# The Great Oppotunities of Customer Reviews Amazon Sentiment Analysis

José Jaén Delgado

September 2021

**Summary** - This **Data Science** project seeks to gain relevant insights from an Amazon customer review dataset using **Data Mining** through a series of **Machine Learning Algorithms** with special focus upon **Natural Language Processing** (NLP). Leveraging user ratings and comments, the **probability distribution of reviews by class** will be studied

**Key Concepts** - Machine Learning (ML), Multinomial Logistic Regression, Data Engineering, TF-IDF, Non-parametric Statistical Inference, Data Mining, Fast Iterative Shrinkage-Thresholding Algorithm (FISTA), NLP

## 1 Introduction

Millions of product reviews are written online by customers on a daily basis, reflecting opinions upon goods and services provided by suppliers. Not only does this feedback pose a great opportunity for improvement to businesses, but it also makes a considerable impact on other potential clients by influencing their buying behavior. Consequently, it would be wise for companies to practice data-driven decision making by getting insights from a thorough review examination. One particular application of this kind of data inspection is **sentiment analysis** or **opinion mining**: "*the computational study of analyzing people's feelings and opinions*" as defined by Ligthart, Catal and Tekinerdogan (2021).

In this Supervised Learning **Big Data Analytics** project a multi-class sentiment analysis of Amazon customers will be carried out to **automate the probabilistic categorization process of comments** and identify the **words that contribute the most to positive reviews** based on an multinomial logistic classification model.

## 2 Data Description

To obtain the product-review dataset I resorted to Julian McAuley and Jianmo Ni's Amazon data website, a superb repository of Recommender Systems Datum, containing millions of product reviews from a wide range of diverse categories. The enormous data-retrieval effort made by the Computer Science Department of University of California San Diego (UCSD) cannot be disregarded and letting them keep a low profile throughout any Data Mining project that profits from its rich information would be unforgivable. Interestingly, McAuley, Ni and Li (2019) experimented themselves with these datasets by concocting a way to justify recommendations using distantly-labeled reviews and fined-grained aspects.

Product-review datasets include reviews (ratings, text, helpfulness votes), product metadata (descriptions, category information, price, brand, image features), as well as links to the items' Amazon web section.

I will analyze **1,689,188 reviews** from the US (2014) related to the *Electronics* category, focusing on users' ratings and comments to predict the probability of identifying the displayed sentiment in the review.

## 3 Programming Software

The UCSD team centralized all the relevant information into a single JavaScript Object Notation (JSON) file, which can be conveniently treated as Python dictionary objects. Hence, it is no surprise that **Python** will be the preferred programming language to elaborate the sentiment analysis task. Particularly, I will be working with **Turi Create**, an open source toolset for creating Core Machine Learning Models and an alternative to the commercial Data Science platform GraphLab Create. For Data Engineering/Manipulation purposes, **SFrame** will be preferred over other popular packages such as Pandas, owing to the out-of-core data structure nature of the former that allows to virtually store any dataframe, independetly of its size, as long as disk space and memory are available. Since we are dealing with a large dataset, SFrame is perfect for the job. Furthermore, Pandas is more RAM demanding and requires another Python package such as scikit-learn for Machine Learning Modeling, whereas Turi Create already contains SFrame and offers multiple ML analytical tools.

Summing up, Turi Create and SFrame are highly-scalable and efficient Python libraries that opportunely serve to attain the objective of this project.

Notwithstanding the aforementioned advantages of Python, statistical software **R** will not be left out, and so the Data Visualizations presented in this project find their inception in said language. Additionally, the dplyr package has made possible a considerable number of Data Exploration queries due to its resemblance to SQL.

Should the reader possess programming knowledge or might they merely be interested in getting a glimpse of the underlying code used for writing algorithms, perform Data Engineering, Data Clearning or Machine Learning Modeling, a link redirecting to the relevant IPython Jupyter Notebook and R Script can be respectively found ***here*** and ***here***.

# 4 Analysis Framework

The approach followed in the present project, similar to Fox and Guestin (2016), permits an iterative process of enhanced Supervised Learning. A collection of features is selected from which a ML Model will learn, and then, as an imperative part of Artificial Intelligence (AI), will efficiently reduce human intervention to a minimum by automating analytical model building based on a specific quality metric to be maximized. Precisely, ML Algorithms will polish the multi-class logistic model by updating estimated parameters in each iteration so as to better fit the data to make accurate predictions.

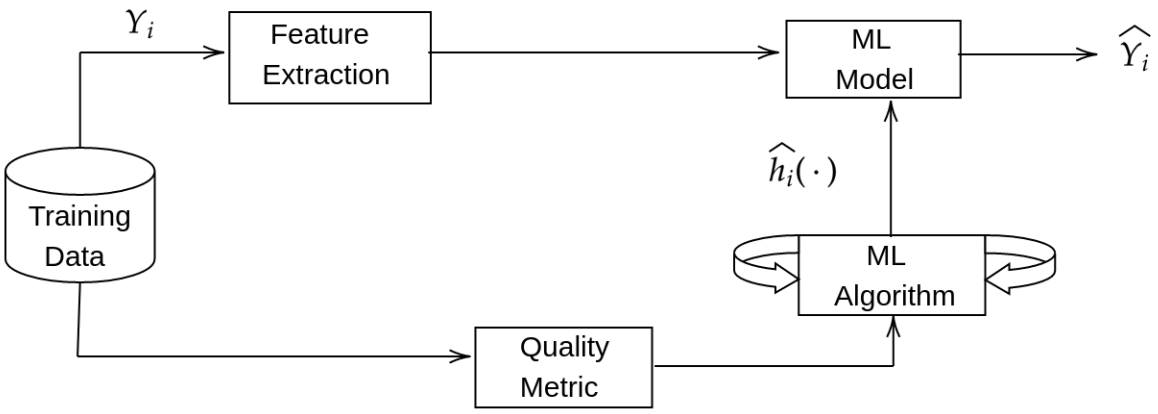


Figure 1: Machine Learning Approach

Focusing now on data composition, the review dataset will be randomly split into three working dataframes: training, validation and testing sets (*holdout validation*). Note that the size of the data allows such framework without negatively impacting the accuracy of the results. An illustrative flow chart is provided below for the sake of synthesizing analytical proceedings, appropriately adapted from StackExchange (2019).

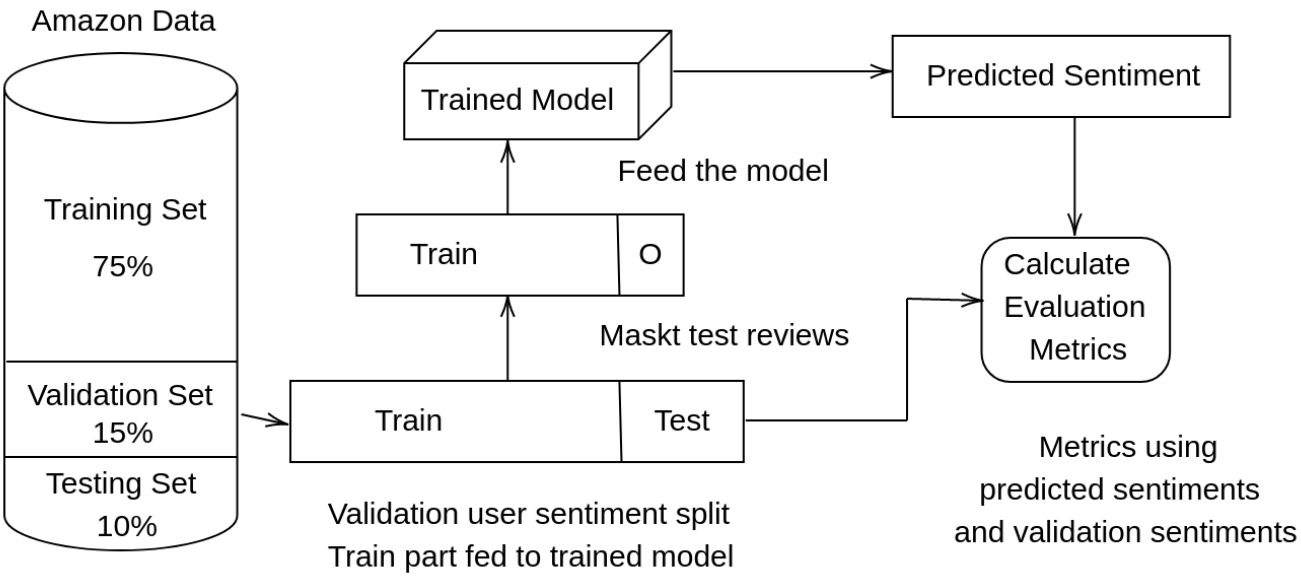


Figure 2: Data Sets Composition and Machine Learning Modeling

The training dataset will be the main one (75% of the data), where we will create our multinomial classifier model and carry out imperative operations. On the other hand, the validation set, stemming from the train set (15% of total data) will assess the performance of our Machine Learning Model, polishing crucial decisions such as Hyperparameter Tuning, penalty values in Regularization, Feature Selection and other technicalities with respect to FISTA. These concepts will be specifically discussed and explained in the ***Machine Learning Approach section***. Evaluating the quality of predictions made by our proposed Machine Learning Model with the test set (remaining 10%) constitutes the final part of the project. Each randomly generated set has its own purpose owing to the urge of preventing overfitting. Indeed, a ML Model that simultaneously learns and gets assesed on a unique dataset is doomed to poor predictive capacity as the training error overcasts the generalization error.

