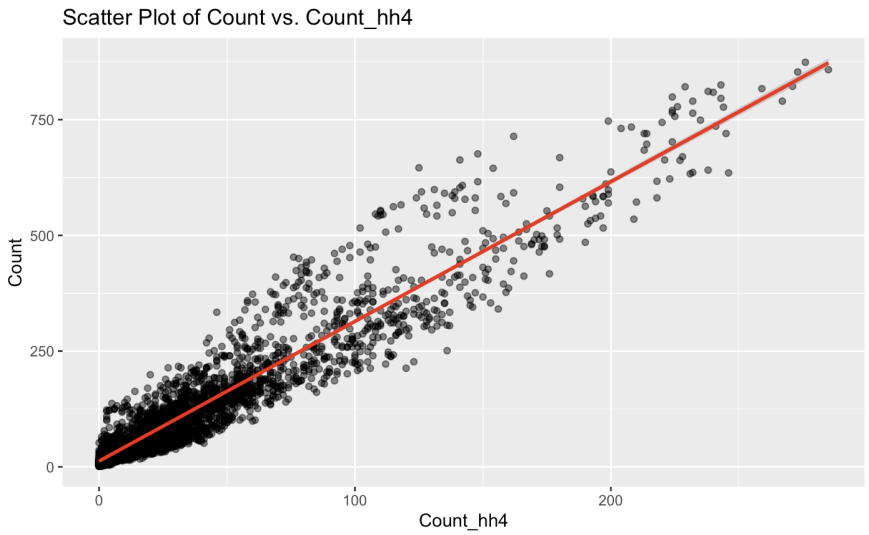

Figure 1: Relationship between Count and Count_hh4

In Figure 2, a scatter plot of Count vs. Count_G was produced to see the relationship between these two explanatory variables. It is shown that there is a positive correlation of R = 0.8974374, with most of the data congregated at the lower values.

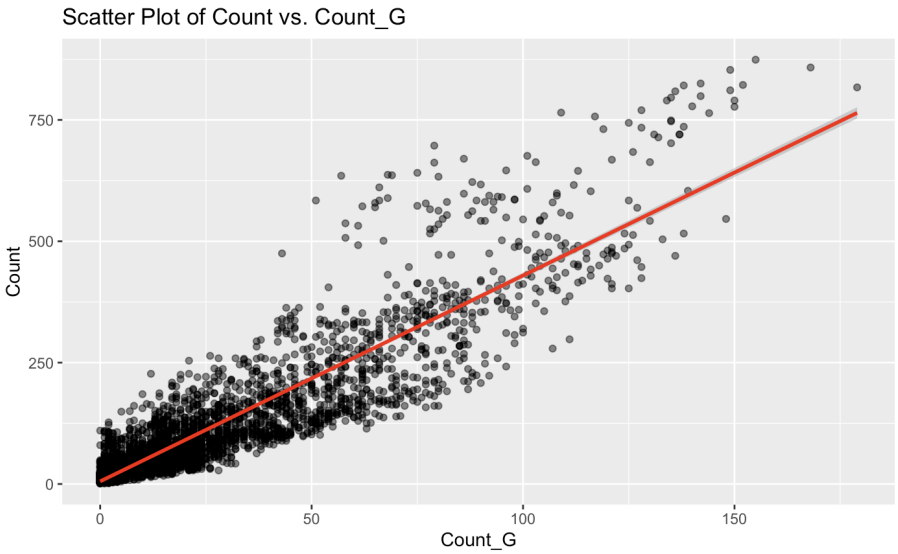

Figure 2: Relationship between Count and Count_G

In Figure 3, a histogram of Count was produced to see the data distribution from the training dataset. It is clearly shown that there is a right skew, suggesting that there should be some pre-processing to account for skewness. Thus, we can apply logistic regression to scale.

