Final Report - Capstone Project

Sriram Kannan

Nissan North America

Key vehicle feature demand based on online comments

Executive Summary

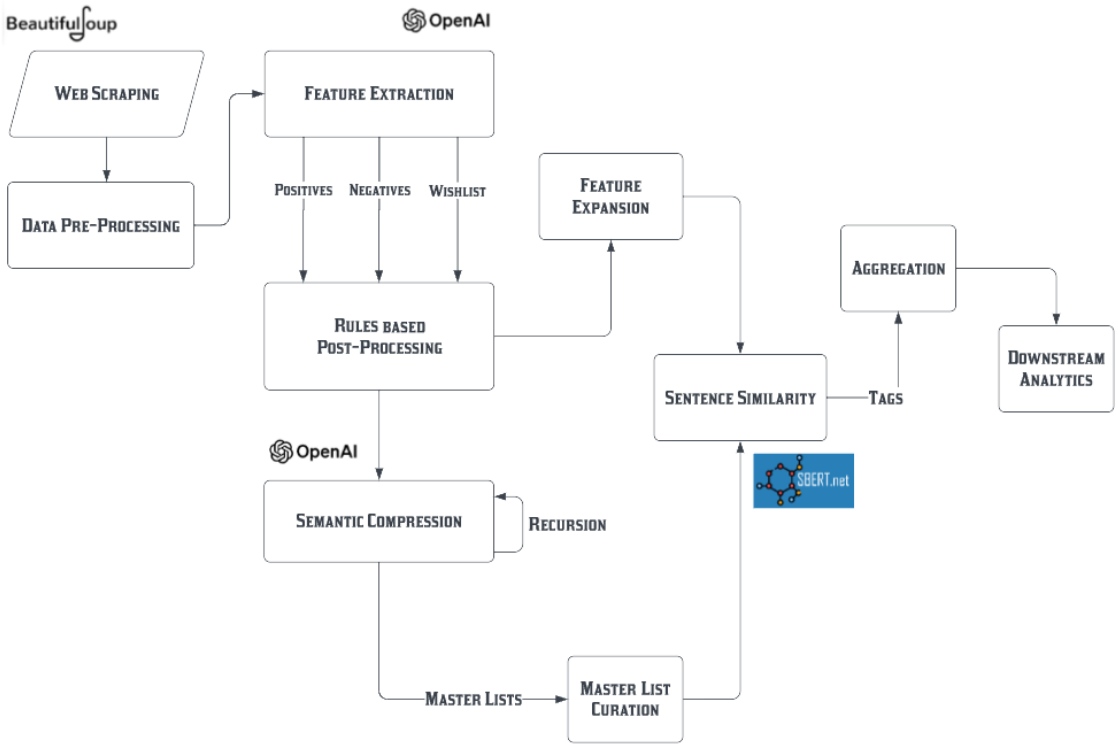
In this project, Transformer Models are utilized to uncover and organically understand automotive pain points faced by customers in the context of Nissan North America's operations and uncovering the features that customers truly care about. This is done by analyzing customer commentary from a variety of online sources which include Youtube, Kelley Blue Book, Edmunds and Cars. The primary information extraction method used was utilizing GPT-3.5 Turbo's chat endpoint (ChatGPT's API) through effectively engineered multi-layered prompts to distill each comment in a consistent manner into categories of "Positives", "Negatives" and "Wishlist". Rules based post-processing of data was then performed on the output. This was followed by grouping together words (representing vehicle features optionally with additional context or a combination of related features) with similar semantics into a list of "master features" through prompt engineering in a recursive fashion over multiple stages which was then manually curated for each of the above categories. Sentence similarity was then performed between each distilled comment and their respective master features to accurately tag them, thereby paving the way for downstream analytics. This allows for granular analysis across market cross-sections such as those across make, segments (Cars, SUVs etc) and models.

Introduction

As automotive experts, it is easy to make assumptions around consumer demand for features based on "cool" ideas that come up or are seen as trendy in the market at large. However, understanding the types of vehicle features that the consumer market at large in different segments care about is a much more involved process which could greatly increase the competitive edge of any vehicle manufacturer, especially considering the fact that vehicle designs are finalized many years in advance and therefore incorrect decisions can have cascading ramifications. Nissan currently uses Survey data as the primarily modality for this information but survey data tends to be very biased towards the negative. Positive reviewers are unlikely to take the time to fill out the survey and the skew is further exemplified by exaggeration in case of negatives. In this project, we utilize Transformer Models through information extraction and semantic similarity tagging to uncover and organically understand automotive pain points faced by customers in the context of Nissan North America's operations and the features that could solve it for them or are in demand in general by analyzing customer commentary from a variety of online sources.