# 5 Data Exploration and Statistical Inference

In the *Data Description* section we mentioned that the total number of reviews amounts to **1,689,188**. Taking a deeper look at the dataset it is revealed that **192,403** distinct clients expressed their opinion about **63,001** different products under the "Electronics" category. Additionally, each unique review is accompanied by its respective discrete rating, which ranges from 1 to 5. Since I will be dissecting comments in a three-division fashion, the Machine Learning Classification Model we will be using requires a *multi-class structure*, ultimately entailing that **one-hot encoding** will be applied not only with respect to two variables but three (One vs Rest approach). Particularizing for our case, the category threshold rule will be similar to Mukherjee et al. (2019): ratings between 1 - 2 will be labeled as *Negative*, scores greater than or equal to 4 are considered *Positive* and three-star views are *Neutral*.

Having established the threshold rule, it is now appropriate to note that **1,356,067** reviews are **Positive**, undoubtedly the dominant group (80.28%), while **142,257** and **190,865** comments are **Neutral** and **Negative**, respectively. It would seem that the motivation to write a review stems mostly from appeased clients rather than dissatisfied ones. Nevertheless, this analysis is superficial as it may well be that certain products provoke a polarizing sentiment among customers and there is also no size normalization. In fact, the most reviewed product presents **4,915** opinions, whereas the least commented one has been reviewed only *five* times. Let us explore the top 10 reviewed products and then proceed with Statistical Inference Techniques (following Ausín Olivera and Alonso, 2019).

Table 1: Top 10 Reviewed Products

| Name | Nº Reviews | Average Rating |
|---|---|---|
| SanDisk 64GB Memory | 4,915 | 4.59 |
| HDMI AmazonBasics Cable | 4,143 | 4.80 |
| Chromecast HDMI | 3,798 | 4.00 |
| Mediabridge HDMI | 3,435 | 4.80 |
| Trascend SDHC Card | 2,813 | 4.66 |
| ErgoFit Headphones | 2,652 | 4.36 |
| DVI HDMI Cable | 2,599 | 4.60 |
| Apple USB Cable | 2,542 | 4.44 |
| Roku 3 Player | 2,104 | 4.42 |
| eneloop AA Batteries | 2,082 | 4.74 |

Disaggregating data shows that Amazon clients seem prone to write comments on presumably top-notch products, owing to the fact that the most reviewed items are also positively rated on average. However, the intricacies of nonlinear relationship between the number of reviews/ratings and clients' buying behavior cannot be ruled out, especially in online transactions as demonstrated by Masłowska, Malthouse and Bernritter (2017). Extrapolating the insights of said research to our analysis, let random variables Average Ratings $\sim X$ and Number of Reviews $\sim Y$: from now on the focal point will be shifted to grouping insights by products, concentrating characteristics by unifying reviews per item and averaging up ratings. Computing Pearson product-moment linear correlation coefficient yields a positive but truly small value (0.052), in other words, the number of times a product gets reviewed seems to be linearly and weakly associated with higher customer regard. Outliers may play a relevant role in distorting the linear relationship between our target variables, which is why visualizing review instances as a function of average ratings might confirm our suspicion. Let us attach a scatterplot defining the aforementioned relation to dissipate sound skepticism around the naive intuition acquired earlier.

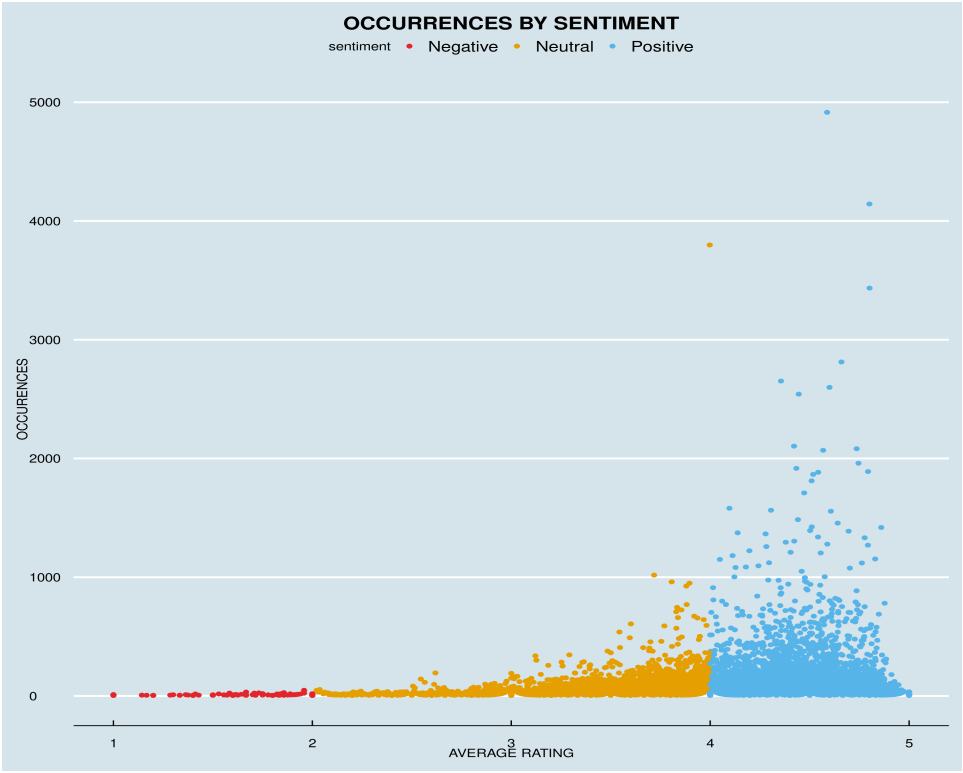


Figure 3: Nº Reviews as a function of Average Ratings

Owing to the eminent data-composition imbalance, the Positive class strictly dominates the rest and so sensible Statistical Inference plays a crucial role in dissociating sampling issues from inner data features. Such difficulty will be addressed in the ***Machine Learning Approach*** section, where we will resort to Random Resampling Techniques to avoid majority class predictions from distorting predictive accuracy. As of now, let us draw our attention to the aforementioned complex relationship between instances of reviews and average ratings.

With respect to this consideration, **Rank Correlation Coefficients**, non-parametric statistical association measures, apply sample-sorting methods in a fashion that overcomes Pearson linear correlation coefficient drawbacks, permitting to identify nonlinear trends. Spearman (1904) proposed a Rank-Order Correlation Coefficient defined as the Pearson correlation coefficient between two variables' distribution functions. Note that it is vital that our data satisfy two assumptions: right variable nature and sampling parity. While the first one requires $X$ and $Y$ to be measured on an ordinar, interval or ratio scale, the second imposes observations to be paired (for each $Y_i$ there is a corresponding $X_i$ and viceversa). Lastly, it is remarkable to recall that Spearman's correlation determines the degree to wich a relationship is monotonic (not necessarily linear), pursuing the strength and direction of such monotonic relationship.

Let the formal definition of Spearman's Rank-Order Correlation Coefficient be expressed as :

$$\rho_s = \rho(F_x(X), F_y(Y))$$

Given a paired sample $\{(\mathrm{Y}_i,\ X_i)\}_{i=1}^{n}$ it is estimated by:

- Calculating ranks $\{R_1,\ R_2,\ ...,\ R_n\}$ for $\{X_1,\ X_2,\ ...,\ X_n\}$
- Calculating ranks $\{S_1,\ S_2,\ ...,\ S_n\}$ for $\{Y_1,\ Y_2,\ ...,\ Y_n\}$
- Applying Pearson correlation coefficient to $\{(\mathrm{R}_i,\ S_i)\}_{i=1}^{n}$

Bearing in mind properties of the linear correlation coefficient it is then easy to conclude that the estimator can be expressed as:

$$\hat{\rho_s} = \frac{\sum_{i=1}^{n}(R_i - \bar{R})(S_i - \bar{S})}{\sqrt{\sum_{i=1}^{n}(R_i - \bar{R})^2}\sqrt{\sum_{i=1}^{n}(S_i - \bar{S})^2}} \tag{1}$$

The resulting estimate of the above coefficient is 0.0096, even smaller than the previous one, which is not surprising given the flat-shape scatterplot. We can advance further and hypothesize about the independence between average rating and number of reviews, parting from an unknown joint distribution $F(x, y)$ and marginal distributions $F_1(x)$ and $F_2(y)$, testing:

$$\begin{aligned} \mathrm{H}_0\text{: } & F(x,y) = F_1(x)F_2(y) \\ \mathrm{H}_1\text{: } & F(x,y) \neq F_1(x)F_2(y) \end{aligned}$$

The test statistic is:

$$S_p = \sum_{i=1}^{n}(R_i - S_i)^2$$

It can be shown that the latter expression is closely related to *Equation 1* (for more details check the corresponding ***mathematical derivation***):

$$\rho_s = 1 - \frac{6S_p}{n^3 - n} \tag{2}$$

Testing this hypothesis on our data translates into a rejection of the null hypothesis, as the data provided enough evidence against the independence of the relevant variables at a 5% significance level (p-value = 0.0156). Thus, although small at first glance, there seems to exist a nonlinear relationship between our variables, as suspected before.