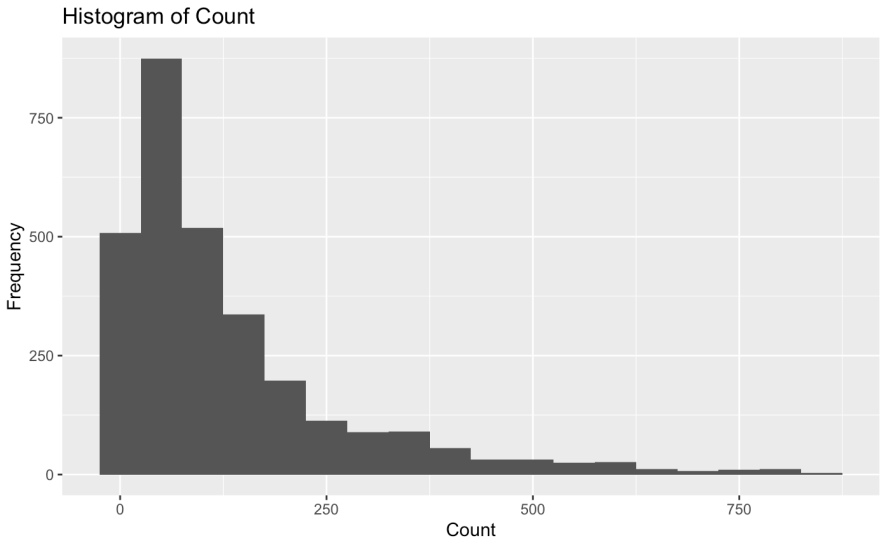

Figure 3: Frequency of Count

In Figure 4, a scatter plot of Log_Total vs. Count was produced to see the relationship between the explanatory and response variables. The scatter plot follows the same behavior as a logarithmic function. It is shown that there is a positive correlation of R = 0.7944187.

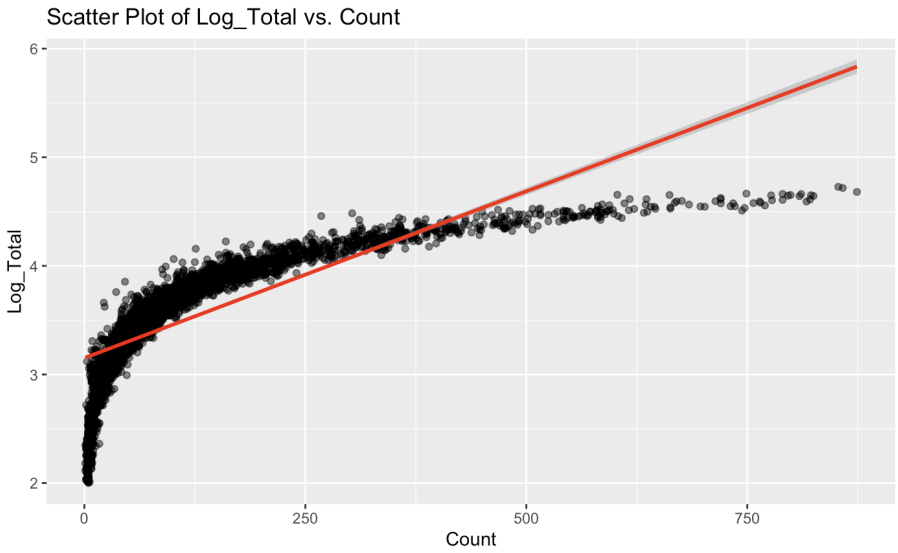

Figure 4: Relationship between Log_Total and Count

In Figure 5, a scatter plot of Log_Total vs. Count_hh4 was produced to see the relationship between the explanatory and response variables. The scatter plot follows the same behavior as a logarithmic function. It is shown that there is a positive correlation of R = 0.761232.

