Guideline: Work Sample # 3

Description: The project focused on predicting **monthly sales on Amazon** by looking at data from a survey of Amazon customers. The goal was to figure out which customer characteristics, like **income, age, and how often they shop**, could help predict how much money Amazon would make from each state every month.

Questions answered:

1. What factors about customers influence how much they spend on Amazon?

2. Can we use these factors to accurately predict Amazon's monthly sales for each state?

Descriptive Statistics:

• The data had some patterns that needed fixing, like certain numbers being too unevenly spread. To deal with this, the project team **transformed** the data, so it was easier to work with.

• They created new categories, like **how many people share an Amazon account** and **the average age of the household**, to help make predictions more accurate.

• The team also used **advanced methods** to make sure different customer traits weren't overlapping too much, which could confuse the predictions.

Source of data: The data comes from a survey/study conducted on approximately 5000 Amazon customers from January 2018 to December 2022. Released by Amazon.

Conclusion: Several models were tested to see which one worked best. The top-performing model used a **random forest method**, which is a way of making predictions by looking at different customer factors and finding patterns. The model that performed the best also **scaled** (or adjusted) the importance of different traits, like **income and education** level, based on actual spending habits from real-world data.

This final model was very good at predicting sales and placed **third in a competition**. The project showed that factors like **age, income, and education** can strongly affect how much people spend on Amazon, and that using these factors in the right way leads to very accurate predictions.

Angelo Desiderio
STATS 101C, Summer 2024
28 July 2024

Kaggle Regression Competition Report:

Predicting Monthly Total Sales on Amazon by State Using the Amazon Survey Data

INTRODUCTION

The data comes from a survey conducted on approximately 5000 Amazon customers from 2018 to 2022, consolidating customer demographics by income, age, order frequency, education, and household size. Our goal is to use this data to predict the log of total monthly sales by state on Amazon. We believe "count" is a significant predictor, as more orders naturally lead to more sales. This report explains our decision to use a random forest model for the Amazon dataset and identifies key variables influencing our response variable. We employ data visualization, preprocessing techniques, and cross-validation to identify the most effective models.

EXPLORATORY DATA ANALYSIS

We started by exploring the customer info data and the orders detail data to understand the distribution of each variable in the training dataset as well as their relationships and interactions with other variables.

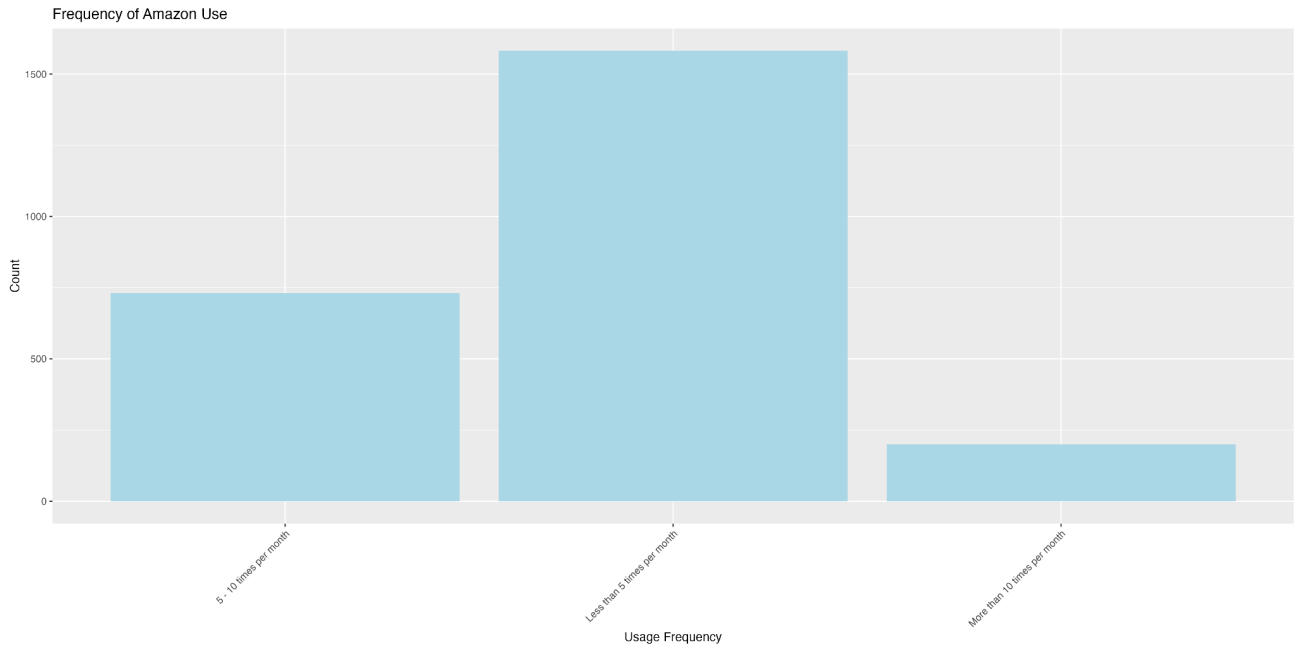

Figure 1: From the customer info data, we can see that the majority of customers use amazon less than 5 times per month.