We can quickly validate the insights presented above with another non-parametric hypothesis test for statistical dependence based on Kendall's tau coefficient. Centered upon probabilistic differences, Kendall (1938) defined it (for a given pair):

$$\tau = \Pr\{(X_k - X_j) \cap (Y_k - Y_j) > 0\} - \Pr\{(X_k - X_j) \cap (Y_k - Y_j) < 0\}$$

Tau can be estimated in the following way (refer ***here*** for the derivation):

$$\hat{\tau} = \frac{n_c - n_d}{\frac{n}{2}(n-1)} \tag{3}$$

Where random vectors come from a bivariate population, $n_c$ is equal to the number of concordant pairs, $n_d$ denotes pair-discordance and $j < k$.

Table 2: Concordant and Discordant pairs

| Pair | Sign |
|---|---|
| $\{X_j < X_k\} \cap \{Y_j < Y_k\}$ | Concordant |
| $\{X_j > X_k\} \cap \{Y_j > Y_k\}$ | Concordant |
| $\{X_j < X_k\} \cap \{Y_j > Y_k\}$ | Discordant |
| $\{X_j > X_k\} \cap \{Y_j < Y_k\}$ | Discordant |

Pair vectors seem to be concordant as the estimate for tau is positive (0.0074) and we also find enough evidence to reject the null hypothesis at a 5% significance level that the distributions of average ratings and number of reviews are independent (p-value = 0.0065).

Another sensible explanation as to why we have opted for Spearman and Kendall correlation coefficients emanates from the fact that the distribution independence hypothesis test requires assuming bivariate normality in the case of using Pearson product-moment correlation coefficient, whereas it is not necessary for Rank-Order coefficients. It is easily expounded by the fact that a non-parametric approach was taken, which does not rely on data belonging to any particular parametric family of probability distributions. Furthermore, had we decided to incorporate Pearson linear correlation coefficient to test said joint distribution independence, an arduous argumentation likely to persuade no reader would have been necessary, as a quick glimpse at the data shows that review instances and average ratings do not quite follow a Gaussian distribution.

The graphs below deny any remote resemblance to a mean-centered and symmetric distribution. Computing third moments, a measure of the lopsidedness of the distribution, reveals a negative-skewed distribution (-1.037) for average ratings and an eminently skewed-right distribution (20.48) in the case of reviews count, implying that the median is greater than the mean for the former variable, and otherwise for the second.


Figure 4: Average Ratings and Nº Reviews Distributions

More analytically, a hypothesis on whether average ratings and number of reviews follow a normal distribution or not can be tested. For such purpose, examining the *goodness of fit* of said distributions to the Gaussian one is crucial. Statistical-judgement making about the population distribution is ubiquitous in non-parametric inference (Genovese et al. 2016), and in this particular case, it is of great interest to compare the empirical distribution $\hat{F}_n(\omega)$ to the theoretical normal distribution $F(\omega)$. This proceeding allows to study the probabilistic divergence between the distribution followed by the data and the Gaussian distribution, which is imperative due to the fact that multiple inference and modeling techniques heavily rely on normally distributed random variables.

Let the empirical distribution be expressed as:

$$\hat{F}_n(\omega) = \frac{\#\text{ observations} \leq \omega}{n}$$

The formal distribution goodness of fit test can be defined as:

$$\mathrm{H}_0\colon\ F = F_0$$
$$\mathrm{H}_1\colon\ F \neq F_0$$

By the Glivenko-Cantelli theorem, which determines the asymptotic behavior of the empirical distribution function as the number of independent and identically distributed observations grows (Tucker, 1959), we can therefore prove that $\hat{F}_n(x)$ uniformly converges to $F(x)$ with probability one, that is to say:

$$\sup_{\omega\in\mathbb{R}} |\hat{F}_n(\omega) - F(\omega)| \to 0$$

Note that the explicit expression of the cumulative theoretical distribution function is $F(\omega) = \Phi$, where:

$$\Phi = \int_{-\infty}^{+\infty} \frac{1}{\sigma\sqrt{2\pi}} e^{-\left(\frac{\omega-\mu}{2\sigma}\right)^2}\, d\omega$$

A common non-parametric test-statistic for testing goodness of fit is the one proposed by Kolmogorov and Smirnov, however, in practice it is necessary to estimate unknown parameters that ultimately characterize the theoretical distribution. A workaround would be estimating both, mean and standard deviation by MLE (Maximum Likelihood Estimation) and using Kolmogorov-Smirnov-Lilliefors statistic:

$$\sup_{\omega\in\mathbb{R}} |\hat{F}_n(\omega) - F_{N(\bar{\omega},s)}(\omega)| \sim \triangle L_n$$

In addition to the aforementioned argument, Glen (2016) poins out that the Lilliefors normality test corrects the Kolmogorov-Smirnov statistic for small values at the tails of probability distributions and can be perfectly used when population mean or standard deviation remain unknown. Moreover, a power-comparison study was carried out by Razali and Wah (2011) showing that in asymmetric distributions Lilliefors test outperforms its uncorrected KS counterpart.

Unsurprisingly, we reject the null hypothesis that our variables follow a normal distribution at a 5% significance level (p-value $<$ 2.2e-16).

Lastly, let us study the number of distinct comments per product, with special consideration towards its rating class. It has been noted that positive reviews clearly dominate the rest, thus imbalancing the data. Nonetheless, no central tendency inference has been performed on average comments per product type. As we have rejected the hypothesis that any of our variables are normally distributed, it is then necessary to call upon non-parametric statistical inference. Note that precisely because of this reason, ANOVA (Analysis of Variance) and thus, Tukey's Honestly Significant Difference test are not appropriate, nor is it Wilcoxon signed-rank test, due to the violation of distribution-symmetry. Consequently, a sign test on the median is proposed, where we will test whether the median of positive reviews is greater than the median of neutral comments or not (where $P$ represents positive comments and $N$ neutral reviews):

$$\begin{aligned} \mathrm{H}_0\text{: } & Q^P_{0.5} = Q^N_{0.5} \\ \mathrm{H}_1\text{: } & Q^P_{0.5} > Q^N_{0.5} \end{aligned}$$

Should the null hypothesis be true, the number of sample "positive" observations greater than $Q^N_{0.5}$ must be approximately equal to the sample "positive" observations smaller than $Q^N_{0.5}$.

Where $\Pr(X_i \geq Q^i_{0.5}) = \Pr(X_i \leq Q^i_{0.5}) = 0.5, \quad i \in \{P,\, N\}, \quad$ and:

$$Z_i = \mathbb{1}\{X_i - Q^N_{0.5}\} = \begin{cases} 1 & \text{if} \quad X_i > Q^N_{0.5} \\ 0 & \text{if} \quad X_i < Q^N_{0.5} \end{cases}$$

Thus, the null hypothesis is true iff $\Pr(Z_i = 1) = \Pr(\mathrm{X} \geq Q^N_{0.5}) = 0.5$

$$\underbrace{\sum_{i=1}^{n} Z_i}_{\text{test statistic}} \sim_{H_0} Bin(n, 0.5) \quad \& \quad \text{p-value} = \Pr\left(\mathrm{Bin}(n, 0.5) \geq \sum_{i=1}^{n} Z_i\right)$$

The data provided enough evidence to reject the null hypothesis that the median of positive reviews equals that of neutral comments at a 5% significance level. Amazon customers' willingness to publish their opinion is associated with higher ratings.

# 6 Machine Learning Approach

In this section every step of the Machine Learning Approach previously presented as a ***flow chart*** is expounded, detailing technical procedures, justifying modeling decisions and explaining critical constraints faced during the development of our Big Data Analysis.

## 6.1 Feature Extraction & Feature Engineering

An imperative initial step of any kind of study is opting for a particular set of relevant explanatory covariates with the objective of better understanding the target variable. In ML the particular label used to denote such vital elements is *feature*, defined by Bishop (2006) as "*an individual measurable property or characteristic of a phenomenon*". Numeric features tend to dominate over their qualitative counterparts due to the intrinsic characteristics of ML Algorithms and Data Science tasks. In fact, quantifiable features are essential in order for any model to properly work. Nonetheless, as pointed out throughout this project, our case is drastically different as we are dealing with Amazon reviews, which is an eminent case of descriptive data as this sample review illustrates:

> "You can't beat the price and the installation was simple as can be. We installed it in our office and it is perfect! No problems and was completed in a matter of minutes"
>
> (Reviewer: "Jasson from GA", Rating: 5.0)

It can be clearly seen that these reviews are an array data structure of bytes, and thus cannot be treated as integers or floats, requiring Data Engineering techniques to be conveniently transformed into a measurable unit.

One plausible approach to overcome the aforementioned issue would be the vectorization of words composing each review, ultimately working with word counts, whose data structure is expressed as integers, rather than a collection of non-operational strings. Harris (1954) referenced this idea on *Distributional Structure*, and nowadays it is known as the "Bag-of-Words" model (BoW), a pretty simplifying text representation used in NLP where the frequency of each word is used as a feature for taining a classifier.