Methodology

The end-to-end process can be subdivided into 5 key aspects - Web Scraping, Information Extraction, Semantic Compression, Sentence Similarity and Downstream Analytics. The entire workflow is depicted in the following flowchart.



Web Scraping

The data utilized for information extraction entirely consists of publicly available comments from automobile review websites and social media. Web Scraping was performed on relevant websites which include Youtube, KelleyBlueBook, Edmunds, Cars and Reddit using Python Web Scraping libraries such as *requests* and *beautifulsoup*. The *YoutubeV3 API* from Google Developers was used in the case of Youtube and the praw API was used in the case of Reddit. The web scraping pipeline was robust such that data cleaning of scraped data was built into it.

In total, we have 120 models of vehicles represented across 11 segments and 23 manufacturers.

Information Extraction

Initially, purely open source tools were considered for the information extraction aspect of this project which include:

1) SpaCy - A robust Natural Language Processing Framework primarily to leverage its Named Entity Recognition (NER) capabilities both through its pre-trained models and custom pipeline capabilities.
2) HuggingFace Transformers Library - Specifically, the BERT (and its variants) Encoder Transformer Model for a Sentiment Analysis pass of the comments as well as its NER capabilities.
3) BERTopic - A powerful topic modeling library which utilizes the latest transformer models providing a lot of additional support for features such as robust visualizations, topic reduction, topic tagging etc.

However, in early March due to the public release of the ChatGPT API, some of the processes could be accomplished with a much higher level of precision using effective Prompt Engineering.

The Information Extraction pipeline is split into 3 parts not necessarily in sequence as due to the nature of tools used some parts woven in and out.

1) Data Cleaning of erroneous records not including Stopwords, End of Sentence Tokens etc as these are handled well and even recommended for good performance with Transformer Models. An example of the cleaning performed was heuristically removing troll comments and comments about the reviewer than the vehicle comments on Youtube by simply omitting comments with lower than 40 words. Due to the use of ChatGPT and Effective Prompt Engineering, it leads to very consistent outputs over different user comments. This is then post-processed through performing a series of rules-based tasks to clean the outputs of the Generated text in order to aid further downstream tasks and analyses.

2) Effective Prompt Engineering which is the backbone of the first pass of information extraction from comments which is geared towards extracting relevant vehicle features categorized broadly as positives, negatives and a wishlist section in a consistent manner with predictable outputs no matter the type and content of the comments involved. It essentially acts as focused summarization which is supercharged to pick out relevant keywords associated with the comment in question (for eg. good fuel efficiency and solid suspension).

Prompt Used for Information Extraction:

{role: system, content: You are a helpful assistant that helps answer questions and perform the specific task being asked.},

{role: user, content: We are attempting to understand what vehicle features customers truly care about and what the pain points are based on customer comments from the perspective of Nissan. Note that the comment is related to automobile reviews hence pay more attention to certain words like Blind Spot Detection, Braking, Steering, Acceleration and other Car related features if they exist in the comment. Also make sure to focus only on the present vehicle being talked about and not on thoughts on similar vehicles owned at a prior time. With that in mind, what are the positives and the negative takeaways the customer had based on the following comment? Did the customer have any specific wishes? If so, list those out as part of a separate wishlist. Note that if the customer has a wish or a requested feature or doesn't list any relevant vehicle features, do not consider that as a positive or a negative. Formulate the answer with Positives, Negatives and Wishlist categories with short keywords per bullet. Do not number the bullet points. Summarize each bullet into specific keywords relating to vehicle features and make it a priority keep it as short as possible, around 3 - 4 words per bullet. Do not add anything else to the response. Comment: [usercomment] }

Note: The prompt shown above was generated through exhaustive testing across many comments through multiple iterations. The specific instructions given above tend to penalize/eliminate certain behaviors exhibited by the model (with chosen hyperparameters) across different comment classes. There were many other versions of the prompt with more specific instructions but led to deteriorating results and thus the above version was settled on for now.

Semantic Compression

Semantic compression is the process by which each category of comments (Positive, Negative and Wishlist) were compressed into a much smaller list of words. This was done through Prompt Engineering using the ChatGPT API in a recursive process across multiple levels where the output of the first level was used as the input of the second level and so on. This is done so as to stay within the 4096 token context length of ChatGPT. This outputs a master list for each of the Positive, Negative and Wishlist categories which can then be manually curated to be used for the sentence similarity portion of the pipeline.