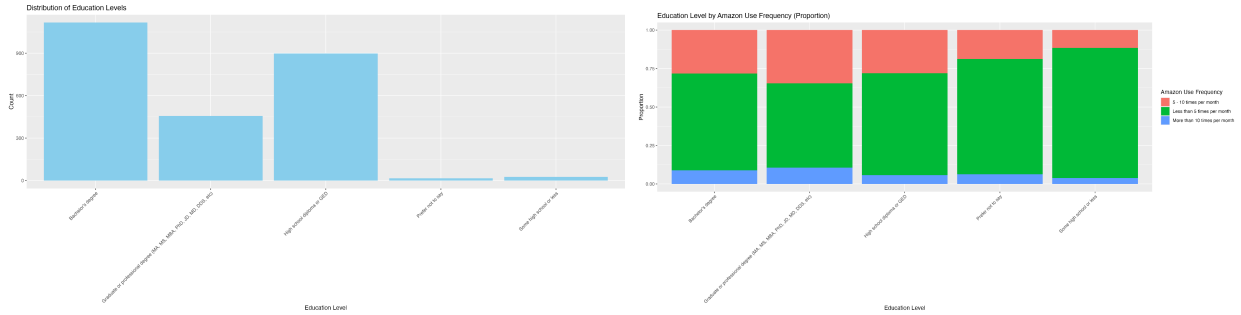

Figure 2: The plot on the left is the distribution of education level among users in the data. We can see that most users have a bachelor's degree or high school diploma. From the plot on the right, we can see that the proportion of customers for each usage frequency does not vary a lot for different education levels.

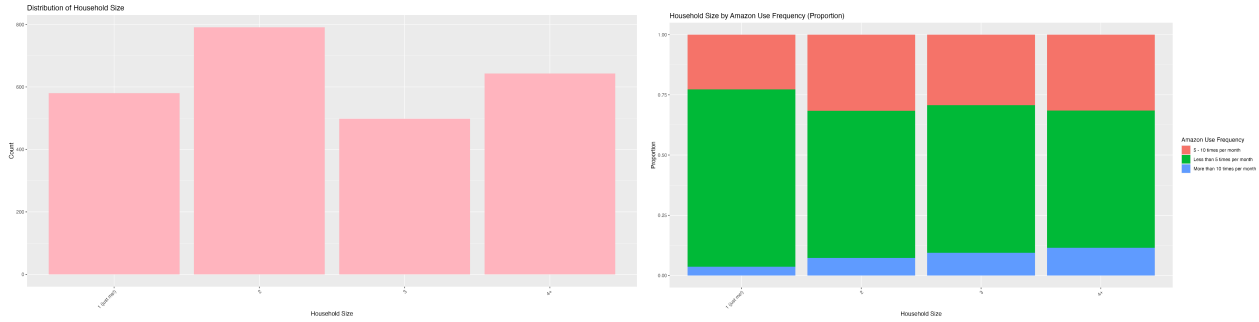

Figure 3: The plot on the left is the distribution of household size among users in the data, we can see that the household size has a quite even distribution. And from the plot on the right, we can see that the usage frequency increases for customers as the size of the household increases.

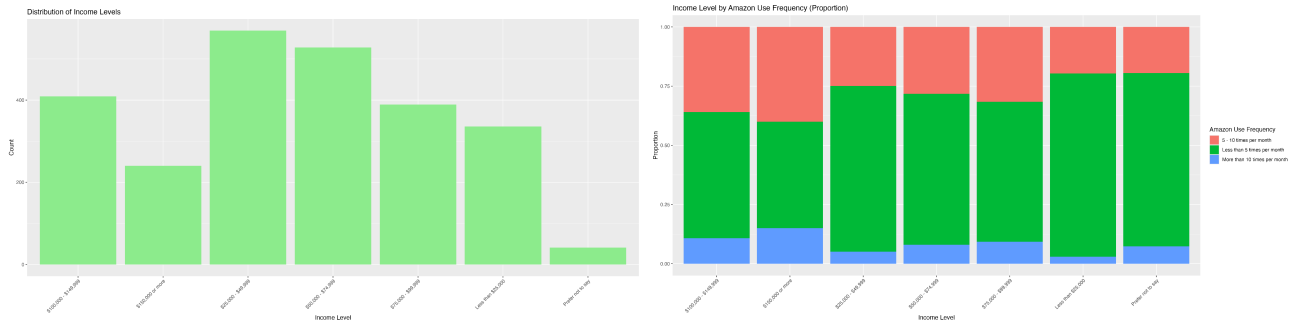

Figure 4: The plot on the left is the distribution of income level among users in the data, we can see that most users have an income between $25k - $75k per year. And from the plot on the right, we can see that customers with income lower than $25k per year tend to use amazon more than people with more than $25k per year, and the usage frequency increases as income increases for customers with income more than $25k per year.

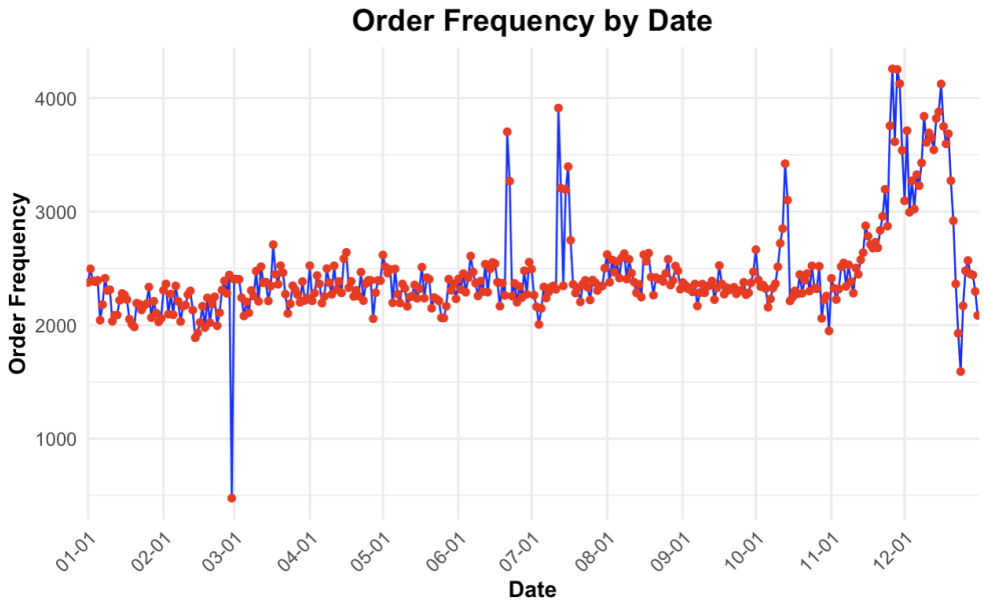

Figure 5: This graph, retrieved from the order details charts, shows consistent order volume from January to October, with spikes in June and July due to Amazon Prime Day. Starting in November, order frequency increases significantly, reflecting the holiday effects of Thanksgiving and Christmas, then drops sharply in December. While this data is not directly related to our model, it illustrates that order volumes vary significantly due to holiday effects.