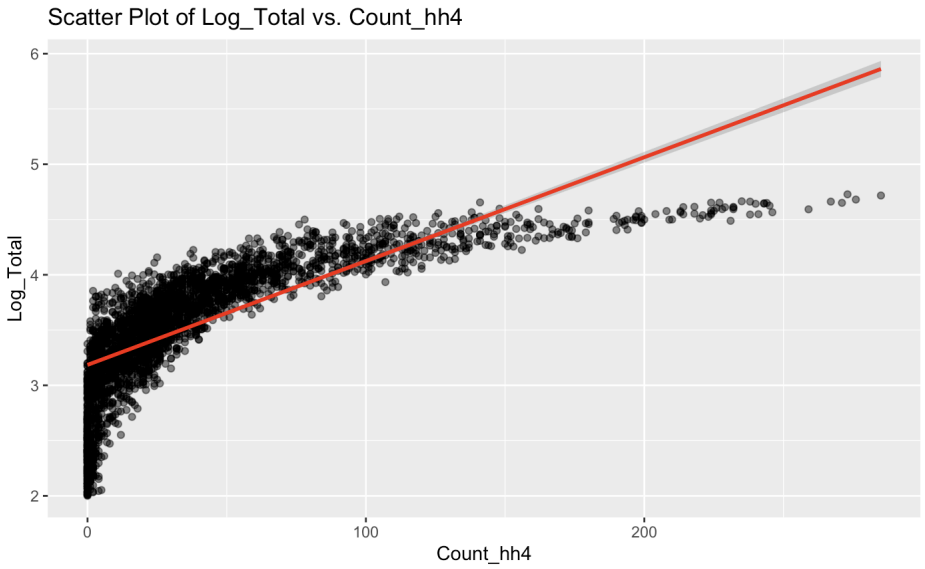

Figure 5: Relationship between Log_Total and Count_hh4

Figure 6 shows a scatter plot of Log_Total vs. Count_G to see the relationship between the explanatory and response variables. The scatter plot follows the same behavior as a logarithmic function. It is shown that there is a positive correlation of R = 0.7784229.

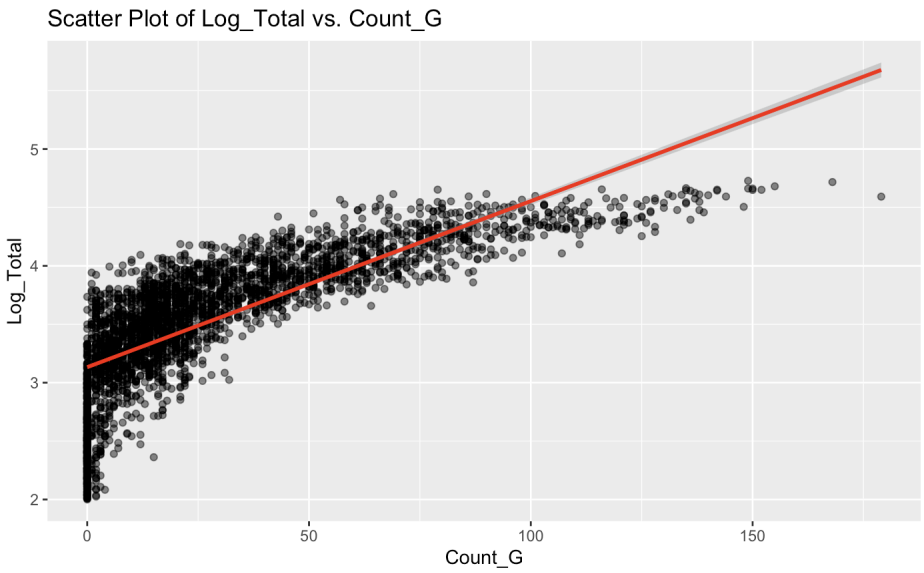

Figure 6: Relationship between Log_Total and Count_G

In Figure 7, a visual correlation map was created to see the relationship between all the variables in the dataset. There are a few variables that have a high correlation with each other, which could lead to multicollinearity problems. Thus, pre-processing should be incorporated.

