




 **ddey117 / ABSA\_Project\_4** Publicforked from [ddeyflatiron/dsc-phase-4-project](#) View license **0** stars  **90** forks Star Watch ▾<> **Code** Pull requests Actions Projects Wiki Security Insights main ▾

...

This branch is [46 commits ahead](#) of ddeyflatiron:main. Contribute ▾ Fetch upstream ▾**ddey117** update ...

20 seconds ago

 **58**[View code](#) **README.md**

# Aspect/Modifier Classification Analysis

## Project Links

Below is the link for the GitHub project page.

[Github link](#)

Import research papers for developement of parser logic. Includes pdf links for spaCy research paper as well as VADER sentiment intensity analyzer. Much work has been done on aspect based sentiment analysis. Please feel free to check out some previous work in the link below.

[Research Papers](#)

For another way to navigate of my overall project, please feel free to check out a HTML version of my project overview at the link below. There is also a link in this directory to see an example of what exactly a Turk worker was looking at when they were labeling data for this project.

[html project directory](#)

## Overview

---

The target for this project is an established e-commerce business with a large amount of review data, such as Amazon.com or other online retailers. The goal of this project is to take advantage of technology and models provided by Spacy combined with a pretrained sentiment intensity classifier provided by the NLTK toolkit in order to perform more fine grained sentiment analysis at scale in an efficient manner. This project takes advantage of the parsing and part of speech tagging capabilities of Spacy's pipeline in order to extract aspect/opinion/sentiment triplets. After the aspects are identified, they can be grouped using unsupervised machine learning clustering techniques; in this case k-means clustering for model speed and simplicity. The business can use the finished product to quickly transform a large amount of informal review data (text data from reviews that may ramble for pages) and transform it into helpful graphs in order to tune into a small number of categories and help funnel resources into areas where they are most needed.

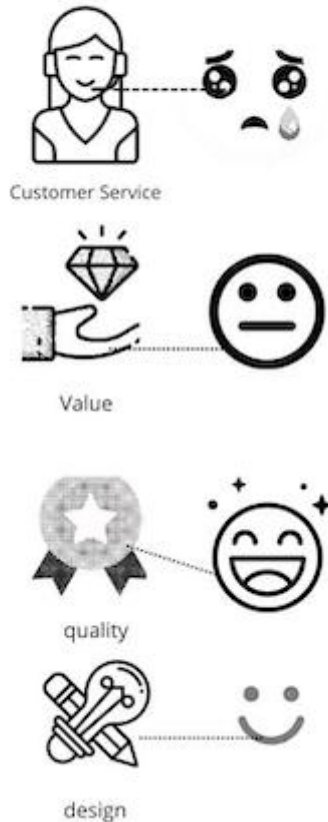
Author: Dylan Dey

The Author can be reached by email: [ddey2985@gmail.com](mailto:ddey2985@gmail.com)

## Buisness Problem

---

## Aspect Category      Sentiment



Sentiment analysis involves computationally identifying and categorizing the sentiment expressed by an author in a body of text. It has a wide range of applications in industry from stock speculation using sentiment expressed in news and blogs, to identifying customer satisfaction from their reviews and social media posts.

Today, most e-commerce website designs include a section where their customers can post reviews for products or services. Customers are free to write how they feel about fine grained aspects of a product at length. From a business perspective, very valuable information can be extracted from this section, such as customers' opinion on a product, understanding of a product, etc..

On Amazon.com the rating can be between 1 and 5 where 1 is the worst and 5 is the best. A customer can leave as lengthy of a review as they wish about a product to explain why a given rating was posted. For example, a customer may give a product a low rating because they didn't like someone they spoke to in customer service but liked everything else about the product. In typical sentiment analysis, these kinds of nuances would be missed since it could only be determined if the overall body of the review contained positive, neutral, or negative sentiment. Valuable information would be left on the table.

There is potentially a disconnect from the amazon review ratings, and the overall sentiment of the body text explaining the review, especially if you begin to break down the text into smaller aspects. Thus, Aspect Based

Sentiment Analysis (ABSA) was chosen to see if a deeper understanding of each product can be gained by breaking down each review into aspect categories to be paired with predicted sentiment, which will then be compared with the overall rating (1-5).