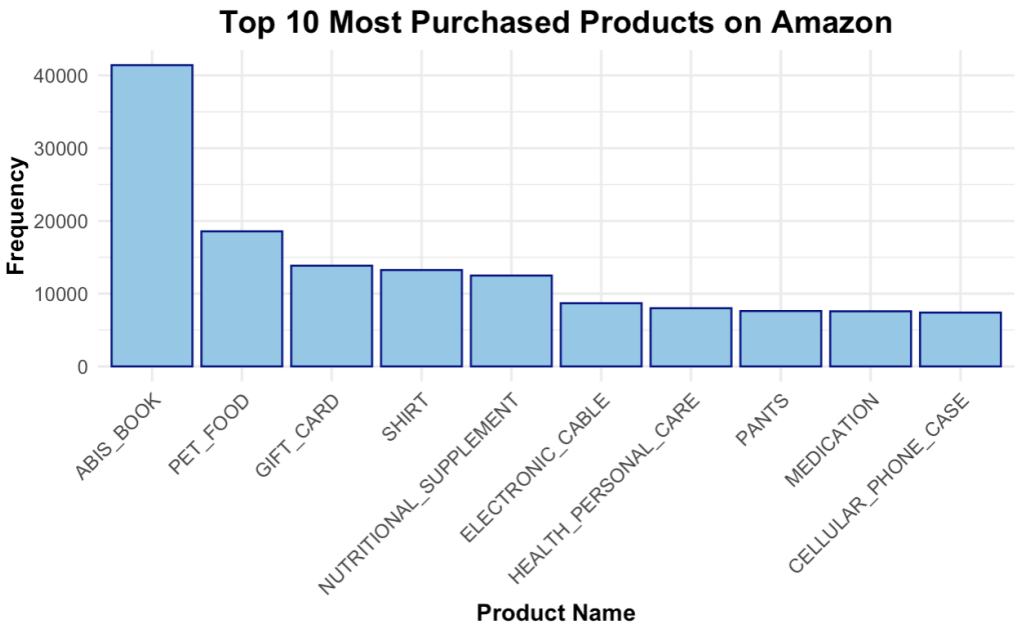

Figure 6: The order details file also shows the top ten most purchased items on Amazon. It highlights that Amazon Business Integrated Search (ABIS) books are purchased significantly more frequently than other products. Other popular products range from 10,000 to 20,000 purchases.

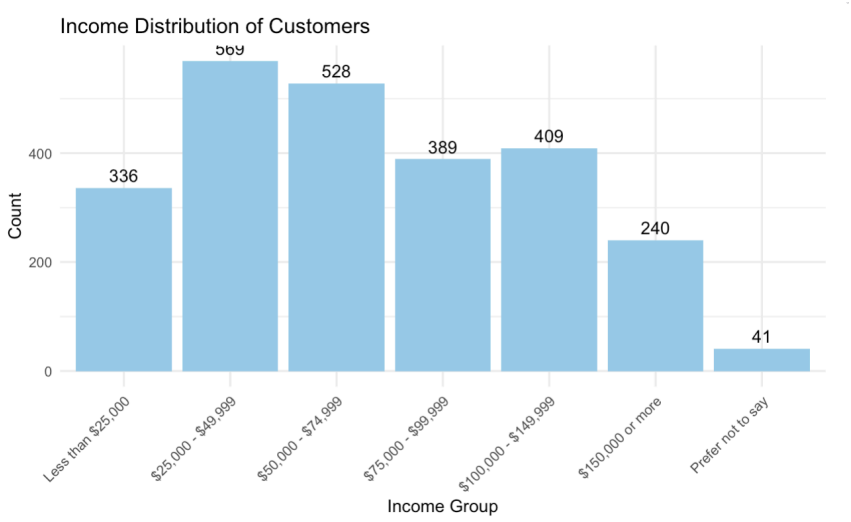

Figure 7: We also explored the customer information dataset to examine the characteristics of Amazon customers involved in this survey. The income distribution shown here is approximately normal, indicating that our clients come from a wide range of income backgrounds.

After learning more about the different variables that are used in the training dataset and their relationship and interactions with each other, we started exploring the potential transformation and combination of different variables for the model training process.

Initially, we planned on using basic multiple regression models. With 34 numeric predictor variables, it was important that we determine their distribution because skewed variables can drastically decrease model performance. Skewed variables have significant outliers in the tail, which means that the outliers exert significant influence on the model compared to other points. From the density plots (Figure 8–9), we observed that all predictor variables were right-skewed. In contrast, we found that the response variable, log_total, is not skewed due to its bell-curve distribution as a result of the log transformation. Realizing this problem, we applied log transformation to all numeric predictors. The result was that the data points follow a bell curve distribution and a significant reduction in skewness compared to the raw data (Figure 10). Due to the large number of predictors, only a few plots showing the before and after distribution for transformed variables are included below.