Prompt Used for Semantic Compression:

{role: system, content: You are a helpful assistant that performs the specific task being asked.},
{role: user, content: Given a list of words in the following list which contain many words with similar semantic meaning, condense those words into a smaller list of words which represent the same meaning. The words are related to vehicle features. Make each word or feature into a bullet point. Focus purely on vehicle features like Fuel Efficiency, Spacious Interiors etc. Try to make each feature as distinct as possible from the others. If similar, try to group them together into a single feature. Avoid long bullet points. Do not number the bullets. Do not add anything else to the response like a Note or a warning. List: [Processed Categories]}

Sentence Similarity

The features from the information extraction pipeline are expanded by simply splitting each positive or negative or wishlist record into unique features to increase the accuracy of sentence similarity. Sentence similarity was performed using the sentence-transformers library using the all-MiniLM-L6-v2 model. Each of the split features in the respective list (+ve, -ve and wishlist) was compared to every feature in the respective master list to accurately tag the features. Aggregation was then performed while maintaining the metadata in order to support downstream analytics.

Analytics

Analytics can be performed through slicing and dicing the data across the following hierarchy:
Make (Nissan vs Competitors for instance), Segments (such as Cars, SUVs etc) and Models to infer the most important/critical features from a customer's viewpoint as well as pain points which vary depending on the granularity of the data considered.

Results

There are many questions which can be answered through this project based on which specific segments or makes we wish to look at. A sample analysis of Nissan compared with Toyota with the combined segments of Compact, Lower Mid Size and Performance which represent their entire suite of Sedans are presented. [Nissan is on the left and Toyota is on the right]

Positives:

Features	Frequency
Great Dealership Experience	558
Great build quality	305
Powerful Engines	280
Attractive Styling	247
Outstanding Road Performance	228
Good Value for Price	212
Smooth driving experience	211
Smooth Ride Quality	192
Spacious Interiors	188
Fuel Efficiency (MPG rating)	177
Sporty Design	173
Excellent Safety Features	161
Excellent Manual Gearbox	150
Reliable and Dependable	140
Comfortable Leather Seats	135
Stylish Exterior	122

Features	Frequency
Fuel Efficiency (MPG rating)	428
Great Dealership Experience	394
Great build quality	263
Reliable and Dependable	249
Smooth driving experience	243
Excellent Safety Features	240
Smooth Ride Quality	210
Attractive Styling	205
Powerful Engines	203
Outstanding Road Performance	195
Good Value for Price	178
Spacious Interiors	171
Comfortable Leather Seats	161
Sporty Design	133
Suitable for Daily Commutes and Long Drives	114
Stylish Exterior	114

Negatives:

Features	Frequency
Affordable Price	348
Easy-to-Use Vehicle Features	329
Smooth Transmission	224
Lack of technology features	179
Engine Power	160
Dated Interior Design	149
Brand quality	128
Ride Quality	110
Comfortable Driving Experience	108
Braking	108
Low visibility out the back	104
Reliability	99
Updated technology	93
Tires	88
Excellent rear visibility	87
Delayed performance - Sluggish Engine or Stalling	86

Features	Frequency
Easy-to-Use Vehicle Features	266
Affordable Price	231
Lack of technology features	166
Engine Power	164
Low visibility out the back	127
Fuel Efficiency	110
Engine Noise	101
Acceleration	81
Brand quality	79
Noisy	79
Customer service	78
Uncomfortable Driver's Seat	78
Dated Interior Design	77
Delayed performance - Sluggish Engine or Stalling	74
Comfortable Driving Experience	72
Spacious Interiors	66

Wishlist:

Features	Frequency
Turbo engine	130
Improved safety features	112
Improved ride quality	111
Powerful engine	107
Better interior materials	64
Optional manual transmission	64
Improved Transmission	57
Improved reliability	54
Lower price point	45
Upgraded model options	38
Advanced braking	36
Updated Infotainment features	29
Stronger front lights	28
More responsive accelerator	27
Hybrid options	27
Improved rear seat space	22

Features	Frequency
Turbo engine	96
Powerful engine	84
Improved safety features	75
Hybrid options	61
Improved ride quality	51
Better interior materials	49
Optional manual transmission	44
Improved rear seat space	33
Fuel efficiency	32
Lower price point	32
More responsive accelerator	25
Improved Transmission	25
Remote start	24
More soundproofing	23
Improved reliability	23
Upgraded model options	22

