



Tesla Sentiment Analysis

Presented by:

Nicolas Caicedo Ortiz

Jose Fernando Alvarez Becerra

Mohan Upadhayay

Iker Perez Sanchez

Professor: Ali Vaisifard

Class: Big Data & Analytics

July 3, 2025

Contents

Introduction	2
Project Goal.....	3
Business Value.....	3
Methodology	3
Data Collection.....	3
Preprocessing.....	5
Cleaning steps	5
Challenges.....	8
Analysis	8
Sentiment Analysis using Vader.....	14
Visualization.....	14
Conclusions & recommendations	18

Introduction

Project Goal

The main goal of this project is to analyze public sentiment around Tesla by leveraging social media data, from Reddit. Our goal is to understand how people talk in the comments regarding Tesla and provide key insights into what aspects they can improve. Leaving apart the fact that this is a final examination, our goal is to get real world experience with data analysis and all the processes that go with it.

Business Value

Tesla operates in a competitive and volatile market. Real-time sentiment analysis helps in:

- **Identifying** risks or emerging controversies.
- **Evaluate** public reaction to events.
- **Hear** what customers are concerned about and their opinions.

Methodology

We used a data pipeline combining APIs, Python, MapReduce, SQL, and Tableau

1. Collected Reddit data using APIs
2. Cleaned and processed the data with MapReduce (Hadoop)
3. Queried processed data in SQL
4. Run sentiment analysis in Python
5. Visualized data results in Tableau

Data Collection

APIs used: Reddit API (via PRAW)

Tools: Python, Hadoop, PostgreSQL

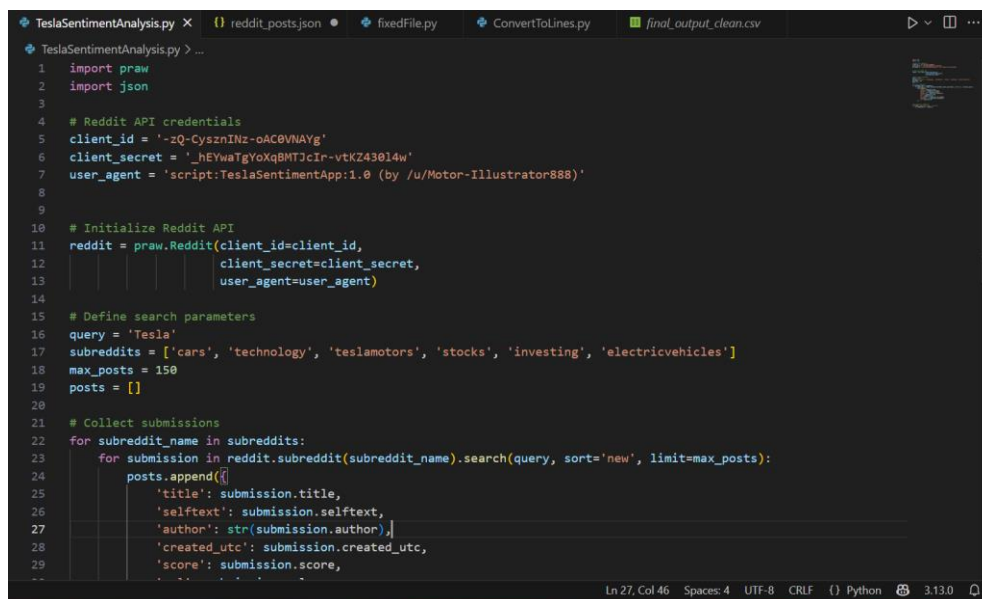
Process:

- **First**, we created an app on Reddit, which is basically a tool that Reddit uses to verify identity by providing Id and password as well as other details.
- **Second**, we run the code in VS code modifying some aspects such as subreddits and max number of posts for subreddit. Including the Client ID, Client Secret and User Agent