After transforming the variables, we generated a heat map of the correlation matrix between the variables. From the plot, we found that most predictors had significant correlations with other predictors. This was a problem, especially if we want to use multiple regression models, because utilizing correlated predictors meant that the slopes for the predictors are poorly estimated and the standard errors are inflated; both outcomes lead to reduction in model predictive performance.

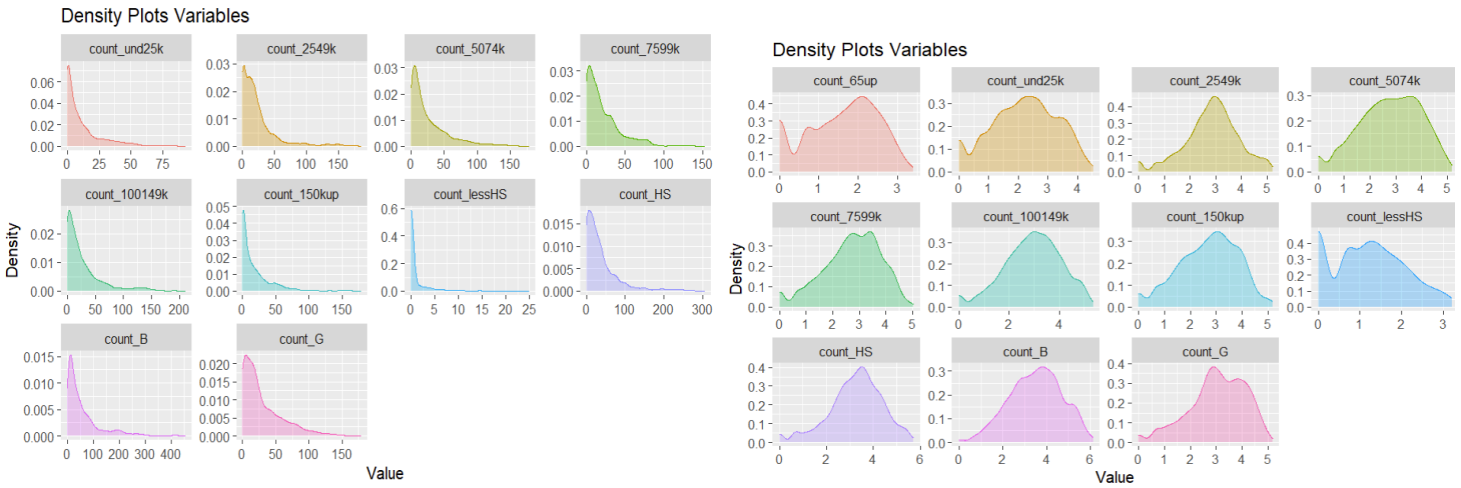

Figure 8: The left plot shows the distribution for variables relating to income. These variables are right skewed with a very long tail. After the log transformation, these variables appear to be in a bell-shaped curve.

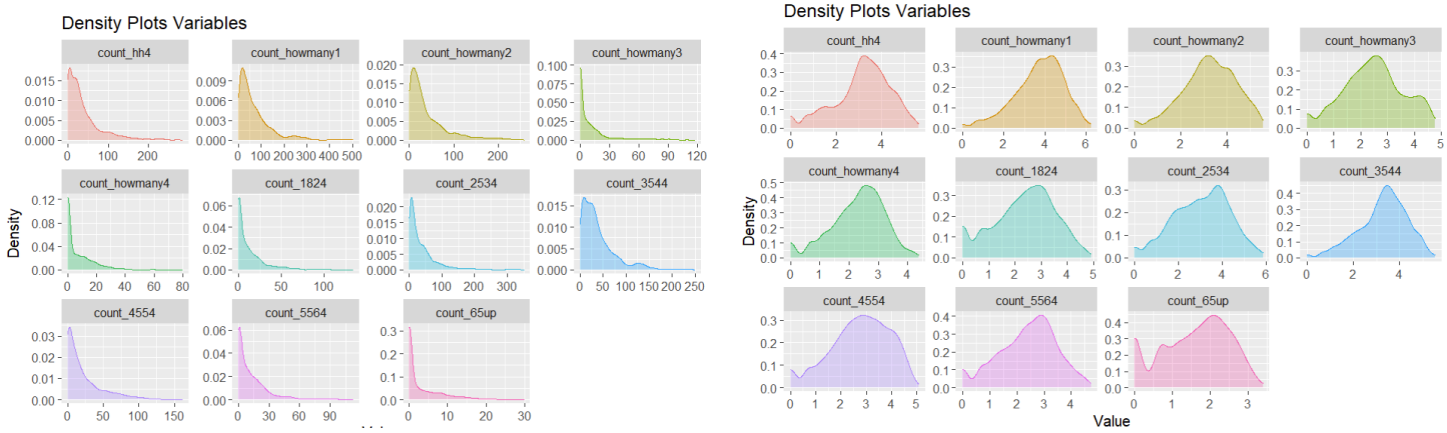

Figure 9: The left plot shows the distribution for variables relating to account sharing and customer age as right skewed. By applying the log transformation, the variables now have a bell curve distribution.

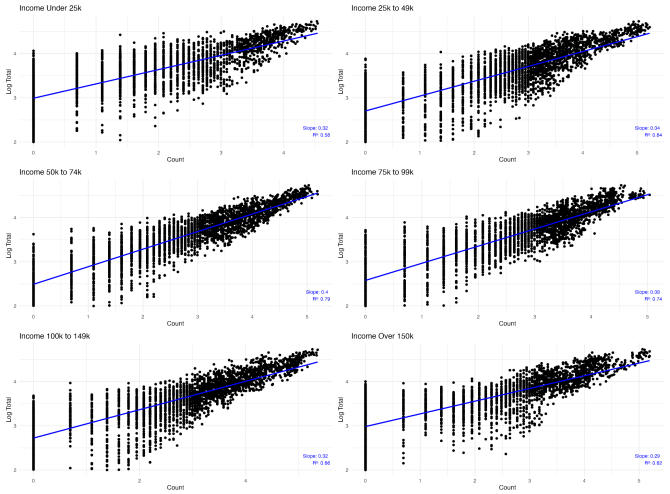

Figure 10: Further analysis of potential linear regression models between predictors relating to income and the response variable shows that the log transformation led to significant linearity of the relationship between predictors and log_total, justifying the log transformation for all numeric predictors.