Challenges

There were a variety of challenges encountered during the projects on multiple fronts. In terms of web scraping, some sites had API support such as Reddit and Youtube while others had to be done through Python Libraries where post-processing of the data was required to a large degree to get it into a usable format. Avoiding getting blocked from certain websites especially during the testing phase (where the initial code is suboptimal and certain errors could cause a massive influx of requests) was also an issue. In terms of Information Extraction, precise Prompt Engineering to generate consistent outputs across the variety of comments (some information rich, others not so much and some being outright useless such as troll comments for instance) was an arduous task. While there are general guidelines provided by OpenAI and the community at large, it's still not fully understood as to how the Large Language Models react to prompting in general. There are times when you'd expect more precise instructions and examples of how certain exceptions are to be handled would produce better results but they in turn don't and in fact greatly regress. Thus it ends up being more of a creative endeavor and the stochasticity of results complicates matters even further. Semantic compression of comments into vehicle keywords and similarity matching

the keywords poses unique difficulties as well in terms of accuracy and information loss where a small loss across this highly layered approach would cause cascading effects in terms of accurately tagging each positive, negative and wishlist feature. The curation of the master list is also extremely important in the accuracy of sentence similarity which requires a domain expert's expertise.

Future Work

Refinement and expansion of the present work can be done in a variety of ways :

1) Collection of more data from the above data sources (Especially Youtube over time). In the interest of time and scope of this project, there was a balancing act in regards to the amount of data scraped from various sources to the desired results, i.e to optimize information gain. But this is always scalable further and while encountering diminishing returns, can always lead to better, more precise results.

2) Incorporation of Sparse data sources with very less information and lacking metadata such as Reddit can be considered. It did not make sense considering the constraints of this project but there's a chance that a certain segment of users are being ignored here. Given enough time, a custom NLP solution could be built to model the topic and subsequently the "(un)desired vehicle features" of different threads within a subreddit while also accurately tagging the metadata. This can further be extended to forums as a whole such as nissanzclub and other such fan forums which also suffer from the very same issue. While diminishing returns is a valid consideration, even a minor improvement in the final inference could have a substantial cascading effect for Nissan's operations as a whole.

3) Prompt Engineering itself is a new but rapidly evolving field with an abundance of new model releases and updates inducing further variance in the same. Considering these Large Language models are trained on enormous quantities of data and reinforcement learning with human feedback while also being extremely massive in scale, they tend to exhibit emergent behavior. While this allows production of incredible results even with simple prompting, truly understanding how the tool can be manipulated to one's precise needs is incredibly difficult which is compounded by the stochastic nature of large language models as a whole. This requires an immense amount of experimentation and exploration to even come close to fully unlocking the potential of these models which is why taking the time to improve upon this aspect could lead to even better results than already produced and could possibly also allow for streamlining some of the secondary tasks.

4) Utilizing more powerful pre-trained models such as GPT-4 for Information Extraction and Semantic compression would greatly improve results through reduction of information loss and each stage and reduce the dependency on recursion. GPT-4 however is about 60 times more expensive than the current GPT3.5 Turbo utilized and does not have API support just yet. But the budget constraints alone put it beyond the scope of the current project. Using larger models like all-mpnet-base-v2 would also increase accuracy in sentence similarity at the cost of 5x the inference time.

5) Curated master lists by a domain expert would also improve the accuracy of tagging. Another method would be stratified semantic compression through recursion which would first tag broad features and then go into granular detail. For example, instead of trying to pinpoint exact features related to seats such as height adjustment, seating material, seat dimensions etc, it's instead broadly classified as just "seat". Then it's further classified into the actual sub-features. This

would greatly help avoid errors and combined with all the methods proposed above would work exceptionally well.

Codebase (GitHub Repository)

The codebase consists of .ipynb Python notebooks organized logically based on specific tasks involved.
Repo Link : https://github.com/sriram-k96/nissan-capstone_project

References

1) OpenAI Documentation : https://platform.openai.com/docs/guides/chat/introduction
2) OpenAI Cookbook : https://github.com/openai/openai-cookbook/blob/main/techniques_to_improve_reliability.md
3) ChatGPT Technical References : https://writings.stephenwolfram.com/2023/02/what-is-chatgpt-doing-and-why-does-it-work/ https://towardsdatascience.com/how-chatgpt-works-the-models-behind-the-bot-1ce5fca96286
4) Natural Language Processing with Transformers : https://learning.oreilly.com/library/view/natural-language-processing/9781098136789/
5) SpaCy Documentation : https://spacy.io/api
6) Beautifulsoup : https://beautiful-soup-4.readthedocs.io/en/latest/
7) Youtube API Documentation : https://developers.google.com/youtube/v3/docs
8) Reddit API Documentation : https://praw.readthedocs.io/en/stable/
9) HuggingFace Documentation and Articles: https://huggingface.co/blog/dialog-agents https://huggingface.co/docs/transformers/model_doc/bert
10) Gensim Documentation: https://radimrehurek.com/gensim/auto_examples/index.html
11) BERTopic Documentation: https://maartengr.github.io/BERTopic/api/bertopic.html
12) Sentence-Transformers Documentation: https://www.sbert.net/