In order to implement this method, a word count algorithm is used and its results stored into a column of the main dataframe in JSON format. The visualization below shows the most common words (excluding stop words) used by Amazon reviewers:

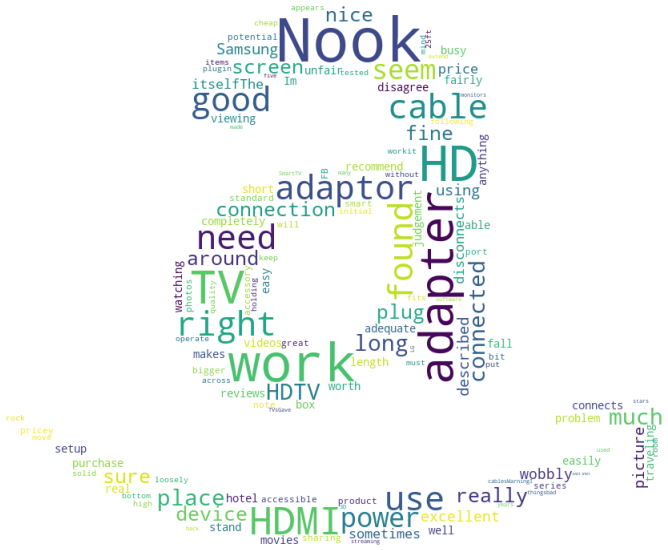


Figure 5: Reviews Word Cloud

It is not surprising that words such as "*HDMI*", "*adapter*", "*HD*", "*TV*" or "*Nook*" stand out, owing to the structure of reviews, in which apart from personal opinions and feelings, a reference to the reviewed electronic article is vital for the sake of context. Predicting the underlying sentiment of an Amazon customer by solely considering the number of word occurrences is rather naive: common words distort a meaningful representation of the relevant subjective features, downweighting truly important lexicon. For instance, prepositions, definite and indefinite articles such as "*in*", "*the*", "*this*" or "*a*" are likely to play a dominant role in any piece of text irrespective of its length, and therefore BoW will incorrectly emphasize such terms. A desired solution would imply focusing on "important words", i.e, those that appear frequently in a given review (common locally) but rarely in corpus (rare globally). Taking into account such trade-off between local frequency and global rarity is key in discounting the importance of ubiquitous but irrelevant words. Thus, a sensible alternative is "Term Frequency - Inverse Document Frequency" (TF-IDF), a statistical measure which allows to reflect the aforementioned insights. Similarly to BoW, it transforms a collection of strings into a numerical value (normally a float) but it does adjust for significant words.

TF-IDF stems from the intersection of two vectors: word count in each observation (Term Frequency) and an heuristic that diminishes the weight of common and (sometimes) meaningless words in the corpus (IDF).

Said heuristic takes the logarithmic value of the ratio between the quantity of total reviews ($N$) and the number of reviews that used the evaluated word added up by one ($1 + \sum_{j=1}^{n}\{w_i \in d_j\}$). Consequently, the more reviews in which word $w_i$ is present, the higher the IDF denominator, and by applying log properties, the resulting TF-IDF is practically null. In contrast, when $w_i$ cannot be detected in many articles $d_j$, the values of the second vector increase, yielding a high TF-IDF.

On balance, our ML Model will learn from a series of quantitative features that are informative of the importance of each individual string forming a review.

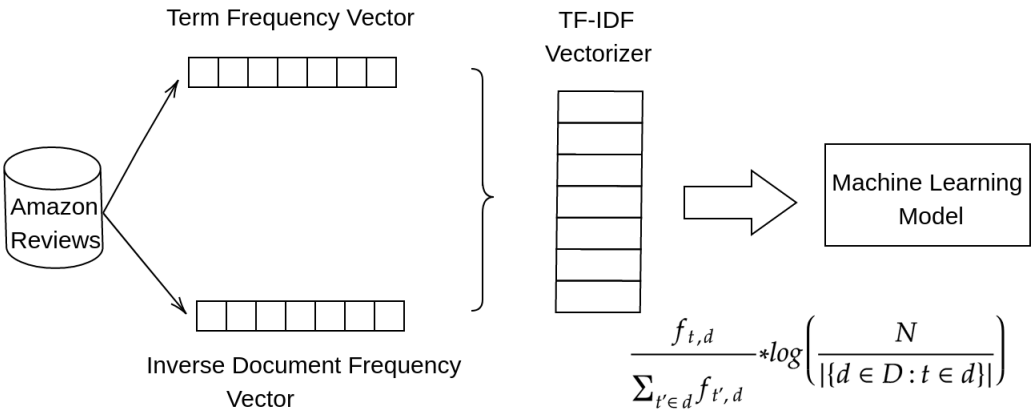


Figure 6: TF-IDF Diagram

Before formally defining our ML Model, it is important to account for the problematic class imbalance mentioned in the *Data Exploration and Statistical Inference* section. A severe skew in the class distribution can easily induce bias in in the data, ultimately provoking pernicious side effects in the ML Algorithms output. For our particular case as more than 80% of the data represent positive reviews, it is possible that the multinomial logistic classifier would not be able to properly learn from minority classes due to their scarcity and therefore shed no light on neutral and negative reviews trigger words. Furthermore, said classifier could still perform relatively well in quality metrics such as accuracy, explained by the fact that predicting $\hat{Y}_i = +1$ turns out to be correct in the vast majority of cases, and so extreme caution should be taken.

Rebalancing class distributions in the training data was achieved through a hybrid random resampling approach where undersampling and oversampling were combined with a NumPy discretized uniform distribution random integer generator. Whereas Random Undersampling implies promiscuously eliminating majority class observations, Random Oversampling builds a synthetic dataset by duplicating data from the under-represented categories. The flow chart below illustrates the underlying Data Engineering mechanics that materialized the aforementioned statistical insights.

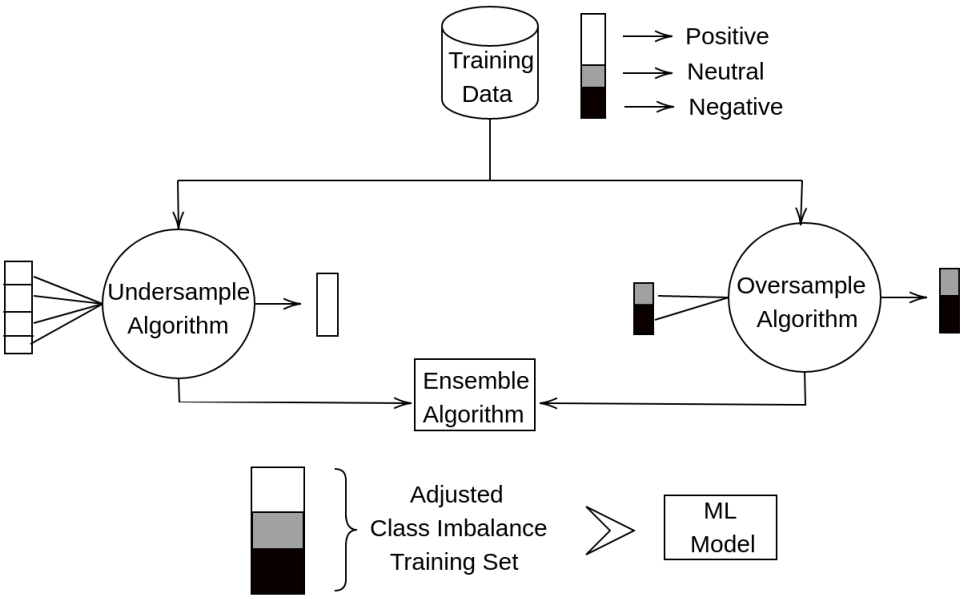


Figure 7: Hybrid Random Resampling Algorithm

Although Batistia et al. (2004) proposed an undoubtedly smarter and more efficient approach known as SMOTE-ENN by integrating Synthetic Minority Oversampling TEchnique (SMOTE) and Edited Nearest Neighbor (ENN), unfortunately it was not possible to apply it to our dataset due to the large number of elements contained within the TF-IDF vectorizer array, ammounting to millions of features in practice, automatically killing the kernel when trying to run said algorithm with scikit-learn. The advantages of SMOTE-ENN technique are clearly summarized by Andhika (2021), pointing out that this resampling technique leverages "*SMOTE ability to generate synthetic examples for minority class and ENN to delete some observations from categories that are identified as having different class between the observation's class and its K-nearest neighbor majority class*".

## 6.2 Machine Learning Model

Given an i.i.d distributed sample of $n$ observations: $\{(\mathrm{y}_i,\, x_i)\}_{i=1}^{n}$

- Let $Y$ be an $n$ $x$ 1 output vector (sentiment)
- Let $H$ be an $n$ $x$ $d$ matrix with $d$ predictive functions of features for $n$ observations
- Let $w$ be an $d$ $x$ 1 vector of unkown parameters to be estimated
- Let $\epsilon$ be an $n$ $x$ 1 vector of disturbances
- Let $h(\cdot)$ be a function of the features denoted by $x_{ji}$

The general overview of the Machine Learning Model is:

$$Y_i = \sum_{j=0}^{d} w_j h_j(x_{ji}) + \epsilon_i$$

Rewriting in matrix notation:

$$\begin{bmatrix} Y_1 \\ Y_2 \\ Y_3 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} 1 & h_1(x_{11}) & h_2(x_{21}) & \dots & h_d(x_{d1}) \\ 1 & h_1(x_{12}) & h_2(x_{22}) & \dots & h_d(x_{d2}) \\ 1 & h_1(x_{13}) & h_2(x_{23}) & \dots & h_d(x_{d3}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & h_1(x_{1n}) & h_2(x_{2n}) & \dots & h_d(x_{dn}) \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \dots \\ \epsilon_n \end{bmatrix}$$

For the sake of maneuverability, we will proceed with a simpler expression:

$$Y = Hw + \epsilon$$

Due to eminent data collection and financial limitations, population parameters require estimation rather than direct computation, working in practice with:

$$\hat{Y} = H\hat{w} + \varepsilon$$

Note that the output vector is but a set of predictions $\hat{Y}$ resulting from the dot product of the feature and estimator matrices $H\hat{w}$. As discussed earlier, customer reviews have been appropriately transformed into quantifiable units through NLP and Data Engineering techniques, therefore each $h_j(x_{ji})$ represents a TF-IDF value.