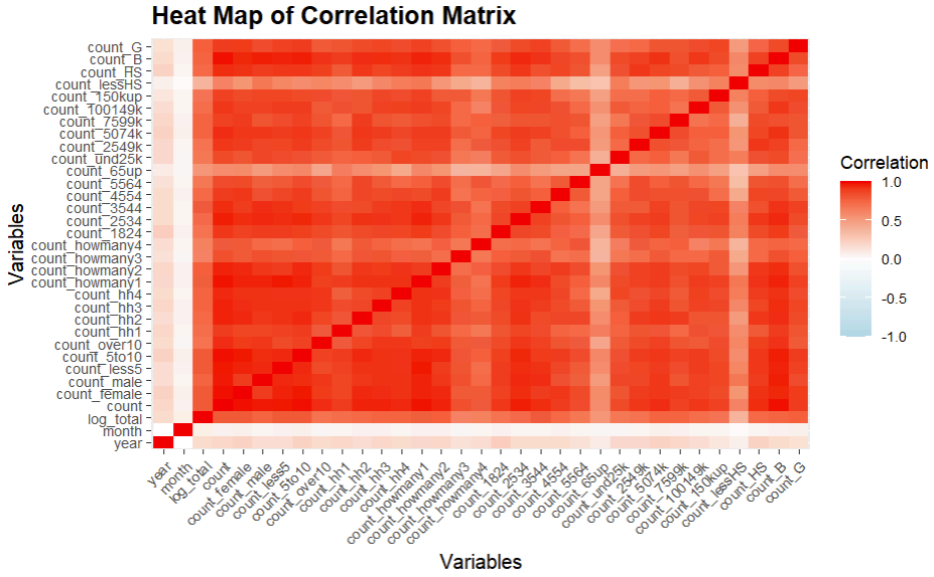

Figure 11: The correlation matrix shows that there are significant positive correlations between most predictors with each other. More importantly, there is also a positive correlation between the response variable and most predictors variables.

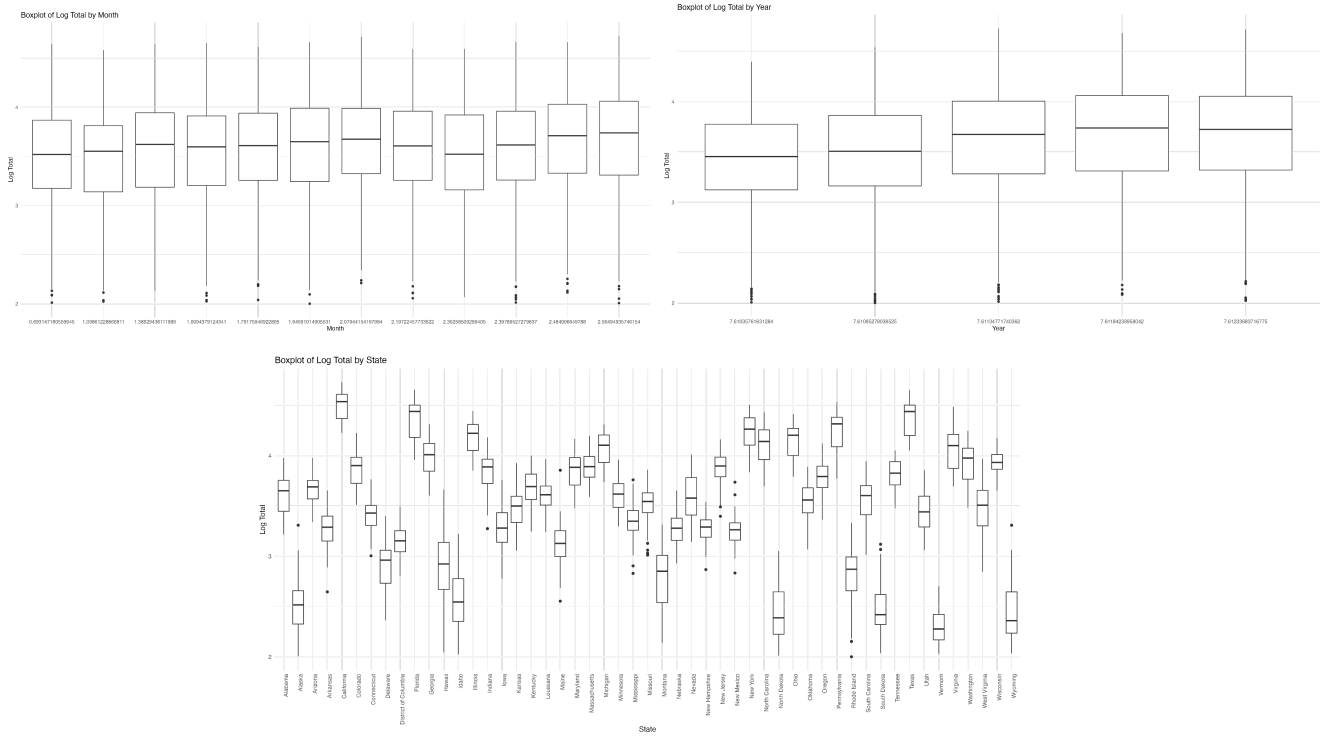

Figure 12: After exploring the relationship between different variables with our response variable log_total, we found that we should set the following three variables, month, year, and state, as dummy variables. By visually inspecting if the distributions of log_total differ across each variable as well as conducting an anova test, we do not see a significant difference that supports using these variables as predictors.

