

Brand Sentiment Analysis using Twitter Mentions

J COMPONENT PROJECT REPORT

Winter 2019-20

Submitted by

KANIKA NARAYAN (17BCE0090)
AKSHATA CHOUDHARY (17BCE0149)

in partial fulfilment for the award of the degree of

B. Tech

in

Computer Science and Engineering

Vellore-632014, Tamil Nadu, India

School of Computer Science and Engineering

June, 2020

TABLE OF CONTENT:

<u>SL. NO.</u>	<u>TOPIC</u>		<u>PAGE NO.</u>
1	PROBLEM STATEMENT		3
2	LINK OF PROJECT GITHUB REPOSITORY		3
3	TEAM DETAILS		4
4	DATASET SOURCE WITH LINK		4
5	MODULES		4
5.1		MODULES OF THE PROJECT	4
5.2		CODE	5
5.3		SNAPSHOTS OF THE CODE	8
6	LIST OF CHALLENGES FACED		12
7	SNAPSHOT OF ERRORS AND FIXES		13
8	REFERENCES		14

1. PROBLEM STATEMENT:

In today's world consumers love to compare online and this extends to their and other consumers' opinion, not only on price or product features, When comparing one brand with another, people will often express something along the lines of 'I love company X but hate company Y' or 'company X's blue widget is so much better than company Y's'.

Under analysis, this could produce a neutral result as the negative sentiment cancels out the positive, when in fact there was a clear expression of positive sentiment about company X and negative sentiment about the other. Being able to spot and disentangle these kinds of issues at the data processing stage is important to establish a robust foundation for the sentiment analysis.

Brands use sentiment analysis to find and measure customer opinions and attitudes towards their brand, products, services, campaigns. Analyzing sentiment on social media provides an excellent source of data and will provide digital consumer insights that can:

- Determine brand reputation
- Improve customer experience
- Stop issues becoming a crisis
- Determine future marketing strategies
- Improve marketing campaigns and product messaging
- Identify brand influencers
- Test business KPIs
- Generate leads
- Understand how your brand reputation evolves over time
- Research your competition and understand how their reputation also evolves over time.

2. LINK TO THE PROJECT GITHUB REPOSITORY:

<https://github.com/kanikanarayan8/NLP-PROJECT.git>

3. TEAM DETAILS:

- 1) KANIKA NARAYAN (17BCE0090)
- 2) AKSHATA CHOUDHARY (17BCE0149)

4. DATASET SOURCE WITH LINK:

The dataset collected by us here is of the analysis of the brand “bmw” starting from date “2019-05-01” until “2020-03-03”. It is set for the geographical location “New York” so that all the tweets obtained are in the English language. Also, the number of tweets is restricted to 100 in order to save time and reduce computational power. The dataset obtained from this is:

https://docs.google.com/spreadsheets/d/e/2PACX-1vTYEgrqC5V1VtOtDRx7jboJvTRMWQc8Gp4X3TkIgSj_dJ05UG-jxDLk3wuob9MjL_7KelHweG3N_1yX/pub?gid=0&single=true&output=csv

For training the model, we have imported the movie review dataset from NLTK for training our model. The NLTK Corpus has a Movie Reviews Corpus with reviews categorized into pos and neg categories, and a number of trainable classifiers, that is each review is categorized as positive or negative.

5. MODULES:

5.1 Modules of the project:

- **Get the tweets from the twitter API by running the GetTweetFinal.py script. This script has the GetOldTweets3 module which fetches tweets based on the input parameters.**

In this module, we have downloaded the tweets based on the parameters that we have mentioned. The parameters used are: the brand name, the start and end date between which the tweets were done, and the geometric location from where the tweet was done. Adding geometric location will provide us with the tweets of one similar language. We have also set the number of tweets that we need to process on.

All the tweets are saved in a CSV file in the working directory.

- **Import the training dataset from NLTK**

For training the NLTK model, we have used an already existing dataset : movie reviews. We have used NLTK to download the movie_reviews dataset.

- **Preprocess data and set parameters for training the model**

Divide the dataset into training and testing dataset(80% dataset is kept for training and 20% is kept for testing).

- **Train and test model on training set**

Train the model using training dataset and testing the model using the testing dataset. We are using Naive Bayes Classifier to classify the tweets into positive or negative emotion based on the positive or negative words obtained from the training dataset.

