**Case Problem 7 – Model with Texts**

**Student Names: Angel Fernandes Date: 13th February 2023**

In this case problem, we will build models to predict the helpfulness of Amazon reviews. The dataset contains 4,915 reviews collected from Amazon under the electronic product category. Below are the definition of each column.

* reviewID: The ID of the review
* reviewerName: The name of the reviewer.
* overall: Product rating (1-5).
* reviewText: The review texts.
* reviewTime: The time the review was made (from Jan 2012 to Dec 2014).
* day\_diff: The age of the review – number of days the review has been on the website. (1 – 1064 days)
* helpful\_yes: The number of times the review was found useful.
* helpful\_no: The number of times people didn't support the review and didn't find it helpful.
* total\_vote: Number of votes given to the review.

We will build two models and compare their performance:

* One base model with the length, age, and rating of the review. Using total votes as a control.
* A model with the above features, plus the sentiment intensity of the review texts.

Please answer all the questions that are in red.

**Part I: Data Cleaning and Feature Extraction**

1. Load the Amazon\_reviews.csv file. Those are 4,915 reviews collected from Amazon. Most of those reviews are relatively short. For very long reviews, the result table may show truncated texts only. You can attach a **Browse node** right after the **Input Data** node to read the full texts.
2. Connect the **Input Data** node to a **Select** node. In the **Select** node, set the type of each variable. I recommend always using this node right after reading the data so that each variable is correctly recognized.

reviewID: v\_string

reviewerName: v\_string

overall: int64

reviewText: The review texts.

reviewTime: Date

day\_diff: int64

helpful\_yes: int64

helpful\_no: int64

total\_vote: int64

1. **Remove the null reviews** by connecting the **Select** node with a **Filter** node. In the **Filter** node, use the following function:

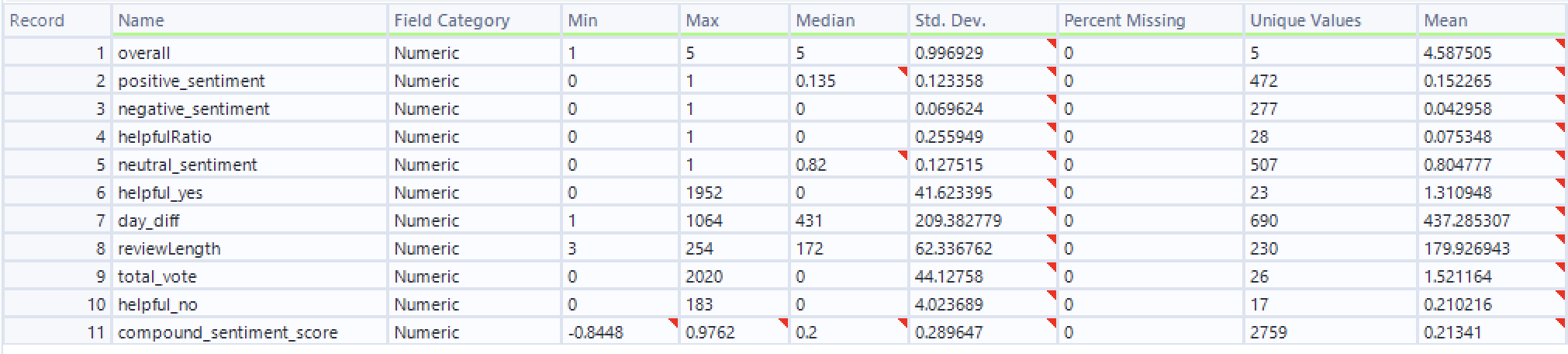
!IsNull([reviewText])

How many null texts were removed? There was 1 record (reviewID = 125)

1. Extract the first feature from the text: Sentiment. Perform **sentiment analysis** to all reviews by connecting the **T** output anchor of the **Filter** node to a **Sentiment Analysis** node. Check “find sentiment at sentence level”.
2. Extract the second feature from the text: Length. Connect the **Sentiment Analysis** node to a **Formula** node. In the **Formula** nod, create a new column named “reviewLength” by using the following formula: Length([reviewText]).
3. Create our dependent variable: Helpfulness. Connect the **Formula** node to a **Formula** node. In the **Formula** nod, create a new column named “helpfulRatio” by using the following formula:

IF [total\_vote]==0 THEN 0 ELSE [helpful\_yes]/[total\_vote] ENDIF

1. Connect the **Formula** node to a **Select** node. In the **Select** node, make sure the type of each variable is correct. Change the types of the newly created variables if needed.
2. Connect the **Select** node to a **Field Summary** node. Check all numeric variables in the node. Run the workflow and check the **O** output anchor. Copy and Paste the summary table below (only include from the first column up to the “Mean” column). What is the average sentiment of all reviews?