PREPROCESSING/RECIPES

step_log():

The step_log() transformation is applied in almost all models we fit. This is because the exploratory data analysis (EDA) graphs show a log relationship between predictors and the response variable. Therefore, we log-transform the predictors to ensure a linear relationship.

step_dummy():

EDA graphs indicate that log_total indeed differs between levels of state, year, and month. Consequently, we convert these categorical variables into dummy variables using step_dummy() to include them in our model.

step_pca/step_corr():

The EDA heat map reveals very high correlations between variables, raising concerns about multicollinearity that could affect our predictions. We attempted step_pca and step_corr in some models to address multicollinearity by creating uncorrelated variables or removing highly correlated ones.

step_mutate():

step_mutate() is used to create new features, such as median_age, which calculates the median age for each observation based on the weighted average from all age groups. We added this feature because a simple linear regression model predicting log_total from median_age showed a statistically significant relationship, justifying its inclusion in the random forest model.

We also use step_mutate() to apply scaling and give greater weight to certain variables in the model. We believe that, while EDA shows all predictors have relationships with the response, certain variables (such as specific age, income, or education groups) can have a greater impact on log_total. This indeed improved our predictions as shown later.

MODEL EVALUATION

In this project, seven models were created, mostly using random forests. The table below summarizes each model's characteristics. Exact hyperparameters for some models are not specified due to the workflow set tuning process, but we have identified them for the best-performing model.

Model Identifier	Model Type	Engine	Recipe	Hyperparameters
scaling_rf	Random Forest	ranger	step_mutate() step_log() step_dummy()	Mtry = 6 Min_n = 12 Trees = 1974
log_rf	Random Forest	ranger	step_log() step_dummy()	Min_n Trees
base_rf	Random Forest	ranger	step_dummy()	Min_n Trees
new_var_rf	Linear Regression	ranger	step_mutate(account_share = , Median_age = …, Household = …) step_log() step_dummy()	Min_n Trees
filter_rf	Random Forest	ranger	step_corr()	Threshold = 0.2874 Min_n Trees
pca_rf	Random Forest	ranger	step_normalize() step_pca() step_log()	Num_comp Min_n Trees
base_lm	Linear Regression	linear_reg	step_dummy()	

Scaling_rf:

This is the best model. Similar to "new_var_rf," it creates a new variable called "account_share." But most importantly, we believe certain variables in a category significantly influence the outcome variable (e.g., wealthier individuals tend to spend more per order, contributing more to log_total than counts in lower income groups). Therefore, we apply greater weights to certain variables in the age, income, and education categories. For each category, we choose a base level and scale the rest of the variables relative to it. The scaling coefficients are derived from the 2022 Consumer Expenditure Surveys by the U.S. Bureau of Labor Statistics. More details are discussed in the final model section.

Log_rf:

Since the graphs of predictors vs. log_total show a logarithmic trend, we transform all numeric predictors by taking their logarithms.

Base_rf:

This is the most basic model. Categorical variables (state, month, year) are converted into dummy variables. This model should be compared with the basic linear regression model(base_lm) to demonstrate that the random forest model significantly outperforms linear regression.

New_var_rf:

In addition to taking logs and creating dummy variables, this model captures underlying trends by creating three new variables:

1. **Account share**: Calculates the total number of people sharing/using Amazon accounts by summing the products of counts in "count_howmany" variables and the number of people sharing the account.
2. **Household**: Calculates the total number of people in a household involved in the survey by summing the products of each "count_hh" and the respective number of household members.
3. **Median age**: Calculates the median age by summing the products of counts in each age category and the respective median age for that category (e.g., 29.5 for the 25-34 age group) and dividing by "counts".

Filter_rf:

To address multicollinearity identified via a correlation heat map, this model removes highly correlated variables using step_corr(), with threshold set as a hyperparameter.

Pca_rf:

To address multicollinearity, this model uses PCA to transform correlated variables into uncorrelated ones. The number of components(num_comp) is a hyperparameter tuned during the process.

Base_lm:

This is the only linear regression model. It should be compared with the "base_rf" model to show that linear regression is not suitable for our complex dataset.

To tune different parameters and compare results across various models, we create a workflow set. We first apply the tune() function to all hyperparameters in each model. Then, we use workflow_map("tune_grid") to perform the tuning and cross validation over 10 folds and rank_results() to find the RMSE for the best models across. Due to the use of the workflow set, extracting the exact values of certain hyperparameters can be challenging.

The hyperparameters for "scaling_rf" are **mtry =6, min_n = 12, and trees = 1974.** The mtry indicates that six features are considered at each split, which is an optimal balance between bias and variance and prevents overfitting. The min_n ensures that each leaf node has at least 12 samples, which ensures each

node has sufficient number of data points to fit. Finally, we have a large number of trees in the model, which enhances the model's predictive power.

We also find that the hyperparameter for the threshold in step_corr is set to 0.2874, meaning that variables with a correlation greater than 0.2874 will be removed from the model. However, given that all predictors have significant correlations with one another, this setting will result in many variables being removed, potentially leading to a decrease in the model's predictive power.