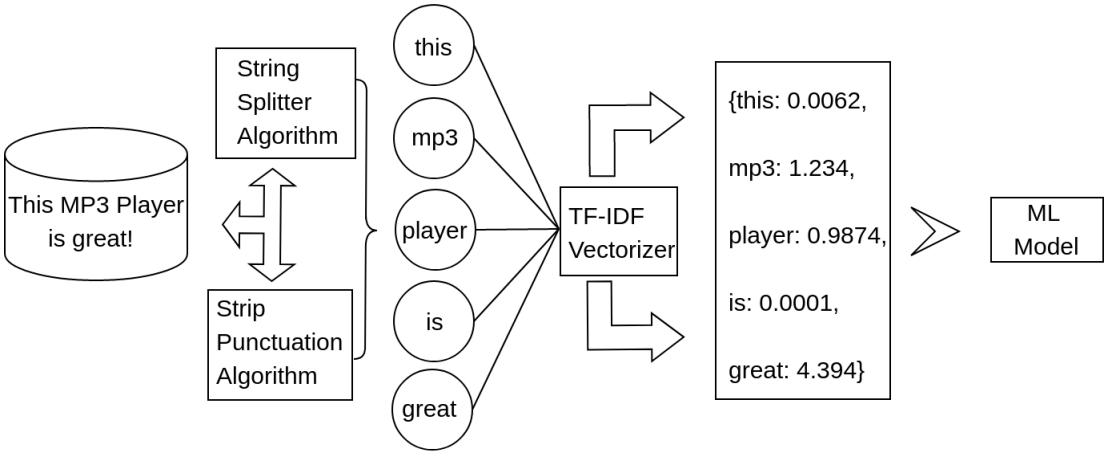


Figure 8: Data Engineering process of Descriptive Features

In addition, it is imperative to recall that we constraint the components of the predicted output vector to be contained within a three unique integer list: +1 (positive reviews), 0 (neutral reviews) and -1 (negative reviews), which is easily applied in Python with an anonymous or lambda function. Rather than predicting the exact type of sentiment displayed in a given review, focus should be shifted to quantifying the probability of a comment pertaining to a specific sentiment category conditional on the linguistic components of said review: $\Pr(Y_i = y_i|X_i, w)$, where $y_i \in$ {-1, 0, 1}, $X_i$ represents the words of the analyzed review and $w$ the ML Model parameter vector for words. Nevertheless, it is unlikely that the dot product $H\hat{w}$ lies between 0 and 1, which are probabilistic boundaries. A specifically designed classifier for dependent binary variables (Turi Create automatically adjusts it for our multi-class perspective) is logistic regression. Apart from its easy interpretation, the underlying reason behind the convenience of the logistic regression classifier is simple: it overcomes the unbounded dot product difficulty as it supplies a link function that can force predicted values to lie between 0 and 1 by taking a sigmoid shape as expressed below:

$$\Pr(Y_i = 1|X_i, w) = \frac{1}{1 + e^{-w^\intercal h(x_i)}}$$

Notice that our model is plugged into the sigmoid function, procuring nonlinear coefficients, which ditches out traditional OLS estimation, necessarily dealing with Generalized Linear Regression, whose models generalize linear regression by allowing the linear model to be related to the response variable via the link function we described above, allowing the magnitude of the variance of each measurement to be a function of its predicted value as defined by Zhao (2012). A likelihood function is concocted from which we will estimate probabilities by maximizing it, aiming to procure the correct $\hat{\Pr}(Y_i = y_i|X_i, w)\ \forall\ i$. We seek to maximize the joint probability distribution of the data, treated as a function of the unknown coefficients.

$$\ell(w) = \Pr(Y_1 = y_1|X_1, w) * \Pr(Y_2 = y_2|X_2, w) * ... * \Pr(Y_n = y_n|X_n, w)$$

$$\ell(w) = \prod_{i=1}^{n} \Pr(Y_i = y_i|X_i, w)$$

Let us take the log of the likelihood function for simplicity:

$$\mathcal{L}(w) = \ln \prod_{i=1}^{n} \Pr(Y_i = y_i|X_i, w)$$

Since we are dealing with a linear classifier and aim to smooth predictions by tackling any inconvenience that induces disturbance, it is then essential to avoid *overfitting*. This last term refers to situations "*when the model memorizes the noise and fits too closely to the training set*, [becoming] *overfitted, unable to generalize well to new data*" (IBM, 2020). This could be achieved by **Regularization**, constraining the weights of the model. The main idea is to add a regularization term to the log-likelihood function, forcing the Learning Algorithm to carry out both, a correct fit to the training data and keeping the weights as small as possible. We will opt for **Elastic Net Regularization**, which combines L1 and L2 Regularization. The former is called Least Absolute Shrinkage and Selection Operator Regression (Lasso regression), and implies incorporating the L1 norm to the main estimation quality metric. In practice it eliminates weights of irrelevant features. The latter, named Tikhonov Regularization, adds a L2 norm to the log-likelihood function. Estimated weights vector is retrieved by optimizing:

$$\hat{w} = \arg\max_{w} \left\{ \ln \prod_{i=1}^{n} \Pr(Y_i = y_i|X_i, w) - \lambda_1 ||w||_1 - \lambda_2 ||w||_2^2 \right\}$$

Where $\lambda_1$ and $\lambda_2$ already account for L1 and L2 regularization mix ratio. Please refer to the ***mathematical derivation section*** for more detailed explanations.

## 6.3 Machine Learning Algorithms

In ML, Gradient Ascent Algorithms are common when working with logistic regression, defined by Géron (2018) as "*an iterative optimization approach, that gradually tweaks the model parameters to* [maximimize] *the* [likelihood] *over the training set, eventually converging to the same set of parameters as* [MLE]". Furthermore, as a not differentiable L1 norm regularization term $||w||_1$ is being introduced to the log-likelihood function, it is then necessary to resort to alternative ML Algorithms such as Fast Iterative Shrinkage-Thresholding Algorithm (FISTA).

---

**Algorithm 1:** Fast Iterative Shrinkage-Thresholding Algorithm (FISTA)

---

**Input:** Training data $S$, Elastic Net Regularization parameters ($\lambda_1$, $\lambda_2$),
Learning Rate $\eta$, Tolerance $\epsilon$, $\mathcal{L}(w)$ and L2 norm Gradient $\Psi(w_j)$

**Output:** Model parameters $\Theta = (w_0, \mathbf{w})$

**Arbitrarily initialize:** $w_0 = \Omega \wedge w_1 = z_1$

**Set:** $\mu_0 = 0,\ \mu_t = \dfrac{1 + \sqrt{1 + 4\mu_{t+1}^2}}{2},\ \gamma_t = \dfrac{1 - \mu_t}{\mu_{t+1}}$

**repeat**

  **for** $(x, y) \in S$ **do**

    $w_0^{(t+1)} \leftarrow w_0^{(t)} + \eta \left( \sum_{i=1}^{n} h_0(x_i) \left[ \mathbb{1}[Y_i = 1] - \Pr(Y = 1 | X_i, w) \right] \right)$

    **for** $j \in \{1, \ldots, p\} \wedge x_j \neq 0$ **do**

      $z_j^{(t+1)} = \text{prox}_{\frac{\lambda}{\beta} ||\cdot||_1} \left[ w_j^{(t)} + \frac{1}{\beta} \Psi(w_j) \right]$

      $w_j^{(t+1)} = (1 - \gamma_t) z_j^{(t+1)} + \gamma_t z_j^{(t)}$

    **end**

  **end**

**until** *convergence tolerance is satisfied*;

---

Beck and Teboulle (2009) showed that by adding acceleration or momentum $\gamma_t$ to the shrinkage operator of the Proximal Gradient Algorithm ISTA, convergence time complexity could be reduced from $\mathcal{O}(1/\text{t})$ to $\mathcal{O}(1/\text{t}^2)$, which is truly a considerable improvement given that ISTA was even more efficient than Subgradient Algorithms with convergence time $\mathcal{O}(1/\sqrt{t})$. Another reason to opt for Proximal Gradient over Subgradient Metod Algorithms is the presensence of self-tunning properties.

Examining our case, $w_0$ is not regularized and the rest of coefficients $w_j$ result from solving a maximization composite problem formed by a smooth function $\Psi(w_j)$ and a not differentiable, easy prox, $\beta$-smooth function $||w||_1$, where $\beta = 1/\eta$.

Note that the optimal Regularization term vector $(\lambda_1, \lambda_2)$ is procured via holdout validation, following the ML approach presented in the *Analysis Framework* section, which involves estimating a multinomial logistic classifier model for each L1 and L2 norm pair value that learns from the randomly resampled training data. Once the models are estimated, they get assessed in the validation set and the selected $\lambda_1$ and $\lambda_2$ will be the ones whose multi-class logistic classifier yields the highest quality metric values (not necessarily all of them at the same time but rather a balanced result). It is imperative to choose exponencially spaced $\lambda_k$ values for $k \in \{1, 2\}$, easily applied in Python with NumPy *logspace* function.