Bayes theorem provides a way of calculating posterior probability $P(c|x)$ from $P(c)$, $P(x)$ and $P(x|c)$. The Naive Bayes Classifier converts data into a frequency table and then fills in a Likelihood Table. Now, it uses the Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

- **Run the saved model on the tweets which we got from Twitter**

The model that is trained, is used to find the sentiment of the tweets that were collected in the previous module.

5.2 Code:

- GetTweetFinal.py: For downloading the tweets based on our parameters for sentimental analysis.

```
pip install GetOldTweets3
#using twitter API to collect dataset
import GetOldTweets3 as got
import sys,getopt,datetime,codecs
#tweets are extracted based on certain criterias: brand name, start and end date
between which tweet was done and the geometric loaction of tweet done
tweetCriteria = got.manager.TweetCriteria().setQuerySearch('bmw')\
                .setSince("2019-05-01")\
                .setUntil("2020-03-03")\
                .setNear("New York")\
                .setMaxTweets(100)
#set geometric location as New York to collect all the tweets in english language
tweet = got.manager.TweetManager.getTweets(tweetCriteria)[0]
```

```

#opening csv file to store tweets
#tweets stored in tweets.csv file
outputFileName = "tweets.csv"
outputFile = codecs.open(outputFileName, "w+", "utf-8")

print('Searching...\n')
#writing the tweets in csv file
def receiveBuffer(tweets):
    for t in tweets:
        outputFile.write(t.text + "\n")
    #clear the internal buffer of the file
    outputFile.flush()
    print('More %d saved on file...\n' % len(tweets))

got.manager.TweetManager.getTweets(tweetCriteria, receiveBuffer)

outputFile.close()
print('Done. Output file generated "%s".' % outputFileName)

```

- NaiveBayesClassifier.py: For training and testing the model and using it for sentimental analysis.

```

import nltk
#training dataset is imported from NLTK
nltk.download('movie_reviews')

import re
import nltk.classify.util
from nltk.classify import NaiveBayesClassifier
from nltk.corpus import movie_reviews

#function to create dictionary of positive and negative words
# we use bag of words model where every word is feature name with a value of true
def extract_features(word_list):
    return dict([(word, True) for word in word_list])

# Load positive and negative reviews using the tags mentioned in the training dataset
positive_fileids = movie_reviews.fileids('pos')
negative_fileids = movie_reviews.fileids('neg')

# Gathering Positive and negative key words in a dictionary by calling the function
features_positive = [(extract_features(movie_reviews.words(fileids=[f])), 'Positive')
for f in positive_fileids]
features_negative = [(extract_features(movie_reviews.words(fileids=[f])), 'Negative')
for f in negative_fileids]

# Split the data into train and test (80/20)

```

```

#threshold set to 0.8 to split the data into 80% training dataset and 20% testing dataset
threshold_factor = 0.8
threshold_positive = int(threshold_factor * len(features_positive)) #length of 80%
positive words+ length of 80% of the negative words for training dataset
threshold_negative = int(threshold_factor * len(features_negative)) #remaining length
of 20% each positive and negative words for testing dataset

#80% of positive words+80% of the negative words as training dataset
features_train      =      features_positive[:threshold_positive]      +
features_negative[:threshold_negative]
#remaining 20% each positive and negative words as testing dataset
features_test       =      features_positive[threshold_positive:]      +
features_negative[threshold_negative:]

print ("\nNumber of training datapoints:", len(features_train))
print ("Number of test datapoints:", len(features_test))

# Training the model using Naive Bayes classifier
classifier = NaiveBayesClassifier.train(features_train)
print ("\nAccuracy of the classifier:", nltk.classify.util.accuracy(classifier,
features_test))

# Sample input reviews for checking the sentimental analysis given by the model
input_reviews = [
    "i dont like the upholstery in this car",
]

# Saved model is run on the tweets which we got from Twitter

#for counting number of positive reviews.
p = 0
#for counting number of negative reviews.
n = 0
with open('tweets.csv',encoding="utf-8") as f:
    #checking sentimental for the first 10 tweets collected
    for i in range(10):
        Review = f.readline()
        print ("\nReview:", Review)
        #calculating probability distribution
        probdist = classifier.prob_classify(extract_features(Review.split()))
        pred_sentiment = probdist.max()
        print ("Predicted sentiment:", pred_sentiment )
        print ("Probability:", round(probdist.prob(pred_sentiment), 2))
        if pred_sentiment=="Positive":
            p+=1
        elif pred_sentiment=="Negative":
            n+=1

    print("Overall response average:", (p/(p+n)))
i = 0