Model Identifier	Metric Score (RMSE)	SE of metric
scaling_rf	0.112830	0.00301
log_rf	0.112872	0.00304
base_rf	0.112930	0.00304
new_var_rf	0.113037	0.00299
filter_rf	0.113769	0.00308
pca_rf	0.120793	0.00289
base_lm	0.144543	0.00497

The table shows the RMSE and SE of the seven tuned models. A comparison between base_lm and base_rf reveals that the RMSE of the linear regression model exceeds that of the random forest model by **28%,** which shows that linear regression is unsuitable for the complex Amazon datasets, so we focus on random forest models. Our attempts to address multicollinearity were not ideal, as both filter_rf and pca_rf produced significantly poorer results compared to base_rf. Creating new variables also proved ineffective, as it increased the RMSE and made our models unnecessarily complex. However, applying a log transformation in log_rf was effective, reducing the RMSE from base_rf by about 0.05%. The most effective approach was scaling certain variables in addition to the log transformation in scaling_rf, which reduced the RMSE from base_rf by about 0.09%. Overall, these results show that a basic random forest model is powerful enough to handle complex data and provide accurate predictions, but applying scaling and log transformations can further enhance performance and boost our Kaggle rankings significantly. :)

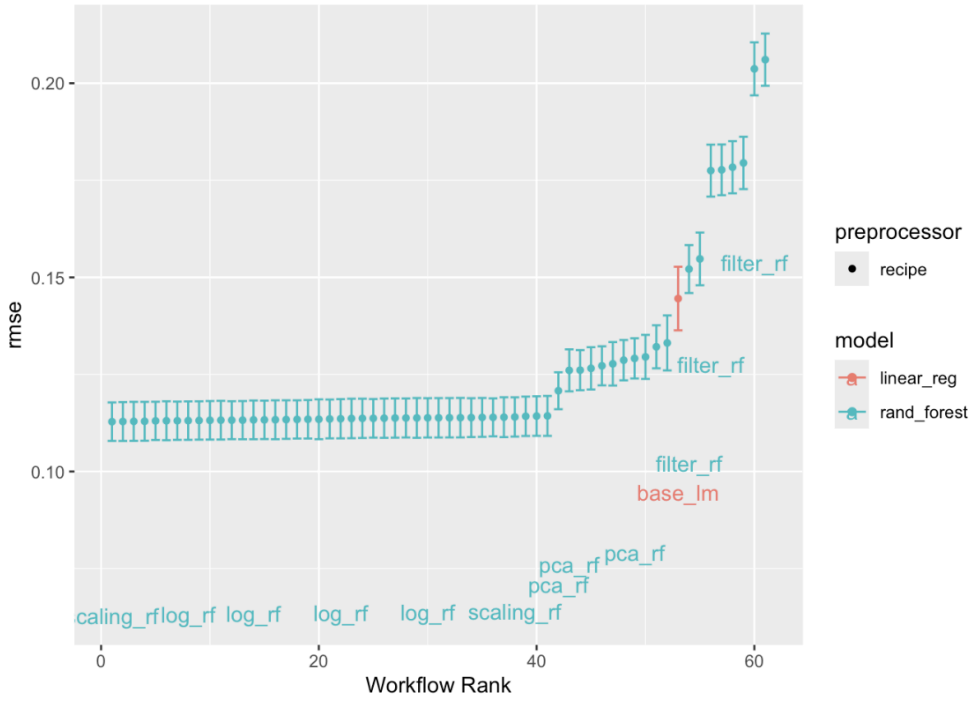

Figure 13: The autoplot shows the RMSE for all models created in the workflow set, with a grid set equal to 10. Workflow ranks 0 to 40 correspond to the top-performing models, with their RMSE values remaining consistent throughout the range of 0.112. Starting from pca_rf, the RMSE gradually increases. Models like filter_rf perform even worse than the base_lm model, reaching RMSE values as high as 0.2.

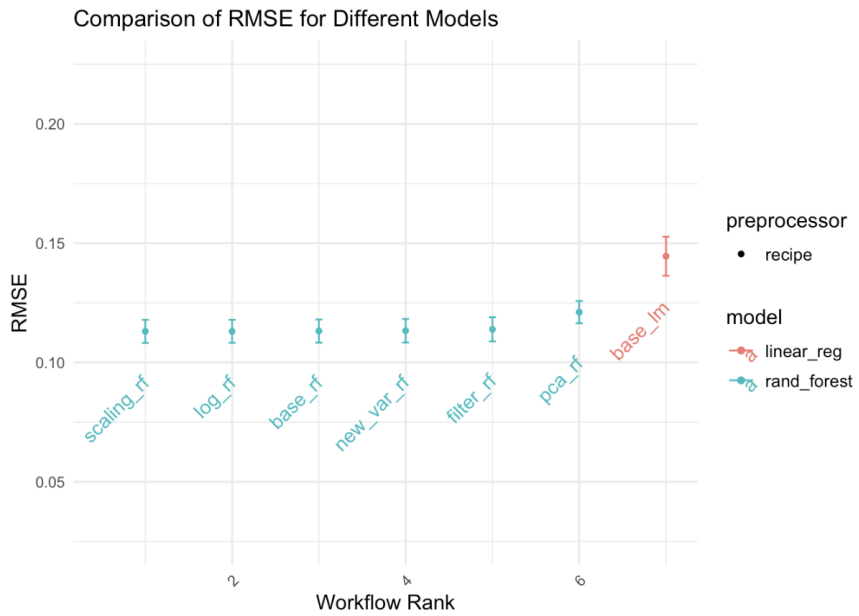

Figure 14: This plot shows the RMSE of the tuned models. The top 5 models have RMSE values and errors all within the range of 0.112. The pca_rf model is the worst-performing random forest model with an RMSE of 0.13, but it is still significantly better than the base_lm model 0.144.

DISCUSSION OF FINAL MODEL

The final model we selected is the scaling_rec model, which achieved an MSE of 0.01710 in the public score and 0.01590 in the private score, securing us 3rd place in the competition. The use of dummy variables for the three categorical variables and log transformations across all predictors undoubtedly improved the model's performance. Additionally, the inclusion of the newly created variable—account share—also enhanced the model, suggesting that more people sharing or using Amazon accounts leads to a higher log_total.