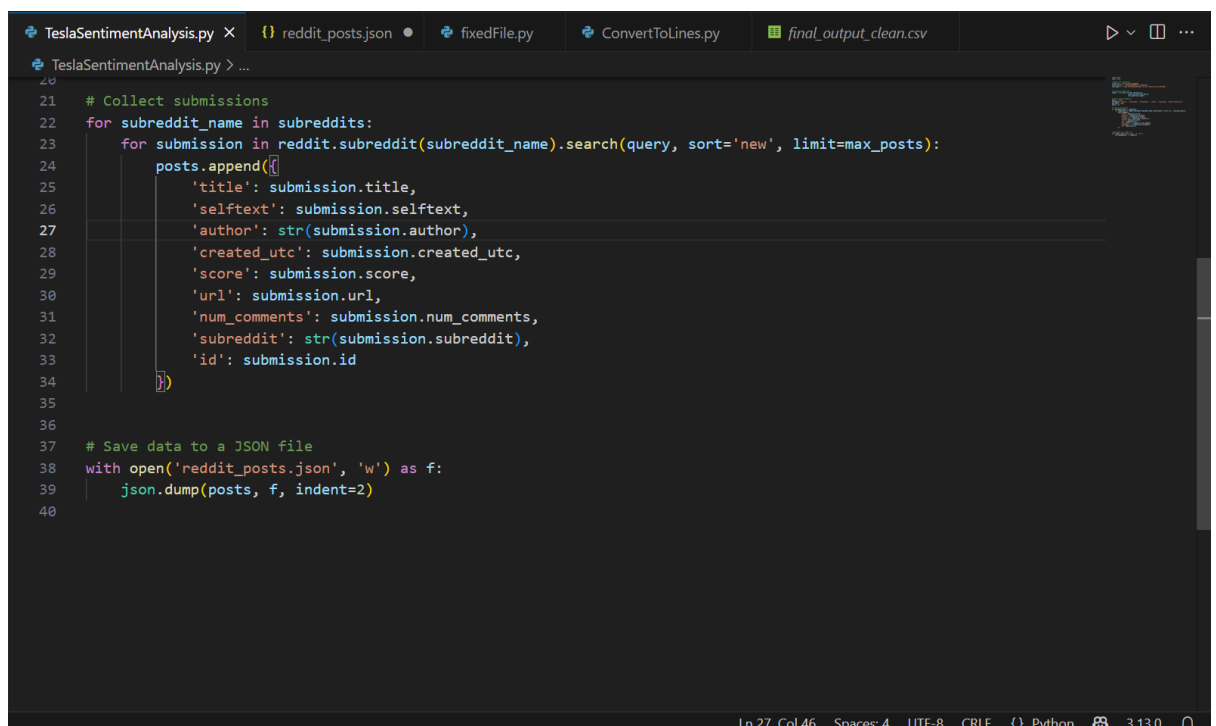
- The query that we used was “Tesla”. Our main idea was to grab posts regarding Tesla from economic topics to technology posts to gather a broader analysis.

Our Subreddits were:

1. Cars
 2. Technology
 3. Teslamotors
 4. Stocks
 5. Investing
 6. Electric Vehicles
- Once the code is executed, the data is saved to a JSON file for further cleaning.



```
TeslaSentimentAnalysis.py x reddit_posts.json fixedFile.py ConvertToLines.py final_output_clean.csv
TeslaSentimentAnalysis.py > ...
1 import praw
2 import json
3
4 # Reddit API credentials
5 client_id = '-zQ-CysznINz-oAC0VNAVg'
6 client_secret = '_hEYwaTgYoXqBMTJcIr-vtKZ49014w'
7 user_agent = 'script:TeslaSentimentApp:1.0 (by /u/Motor-Illustrator888)'
8
9
10 # Initialize Reddit API
11 reddit = praw.Reddit(client_id=client_id,
12                     client_secret=client_secret,
13                     user_agent=user_agent)
14
15 # Define search parameters
16 query = 'Tesla'
17 subreddits = ['cars', 'technology', 'teslamotors', 'stocks', 'investing', 'electricvehicles']
18 max_posts = 150
19 posts = []
20
21 # Collect submissions
22 for subreddit_name in subreddits:
23     for submission in reddit.subreddit(subreddit_name).search(query, sort='new', limit=max_posts):
24         posts.append({
25             'title': submission.title,
26             'selftext': submission.selftext,
27             'author': str(submission.author),
28             'created_utc': submission.created_utc,
29             'score': submission.score,
```



```
TeslaSentimentAnalysis.py x reddit_posts.json fixedFile.py ConvertToLines.py final_output_clean.csv
TeslaSentimentAnalysis.py > ...
20
21 # Collect submissions
22 for subreddit_name in subreddits:
23     for submission in reddit.subreddit(subreddit_name).search(query, sort='new', limit=max_posts):
24         posts.append({
25             'title': submission.title,
26             'selftext': submission.selftext,
27             'author': str(submission.author),
28             'created_utc': submission.created_utc,
29             'score': submission.score,
30             'url': submission.url,
31             'num_comments': submission.num_comments,
32             'subreddit': str(submission.subreddit),
33             'id': submission.id
34         })
35
36
37 # Save data to a JSON file
38 with open('reddit_posts.json', 'w') as f:
39     json.dump(posts, f, indent=2)
40
```

```
{} reddit_posts.json > {} 143 > id
1  [
2    {
3      "title": "Toyota RAV4 topples Tesla Model Y as world\u2019s best-selling car",
4      "selftext": "",
5      "author": "Car-face",
6      "created_utc": 1751017641.0,
7      "score": 969,
8      "url": "https://www.carexpert.com.au/car-news/toyota-rav4-topples-tesla-model-y-as-worlds-best-selling-car",
9      "num_comments": 127,
10     "subreddit": "cars",
11     "id": "1llptrd"
12   },
13   {
14     "title": "JayEMM On Cars | Follow link (ctrl + click) | siness Practices of BYD",
15     "selftext": "Video link: https://www.youtube.com/watch?v=kBbICrsk7RM\n\nThe complete title was a bit click-baity",
16     "author": "hi_im_bored13",
17     "created_utc": 1750641737.0,
18     "score": 349,
19     "url": "https://www.reddit.com/r/cars/comments/1li438m/jayemm_on_cars_the_unbelievable_business/",
20     "num_comments": 139,
21     "subreddit": "cars",
22     "id": "1li438m"
23   },
24   {
25     "title": "Where are all the colors. Analysis of every paint option available today.",
26     "selftext": "A game I often play on my commute home every day is to try and find color options of cars that I'm",
27     "author": "caterham09",
28     "created_utc": 1749849051.0,
29     "score": 379,
```

Preprocessing

Cleaning steps

1. First, we transfer the JSON file from our windows environment to the Ubuntu environment. We did this with the help of Google Drive.
2. Second, we downloaded the file in our Ubuntu environment.
3. Third, we started Hadoop by first starting the **dfs** and **yarn** daemons. Once they were running, we inserted the JSON file with this command.

hdfs dfs -mkdir -p /input

hdfs dfs -put ConvertedFile.json /input/

Once the file is already inside Hadoop we proceed to the Mapping and Reducing stages. We open the nano text-editor to insert the code for Mapper and Reducer.

Mapper Code:

```
#!/usr/bin/env python3
import sys
import json
import re

# Define stopwords
stopwords = {
    "the", "is", "at", "which", "on", "a", "an", "and", "this", "that", "of", "to", "in", "for", "it",
    "i", "you", "we", "they", "he", "she", "but", "or", "so", "be", "was", "were", "has", "have", "had"
}

# Function to clean and normalize text
def clean_text(text):
    text = text.lower()
    text = re.sub(r"http\S+", "", text) # Remove URLs
    text = re.sub(r"[^\w\s]", "", text) # Remove punctuation/emojis
    words = text.split()
    words = [word for word in words if word not in stopwords]
    return " ".join(words)

# Process each JSON line
for line in sys.stdin:
    try:
        data = json.loads(line)

        # 1. Post ID
        post_id = data.get("id", "").strip()

        post_id = data.get("id", "").strip()

        # 2. Author
        author = data.get("author", "").strip()

        # 3. Created UTC timestamp
        created_utc = data.get("created_utc", "")

        # 4. Subreddit
        subreddit = data.get("subreddit", "").strip()

        # 5. Score
        score = data.get("score", 0)

        # 6. Number of comments
        num_comments = data.get("num_comments", 0)

        # Combined + cleaned text
        title = data.get("title", "")
        selftext = data.get("selftext", "")
        combined_text = f"{title} {selftext}".strip()

        if not combined_text:
            continue # Skip empty posts

        cleaned = clean_text(combined_text)

        # Output: tab-separated
        print(f"{post_id}\t{author}\t{created_utc}\t{subreddit}\t{score}\t{num_comments}\t{cleaned}")
```

```
# Combined + cleaned text
title = data.get("title", "")
selftext = data.get("selftext", "")
combined_text = f"{title} {selftext}".strip()

if not combined_text:
    continue # Skip empty posts

cleaned = clean_text(combined_text)

# Output: tab-separated
print(f"{post_id}\t{author}\t{created_utc}\t{subreddit}\t{score}\t{num_comments}\t{cleaned}")

except Exception:
    continue
```