It is often difficult to efficiently get useful data from a large collection of text data. A lot of e-commerce websites have thousands of reviews and more incoming all of the time. Thousands of reviews with hundreds of words of mostly unhelpful information seems fairly unmanageable to most companies. While the reviews are rather informal, if they are carefully broken down there is information worth saving before generalizing again for efficiency. Aspect Based Sentiment Analysis can transform a messy collection of thousands of informal reviews into a neat and manageable collection of a few aspect categories, in this case 4 different categories using the out of box Aspect/Opinion/Sentiment Triplet Extractor. Each category will have an associated degree of sentiment related to it, and therefore graphics can easily be prepared and presented to digest more precisely what it is that customers do and do not like about a product in a quickly digestible format in real time. By breaking it down into these categories, say for example Product Design, Value, Quality, and Customer Support, the mass of text data has now been transformed into a numerical representation of sentiment towards broad categories of a product that can be directly improved upon by the company. If a product scores very high sentiment for value and design but lower scores for customer support, then a company knows it doesn't need to invest more money into improving the product and actually needs to focus on improving how its forward facing employees interact with customers.

## The Data

---

Helpful links:

[ReadMe file for Amazon Product Reviews MetaData](#)

The Amazon Customer Reviews (Product Reviews) contains over 130+ million customer reviews available to researchers in TSV files in the amazon-reviews-pds S3 bucket in AWS US East Region, as per the provided readme file. The reviews were collected from 1995 to 2015. See the provided link for associated metadata. This project focuses on the dataset given by pulling "[https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon\\_reviews\\_us\\_Electronics\\_v1\\_00.tsv.gz](https://s3.amazonaws.com/amazon-reviews-pds/tsv/amazon_reviews_us_Electronics_v1_00.tsv.gz)" from the S3 bucket.

Product\_id "[B0001FTVEK](#)" was chosen to showcase the triplet extractor as it had a large amount of verified reviews and a pair of headphones seemed like a reasonable choice for aspect based sentiment analysis.

### Clean Data

Text data trends towards exponential growth with increasing dataset size. Therefore, text cleaning and preprocessing was a major consideration of this project. Please refer to my [Text Preprocessing Toolset](#) that I created to use for this and other projects that involve text data preprocessing.


## Unlabeled Data Created Through Unsupervised Learning

This project showcases an out of box product for extracting opinion/aspect/sentiment triplets from a large amount of messy text data and converting it into a neat set of categories for analysis. To do this, however, it takes advantage of some simple clustering techniques from the sklearn cluster library. For this project, kmeans clustering was chosen for speed and simplicity. Error analysis will be discussed later in more detail in regards to how the model performs with clustering the reviews appropriately into categories and what issues it may run into when parsing internet language. Error analysis for the [SentimentIntensityClassifier](#) offered by the Natural Language ToolKit (NLTK library) will be tested against this 'newly' generated data from my unsupervised learning will be performed by comparing to a separate set of hand labeled aspect/modifier pairs by humans in an experimental setting.

### experimental setup

Using the following [Turk\\_Form\\_HTML](#) I crowdsourced some labels from humans using Amazon Mechanical Turk to compare to my model using the SentimentIntensityAnalyzer for each aspect/modifier pair extracted from the Amazon reviews. Amazon Mechanical Turk works by quickly dispersing large amounts of data to a large number of people in order to complete simple tasks for a reward. This experiment was set up to reward a penny for each aspect/modifier pair labeled for sentiment from very negative to very positive with an option for NA from a drop down menu (see html above for reference). In total, 410 workers submitted 6107 non-null aspect/opinion pairs for sentiment intensity pertaining to 1438 unique aspects. Duplicate pairs of aspect/opinion pairs were included to inspect variance of submission from human labels and machine labels for each opinion pair. No qualifications or screening was put in place before the workers were chosen, but I did review sections of the data and accept or reject what seemed reasonable. An additional 860 aspect/modifier pairs were hand labeled by a family member who only knew that the labels were extracted from amazon reviews about headphones and followed a similar template as the turk HTML.

