Sephora Product Ratings Regression

Abigail Ajamian
Course D214

Agenda

- Introduction
- Research Question / Hypotheses
- Analysis Process
- Findings
- Limitations
- Proposed Actions
- Expected Benefits

Introduction

Abigail Ajamian

- WGU Data Analytics Graduate Student
- Experience: 1.5 years including an internship working with a non-profit organization.

Research Question / Hypotheses

Can an ordinal logistic regression be performed to determine product characteristics that influence the rating of a product?

H_0: An ordinal logistic regression model cannot be made using this data to determine product characteristics that influence the rating of a product.

H_1:An ordinal logistic regression model can be created to determine product characteristics that influence the rating of a product with accuracy> 70%.

Analysis Process

Data Collection

- The dataset was collected from Kaggle.com
- Data Quality: Acceptable
- Kaggle's Usability Score: Completeness = 100%
  Credibility = 100%
  Compatibility = 100%
- Data Sparsity = 19.35%

Analysis Process

Data Preparation

- Detect duplicates, missing values, and outliers
- Duplicates → 0 Detected
- Missing Values →

  Dropped columns with missing values above 50% of observations.

  Dropped 3% of observations in Ratings columns.

  Imputed with mode for Secondary Category column.

  Fill missing values with 0 for the Size column to represent no size.

- Outliers →

  Price (USD): Max outlier ($1,900) imputed with median.

  Child Count: All outliers with z score > 3 were imputed with the median.

```python
# Label encode primary_category
encoder = LabelEncoder()
df['primary_category_label'] = encoder.fit_transform(df['primary_category'])
```

```python
# Label encode secondary_category

df['secondary_category_label'] = encoder.fit_transform(df['secondary_category'])
```

```python
# Change labels for size
# no size = 0 , 0.5 oz/ 15 mL = 1, 1 oz/ 30 mL = 2, 1.7 oz/ 50 mL = 3, 3.4 oz/ 100 mL = 4 , other = 5
df['size'] = df['size']
dict_over = {'size':{"0.5 oz/ 15 mL":1,"1 oz/ 30 mL": 2,"1.7 oz/ 50 mL":3,"3.4 oz/ 100 mL":4}}
df.replace(dict_over, inplace = True)
df['size']

for i in df['size']:
    if i not in (0,1,2,3,4):
        df['size'] == df['size'].replace(i,5, inplace= True)
print(df['size'])
```

Analysis Process

Data Preparation cont.

- Data Wrangling: Encode categorical variables to be represented by numbers (0=No 1=Yes).

  Primary Category : Used LabelEncoder() from sklearn.preprocessing package. The labels were stored in a new column with the variable name plus "label" at the end.

  Secondary Category: Used LabelEncoder() from sklearn.preprocessing package. The labels were stored in a new column with the variable name plus "label" at the end.

  Size: To encode size, the top 4 size values were represented by 1-4; if a product had no size, it was represented by 0, and 5 represented the products with other sizes.

Analysis Process

```python
# Use Kmeans clustering to identify groups of ratings
# 5 clusters for ratings 1-5

X = df[['rating']].values.reshape(-1,1) #reshape single-column bc kmeans expects 2D array

kmeans = KMeans(n_clusters=5,random_state=0)
kmeans.fit(X)

labels = kmeans.labels_
centroids = kmeans.cluster_centers_
```

```python
#Change labels
kmeans.labels_[kmeans.labels_ == 0] = 10
kmeans.labels_[kmeans.labels_ == 1] = 7
kmeans.labels_[kmeans.labels_ == 2] = 9
kmeans.labels_[kmeans.labels_ == 3] = 6
kmeans.labels_[kmeans.labels_ == 4] = 8
```

```python
#Change labels to represent 1-5 stars
kmeans.labels_[kmeans.labels_ == 10] = 5
kmeans.labels_[kmeans.labels_ == 9] = 4
kmeans.labels_[kmeans.labels_ == 8] = 3
kmeans.labels_[kmeans.labels_ == 7] = 2
kmeans.labels_[kmeans.labels_ == 6] = 1
```

```python
df = df.drop(['product_id','brand_name','loves_count','product_name','z_child'], axis=1)
```

Data Preparation cont.

- Since analysis is an ordinal regression(order matters), the dependent variable needed to be grouped.
- Kmeans clustering was utilized to accomplish grouping.

  $K = 5 \rightarrow$ *5 groups relabeled as1-5*
- This was stored in a new column named rating clusters
- The last data preparation step is to remove all unneeded columns.

Analysis Process

Exploratory Data Analysis (EDA)

- Why? : Create visualization to gain insight into patterns that can be found in the dataset.
- Univariate Analysis:

A visual was created to view each individual variable.

- Bivariate Analysis:

A visual that compares each predictor variable with the dependent variable was executed.

- Findings will be discussed further in the presentations

Analysis Process

Ordinal Regression

- The datasets were split into training and testing sets with a 70-30 split.

x_train: 70% of the data from predictor variables

x_test: 30% of the data from predictor variables

y_train: 70% of the data from the dependent variable

y_test: 30% of the data from the dependent variable

- Python's mord package was used to execute the initial ordinal logistic regression model using LogisticIT().