Reducer Code:

```
#!/usr/bin/env python3
import sys

# This reducer just forwards the mapper output line by line
for line in sys.stdin:
    print(line.strip())
```

- The next step is to run Hadoop Streaming Job with this command

```
hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-  
*.jar \
```

```
-files mapperTeslaSent.py,reduceTeslaSent.py \
```

```
-input /input/ConvertedFile.json \
```

```
-output /output/cleaned_output \
```

```
-mapper mapperTeslaSent.py \
```

```
-reducer reduceTeslaSent.py
```

```
253 hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming-*.jar -files mapperTeslaAnalysis.py,reduceTeslaAnalysis.py -input /input/ConvertedFile.json -output /output/  
t/cleaned_output -mapper mapperTeslaAnalysis.py -reducer reducerTeslaAnalysis.py
```

- The final step was to save the data cleaned to later export it to PostgreSQL. We used this command:

```
hdfs dfs -get /output/cleaned_output ./cleaned_output
```

- Before we exported the data to PostgreSQL we wanted to make sure that the cleaned output made by Hadoop is in a clear and understandable format, so we added the header of each field like: Post Id, Number of comments, Score etc and we also formatted the data in a csv form. We used the following codes to make this possible.

```
# Read the original file with tabs and write a new one with commas
with open("final_output.csv", "r", encoding="utf-8") as infile, \
    open("final_output_fixed.csv", "w", encoding="utf-8") as outfile:

    for line in infile:
        fields = line.strip().split("\t") # Divide by tabs
        cleaned_fields = [field.replace(",", " ") for field in fields] # Replace commas inside fields with spaces
        outfile.write(",".join(cleaned_fields) + "\n") # Join the fields with commas and write the line to the file
```

```

1 final_output_fixed.csv
2 post_id author created_utc subreddit score num_comments cleaned_text
3 1cigtzu,Appropriate_Door_524,1714659474.0,cars,644,182,la times source tesla fire entire supercharger team mostly fir
4 1clndy5,Doppelkupplungs,1715013023.0,cars,547,99,tesla lays staff in software service teams electrek reports
5 1cmro2l,Krankjanker,1715129179.0,cars,20,48,lucid tesla rivian fisker another good bad day standalone electric car ma
6 1cnzu7a,TurboSonic1,1715268072.0,cars,235,73,tesla cuts thousands us job listings following massive layoffs
7 1codw8i,BBQCopter,1715305035.0,cars,1179,216,hertz charges tesla model renter fee gas wont back down
8 1coq7ab,mostlyBadChoices,1715349488.0,cars,0,24,diehard petrolhead drives tesla first time in his life driving answer
9 1covv57,hplaptop12,1715364112.0,cars,481,142,elon musk manually approving service tickets tesla apparently denies cyb
10 1cpcgli,thentherestattoo,1715414736.0,cars,1087,172,tesla tells cybertruck owner coolant leaks arent covered warrant
11 1cr64tk,WUI9EMJ,1715623951.0,cars,358,50,tesla rehires supercharger workers weeks musks cuts

```

```

1 ConvertToLines.py > ...
2 input_path = "final_output_fixed.csv"
3 output_path = "final_output_clean.csv"
4
5 with open(input_path, "r", encoding="utf-8") as infile:
6     lines = infile.readlines()
7
8 # Add Headers
9 header = "post_id,author,created_utc,subreddit,score,num_comments,cleaned_text\n"
10
11 # Save with clean Header
12 with open(output_path, "w", encoding="utf-8") as outfile:
13     outfile.write(header)
14     for line in lines[1:]:
15         outfile.write(line)