All labels were generated using my triplet extractor on the dataset describing Product\_id "[B0001FTVEK](#)" and randomized for different aspect/modifier pairs before sending out to humans for rating for sentiment.

A large collection of amazon reviews that fall under the "electronics" category. For this project, product\_id "B0001FTVEK" was chosen as it had a large amount of verified reviews and a pair of headphones seemed like a reasonable choice for aspect based sentiment analysis.  [large\\_amz\\_tsv\\_output.jpg](#)

A simple package I created for preprocessing text data. [ddey117 Preprocessing Library](#)

## Explaining The Parser

---

![[parser.jpg]](Modifieraspectopinion extractor.jpg)

### Clustering and Polarity

A large number of amazon reviews produce a large number of aspect-modifier pairs. These pairs ultimately seemed to diverge to common topics, and therefore it would make sense to use machine learning to automatically figure out these categories for us. This leads to a better summation of insight from the total pool of customers who were kind enough to leave a review. Polarity scores are also averaged out of every cluster to give a quantifiable explanation to opinion to distinct categories of a given product.

### Word Vectors and Clustering

In order to work with any amazon review data, first the text data must be converted into something that a machine can recognize. The most famous implementation of words vectors is the word2vec project. However, spaCy vectorization was chosen for this projec as it provides fast and easy access to over a million unique word vectors, and its multi-task CNN model is trained on 'web' data and not 'newspaper' data as in other libraries like NLTK.

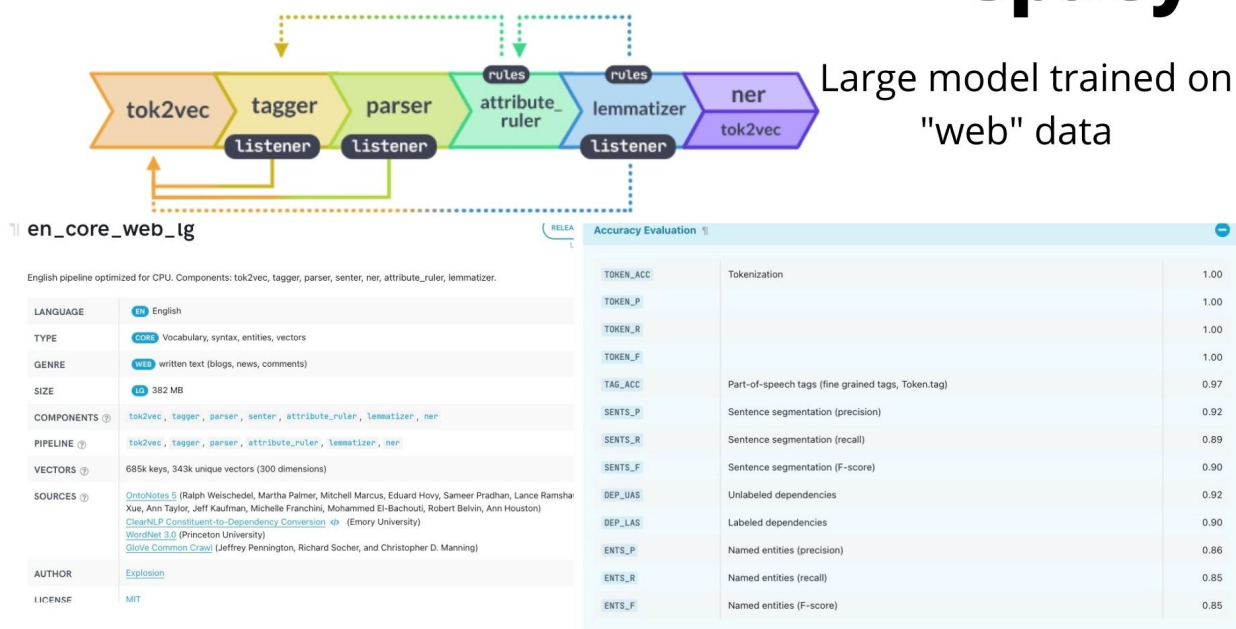
The word vectors were then grouped using K-Means clustering algorithm in Scikit-Learn. Other clustering algorithms such as DBSCAN were tested. However, K-Means gave optimal results with four clusters. The clusers were labeled with input from a user after suggesting the top most common word for each cluster.

