Austin Bell (alb2307) and Cedric Jouan (cj2567)

Arbitrary Aspect Identification, Extraction and Ranking

**Introduction**

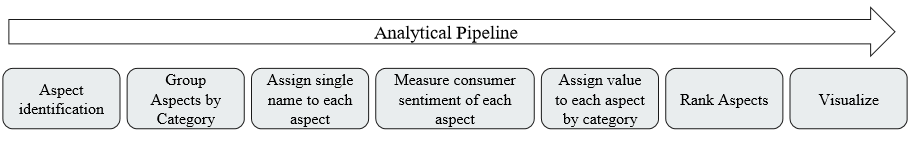
Why does one customer choose to purchase one item over another? If two items are truly substitutes (i.e., they perform the same function), there must be some defining features or aspects that lead to purchase of specific products. These are common questions asked across all industries and even across microeconomic academic circles.

Traditionally, when trying to identify the causal factors of a consumer’s purchase two methods are typically utilized:

1. Experiments with a control group and treatment group
2. Surveying customers followed by a conjoint analysis

Both of which come with strong disadvantages. Traditional experiments are incredibly costly and time consuming outside of the software industry (i.e., AB testing), whereas survey analysis introduces strong bias where a participant may answer in the form that they believe the surveyor wants them to.

Our research develops a complete methodology for understanding the relative importance of a product’s aspects within the retail sector. Our method leverages the unstructured text in product reviews to identify aspects, clustering algorithms to select the common aspect phrases, sentiment analysis to determine the user’s opinion towards the aspect, and then uses a probabilistic ranking model to evaluate each feature’s importance. We apply our model across a wide variety of unrelated product categories to demonstrate the generalizability of our model across the retail sector. Specifically, our model leverages the following pipeline:



*Figure 1:General Analytical Pipeline*

The result is that for each product, we will have identified the key product aspects and their relative importance. Since we identify aspects at the category level, we then are able to easily compare aspects and aspect importance over a wide range of products.

We must acknowledge that this is just the first step in causal analysis to understanding consumer reasons for purchases, but we must impart the two key advantages over traditional approaches to gathering such data:

1. The analysis time after model development is extremely quick, representing a significant improvement compared to experiments or surveys
2. Potential surveyee bias has effectively been removed by leveraging open-ended unstructured reviews volunteered by purchasers of the product without knowledge of this research

**Relevant Work**

Due to the multi-layered approach of our work, there exists multiple bodies of literature that overlap with our analysis including aspect extraction, sentiment analysis, and aspect ranking. We provide a basic overview of some of the work from each of these groups that have influenced our analysis.

Aspect Extraction:

The first portion of our framework requires identifying and extracting the aspects. Much of the past research have extracted aspects based on the frequency of nouns or noun chunks. For each review or product, word counts and TF-IDF weightings are calculated and the top *n* aspects are extracted. While this results in extracting all possible aspects, this methodology results in a significant amount of noise. Past research has typically accounted for this additional noise through leveraging data external to the unstructured product reviews such as leveraging a Pros and Cons input (Zha et al, 2014), through leveraging the “Product Features” information found on Amazon product pages (Najmi, 2015), or through utilizing a language model (Wu 2009).

Our goal was to design a framework that could be applied independent of data source where the only required input were the actual free text product reviews. Therefore, we opted for a more rule based system.

This requires an assumption that aspects are discussed in grammatically similar ways such that linguistic rules can be effectively applied, which research has shown to hold true (Liu, 2010). Liu’s research identifies that there are two key types of aspects: 1) explicit (or direct opinions) and 2) implicit (or indirect opinions). Mining implicit aspects is significantly harder than mining explicit aspects.

There is a significant body of work that utilizes this underlying grammatical and dependency structure to extract product. In Wu et al, (2009), their team generates dependency phrase trees and extracts noun phrases and verb phrases relevant to aspects. Additionally, multiple teams (Poria, 2014; Pekar, 2014; and Maharani, 2015) utilize a series of expert defined grammatical and linguistic rules to extract product aspects with very positive results. The effectiveness of dependency parsing is also demonstrated by its heavy use in opinion mining (Zhi et al, 2018; Caro and Grella, 2013)

Sentiment analysis

Sentiment analysis is maybe one of the most documented problem in natural language processing. The rich literature around this topic gave us a great diversity of methods to build this part of the framework. Given that our input dataset contains sentiment labels and willing to fit the model to our specific corpus, we mainly focused on supervised models.