```

```

1 final_output_clean.csv
2 post_id,author,created_utc,subreddit,score,num_comments,cleaned_text
3 1cigtzu,Appropriate_Door_524,1714659474.0,cars,644,182,la times source tesla fire entire supercharger team mostly fir
4 1clndy5,Doppelkupplungs,1715013023.0,cars,547,99,tesla lays staff in software service teams electrek reports
5 1cmro2l,Krankjanker,1715129179.0,cars,20,48,lucid tesla rivian fisker another good bad day standalone electric car ma
6 1cnzu7a,TurboSonic1,1715268072.0,cars,235,73,tesla cuts thousands us job listings following massive layoffs
7 1codw8i,BBQCopter,1715305035.0,cars,1179,216,hertz charges tesla model renter fee gas wont back down
8 1coq7ab,mostlyBadChoices,1715349488.0,cars,0,24,diehard petrolhead drives tesla first time in his life driving answer
9 1covv57,hplaptop12,1715364112.0,cars,481,142,elon musk manually approving service tickets tesla apparently denies cyb
10 1cpcgli,thentherestattoo,1715414736.0,cars,1087,172,tesla tells cybertruck owner coolant leaks arent covered warrant
11 1cr64tk,WUI9EMJ,1715623951.0,cars,358,50,tesla rehires supercharger workers weeks musks cuts

```

Now that the data is clean and correctly formatted. We are ready to transfer the data to insights. Next step is PostgreSQL

Challenges

This stage of cleaning the data was the most difficult part of this project because we ran the mapreduce job in a computer and Hadoop was not providing the expected results. It was stuck at the stage of Reducing, even though Mapping was 100% completed. The solution was to do the process again in another computer and it finally worked.

Analysis

This was the most interesting stage of the project because we got to know what is happening in the data. Below you can find the SQL queries as well as the results:

1

2

3

4

5

6

7

8

9

10

11

12

```

SELECT
  to_timestamp(created_utc)::date AS date,
  sentiment,
  COUNT(*) AS count
FROM posts
WHERE subreddit ILIKE '%technology%'
GROUP BY date, sentiment
ORDER BY date;

```

Data Output Messages Notifications

≡

📄

▼

📋

▼

🗑️

🗄️

⬇️

📈

SQL

	date date	sentiment text	count bigint
1	2025-03-30	negative	1
2	2025-03-31	neutral	1
3	2025-04-01	negative	2
4	2025-04-02	positive	1
5	2025-04-03	neutral	1
6	2025-04-04	neutral	3
7	2025-04-06	positive	1
8	2025-04-07	neutral	1
9	2025-04-08	positive	1
10	2025-04-09	negative	1

Total rows: 101

Query complete 00:00:00.226

1.

Sentiment Count per day for the technology subreddit.

```
1  SELECT subreddit, SUM(num_comments) AS total_comments
2  FROM posts
3  GROUP BY subreddit
4  ORDER BY total_comments DESC
5  LIMIT 6;
```

Data Output Messages Notifications



	subreddit text	total_comments bigint
1	technology	45961
2	stocks	21768
3	electricvehicles	19274
4	cars	19148
5	teslamotors	19135
6	investing	12502

Total rows: 6 Query complete 00:00:00.149

2. **Top 6 subreddits with the most number of total comments.**

Query Query History

```
1 SELECT
2   subreddit,
3   AVG(score) AS avg_score
4 FROM posts
5 GROUP BY subreddit
6 ORDER BY avg_score DESC
7 LIMIT 6;
```

Data Output Messages Notifications

	subreddit text	avg_score numeric
1	technology	5255.1575342465753425
2	stocks	756.4391891891891892
3	cars	436.4333333333333333
4	teslamotors	341.2080536912751678
5	electricvehicles	227.2953020134228188
6	investing	201.8175675675675676

Total rows: 6 Query complete 00:00:00.265

3.

Top 6 subreddits with the highest average score

```

1  SELECT subreddit, sentiment, COUNT(*) AS count
2  FROM posts
3  GROUP BY subreddit, sentiment;
4
5

```

Data Output Messages Notifications

SQL

	subreddit text	sentiment text	count bigint
1	electricvehicles	neutral	42
2	cars	neutral	46
3	technology	neutral	65
4	investing	negative	31
5	teslamotors	negative	21
6	cars	positive	55
7	stocks	neutral	2
8	cars	negative	49
9	electricvehicles	negative	39
10	technology	negative	55

Total rows: 18 Query complete 00:00:00.188

4. **Sentiment counts per subreddit**

```

1  SELECT
2      to_timestamp(created_utc)::date AS date,
3      sentiment,
4      COUNT(*) AS count
5  FROM posts
6  GROUP BY date, sentiment
7  ORDER BY date;
8
9

```

Data Output Messages Notifications

SQL

	date date	sentiment text	count bigint
1	2024-05-02	negative	1
2	2024-05-06	neutral	1
3	2024-05-08	positive	1
4	2024-05-09	negative	1
5	2024-05-10	negative	2
6	2024-05-10	positive	1
7	2024-05-11	neutral	1
8	2024-05-13	negative	1
9	2024-05-13	neutral	1
10	2024-05-14	neutral	1

Total rows: 450 Query complete 00:00:00.252

5.