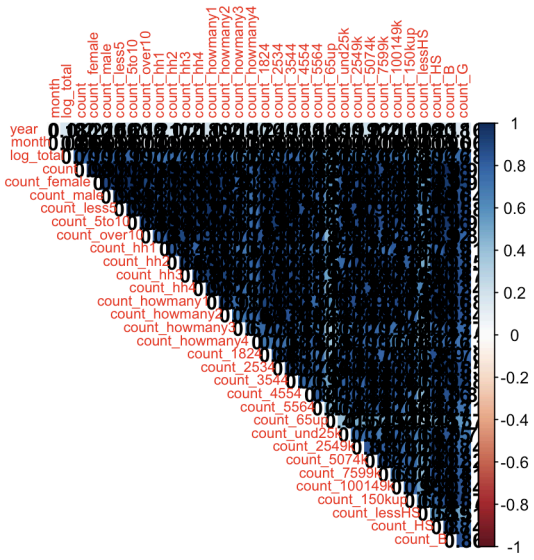

Figure 7: Correlation Map

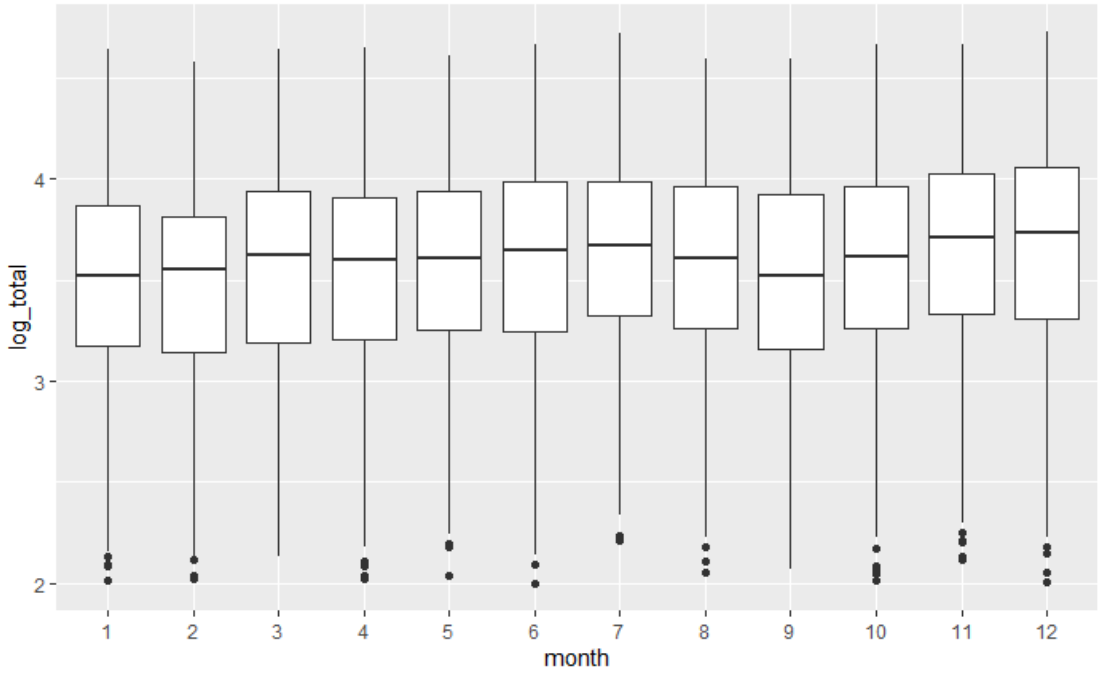

Figure 8: Boxplots of Each Month

In Figure 8, boxplots for each month were constructed to determine if the difference in log_total in December and January were significant, which they were not. Thus, the most ideal approach for this predictor was to normalize it with the other numeric predictors to minimize the scale differences.

Pre Processing

From the various models that were built by the team members, pre-processing was implemented to improve the validity of the predictions produced.

For example, numeric predictors that correlate greater than 0.8 were removed to prevent overfitting and multicollinearity problems. Therefore, cleaning the data by removing highly correlated variables will improve the model by making it more generalizable to which the model can predict new data.

In addition to that, step_dummy was called to convert variables of qualitative form to quantitative form in binary format. This would improve the model by providing more ingredients to the recipe, which would give deeper analytical insight into how explanatory variables affect the response variable.

Also, logarithmic transformations were applied, specifically to the numeric predictor variables to fix the skewness of data in variables. This is demonstrated by the right skewness based on Figure 3. This would enhance the model by reducing the impact of outliers from the explanatory variable affecting the potential prediction in the response variable.

Candidate Models

Model 1:
The model used was a random forest model with the "ranger engine" to predict the variable log_total. This model was used because its flexibility allows for a model with more accurate predictions while simultaneously reducing the risk of overfitting through the usage of numerous trees. The recipe used in this model includes removing numeric predictors that correlate greater than 0.8 and then performing a logarithmic transformation of all numeric predictors to reduce skewness. We then create dummy variables for our categorical predictors which allow our model to handle categorical data appropriately.

Model 2:
The random forest model was used with the "ranger" engine and regression type to predict a continuous target variable, log_total. Random forest model helps construct multiple decision trees during training which helps produce a more accurate model prediction. The "ranger" engine is particularly useful for handling larger datasets for its efficiency and speed. It also reduces the risk of overfitting and can provide insight into the importance of different features in the model. The recipe used in the model includes the normalization of all numeric predictors to help improve the model's performance and the dummy variables for all categorical predictors, enabling the model to handle categorical data effectively. The model then uses these preprocessing steps with the random forest model.