In 2011 X Glorot et al. explored a deep learning approach to perform domain adaption for large scale sentiment analysis. Other research focus more on finding the most appropriate neural structure for text understanding, text classification and sentiment analysis (Maryem Rhanoui et al, 2019; Tao Chen et al, 2017; [Lei Zhang](https://onlinelibrary.wiley.com/action/doSearch?ContribAuthorStored=Zhang%2C+Lei) et al, 2018). More recent research on Universal Language Model fine tuning (Jeremy Howard and Sebastian Ruder, 2018) expend the learning transfer for text understanding and lead to astonishing results in text classification and sentiment analysis. In the same idea of using transfer learning for sentiment analysis, [Chi Sun](https://arxiv.org/search/cs?searchtype=author&query=Sun%2C+C) et al (2019) proposed a fine tuning of the Bert language model for Aspect Based Sentiment Analysis.

Aspect Ranking

There are typically two approaches to ranking extracted aspects and their corresponding opinions: 1) some variant of a weighted average of opinions and frequencies for each aspect and 2) a probabilistic ranking algorithm. While we employ both approaches in our final evaluation, we focus primarily on the probabilistic ranking algorithm.

For a probabilistic ranking algorithm to work, we must assume the following: a reviewer, when deciding to leave a review, first knows what aspects they plan to comment on, second they leave their review discussing these aspects, and finally, the overall score attributed to the product is a weighted aggregation of the individual opinions on each aspect. The goal of a probabilistic ranking algorithm is to then uncover the weights used in the weighted aggregation. All other data is considered known due to inclusion in the product review (overall score) or the sentiment analysis (aspect specific opinions). Thus, in past research it is common to use a variant of the expectation maximization algorithm (Zha et al., 2014; Yu et al., 2011), where the weights are considered the latent variables.

**Data**

Our project leverages millions of unstructured Amazon product reviews developed by Julian McCauley and his team. The Amazon Product reviews data consists of two primary datasets: 1) Product Reviews and 2) Metadata. Both of which are utilized.

After De-duplicating, the Product Reviews dataset includes 88 million observations. Within our analysis we utilize the following columns:

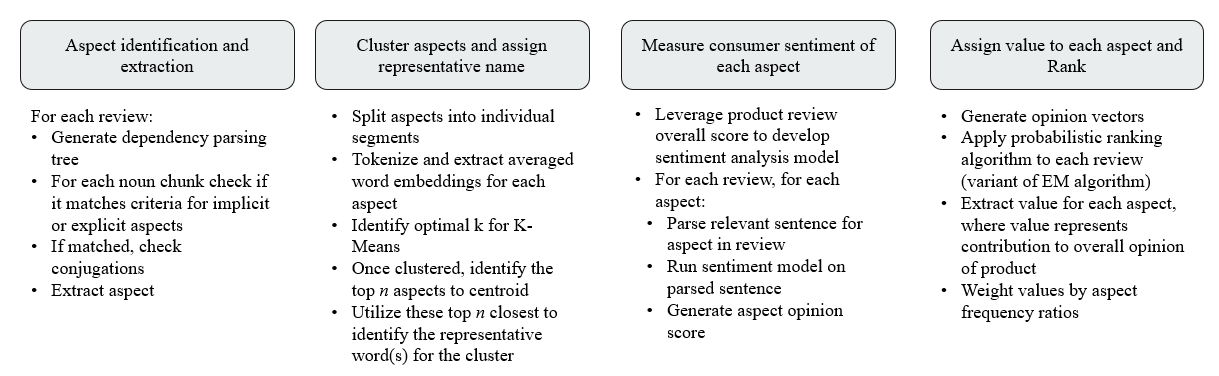
* ASIN: Product ID number
* Reviews: Unstructured review text
* Overall: number of stars (i.e., score) given to the product for the review

Additionally, the Metadata dataset consists of approximately 9 million products. From this we utilize the following columns:

* ASIN: Product ID number
* Categories: Complete list of categories associated with the product

Note, that due to time constraints, we have limited the number of reviews for each top-level category to 1 million. We incorporate 30 top-level categories in our analysis.