Sentiment counts per day

Sentiment Analysis using Vader

```
#!/usr/bin/env python3
import sys
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

"""
Post-processing script:
- Reads cleaned Hadoop output (TSV lines)
- Applies VADER sentiment analysis on the cleaned text
- Outputs tab-separated values with sentiment label only
"""

analyzer = SentimentIntensityAnalyzer()

for line in sys.stdin:
    parts = line.strip().split("\t")
    if len(parts) < 7:
        continue # Skip malformed lines

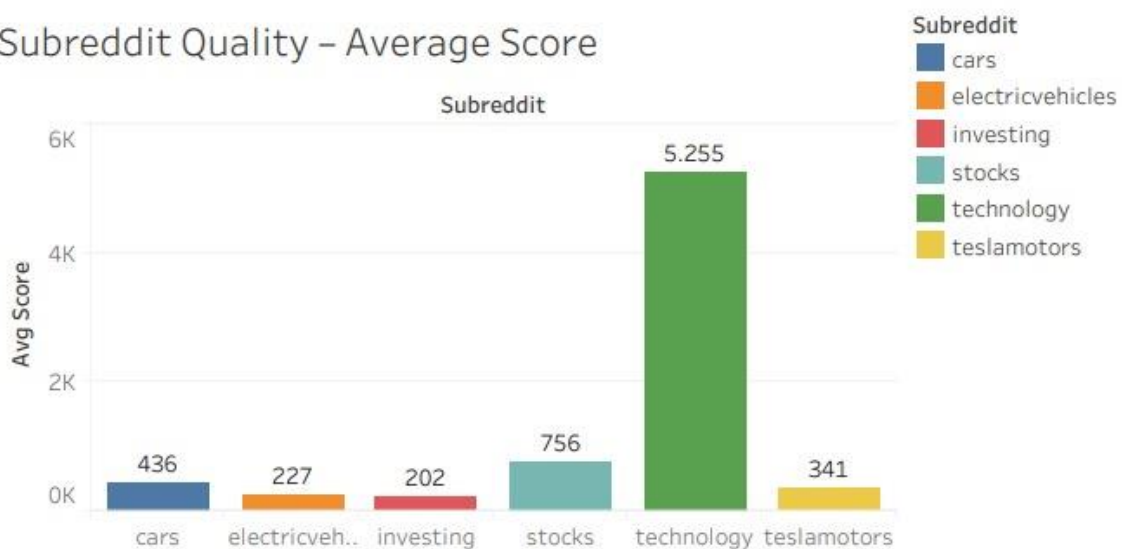
    post_id, author, created_utc, subreddit, score, num_comments, cleaned_text = parts

    # Get VADER compound score
    compound = analyzer.polarity_scores(cleaned_text)["compound"]

    # Assign sentiment label
    if compound >= 0.05:
        sentiment = "positive"
    elif compound <= -0.05:
        sentiment = "negative"
    else:
        sentiment = "neutral"

    # Print all fields + sentiment label
    print(f"{post_id}\t{author}\t{created_utc}\t{subreddit}\t{score}\t{num_comments}\t{cleaned_text}\t{sentiment}")
```

Visualization



Conclusions & recommendations

- As the data says Tesla does have positive comments in relation to investments and stocks, but they are mostly receiving a considerable amount of negative and neutral comments about aspects related to the product itself (Car).
- A great majority of users are leaving their opinions on things related to the technology aspect of Tesla. These comments are in the vast majority negative or neutral.
- As a recommendation to Tesla, they need to observe in more detail what specific aspects are negative regarding their technology, for example if the autopilot mode is not working as expected. This recommendation is key because, as the graph states, most people disagree with facts related to tech.