1. Because I plan to use both rating (the “overall column”) and sentiment score (the “compound\_sentiment\_score” column) in my regression later, I want to first check their correlations. Because usually, when someone provides a review, the sentiment in the texts can be correlated with the numeric rating. Connect the **Select** node to a **Pearson Correlation node.** Usually, a correlation above 60% can be concerning and introduce multicollinearity problems. Select the two variables to make a correlation table. What is the correlation between the rating and sentiment score? The correlation is 0.274343112596824

**Part II: Prediction and Evaluation**

1. Make the **testing and training datasets** by connecting the **Select** node to a **Create Samples** node. Make a 60-40 split.
2. Connect the **“E” Output Anchor (Estimation/Training dataset)** of the **Create Samples** node to two **Linear Regression** nodes.
3. In the first **Linear Regression** node, name it “RegressionwithoutText”. Select “helpfulRatio” as the target. For predictors, select 4 fields: overall, day\_diff, total vote, and ReviewLength. Connect two **Browse** nodes to the **R** and **I** output anchors of the **Linear Regression** node.
4. In the second **Linear Regression** node, name it “RegressionwithText”. Select “helpfulRatio” as the target. For predictors, select 5 fields: overall, day\_diff, total vote, reviewLength, and compound\_sentiment\_score. Connect two **Browse** nodes to the **R** and **I** output anchors of the **Linear Regression** node.
5. Add the **VIF** nodes to test collinearity: Download the “Variance+Inflation+Factors+Sample.yxzp”. I found this sample workflow with the VIF macro by typing “vif” in the search bar, and clicking the [third link](https://community.alteryx.com/t5/Community-Gallery/Variance-Inflation-Factors-Sample/ta-p/878728), which directed me to the Alteryx Community Gallery. I then downloaded the Variance Inflation Factors Sample workflow. You can directly download the yxzp file I attached. But in the future, you can find needed workflows or macros by searching in the community gallery.

Graphical user interface, text, application, email

Description automatically generated

1. After downloading the yxzp file, double-click it. Click OK to import the workflow. There may be an error but you can ignore it for now. You should be able to open the “Variance Inflation Factor Sample.yxmd”. You should be able to see how the VIF node is used in the example. Copy and paste the **VIF** node and connect it to the **“O”** output anchor (where the trained models reside) of both of the **Linear Regression** nodes. Then connect the **D** output anchors of the two **VIF** nodes to two **Browse** nodes.
2. Connect the **“O”** output anchors of both of the **Linear Regression** nodes to a **Union** node – this allows the two models to be compared. Connect the **Union** node to the **“M”**  (model) input anchor of a **Model Comparison** node. Connect the **V Output Anchor (Validation/Testing dataset)** of the **Create Samples** node to the **D** input anchor of the **Model Comparison** node. Connect the **R** output anchor of the **Model Comparison** node to a **Browse** node.
3. Run the workflow. For each model, click the two **Browse** nodes. Evaluate the trained model below:

The adjusted r square of both models (I output anchor):

The mean absolute error of both models when applying to the training dataset (I output anchor):

Copy and paste the coefficient tables of the two models (R output anchor). Discuss the significant predictors of model models and inteprete their impact (positive vs. negative) on helpfulness:

1. Click the two **Browse** nodes of the two **VIF** nodes. Check the GVIF table. Any concerning VIF numbers? (Values of more than 4 or 5 are sometimes regarded as being moderate to high, with values of 10 or more being regarded as very high.)
2. Check the **Browse** node of the **Model Comparison** node. Report the correlation (predicted vs actual) and MAE of the two models. A high correlation and a low MAE indicate a model with better predictive performance.
3. Compare the two models in terms of:

Their predictive performance:

Any overfitting or underfitting problems (you obtained the MAE of the training dataset in step 17, and the MAE of the testing dataset in step 19).

1. Currently, our models do not have great performance due to a limited number of predictors. Can you identify any other predictors to collect/develop? Identify both text-related and non-textual features.
2. Save your workflow and provide a screenshot of your whole workflow below.

**Deliverable:**

Submit this word document with your answers and your workflow file. One submission per team.