**Algorithm 2:** Holdout Validation Algorithm

**Input:** Validation data $\mathcal{V}$, Multinomial Logistic Classifier $\psi$,
Elastic Net Regularization Parameters $(\lambda_1, \lambda_2)$
**Output:** $(\lambda_1^*, \lambda_2^*)$
**Initialize:** $\Upsilon$ = Exponentially Spaced $\lambda_k$ where $k \in \{1, 2\} \wedge$ models $= \emptyset$
**repeat**
  **for** $\lambda_{1i} \in \Upsilon$ **do**
    **for** $\lambda_{2j} \in \Upsilon$ **do**
      model $= \psi(\mathcal{V}, \lambda_{1i}, \lambda_{2j})$
      models.append(model)
      Quality Metrics Vector: $QMV_{\mathcal{V}}$(Accuracy, Precision, Recall, F1 Score)
      model$^*$ = models.max($QMV_{\mathcal{V}}$)
    **end**
  **end**
**until** *nested for loop completed*;

Lastly, as a way to learn from both, satisfied and dissatisfied Amazon customers, a k-nearest neighbors Algorithm (K-NN) is proposed to solve the optimization problem of finding the closest comments to the most positive and most negative reviews in our dataset. Specifically, a 3-NN search will be carried out to gain insights from the expressions, words and articles contained in the data, clarifying and solving doubts around consumers' behavior.

**Algorithm 3:** K-Nearest Neighbors Algorithm (K-NN)

**Input:** Training data $S$, Review corpus $X^{NN} = (X_1, X_2, \ldots, X_n)$,
Query Review $X_q$

**Output:** List of $k$ similar reviews $\Theta = (X_1, X_2, \ldots, X_k)$

**Initialize:** dist2kNN = sort$(\delta_1, \ldots, \delta_k)$

**repeat**
  **for** $(i = k + 1, \ldots, N) \in S$ **do**
    $\delta = \text{dist}(X_i, X_q)$
    **if** $\delta <$ dist2kNN[$k$] **then**
      find $j$ such that $\delta >$ dist2kNN[$j$ - 1] but $\delta <$ dist2kNN[$j$]
      remove furthers review and shift queue:
        dist2kNN[$j$+1:$k$] = dist2kNN[$j$:$k$-1]
        dist2kNN[$j$] = $\delta \wedge X^{k-NN}[j] = X_i$
    **end**
  **end**
**until** *updated k-nn found*;

# 7 Sentiment Analysis Results

Having explained the methodology employed throughout this Big Data Analytics project, it is now time to share our main findings, pointing out insightful breakthroughs. Although the ML Model has learned from the Randomly Resampled training dataset, another specification run on imbalanced data is used for comparison purposes.

Quality metrics to be studied on test data are Accuracy (percentage of correctly predicted review classes), Precision (positive predictive value capacity), Recall (true positive rate or sensitivity) and F1 Score (harmonic mean of the model's precision and recall). Whereas the first one does not quite account for precise predictions, the latter are significantly more robust:

$$\text{Precision} = \frac{\text{TP}}{\text{TP + FP}} \qquad \text{Recall} = \frac{\text{TP}}{\text{TP + FN}}$$

$$\text{F1 Score} = 2 \left( \frac{\text{Precision * Recall}}{\text{Precision + Recall}} \right)$$

Where TP denotes "*True Positives*", FP stands for "*False Positives*" and FN represents "*False Negatives*"

Looking at the table below, it can be concluded that our *hybrid rebalancing approach* has proven to be effective. Notice how a strongly class-skewed dataset has eminently harmed predictive performance.

Table 3: ML Model Test Data Evaluation

| | Imbalanced Data | Random Resampling |
|---|---|---|
| Feature | TF-IDF | TF-IDF |
| $\lambda_1$ | 10 | 1284.43 |
| $\lambda_2$ | 10 | 356.87 |
| Accuracy | 0.8 | 0.83 |
| Precision | 0.24 | 0.61 |
| Recall | 0.4 | 0.6 |
| F1 Score | 0.3 | 0.604 |

By selecting $\hat{Y}_i = +1 \ \forall i$, a model learning from uncleaned data correctly predicts 80% of classifications due to overrepresentation of the positive category (it is no coincidence that accuracy coincides with its class weight). Nevertheless, an underlying deficient modeling approach is uncovered when analyzing other indicators. A confusion matrix stemming from the preferred ML Model on test data graphically summarizes its predictive quality:

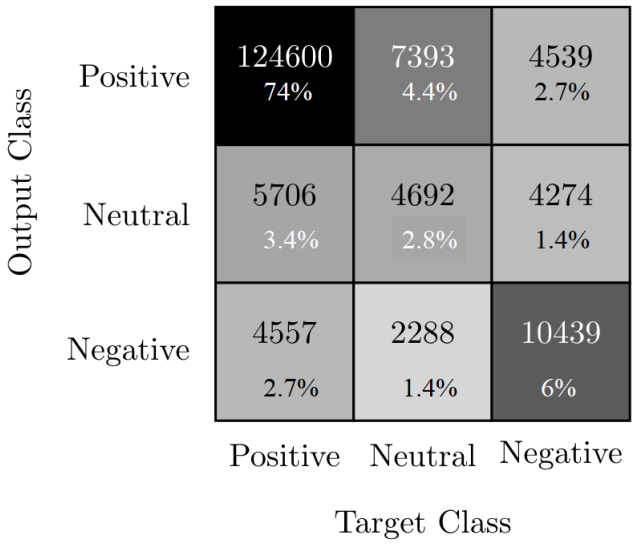


Figure 9: Confusion Matrix

Overall performance of the ML Model is considerably high given the multiclass nature of the logistic classifier, which surely increases modeling complexity. Obtaining a 60% score in Precision and Recall demonstrates that despite facing a challenging business task, our specification is more than capable of identifying review categories (Recall) and displaying relevant results (Precision). A more detailed analysis shows that the vast majority of positive and negative reviews are correctly classified by the model, while predictive precision is lower with respect to neutral comments, an expected outcome given the difficulty of defining an anodyne product. Moreover, well-known Machine Learning Text Analysis company MonkeyLearn (2021) precisely highlights the intricacies of neutral-class reviews, devoting a specific bullet point to tackle it.

Moving on to positive and negative review trigger expressions, a summary table collecting top contributing words is presented below.

Table 4: Top 5 Most Positive and Negative Words

| | Imbalanced Data | | Random Resampling | |
|---|---|---|---|---|
| | Positive Words | Negative Words | Positive Words | Negative Words |
| Top 1 | $w_0$ (0.018) | **americanas** (-0.002379) | **highly** (0.0359) | **returning** (-0.0223) |
| Top 2 | **great** (0.0095) | **socapu** (-0.002379) | **great** (0.032) | **returned** (-0.0221) |
| Top 3 | **works** (0.0088) | **watchvb0w36gayzia** (-0.002379) | **perfect** (0.0269) | **return** (-0.0195) |
| Top 4 | **this** (0.0076) | **chromebookslets** (-0.002379) | **pleased** (0.0263) | **waste** (-0.0191) |
| Top 5 | **recommend** (0.0074) | **onessucks** (-0.002379) | **perfectly** ( 0.0259) | **useless** (-0.018) |

Note: Numbers in parenthesis represent weight value for each word

Our preferred multinomial classifier successfully identifies the relevant linguistic elements for describing customers' emotions. Adjectives such as "*great*", "*perfect*" or "*pleased*" require no explanation, adverbs "*highly*" and "*perfectly*" are normally used in positive contexts. Interestingly, different tenses of the verb "*return*" dominate the top three negative connotation words rank, surely because returning a product implies dissatisfaction or malfunctioning, while"*waste*" and "*useless*" commonly express poor quality.

Once again, resampling the data has considerably paid off as the imbalanced specification miserably fails to retrieve sensible results. Note that the intercept $w_0$ contributes the most to writing a positive comment (literally an empty review scores the highest!) and a stop word like "*this*" has surprisingly made its way to a category it certainly does not belong to. Top negative words are simply chaotic, full of spam jargon ("*watchvb0w36gayzia*") from a 50-page PDF filler, incorrectly written words like "*socapu*" instead of "*System-On-a-Chip*" (SoC) and "*Accelerated Processing Unit*" (APU), and "*chromebookslets*".

Using the Random Resampling ML Model we can retrieve the most positive and negative reviews based on their conditional probability value. Displaying the entirety of both consumer comments is practically impossible due to their extensive length, thus, a shorter modified version is provided as a summary:

> "This Xcenior 36 is very roomy and can carry one or two fullframe DSLRs with battery grips along with five to eight lenses. This bag has a perfect sturdy build quality with tons of padding everywhere. The dividers have a great combination of being both reasonably stiff and thickly cushioned. Highly recommended"
>
> (Most Positive Review)

> "If you have an ATI card you will spend days on this unit and it will likely never work. When Hauppauge gets their act together I will up this to a five star review, until then this is likely a useless box for many people especially for premium content"
>
> (Most Negative Review)

Apart from product characteristics and drawbacks pointed out by consumers, it can be noticed that top trigger words from our ML Model are present in the most polarized reviews. As users provide valuable information to both, suppliers and Amazon, great business opportunities can arise by analyzing similar comments. The former could employ review analysis to improve or fix flaws as well as further polish top-notch features, while distributors such as Amazon can predict customer sentiment and select which reviews are shown in the comments section. For such information retrieval purpose, the proposed K-NN Algorithm in the preceding section of utmost importance.