**Methodology**



*Figure 2:Detailed Analytical Pipeline*

Our approach consists of four discrete tasks:

1. Aspect identification and extraction
2. Cluster aspects and assign representative names
3. Measure consumer sentiment of each aspect
4. Assign value to each aspect and rank

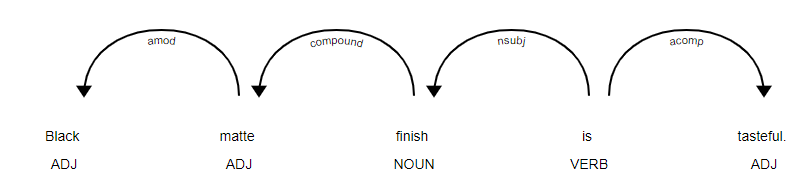
An overview for each discrete task is detailed in figure 2 above. The rest of our methodology section dives into each task in significantly more detail.

Aspect Identification and Extraction

As discussed in Liu, 2010, we assume that product aspects are discussed in linguistically and grammatically similar ways within product reviews. Liu writes that a direct opinion is represented as a quintuple vector (oj, fjk, ooijkl, hi, tl), where o*j* is an object, fjk is a feature of the object oj, ooijkl is the orientation or polarity of the opinion on feature fjk of object oj, hi is the opinion holder and tl is the time when the opinion is expressed by hi.

Our analysis leverages unstructured product reviews, therefore, we do not need to uncover all items within the quintuple vector. For example, the opinion holder (hi) is the username who wrote the product review and the time when the opinion is expressed (tl) is the timestamp associated with the review. Therefore, neither of these items need to be extracted from the unstructured text. Additionally, since product reviews are explicitly associated with a single object then oj is already known. However, we find that users still refer to the product by name or general reference. In conclusion, we find that the inclusion of object (oj) is optional in our direct opinion vector.

Following past research, we developed a rule based system to extract aspects. These linguistic rules developed according to Liu’s research on opinion mining are applied to the dependency parsing trees generated by SpaCy. In figure 3 below, we demonstrate the result of SpaCy’s dependency parsing algorithm on the sentence “Black matte finish is tasteful.”



*Figure 3: Dependency Parsing Tree on Sample Sentence*

For each review, we split the sentences using SpaCy’s sentence tokenizer, and generate a tree similar to figure 3. Once the tree is generated, we apply our set of linguistic rules[[1]](#footnote-1):

* Identify all noun chunks in the sentence
* Identify whether the noun chunk is connected to the rest of the sentence via a linking verb (“is”, “was”, “has”, “had”, etc.)
* Identify whether the connection is to another noun chunk or descriptor
* If the above rules match then we extract all noun chunks and / or descriptors
* Recursively apply rules to extract conjunctions

Cluster Aspects and Assign Representative Names

After all aspects have been extracted successfully, our next goal is to group semantically similar aspects with one another such that we can assign a single name to word(s) that reference the same aspect. For example, we may have extracted two aspects “The battery life” and “the phone battery”. These two extracted aspects clearly refer to the same aspect “battery” – our goal with this section is to first identify that these two aspects are the same then assign each of them a common name for future analysis. We accomplish this through a two-step process:

* K-Means clustering
* Assign a common name using TF-IDF weights

First, we generate the averaged word embeddings of each aspect then identify our clusters via K-Means for each top-level category. GloVe 100 dimensional embeddings and K-Means random initialization are used purely for computational speed. We limit the number of potential clusters to 100 or less for each top-level category. While, we understand that a higher number of clusters may lead to better clustering results, our work showed that the additional clusters created were small and would not be considered important aspects during our final analysis. Therefore, we felt that we were not losing much granularity by limiting to 100 clusters.

Second, we identified the most reasonable common names for each cluster. To accomplish this, we first subset each cluster to the closest 100 aspects to the corresponding cluster centroid. For this group of 100 aspects, we computed the TF-IDF scores and selected the top 5 words with the highest TF-IDF scores. These top 5 words were conjoined to form the single name representative of that particular semantic group.

Measure Consumer Sentiment of each Aspect

Ones we have the different aspects discussed within each review; we want to find what are the sentiments of the customers regarding these aspects. To do so, for every aspect of a given review, we first need to extract the part of the review that deals with this aspect. Then we predict a sentiment level on this portion of the review using a deep learning sentiment analyser. That way, for every review, we create an “opinion vector” where the elements are the sentiment scores for the different aspects of the category. This score is zero if the aspect is not discussed in the review and the sentiment level outputted by the sentiment analyser scaled between -1 and 1 if the aspect is discussed in the review.