Below Is the pipeline design for spaCy and a description for the size and sources for the model loaded to run this project and parse the amazon reviews.

[en\\_core\\_web\\_large](#)

## CPU pipeline design

spaCy



"spaCy uses the terms **head** and **child** to describe the words connected by a single arc in the dependency tree. The term **dep** is used for the arc label, which describes the type of syntactic relation that connects the child to the head. As with other attributes, the value of `.dep` is a hash value. You can get the string value with `.dep_`." [Navigating The Parse Tree](#)

**First Rule of Dependency Parser:** The Aspect (A) token is a subject noun with a child modifier (M) that has a relation of `amod` (adjectival modifier). This just means that the aspect and opinion share a simple adjective/noun relationship that can be extracted. However, there are certain caveats that need to be kept in mind when parsing the tree for this rule.

- First, it is important to check to see if there is an additional adverbial modifier that could adjust the intensity of the sentiment implied by the adjective and adverb combination in regards to the subject/aspect. This is important to keep in mind as we are taking advantage of NLTK vader sentiment intensity analyzer which can make use of additional adverbs to get a better understanding of sentiment.

- 

Another important thing to keep in mind when parsing for this rule is to be aware of the possibility of negating the adjective with 'no' as a determiner. `</span>`

## First Rule Examples



**Example1:** The comfortable headphones.

**Example2:** The most comfortable headphones.

**Example3:** No comfortable features.

- det = determiner
- A = aspect
- M = modifier
- amod = adjectival modifier

**Second Rule of Dependency Parser:** The aspect (A) is a child of something with a relation of nominal subject (nsubj.) while the modifier (M) is a child of the same something with a relationship of direct object. In this case, the adjective would be acting as the determiner of the clause. For simplicity's sake, it was determined to assume that each verb will have only one NSUBJ and DOBJ. This is a fair assumption for the application of this project, because even if there are multiple subjects, they will both be reviewing the same thing and will likely share the same opinion as it is written as a single review. For example, if an author were to say "My wife and I bought the awesome headphones", we still only want to extract the keywords 'awesome' and 'headphones.' If this sounds confusing, hopefully the example below will help clarify.

### Second Rule Example

**Example:** I bought the awesome headphones.

- nsubj = nominal subject
- dobj =headphones
- det= awesome

**Third Rule of Dependency Parser:** The modifier (M) is a child of something with a relation of an adjectival complement (acomp), while the aspect (A) is a child of that same something with a relation of nominal subject (nsubj).

- This rule needs to handle special cases in which the child is tagged as a modal verb with an auxiliary dependency. This would flag for phrases such as "the sound of the speakers could be better." For special cases like this, the parser will add a negative prefix before scoring the aspect/modifier pairs for sentiment.

### Third Rule Examples



**Example1:** Barb is happy about the sound quality.

**Example2:** This could be better.

Example2 would be extracted as A= "this" and M= "not better"

- A = aspect
- M = modifier

**Fourth Rule of Dependency Parser:** The aspect (A) is a child of something with a relationship of passive nominal subject (nsubjpass) while the modifier (M) is a child of that same something with a relationship of adverbial modifier (advmod). In other words, the modifier is an adverbial modifier to a passive verb.

- nsubjpass: A passive nominal subject is a noun phrase which is the syntactic subject of a passive clause.
- This step of the parser will also check to add a negative prefix before extracting and scoring for sentiment if necessary

#### Fourth Rule Examples

**Example1:** The headphones died quickly.

- A = aspect
- M = modifier

**Fifth Rule of Dependency Parser:** The aspect (A) is a child of the modifier with a relationship of nominal subject, while the modifier has a child with a relation of copula(cop). Here the parser is looking for the complement of a copular verb. An often used copula verb is the word "is," as in the phrase "Bill is big."