Oftentimes, the distance metrics used for evaluating proximity in K-NN Algorithms is Euclidean distance, expressed as:

$$D = \sqrt{\sum_{j=1}^{d} a_j (x_{ij} - x_{qj})^2}$$

However, Euclidean distance is sensitive to document length, translating this side effect to bias towards short articles. This arises as TF-IDF is proportional to word frequencies and more prolonged articles logically present longer TF-IDF vectors. To remove this bias, we turn to cosine distances, allowing comparison among heterogeneous review lengths:

$$D(x, y) = 1 - \frac{x^\intercal y}{||x||\,||y||}$$

In spite of following a 3-KNN Algorithm approach, only the two most similar reviews of each class will be shown for clarity and presentation purposes. The results of said K-NN search for the top most positive review are presented below:

> "I consider my Vanguard Xcenior bag the best investment I ever made in a piece of photographic equipment. The seals and zippers are of such high quality that the Vanguard Xcenior bag is a perfect 10"
>
> (First Nearest Neighbor Positive Review)

> "The Canon Rebel and Nikon D thousand series will fit perfectly with their light compact bodies and lenses in this handsome well built camera bag. The upper compartment for stuff is a great idea"
>
> (Second Nearest Neighbor Positive Review)

Again, top trigger words in our model appear in the retrieved reviews, demonstrating the validity of the proposed ML Modeling procedure. Additionally, as already mentioned marketing insights provided by customers could be used by suppliers and Amazon to focus on relevant improvements in case of the former, and select displayed sample reviews in the latter.

Moving on to the negative class, the closest reviews retrieved by the K-NN Algorithm are:

> "There are several unresolved technical issues amongst the several extra features of this HVR1950 and the instructions are not comprehensive. I will be returning my two models, since I spent hours with screen captures and video captures of the repeated malfunctions and emailed them to Hauppauge but they never responded"
>
> (First Nearest Neighbor Negative Review)

> "I have a brand new computer and never had any problems until now installing software or hardware It just did not work on my computer I guess windows 7 does not recognize it. Tried to get support from the company but email support is all that is available and they tell you it may be up to 48 hours before a retuning the product is possible"
>
> (Second Nearest Neighbor Negative Review)

# 8 Conclusion

On balance, it seems that satisfied Amazon clients are more likely to express their opinion than dissatisfied ones, given the uncontested dominance of positive comments. Such class distribution imbalance was effectively tackled by a hybrid random resampling algorithm, which allowed to properly predict customers' opinions and successfully identify top trigger words for extreme sentiments. Linguistic elements such as "*perfect*", "*great*", "*return*" or "*waste*" contribute to drive users' subjective view on electronic items. By performing review retrieval tasks, important insights about product flaws and appreciated features were found, which ultimately benefits both, suppliers and distributors by procuring enriching feedback to act upon.

# 9 Mathematical Proofs

## 9.1 Spearman Rank-Order Correlation Coefficient

Note that we can rewrite ***Equation 1*** as though it were a regular Pearson linear correlation coefficient by replacing vectors $(R_i,\, S_i) \leftrightarrow (X_i,\, Y_i)$, $\forall\ i$

$$\hat{\rho}_s = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Focusing on the denominator and leveraging there are no ties and the **indices** of the ranks satisfy $R_i = S_i$, $\forall\ i$ since we are dealing with paired random variables:

$$\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2} = \sum_{i=1}^{n}(x_i - \bar{x})^2$$

Applying basic notable products properties:

$$\sum_{i=1}^{n}(x_i - \bar{x})^2 = \sum_{i=1}^{n}(x_i^2 + \bar{x}^2 - 2x_i\bar{x}) = \sum_{i=1}^{n}x_i^2 + \sum_{i=1}^{n}\bar{x}^2 - 2\sum_{i=1}^{n}x_i\bar{x}$$

Let us remind basic basic properties of the measure of central tendency:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n}x_i \qquad\qquad \sum_{i=1}^{n}x_i = n\bar{x}$$

$$\sum_{i=1}^{n} x_i^2 + \sum_{i=1}^{n} \bar{x}^2 - 2\sum_{i=1}^{n} x_i\bar{x} = \sum_{i=1}^{n} x_i^2 + n\bar{x}^2 - 2n\bar{x}^2$$
$$= \sum_{i=1}^{n} x_i^2 + n\bar{x}^2 - 2n\bar{x}^2 = \sum_{i=1}^{n} x_i^2 - n\bar{x}^2$$

Note that $x_i \in \{1, 2, 3, ..., n\}$ and so $x \in \mathbb{N}$. Consequently:

$$\sum_{i=1}^{n} x_i = \frac{n(n+1)}{2} \qquad \sum_{i=1}^{n} x_i^2 = \frac{n(n+1)(2n+1)}{6} \qquad \sum_{i=1}^{n} x_i^2 = \underbrace{\sum_{i=1}^{n} \frac{x_i^2 + y_i^2}{2}}_{\text{Since pairs match!}}$$

$$\sum_{i=1}^{n} x_i^2 + n\bar{x}^2 - 2n\bar{x}^2 = \sum_{i=1}^{n} x_i^2 - n\bar{x}^2 = \frac{n(n+1)(2n+1)}{6} - n\left(\frac{n+1}{2}\right)^2$$
$$= n(n+1)\left(\frac{2n+1}{6} - \frac{n+1}{4}\right) = n(n+1)\left(\frac{n-1}{12}\right)$$

Moving on to the numerator:

$$\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) = \sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i\bar{y} - \sum_{i=1}^{n} y_i\bar{x} + \sum_{i=1}^{n} \bar{x}\bar{y}$$
$$= \sum_{i=1}^{n} x_i y_i - \bar{y}\sum_{i=1}^{n} x_i - \bar{x}\sum_{i=1}^{n} y_i + n\bar{x}\bar{y}$$

$$\sum_{i=1}^{n} x_i y_i - \bar{y}\sum_{i=1}^{n} x_i - \bar{x}\sum_{i=1}^{n} y_i + n\bar{x}\bar{y} = \sum_{i=1}^{n} x_i y_i - \underbrace{\bar{y}n\bar{x} - \bar{x}n\bar{y}}_{2n\bar{x}\bar{y}} + n\bar{x}\bar{y}$$
$$= \sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y}$$

$$\sum_{i=1}^{n} x_i y_i - n\bar{x}\bar{y} = \sum_{i=1}^{n} x_i y_i - n\left(\frac{n+1}{2}\right)^2 = \sum_{i=1}^{n} x_i y_i - n\left(\frac{n+1}{12}\right) 3(n+1)$$
$$= \frac{n(n+1)}{12}(-3(n+1)) + \sum_{i=1}^{n} x_i y_i$$
$$= \frac{n(n+1)}{12}\left((n-1) - (4n+2)\right) + \sum_{i=1}^{n} x_i y_i$$
$$= \frac{n(n+1)(n-1)}{12} - \underbrace{\frac{n(n+1)(2n+1)}{6}}_{\sum_{i=1}^{n} x_i^2} + \sum_{i=1}^{n} x_i y_i$$
$$= \frac{n(n+1)(n-1)}{12} - \underbrace{\sum_{i=1}^{n} \frac{x_i^2 + y_i^2}{2} + \sum_{i=1}^{n} x_i y_i}_{-\frac{a^2+b^2}{2} - ab = -\frac{a^2+b^2-2ab}{2}}$$
$$= \frac{n(n+1)(n-1)}{12} - \sum_{i=1}^{n} \frac{x_i^2 - 2x_i y_i + y_i^2}{2}$$
$$= \frac{n(n+1)(n-1)}{12} - \sum_{i=1}^{n} \frac{(x_i - y_i)^2}{2} = \frac{n(n+1)(n-1)}{12} - \frac{S_p}{2}$$

Let $N$ denote the numerator and $D$ the denominator:

$$N = n(n+1)\left(\frac{n-1}{12}\right) \qquad D = \frac{n(n+1)(n-1)}{12} - \frac{S_p}{2}$$

$$\text{Where } n(n+1)\left(\frac{n-1}{12}\right) = \frac{n(n^2-1)}{12} \quad \text{Since} \quad (a+b)(a-b) = a^2 - b^2$$

$$\hat{\rho}_s = \frac{N}{D} = \frac{\frac{n(n^2-1)}{12} - \frac{S_p}{2}}{\frac{n(n^2-1)}{12}} = 1 - \frac{6S_p}{n(n^2-1)}$$

Finally we get ***the desired expression***:

$$\hat{\rho}_s = 1 - \frac{6S_p}{n^3 - n}$$

## 9.2 Kendall Rank-Order Correlation Coefficient

Let $n_c$ - $n_d$ denote the symmetric difference distance between two sets of ordered pairs, we normalize such symmetric difference with the number of maximum pairs which can differ between said sets.