As mentioned above, for a given review and a given aspect discussed within this review we first have to identify the portion of the review that deals with this aspect. Our first approach was simply to extract the sentence where we extracted the “aspect chunk” during the Aspect extraction phase. However, in many cases two aspects was discussed within the same sentence. To solve this problem, we build a simple algorithm that split complex sentences with several clause on multiple simple sentences with one clause. This algorithm was based on dependency trees. Assuming that a simple sentence has the structure Subject + verb + complements, we search for the “is subject to” dependences in the complex sentence and recreate the simple sentences by recombining the sub trees dominated by each verb. Such split allows us to extract one coherent sentence of each aspects discussed in a review.

The second step is to give a sentiment score for these aspect-related sentences. After exploring several solutions of the sentiment analyser (lexicon models, SVM (MLlib), ULMfit (fast.ai), different other deep learning models) we decided to train a CNN-BiLstm model with a pretrained embedding (glove 100 dimensions). This choice was motivated by two considerations: first, our input dataset gave us enough labelled data to train a very performant algorithm that perfectly fit the domain mixture of our corpus. Second, once the model trains the pre-processing and the prediction are easily and efficiently integrated in the spark pipeline.

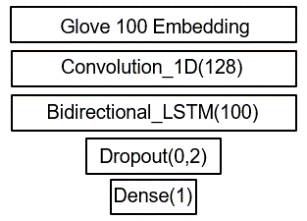
The model has been trained on approximatively 500 000 reviews equally extracted from the different categories of our dataset. The structure of the model is show on Figure 4 aside.

Figure 4: Structure of the sentiment analyser

Once the model trained and saved, we were able to apply it to the whole dataset creating one “opinion vector” per review.

Assign Value to each Aspect and Rank

We take advantage of two separate approaches to measure the value or weight of each product’s aspects. The first is a regression analysis where the weights correspond to resulting coefficients and the p-values of the coefficients provides confidence interval for the weights. Additionally, we use the probabilistic ranking algorithm outlined in Zha et al. (2014). The results of both methodologies are compared and analysed later in this report.

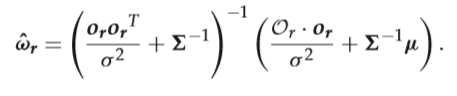
The Linear regression take as independent variables the opinion vectors that aggregate the sentiments of the different aspects discussed in each review and as, dependent variable, the overall rating of the review. Such regression assumes that the rating given by the customers follows a gaussian distribution around the weighted sum of the sentiments discussed in the review.

The weights obtained (i.e., the coefficients of the fitted regression) represent the relative importance of each aspect in the overall rating. Two points should be raised here. First, in this method the frequency of the aspect has no influence on the coefficients of the regression. In other words, using this method, the frequency does not impact the importance of an aspect. To address this issue, we can weight the regression coefficients by the frequency of the relevant aspect. Second, one regression output one weight per aspect. In order to compare the importance weights of different objects within a category, multiple linear regression should be fitted for each sub-categories.

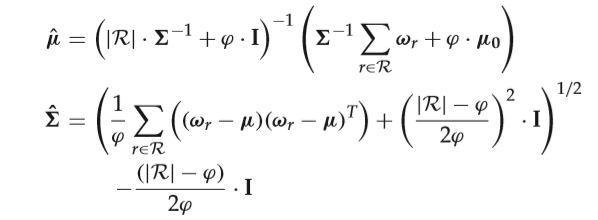
The probabilistic ranking algorithm is a variant on the expectation maximization algorithm designed specifically for the use of ranking aspects in free text product reviews. At a very basic level, the EM algorithm is used to iteratively infer latent variables through a series of updates when the probability distribution is known. It consists of two steps: 1) the expectation step where we estimate our latent variables and 2) the maximization step where we maximize the parameters of our distribution. In the case of our product aspect analysis, the latent variables that we are trying to infer are the product aspect weights. These weights, in other words, can be considered as the amount that each opinion for a specific aspect contributes to the overall opinion of each review.

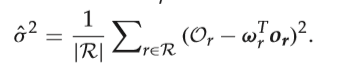
For a complete proof and derivation of the probabilistic ranking algorithm please refer to Zha et al. (2014). Below, we outline just the update equations for the expectation and maximization steps.