- Assumption - A verb will have only one NSUBJ and DOBJ
- cop: copula A copula is the relation between the complement of a copular verb and the copular verb. (We normally take a copula as a dependent of its complement.

#### Fifth Rule Example

**Example1:** The sound is awesome.

- A = aspect
- M = modifier

## Sixth Rule of Dependency Parser: Aspect/modifier are children of an interjection

- NTJ (interjections like bravo, great etc)

### Sixth Rule Example

**Example1:** Bravo, headphones.

- A = aspect
- M = modifier

**Seventh Rule of Dependency Parser:** This rule is similar to rule 5, but makes use of the attr (attribute) tag instead. It seems to function similarly, in which an attribute is considered a noun phrase following a copular verb

- ATTR - link between a verb like 'is/seem/appear/became' and its complement

### Seventh Rule Example

**Example1:** This is garbage.

- A = aspect
- M = modifier

**For all Parsing:** SpaCy has a large library of named entities it can recognize and tag. This logic is added for each step in the model.

Please feel free to review the following sections above under the spaCy documentation for pos\_tagging if you would like to get an understanding of how the parser was designed in spaCy. Each link should be a direct link to the appropriate topic.

[Dependency Parsing with spaCy](#)

[Navigating The Tree](#)

[Named Entity Recognition](#)

Hutto, C.J. & Gilbert, E.E. (2014). VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text. Eighth International Conference on Weblogs and Social Media (ICWSM-14). Ann Arbor, MI, June 2014.

[VADER sentiment](#)

The reasearch paper was published on release of the VADER intensity sentiment analyzer. Please feel free to read to get a better understanding of how this tool was developed before being taken advantage of in this project.

#### TABLE OF REFERENCE:

##### **AMOD**

adjectival modifier

##### **ADVMOD**

adverbial modifier

example: Genetically Modified Food, Less often

##### **NSUBJ**

"Nominal subject (nsubj) is a nominal which is the syntactic subject and the proto-agent of a clause. That is, it is in the position that passes typical grammatical test for subjecthood, and this argument is the more agentive, the do-er, or the proto-agent of the clause. This nominal may be headed by a noun, or it may be a pronoun or relative pronoun or, in ellipsis contexts, other things such as an adjective." Taken from the documentation.

example: Genetically Modified Food, Less often

##### **DOBJ**

The direct object of a VP is the noun phrase which is the (accusative) object of the verb

##### **DET**

Determiner. "The English DET covers most cases of Penn Treebank DT, PDT, WDT. However, when a Penn Treebank word with one of these tags stands alone as a noun phrase rather than modifying another word, then it becomes PRON." Taken from the documentation.

##### **ACOMP**

Adjective complement. A phrase that modifies an adjective.

##### **cop**

"A cop (copula) is the relation of a function word used to link a subject to a nonverbal predicate, including the expression of identity predication (e.g. sentences like "Kim is the President"). It is often a verb but nonverbal (pronominal) copulas are also frequent in the world's languages. Verbal copulas are tagged AUX, not VERB. Pronominal copulas are tagged PRON or DET." From the documentation.

INTJ

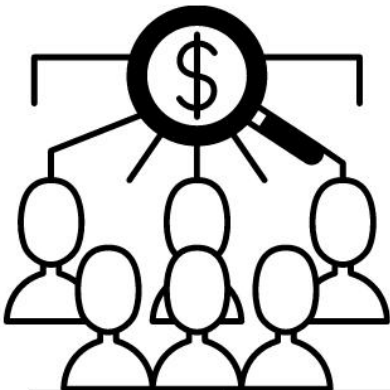
interjection. An interjection is a word that is used most often as an exclamation or part of an exclamation.

Below is an example of the extractor being run to cluster the aspects and return a bar graph of total sentiment (total summation of negative and positive from the VADER sentiment intensity analyzer) as well as a DataFrame with cluster names for further analyses. You can see that for the product\_id "B0001FTVEK", which are RS120 Wireless Headphones, there is a lot of positive sentiment for the value and sound\_quality categories as compared to the headphone\_design and hiss/tech\_diff categories. The hiss category is low enough that it should be the major focus for the company to funnel resources in response to customer demand for improving their product. If they can focus first on fixing the hiss mentioned in many amazon reviews, they can also put a few extra resources into improving some other design aspects of the headphones, such as the batteries or cradle.