Note that the total number of bivariate vectors we can form given $n$ numbers is given by:

$$\binom{n}{2} = \frac{n!}{2!(n-2)!} = \frac{n(n-1)(n-2)(n-3)\dots(n-k+1)}{2*1*(n-2)(n-3)\dots(n-k+1)} = \frac{n(n-1)}{2}$$

Thus, we attain the expression presented in ***Equation 3***:

$$\hat{\tau} = \frac{n_c - n_d}{\binom{n}{2}} = \frac{n_c - n_d}{\frac{n}{2}(n-1)}$$

## 9.3 Maximum Likelihood for the ML Algorithm

Let us consider the log-likelihood optimization problem of Elastic Net Regularization for a binary logistic classifier (scalable to multinomial logistic regression with One vs Rest):

$$\hat{w} = \arg\max_{w} \left\{ \ln \prod_{i=1}^{n} \Pr(Y_i = y | X_i, w) - \lambda_1 \left|\left|w\right|\right|_1 - \lambda_2 \left|\left|w\right|\right|_2^2 \right\}$$

$$\left|\left|w\right|\right|_1 \coloneqq \sum_{j=1}^{d} |w_j| \qquad\qquad \left|\left|w\right|\right|_2^2 \coloneqq \sum_{j=1}^{d} w_j^2$$

Let random event A denote $\{y_i = 1\}$ and considering $A^c = \{e \in \Omega : e \notin A\}$:

$$A = \Pr(Y_i = 1 | X_i, w) = \frac{1}{1 + e^{-w^\intercal h(x_i)}}$$

$$A^c = \Pr(Y_i = -1 | X_i, w) = 1 - \Pr(Y_i = 1 | X_i, w) = \frac{e^{-w^\intercal h(x_i)}}{1 + e^{-w^\intercal h(x_i)}}$$

Focusing on the first term of the log-likelihood function notice that applying logarithm properties: ln(ab) = ln(a) + ln(b)

$$\begin{aligned} \mathcal{L}(w) &= \ln \prod_{i=1}^{n} \Pr(Y_i = y | X_i, w) \\ &= \sum_{i=1}^{n} \ln \Pr(Y_i = y | X_i, w) \end{aligned}$$

Introducing an indicator function:

$$\mathcal{L}(w) = \sum_{i=1}^{n} \ln \Pr(Y_i = y | X_i, w)$$

$$\Updownarrow$$

$$\sum_{i=1}^{n} \left[ (\mathbb{1}[Y_i = 1]) \ln \Pr(Y = 1 | X_i, w) + (\mathbb{1}[Y_i = -1]) \ln \Pr(Y = -1 | X_i, w) \right]$$

Bearing in mind the probability facts shown earlier:

$$\sum_{i=1}^{n}\left[(\mathbb{1}[Y_i=1])\ln\left[\frac{1}{1+e^{-w^\intercal h(x_i)}}\right]+(1-\mathbb{1}[Y_i=1])\ln\left[\frac{e^{-w^\intercal h(x_i)}}{1+e^{-w^\intercal h(x_i)}}\right]\right]$$

Note that:

$$\text{i) } \ln\left[e^{-w^\intercal h(x_i)}\right]=-w^\intercal h(x_i)$$

$$\text{ii) } \ln\left[\frac{e^{-w^\intercal h(x_i)}}{1+e^{-w^\intercal h(x_i)}}\right]=\ln\left[e^{-w^\intercal h(x_i)}\right]-\ln\left[1+e^{-w^\intercal h(x_i)}\right]$$

$$\text{iii) } \ln\left[\frac{1}{1+e^{-w^\intercal h(x_i)}}\right]=\ln(1)-\ln\left[1+e^{-w^\intercal h(x_i)}\right]=-\ln\left[1+e^{-w^\intercal h(x_i)}\right]$$

Rearranging:

$$\sum_{i=1}^{n}\left[-\mathbb{1}[Y_i=1]\ln\left[1+e^{-w^\intercal h(x_i)}\right]+(1-\mathbb{1}[Y_i=1])\left(-w^\intercal h(x_i)-\ln\left[1+e^{-w^\intercal h(x_i)}\right]\right)\right]$$

$$\hookrightarrow \mathcal{L}(w)=\sum_{i=1}^{n}\left[-(1-\mathbb{1}[Y_i=1])w^\intercal h(x_i)-\ln\left[1+e^{-w^\intercal h(x_i)}\right]\right]$$

Proceeding with the derivative, let us consider the individual case of a single data point for the sake of simplicity:

$$\frac{\partial\mathcal{L}(w)}{\partial w_j}=-(1-\mathbb{1}[Y_i=1])\frac{\partial w^\intercal h(x_i)}{\partial w_j}-\frac{\partial\ln\left[1+e^{-w^\intercal h(x_i)}\right]}{\partial w_j}$$

$$\frac{\partial w^\intercal h(x_i)}{\partial w_j}=h_j(x_i) \qquad \frac{\partial\ln\left[1+e^{-w^\intercal h(x_i)}\right]}{\partial w_j}=-h_j(x_i)\underbrace{\frac{e^{-w^\intercal h(x_i)}}{1+e^{-w^\intercal h(x_i)}}}_{\Pr(Y=-1|X_i,w)}$$

$$\frac{\partial \mathcal{L}(w)}{\partial w_j} = -(1 - \mathbb{1}[Y_i = 1])h_j(x_i) + h_j(x_i)\Pr(Y = -1|X_i, w)$$

$$= h_j(x_i)\left(\mathbb{1}[Y_i = 1] - \Pr(Y = 1|X_i, w)\right)$$

Lastly, considering that generalizing to all data points requires summing them up:

$$\nabla \mathcal{L}(w) = \sum_{i=1}^{n} h_j(x_i)\left[\mathbb{1}[Y_i = 1] - \Pr(Y = 1|X_i, w)\right]$$

The case of Tikhonov Regularization is fairly simple:

$$\lambda \frac{\partial \, ||w||_2^2}{\partial w_j} = \lambda \frac{\partial}{\partial w_j} \sum_{j=1}^{d} w_j^2 = 2\lambda_2 w_j$$

$$\Psi(w_j) \coloneqq \nabla \mathcal{L}(w) - 2\lambda_2 w_j$$

Dealing with the derivative of L1 norm is much more complex as it actually leads to non-unique solutions, entailing that a closed-form or Newtonian approach must be discarded. Consequently proximal methods such as Subgradient are essential. For instance, FISTA defines a proximal operator and then iterates (check pseudocode here), resembling a Coordinate Ascent (Descent) Algorithm. The inception of such issue is the absolute value function, which has a kink at the origin as the derivative from the left does not equal its right counterpart. Subgradients generalize gradients to non-differentiable points: $V \in \partial g(x)$ if $g(b) \geq g(a) + \nabla g(a)(b - a)$.

Thus any plane $V \in$ [-1, 1] that lower bounds $|w_j|$ is a plausible solution:

$$\hat{\nabla w_j} = \begin{cases} \Psi(w_j) + \lambda_1 & \text{if} \quad w_j < 0 \\ [\Psi(w_j) + \lambda_1, \Psi(w_j) - \lambda_1] & \text{if} \quad w_j = 0 \\ \Psi(w_j) - \lambda_1 & \text{if} \quad w_j > 0 \end{cases}$$

# 10 References


ANDHIKA, R.A (2021). *Imbalanced Classification in Python: SMOTE-ENN Method*. Retrieved from Towards Data Science: https://towardsdatascience.com/imbalanced-classification-in-python-smote-enn-method-db5db06b8d50

AUSÍN OLIVERA, M.C., & ALONSO, A.M. (2019). *Statistical Inference Techniques*. Universidad Carlos III de Madrid

BECK, A., & TEBOULLE, M. (2009). A Fast Iterative Shrinkage- Thresholding Algorithm for Linear Inverse Problems. *SIAM Journal on Imaging Sciences*. Society for Industrial and Applied Mathematics

BISHOP, C. (2006). *Pattern recognition and machine learning*. Springer

FOX, E., GUESTRIN, C. (2016). *Machine Learning Specialization*. University of Washington

GENOVESE, C., PERONE-PACIFICO, M., VERDINELLI, I., WASSERMAN, L. (2016). Nonparametric Inference For Density Modes. *Journal of the Royal Statistical Society*

GÉRON, A. (2018). *Hands-on Machine Learning With Scikit-Learn, Keras & TensorFlow*. O'Reilly

GLEN, S. (2016). Lilliefors Test for Normality Exponential Distributions. *Statistics How To*

KENDALL, M. (1938). A New Measure of Rank Correlation. *Biometrika*

LIGTHART, A., CATAL C., & TEKINERDOGAN B. (2021). Systematic reviews in sentiment analysis: a tertiary study. *Artificial Intelligence Review*

MASłOWSKA, E., MALTHOUSE, E.C., BERNRITTER, S.F. (2017). Too Good To Be True: The Role of Online Reviews' Features in Probability to Buy. *International Journal of Advertising*

MONKEYLEARN. (2021). *Sentiment Analysis: A Definitive Guide*. Retrieved from: https://monkeylearn.com/sentiment-analysis/

MUKHERJEE, A., MUKHOPADHYAY, S., PANIGRAHI, P., Goswami, S. (2019). Utilization of Oversampling for multiclass sentiment analysis on Amazon Review Dataset. *Institute of Electrical and Electronics Engineers*

NI, J., LI, J., & MCAULEY, J. (2019). Justifying Recommendations using Distantly-Labeled Reviews and Fine-Grained Aspects. *Association for Computational Linguistics*

RAZALI, N., WAH, Y.B. (2011). Power comparisons of Shapiro-Wilk, Kolmogorov-

Smirnov, Lilliefors and Anderson-Darling tests. *Journal of Statistical Modeling and Analytics*

SPEARMAN, C. (1990). The proof and measurement of association between two things. *American Journal of Psychology*

STACKEXCHANGE (2019). *Drawing a flow chart using tikz package.* Retrieved from: https://tex.stackexchange.com/questions/503128/how-to-draw-a-flow-chart

TUCKER, H. (1959). A Generalization of the Glivenko-Cantelli Theorem. *The Annals of Mathematical Statistics*

ZHAO, Y. (2012). *R and Data Mining: Examples and Case Studies.* Academic Press