Expectation[[2]](#footnote-2):



Maximization[[3]](#footnote-3):



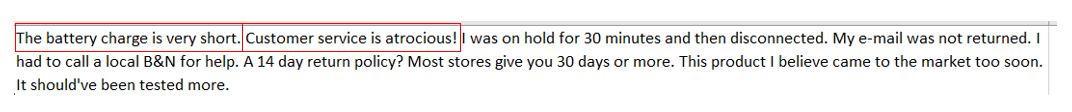


**Results**

In this section, we first outline the results of each our four discrete steps. Afterwards, we do a deep dive into the results of our analysis by displaying various results and any corresponding insights.

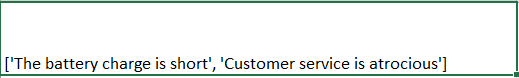
Aspect Identification and Extraction

Our team was able to successfully identify and extract aspects from all open text reviews. For details on number of aspects per category, please refer to Appendix A We first start with a raw review:



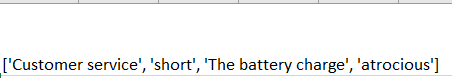
*Figure 5: Raw Review with highlighted aspects*

Our aspect identification rules successfully identify the above aspects and extracts them into their own column:



*Figure 6: Extracted Aspects*

At which point, we split these aspects into their individual aspects:



*Figure 7: Extracted Aspects split into individual aspects*

Cluster Aspects and Assign Representative Names

**Individual Aspect Clustered Aspect**

Once aspects were successfully extracted, we clusters our aspects for each top-level category. For details related number of unique clusters per category, please refer to Appendix A.

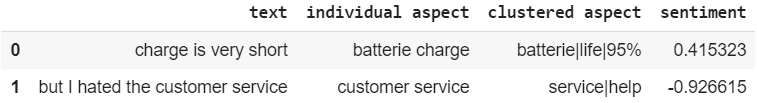
A sample of extracted aspects and their assigned names is displayed in table 1. Some aspects do not consist of 5 words, which is a result of subsetting the choice to the closest 100 aspects. Often times, we find that the closest 100 aspects are one or two aspects repeated over and over.

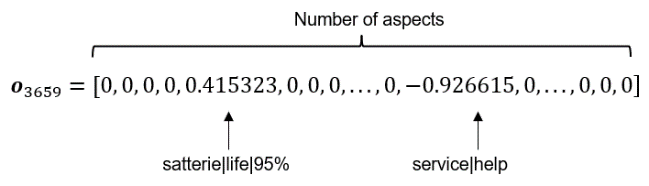
*Table 1: Sample of Names assigned to Clusters*

Measure Consumer Sentiment of each Aspect Results

Before jumping in the implementation of the sentiment analysis in the global framework we should give few results on the sentiment analyser. The target value was binary: 1 if the rating was 4 or 5 and 0 if the rating was 1 or 2. After rebalancing (both in term of category and label) we trained the model on 500k+ samples and obtained an AUC score of 0.96 out of sample. The performance was stable and good enough.

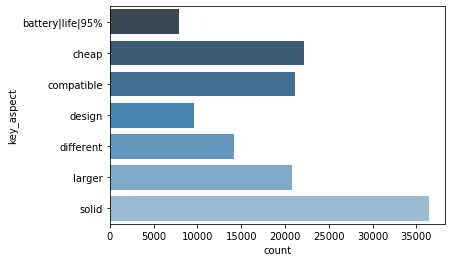
We successfully targeted the right portion of the review that deals with corresponding aspects and applied our sentiment prediction on this text.



Figure 8: Sample output of sentiment analysis

After aggregation we obtain one opinion vector for each review:

Assign Value to each Aspect and Rank Results

For clarity reasons we will focus here on the “Cell Phones and Accessories” category. We also selected a list of seven aspects. This restriction allows us to better visualize and interpret our results however the same discussion could be transposed to other aspects within this same category and to the other categories.

Let first visualize the frequencies (as seen in figure 9) and our computed importance values (as seen in figure 10 and 11) at the higher category level (for all the “Cell Pones and Accessories” reviews) for the selected aspects. Below, we display the results of our two ranking algorithms: on the left, the results of the probabilistic ranking algorithm, and on the right, the results of our linear regression.

Figure 9 Aspects Frequencies over All Phone Category

The first thing to observe is that the importance weights are very different from the frequencies. This observation shouldn’t surprise us, the important features are not necessary the most commented features and this is the main goal of this work to truly reflect the importance of each aspect.