  Initial models accuracy = 39%

	precision	recall	f1-score	support
1	0.00	0.00	0.00	51
2	0.00	0.00	0.00	288
3	0.00	0.00	0.00	566
4	0.39	0.98	0.56	946
5	0.56	0.08	0.14	614
accuracy			0.39	2465
macro avg	0.19	0.21	0.14	2465
weighted avg	0.29	0.39	0.25	2465

Analysis Process

Ordinal Regression cont.

- Feature reduction was implemented for further exploration. This was done by using SequentialFeatureSelector() from mlxtend.feature_selection.
- This detected the top 4 predictor variables based on the accuracy score.

Top 4: brand_id, size, new, secondary_category (labeled)

- These top 4 created a reduced ordinal logistic model using the same LogisticIT function.

Reduced model accuracy = 38%

	precision	recall	f1-score	support
1	0.00	0.00	0.00	41
2	0.00	0.00	0.00	270
3	0.00	0.00	0.00	619
4	0.38	0.97	0.55	929
5	0.27	0.04	0.07	606
accuracy			0.38	2465
macro avg	0.13	0.20	0.12	2465
weighted avg	0.21	0.38	0.22	2465

Analysis Process

```
# Create a logistic regression model
model = LogisticRegression(multi_class='multinomial', solver='lbfgs')
```

```
# Fit the model on the training data
model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = model.predict(X_test)

# Evaluate the model's accuracy
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

Accuracy: 0.3905109489051095
```

Ordinal Regression cont.

Multinomial logistic regression to determine if the dataset satisfies the proportional odds assumption. (Python does not have the Brant-Wald Test)

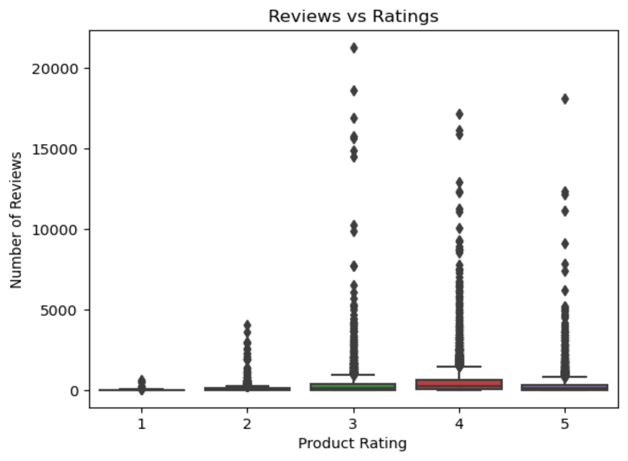
LogisticRegression() from sklearn.linear_model

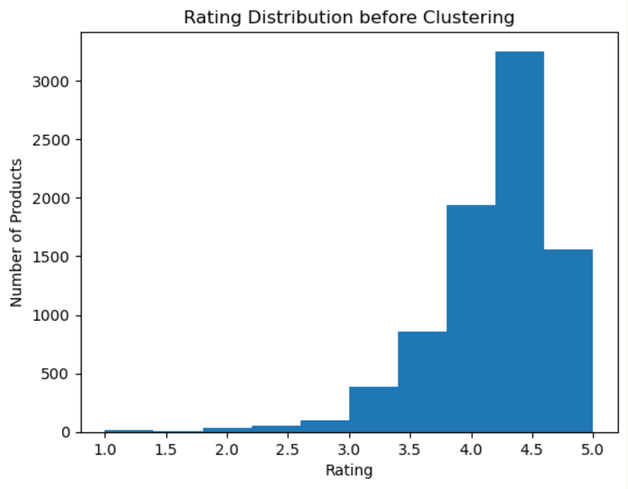
Predictions from model was compared to actual y_test to determine accuracy.

Model accuracy = 39%

Concluded that the proportional odds assumption does not cause a poor fit of the ordinal regression model.

Findings





Univariate/Bivariate Analysis

- Products with higher ratings often have a higher number of reviews.
- Product rating distribution is left skewed, with the peak being around the ratings 4-4.5.

Ordinal Regression

- Failed to reject the null hypothesis
- Model's best accuracy = 39%

Limitations

- An ordinal logistic model assumes proportional odds (Lee, 2019).
- Python does not include a Brant-Wald Test.
- Relatively small sample size.
- Kaggle's usability score does not consider data sparsity, which is 19.35% for this dataset.

Proposed Actions

Collect additional data on Sephora products that can be joined with this dataset and then complete the ordinal logistic regression again.

Use the written reviews of the products to create a sentiment analysis.

Benefits

The expected benefits of this study have to do with informing Sephora of the relationship between the independent variables used in the ordinal regression and product ratings.

None of the 11 independent variables should not be utilized in business decisions to achieve a product catalogue that is higher rated.

This helps Sephora stakeholders and marketers remove these 11 independent variables from consideration.

Sources

- Lee, E. (2019, May 29). Ordinal logistic regression on World happiness report. Medium. https://medium.com/evangelinelee/ordinal-logistic-regression-on-world-happiness-report-221372709095