Model 3:
The model selected is a random forest model with a regression type and the "ranger" engine. It is used to predict the value log_total from the rest of the variables known as the predictors. This type of model is used because a random forest model is simple and flexible enough to capture accurate results for regression. Additionally, variance is reduced as the predictions of multiple trees are observed, which decreases the likelihood of overfitting. The hyperparameter, "trees", is used to indicate how many trees are observed. During the recipe, a logarithmic transformation is used to reduce right skewness. Possible interaction terms between count_female and variables starting with "count_hh" are also explored. Any numeric variables with extremely high correlations with each other, with the threshold being 0.8, are removed. Then, scaling and centering these numeric predictors help to normalize the data. Finally, any categorical variable is converted to have a numerical value based on the category.

Model 4:

A regression random forest model with the "ranger" engine was selected. The random forest model was selected because of the high number of predictors and the potential non-linear relationship between the predictors and log_total. The recipe includes creating a dummy variable of the categorical predictor "q_demos_state", inputting any missing data by using the mean of their respective variable, and normalizing all the predictors beginning with "count" (such as "count_hh" or "count_B" to reduce any scalar weight when modeling. The hyperparameters, min_n and trees, were also tuned by selecting the most ideal values from a randomly generated grid of 20 levels.

Model 5:

Another regression random forest model was created from the provided train dataset to predict log_total from the test dataset. This random forest identifier was chosen because it is a fundamentally universal model to fit coefficients to create relationships between explanatory and response variables with more complexity. A logarithmic transformation was applied to the Count explanatory variable to accommodate for the right skewness. The numerical predictors were also centered and scaled to normalize the training data for more refined predictions. Hyperparameters of mtry, trees, and min_n were chosen to further refine the model.

Table 1: Candidate Model Summary

Model / Author	**Model Identifier**	**Type of Model**	**Engine**	**Recipe Used**	**Hyperparameters**
Model 1 / Daniel	rand_forest	Regression	Ranger	log_total ~ . %>% step_corr(all_numeric(),-all_outcomes(), threshold = 0.8) %>% step_log(all_numeric(),-all_outcomes()) %>% step_dummy(all_nominal_predictors())	trees = 500
Model 2 / Khang	rand_forest	Regression	Ranger	log_total ~., data = train) %>% step_normalize(all_numeric_predictors()) %>% step_dummy(all_factor_predictors())	
Model 3 / Andy	rand_forest	Regression	Ranger	amazon_recipe <- recipe(log_total ~., data = amazon_train) %>% step_log(count, base = 10) %>% step_interact(~ count_female:starts_with("count_hh")) %>%	trees = 1000

				step_corr(all_numeric_predictors(), threshold = 0.8) %>% step_scale(all_numeric_predictors) %>% step_center(all_numeric_predictors()) %>% step_dummy(all_nominal_predictors())	
Model 4 / Jessie	rand_forest	Regression	Ranger	amz_basic_rec <- recipe(log_total ~ ., data = amz_training) %>% step_dummy(q_demos_state) %>% step_impute_mean() %>% step_normalize(all_numeric_predictors()) %>% step_corr(all_numeric_predictors(), threshold = 0.90)	Min_n = 11 trees = 429 (via random_grid)
Model 5 / Chris	rand_forest	Regression	Ranger	recipe(log_total ~ ., data = train) %>% step_log(count, base = 10) %>% step_center(all_numeric_predictors()) %>% step_scale(all_numeric_predictors()) %>% step_corr(all_numeric_predictors(), threshold = 0.8)	mtry = 5, trees = 500, min_n = 3

Model Evaluation and Tuning

For the 5 models, v-fold cross-validation was performed to split the training dataset. By performing v-fold cross-validation, the produced regression models can be controlled on how cross-validation and resampling will perform. Specifically, we set v = 10 and also set the strata argument to be the "log_total" variable from our data set. From there, the performance metric is calculated for each of the models to determine the accuracy of the models shown in Table 2.

Table 2: Performance of Models

Model	**Model Identifier**	**Metric score**	**SE**
Model 1	rand_forest	0.1599200	0.003598526
Model 2	rand_forest	0.1128934	0.003002182
Model 3	rand_forest	0.1138717	0.002130406
Model 4	rand_forest	0.1214912	0.002054271

Model 5	rand_forest	0.1146292	0.00310722

From the results in Table 2, we can see that Model 2 has the lowest metric score, which is the root mean square error (rmse).

Autoplot

To further understand the efficiency of the models, data visualization was produced of autoplot of models based on Table 2. It is shown that Workflow Rank 1 is the lowest, with the lowest rmse that was lower than 0.12. Overall, the rmse of the models ranged around 0.11 - 0.17.