Additionally, we note that the results are very different from one another, which is quite surprising. Further analysis is needed to uncover what is driving these differences. However, for the sake of brevity and this report we continue our analysis focusing just on the linear regression results.

A brief description of the weights in figure 11 is as follows: the aspects that contribute positively to opinions with the “Cell Phones” category are battery life (“battery|life|95%”), design, size (“larger”), and durability (“solid”), whereas the aspects that contribute most to negative opinions are price (“cheap”), and compatibility. Product uniqueness (“different”) does not appear to have any effect. The most impactful positive aspect was design and the most impactful negative aspect was price.

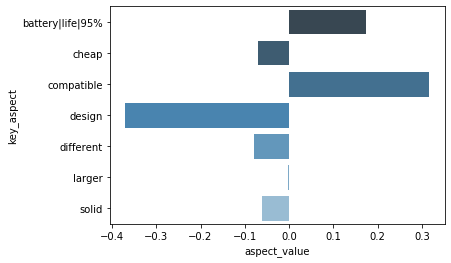
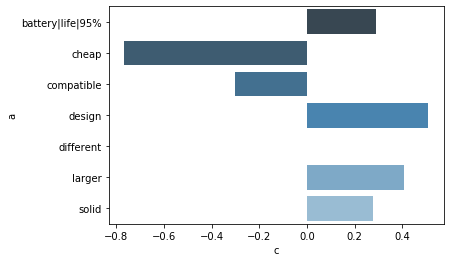
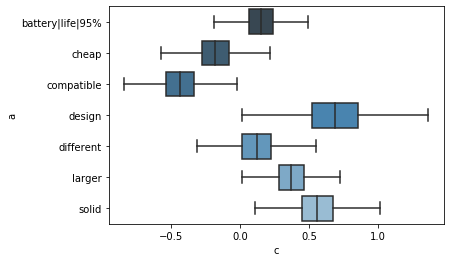


Figure 0: Aspects values for the global Phone Category using the linear regression.

Figure 9: Aspects values for the global Phone Category using the probabilistic method.

We can now dive deeper into our dataset and apply the linear regression algorithm to two different sub-categories within the Phone and Accessories category. To do this comparison we chose the “Cases” sub-category and the “Batteries” sub-category.



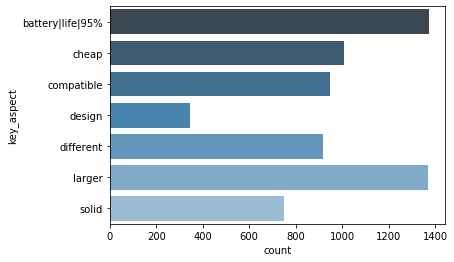
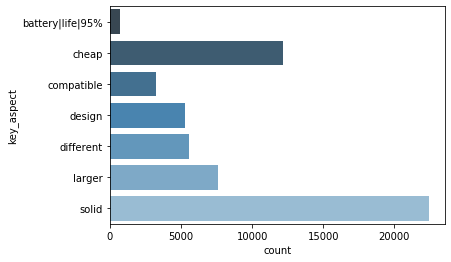
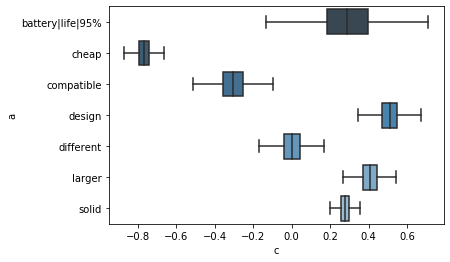


Figure 2 Linear regression weights over the Batteries sub-category.

Figure 1 Aspect’s Frequencies over the Batteries sub-category.

Figure 4 linear regression weights over the Cases sub-category.

Figure 13 Aspect’s Frequencies over the Cases sub-category.

The frequencies and importance weights plotted above lead to many valuable insights about the products within these sub-categories.

From the Batteries category, we observe that the most frequent aspect is the battery life time however, our model predicts that the battery is not a critical factor for customer’s scoring. This observation could reflect the fact that, battery’s lifetimes are aligned across the market and therefore won’t really differentiate one battery to another. However, the customer is more impacted by compatibility issues or the solidity of the Battery. The implication here is that a lack a compatibility with the desired device significantly increases the likelihood of a user to leave a negative review. Whereas, if a battery is durable and lasts for a long time, the user will write a positive review.