 [Extractor Example](#)

RESULTS

Comparison of Turk Data to Extractor Data



VS



	Neutral	Positive	Very Positive	Negative	Very Negative
Precision	36.20%	51.64%	35.76%	35.52%	23.86%
Accuracy	23.58%	7.76%	3.68%	2.78%	0.41%

The precision values explain how many times the machine label correctly line up with the human labels for each aspect/opinion pair when guessing for that appropriate sentiment. That is, if the machine were to guess only for negative sentiment, it was in agreement with humans 35% of the time. The accuracy scores represent how many times the machine were right overall for the entire dataset when predicted a certain class. Due to the nature of the dataset, accuracy scores will be low for a lot of the classes strictly because of class imbalances. Therefore, precision is a better metric for judging the performance of my model. However, there were some major issues with the overall experimental setup that need to be discussed and analyzed.

## Reliability of Experimental Setup

To quickly visualize the variance among the different Amazon Turk workers assigned to labeling the aspect/opinion pairs, a function was utilized to create a bargraph that displays the different labels each worker voted for each aspect/opinion set that appeared in the dataset more than 10 times. It is very apparent that the Amazon turks had a lot of trouble coming to any agreement on sentiment. The task was setup to reward workers to label data as quickly as possible without much safeguard to the quality of the work being submitted other than a quick overview by myself. This cast a large shadow of doubt on the reliability of this data to be used as a way to reliably test the accuracy of my model. In comparison, you can see that the extractor chooses the same sentiment for a unique aspect/opinion pair every single time it shows up in the dataset. While the human data has been revealed to be severely flawed, it has brought up a shining example as to why machine learning may be a better substitute for labeling large amounts of tedious data in the first place.

## Partition Sum Of Squares to Measure Dispersion of Turk Data

---

### Partition Sum of Squares

For simplicity, sum of square statistical measurements were calculated for each aspect in the human labeled data as well as the machine labeled data that had more than 10 votes. This was chosen over entropy as a calculation for dispersion as the "sum of squares" for these data is a close enough estimate for the variance in categorical data for this experiment. Every single human labeled aspect with multiple aspects had a non-zero value for residual sum of squares, while every machine labeled aspect had zero residual error. This shows the difference in variance statistically and mathematically very clearly between the two and shows the unreliability of the turk data. See the figure below for the total range in the residual sum of square values for the tested aspects in each group.

## Further Discussion and Future Work

1. Due to a lack of funding from grants and some other issues in exiremental design, I feel the most appropriate next step would be to repeat the experiment with more carefully collected labeled data. Increasing the reward per label and the requirements for submission (such as proving a proficiency in english) I believe cam substantiely improve the quality of the human labeled data and reduce the variance in this data significantly. Sourcing reliable test data is the number one priority in continuing future work for this project.
2. Integrate into AWS for scaling
3. integrate into a SQL server for data management
4. incoroprate a flash based UI for viewing results at scale

## THANK YOU

[Blog](#) | [GitHub](#) | [PreProcess Github](#)

Dylan Dey

email: [ddey2985@gmail.com](mailto:ddey2985@gmail.com)

## Repository Structure

Describe the structure of your repository and its contents, for example:

— README.md	<— The top-level README for reviewers of this project
— Exploratory_Notebook.ipynb	<— exploratory notebook
— Exploratory_Notebook.pdf	<— PDF version of exploratory notebook
— Sentiment_Modeling.ipynb	<— modeling notebook
— Sentiment_Modeling.pdf	<— modeling notebook pdf
— Project_Presentation.pdf	<— project presentation pdf
— data	<— Both sourced externally and generated from code
— images	<— Both sourced externally and generated from code

## Releases

No releases published  
[Create a new release](#)

---

### Packages

No packages published  
[Publish your first package](#)

---

### Languages

● Jupyter Notebook 94.6%    ● HTML 5.4%