All consumer units	Under 25 years	25-34 years	35-44 years	45-54 years	55-64 years	65 years and older	65-74 years	75 years and older
$72,967	$46,359	$67,883	$86,049	$91,074	$78,079	$57,818	$60,844	$53,481

Most importantly, we find that applying scaling to variables in the age, education, and income categories works well. The scaling coefficients are calculated using data from the 2022 Consumer Expenditure Surveys by the U.S. Bureau of Labor Statistics. For example, we first combine age categories using weighted averages to match the Amazon dataset categories, set the expenditure for those under 25 years ($46,359) as the base level, and divide the expenditures of other categories by this amount to obtain coefficients of 1.459, 1.849, 1.965, and 1.685. This suggests that people in the age groups 35-44 and 45-54 contribute more to log_total compared to other age groups. Similarly, scaling coefficients for income increase from 1.365 to 4.17 as income levels increase. For education, individuals with high school, bachelor, and graduate degrees spend approximately 10 times more than those with less than a high school education.

Overall, the strength of our model is random forest's ability to handle the complexity of the dataset effectively through bootstrapping, feature selection to resolve interactions, and averaging over numerous decision trees to achieve high accuracy. Our use of log and dummy variables also accurately captures the underlying trends between predictors and the target variable. Scaling and the creation of new variables further enhance performance. However, despite the fact that scaling decreases RMSE, we lack evidence that the general expenditure trends can be applied to Amazon customers. Finally, the use of a large number of trees can potentially cause overfitting.

Throughout the competition, we attempted but failed to explore many underlying patterns between predictors and the response variable. We believe this is because the consolidated survey summary is difficult to organize and use for accurate predictions. For example, we cannot isolate one level in a category to examine its effect, as all entries are consolidated. If we continue working with this type of data, we would hope for additional data related to price, such as counts by price range.

Another possible approach is breaking the summary back into individual records. By doing so, we could treat many variables as categorical and predict individual Amazon spending based on their profile. We could then combine these individual predictions to obtain the log_total.

APPENDIX I: FINAL ANNOTATED SCRIPT

```
library(tidymodels) # read in libraries
library(tidyverse)


train <- read_csv("train.csv") # read in training and testing set
test <- read.csv("test.csv")
# deselect order_totals from training set
train <- train %>% select(-order_totals)

set.seed(2024)
# create training folds with folds of 10, stratifying on log_total
train_folds <- vfold_cv(train, v = 10, strata = log_total)

# create final recipe
final_rec <- recipe(formula = log_total ~.,
                        data = train) %>%
# set state, year, month to factors, so we can apply step_dummy()
  step_mutate(q_demos_state = as.factor(q_demos_state),
              year = as.factor(year),
              month = as.factor(month)) %>%
# apply scaling to variables in the income category, with
count_und25k as the base level
  step_mutate(count_2549k = 1.365 * count_2549k,
              count_5074k = 1.794 * count_5074k,
              count_7599k = 2.033 * count_7599k,
              count_100149k = 2.569 * count_100149k,
              count_150kup = 4.17 * count_150kup) %>%
# apply scaling to variables in the age category, with count_1824
as the base level
  step_mutate(count_2534 = 1.459 * count_2534,
              count_3544 = 1.849 * count_3544,
              count_4554 = 1.965 * count_4554,
              count_5564 = 1.685 * count_5564,
              count_65up = 1.247 * count_65up) %>%
# apply scaling to variables in the age category, with count_lessHS
as the base level
  step_mutate(count_HS = count_HS * 12.1,
              count_B = count_B * 10.759,
              count_G = count_G * 10.655
  ) %>%
# create new variable account_share
  step_mutate(account_share = count_howmany1 * 1 + count_howmany2 *
2 + count_howmany3 * 3 + count_howmany4 * 4) %>%
```

```
  step_dummy(all_factor_predictors())

# create the random forest model with tuned parameters
rf_model <- rand_forest(
  Mtry = 6,
  min_n = 12,
  trees = 1973) %>%
  set_engine("ranger") %>%
  set_mode("regression")

# create workflow by adding the recipe and model
rf_wflow <- workflow() %>%
  add_recipe(final_rec) %>%
  add_model(rf_model)

# Cross Validation using fit_resamples(this step is unnecessary
since we already choose the final model)
results <- rf_wflow %>%
   fit_resamples(resamples = train_folds)

# A look at the rmse from cv
metrics <- collect_metrics(results)
metrics %>%
  mutate(mean = format(mean, digits = 6))

# Create a fitted model using the whole training set
rf_workflow_fit <- rf_wflow %>% fit(data = train)

# Use the fitted model to make predictions on the test data
model_predictions <- predict(rf_workflow_fit, new_data = test)

# Bind id column with predictions, and output as a csv file
test <- test %>% select(id) %>% bind_cols(model_predictions)
test <- test %>% rename(log_total = .pred)
write_csv(test, "final_predictions.csv")
```

WORKS CITED

U.S. Bureau of Labor Statistics. (2022). Mean Item Share, Average and Standard Error by Reference Person Age Ranges, 2022. BLS. https://www.bls.gov/cex/tables/calendar-year/mean-item-share-average-standard-error/reference-person-age-ranges-2022.pdf

U.S. Bureau of Labor Statistics. (2022). Aggregate Group Share by Consumer Unit Education, 2022. BLS. https://www.bls.gov/cex/tables/calendar-year/aggregate-group-share/cu-education-highest-2022.pdf

U.S. Bureau of Labor Statistics. (2022). Mean Item Share, Average and Standard Error by Income Before Taxes, 2022. BLS. https://www.bls.gov/cex/tables/calendar-year/mean-item-share-average-standard-error/cu-income-before-taxes-2022.pdf