If we compare to battery buyers, cases buyers are much more price sensitive. Since the reviews has been written after buying the “price aspect” weight gives an a posteriori vision of the price sensitivity.

Additionally, despite having incredibly low frequency, when mentioned in the case category, the battery life leads to some positive reviews (we not that there is a rather large confidence interval here). The implication could be that while most cases do not require a battery, there are some that do – then for this small subset, the battery life appears to be very important.

This analysis could be conducted at any granularity level of the dataset and can therefore provide a great diversity of insights on customer’s behaviour.

**System Overview**

Our framework contains four phases: aspect extraction, aspect clustering, sentiment analysis and aspect ranking. Constrained by the important size of the input data (almost 150millions reviews) we used Pyspark all along our pipeline to parallelize every computation.

During the aspect extraction phase, we used spacy to parse the reviews and define the extraction strategies. Moving to the aspect clustering phase, we relied on the MLlib library to implement a K-Mean clustering algorithm and the TF-IDF feature extraction functions. During this same phase we used the glove pretrained embedding proposed by the spark-nlp library to vectorise the individual aspect and feed the clustering algorithm. We used the same library to build the features fed to the sentiment analyser. Indeed, after training the BiLSTM sentiment analyser using keras on a GPU boosted instance, we parallelized the prediction by building the features (series of GloVe100 word representation) thanks to spark-nlp. Finally, efficient numpy functions were good enough to handle the probabilistic aspect ranking and the statmodel library allows us to create importance weights with the linear regression.

**Future Work and Considerations**

We understand that while we believe that our results are significant, there is still room for improvements. We will use this section to highlight a few of the immediate improvements that we would like to make.

We would like to improve our initial aspect extraction algorithm. We understand that we simplified some of the linguistic rules surrounding our aspect extraction algorithm to increase interpretability. This likely has led to the inclusion of some noise (i.e., including words or phrases in our results that are not necessarily aspects). Additionally, we do not treat explicit and implicit aspects differently and treat them as a single entity throughout the entire analysis. Finally, when a descriptor is applied to an aspect (e.g., “short” in “The customer service is short”), often times it should not be considered as an implicit aspect. It is only when the descriptor is applied to the actual object (e.g., “large” in “the phone is large”) that we would want to consider it an aspect. A recent update to our underlying data includes a description of each product. We would wish to use this description field to identify when extracted aspects are discussing the actual object and only then would we like to extract descriptors.

Next, we would like to incorporate a variety of improvements to the clustering portion of the algorithm. First, we believe that leveraging a more sophisticated clustering algorithm (such as DBSCAN) would results in significantly improved results. DBSCAN is capable of acknowledging and ignoring outliers during the clustering process, of which we believe there are a lot. Finally, after clustering – we would like to have implemented an algorithm that identifies and remove intra-cluster outliers. Since we limit our analysis to at most 100 clusters, it is very likely that some semantic groups include aspects that are not particularly relevant to the assigned common name. Through removing outliers within each cluster, we can reduce the likelihood that we are over-weighting the importance of some aspects.

**Conclusion**

In conclusion, our framework has led to some very significant results. Across most of the Amazon product library, we are now able to understand what product aspects are most and least important to purchasers of the product. While there is definitely room for improvement, the insights that can be extracted from the framework as is are already quite valuable. For the sake of brevity, we have only showed the results within one very small portion, but similar results can be generated across every product category within Amazon.

Our belief is that this framework can be fine-tuned and extended to be useful across a wide variety of applications. The first is that this could serve as an input to improving search and preference functions within Amazon. By understanding which products are affect most by which aspects, the product buyer could search for “phones with great battery life” rather than search for just “phones”. Additionally, we believe that this is the first step to a causal analysis as to why consumers purchase certain products over others.

**Individual Contribution**

The following split occurred over all parts of the project including presentations, code, and report

* Austin (alb2307): Aspect Extraction, Clustering, and Probabilistic Ranking Algorithm
* Cedric (cj2567): Sentiment analysis, linear regression, report visualizations

Additional splits:

* Youtube Video: Austin
* Report Introduction: Austin
* References: Cedric
* System Overview: Cedric

**References**

Maryem Rhanoui, [Mounia Mikram](https://sciprofiles.com/profile/414675), [Siham Yousfi](https://sciprofiles.com/profile/author/cjdSamZhaWVjZzZVZG1JT2loTHYvc0tVOHVmOWE3ckFQRGM5T1FKOG1tbz0=), [Soukaina Barzali](https://sciprofiles.com/profile/author/amRDc0hJNGhVUmFtNEdYQWsrZTFZMHFNSUZBUHY2RVhJcHMyWTA5OFp0MD0=). A CNN-BiLSTM Model for Document-Level Sentiment Analysis (2019).