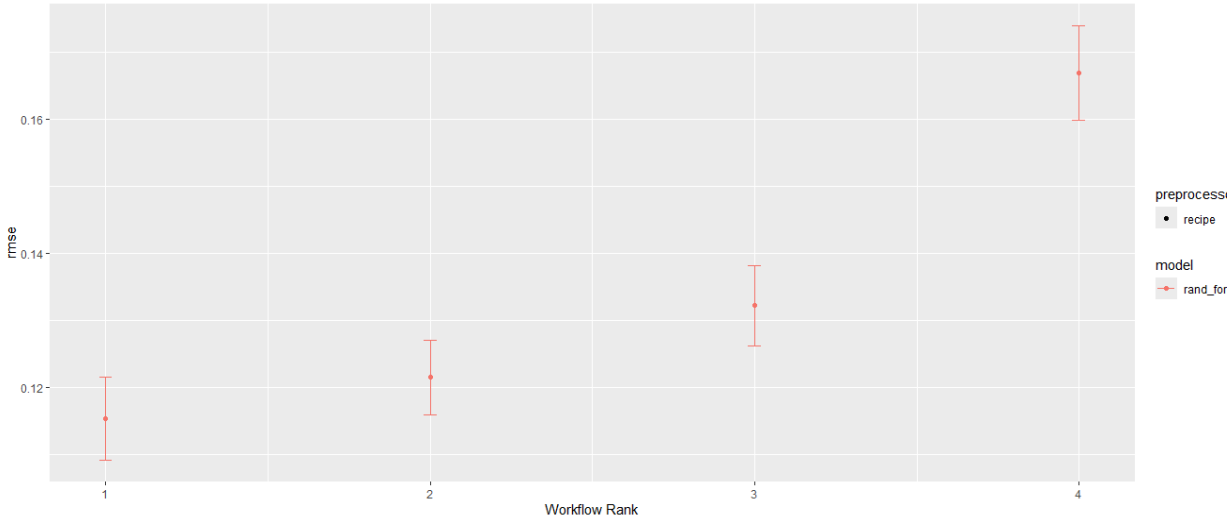

Figure 9: Autoplot of Models

Discussion of Final Model

The final model selected was model 2, a random forest model with a regression type and the "ranger" engine. This choice leverages the strengths of random forest, particularly their ability to handle complex relationships within the data by constructing multiple decision trees. The "ranger" engine helps provide a more effective way of managing large datasets with its speed and efficiency. Some key advantages of this model include its accuracy and robustness, as well as its ability to indicate feature importance. This helps provide valuable insights into the predictors of the target variable. Additionally, the averaging of multiple trees helps mitigate the risk of overfitting, which is considered a significant concern in more complex models. Decreasing the risk of overfitting will consequently be advantageous as it reduces the variance of the model. Also, using a random forest model helps in being more flexible for regression models. The model also involves a recipe where we normalize the numeric predictors, which can help decrease the impact of outliers and help analyze the data more easily by bringing values to a

common scale. However, the model also has some limitations. One weakness of this model is its lack of interpretability compared to simpler models like linear regression. While random forests can highlight which features are important, they do not show the relationship between predictors and the target variable. In addition, even with the efficiency of the “ranger” engine, the computational complexity of random forest models can be high when dealing with larger datasets. Additionally, the model does not have a threshold for multicollinearity, which may cause it to overrepresent correlated predictors.

To further improve the model, hyperparameter tuning, such as adjusting the number of trees, the number of features at each split, and the minimum node size could all help enhance performance. Additionally, various techniques such as creating interaction terms or transforming variables, could also help the model create more complex patterns in the data. Incorporating more data from the other datasets could potentially enhance the model’s accuracy. For example, product information about the actual items ordered such as its use or type could prove to be valuable. It could be used to improve the predictive model by providing additional features that capture additional characteristics, potentially leading to more accurate predictions. These improvements could help in making the model more robust, accurate, and interpretable.

Appendix

Daniel: candidate models, model evaluation and tuning, discussion of final model

Khang: candidate models, model evaluation and tuning, discussion of final model

Andy: candidate models, model evaluation and tuning, discussion of final model

Jessie: exploratory data analysis, candidate models, discussion of final model, model evaluation and tuning

Chris: introduction, explanatory data analysis, pre processing, candidate models, model evaluation and tuning