[Tao Chena](https://www.sciencedirect.com/science/article/pii/S0957417416305929" \l "!), [Ruifeng Xuab](https://www.sciencedirect.com/science/article/pii/S0957417416305929#!), [Yulan Hec](https://www.sciencedirect.com/science/article/pii/S0957417416305929#!), [Xuan Wanga](https://www.sciencedirect.com/science/article/pii/S0957417416305929#!). Improving sentiment analysis via sentence type classification using BiLSTM-CRF and CNN (2017).

[Lei Zhang](https://onlinelibrary.wiley.com/action/doSearch?ContribAuthorStored=Zhang%2C+Lei), [Shuai Wang](https://onlinelibrary.wiley.com/action/doSearch?ContribAuthorStored=Wang%2C+Shuai), [Bing Liu](https://onlinelibrary.wiley.com/action/doSearch?ContribAuthorStored=Liu%2C+Bing). Deep learning for sentiment analysis: A survey (2019).

Jeremy Howard and Sebastian Ruder. Universal Language Model Fine-tuning for Text Classification (2018).

Chi Sun, Luyao Huang, Xipeng Qiu. Utilizing BERT for Aspect-Based Sentiment Analysis via Constructing Auxiliary Sentence (2019).

Xavier Glorot, Antoine Bordes, Yoshua Bengio. Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach (2011).

Ivan Cruz-Garcia, Alexander Gelbukh, and Grigori Sidorov. Implicit aspect indicator extraction for aspectbased opinion mining (2014).

Erfan Najmi. CAPRA: a comprehensive approach to product ranking using customer reviews (2015)

Zheng-Jun Zha, Jianxing Yu, Jinhui Tang, Meng Wang, [Tat-Seng Chua](https://ieeexplore.ieee.org/author/37338664500). Product aspect ranking and its applications (2014)

Yuanbin Wu, Qi Zhang, Xuanjing Huang, Lide Wu. Phrase Dependency Parsing for Opinion Mining (2009).

Bing Liu. Sentiment Analysis and Subjectivity (2010).

Soujanya Poria, Erik Cambria, Lun-Wei Ku, Chen Gui, Alexander Gelbukh. A Rule-Based Approach to Aspect Extraction from Product Reviews (2014).

[Viktor Pekar](https://www.aclweb.org/anthology/people/v/viktor-pekar/), [Naveed Afzal](https://www.aclweb.org/anthology/people/n/naveed-afzal/), [Bernd Bohnet](https://www.aclweb.org/anthology/people/b/bernd-bohnet/). UBham: Lexical Resources and Dependency Parsing for Aspect-Based Sentiment Analysis (2014).

Jianxing Yu, Zheng-Jun Zha, Meng Wang, Tat-Seng Chua. Aspect Ranking: Identifying Important Product Aspects from Online Consumer Reviews (2011).

Warih Maharani, Dwi H. Widyantoro, Masayu Leylia Khodra. Aspect Extraction in Customer Reviews Using Syntactic Pattern (2015).

Lei Zhang and Bing Liu. Aspects Opinion Mining Based on Word Embedding and Dependency Parsing (2018).

[Luigi Di Caroa](https://www.sciencedirect.com/science/article/abs/pii/S0920548912001237" \l "!), [Matteo Grellab](https://www.sciencedirect.com/science/article/abs/pii/S0920548912001237#!). Sentiment analysis via dependency parsing (2013).

1. This is an overly simplistic review of our applied linguistic rules. For a more detailed discussion, please refer to Poria, et all (2014) [↑](#footnote-ref-1)
2. *w*r is our weight vector for a specific Review (r). or is the opinion vector for each of our aspects for a specific Review. Or is the overall opinion of a specific Review. [↑](#footnote-ref-2)
3. µ is the mean of our gaussian (each aspect mean initialized as their corresponding frequency ratio). Σ is the covariance matrix. σ2 is the overall variance. ϕ is a constant. **I** is a identity matrix. |R| is the number of total reviews. [↑](#footnote-ref-3)