```

```

input_reviews = [
    "i dont like the upholstery in this car",
]

p = 0
n = 0

i = 0

#checking sentimental analysis for the given input
for review in input_reviews:

    print ("\nReview:", review)
    probdist = classifier.prob_classify(extract_features(review.split()))
    pred_sentiment = probdist.max()

    print ("Predicted sentiment:", pred_sentiment )
    print ("Probability:", round(probdist.prob(pred_sentiment), 2))

```

5.3 Snapshots of output:

- GetTweetFinal.py:

```
In [1]: pip install GetOldTweets3
```

```

Collecting GetOldTweets3
  Downloading https://files.pythonhosted.org/packages/ed/f4/a00c2a7c90801abc875325bb5416ce9090ac86d06a00cc887131bd73ba45/GetOldTweets3-0.0.11-py3-none-any.whl
Requirement already satisfied: lxml>=3.5.0 in c:\users\dell\anaconda3\lib\site-packages (from GetOldTweets3) (4.3.4)
Collecting pyquery>=1.2.10 (from GetOldTweets3)
  Downloading https://files.pythonhosted.org/packages/78/43/95d42e386c61cb639d1a0b94f0c0b9f0b7d6b981ad3c043a836c8b5bc68b/pyquery-1.4.1-py2.py3-none-any.whl
Collecting cssselect>0.7.9 (from pyquery>=1.2.10->GetOldTweets3)
  Downloading https://files.pythonhosted.org/packages/3b/d4/3b5c17f00cce85b9a1e6f91096e1cc8e8ede2e1be8e96b87ce1ed09e92c5/cssselect-1.1.0-py2.py3-none-any.whl
Installing collected packages: cssselect, pyquery, GetOldTweets3
Successfully installed GetOldTweets3-0.0.11 cssselect-1.1.0 pyquery-1.4.1
Note: you may need to restart the kernel to use updated packages.

```



```
In [2]: #using twitter API to collect dataset
import GetOldTweets3 as got
import sys, getopt, datetime, codecs
#tweets are extracted based on certain criterias: brand name, start and end date between which tweet was done and the geometric location
tweetCriteria = got.manager.TweetCriteria().setQuerySearch('bmw')\
    .setSince("2019-05-01")\
    .setUntil("2020-03-03")\
    .setNear("New York")\
    .setMaxTweets(100)

#set geometric Location as New York to collect all the tweets in english language
tweet = got.manager.TweetManager.getTweets(tweetCriteria)[0]
#opening csv file to store tweets
#tweets stored in tweets.csv file
outputFileName = "tweets.csv"
outputFile = codecs.open(outputFileName, "w+", "utf-8")

print('Searching...\n')
#writing the tweets in csv file
def receiveBuffer(tweets):
    for t in tweets:
        outputFile.write(t.text + "\n")
        #clear the internal buffer of the file
        outputFile.flush()
    print('More %d saved on file...\n' % len(tweets))

got.manager.TweetManager.getTweets(tweetCriteria, receiveBuffer)


outputFile.close()
print('Done. Output file generated "%s".' % outputFileName)

Searching...

More 100 saved on file...

Done. Output file generated "tweets.csv".
```

Tweets.csv where we collected all the tweets

 jupyter tweets.csv ✓ a few seconds ago Logout

File Edit View Language current mode

```

1 Congrats
2 M3 on the lift this morning finishing up with rear sub frame replacement #bmw #bmwm3 #m3 #wedoitall #wemakeitbetter #nolimits
  #azevedomotorsports #lovelyjob #lindenj #newjersey #portuguese @Azevedo Motorsports https://www.instagram.com/p/B901hnxn6uQ/?
  igshid=16k81lqvzxl6c
3 Happy Monday ... Here is the completed M coupe we delivered last week... Car came in for some minor insurance work and ended up getting a
  full restoration .. #Bmw #bmwcoupe #restoration #wemakeitnewagain #nolimits.. https://www.instagram.com/p/B90wfM0nwvF/?igshid=10mpbw02l3lyi
4 ///M Power Donuts mli2.0 yousef.zarrouk @ffreshboii96 Tag us #bmw_world_ua @New York, New York https://www.instagram.com/p/B90oPkwFviN/?
  igshid=1xdkfx69zxv35
5 Sickkkk M2 _leo.oliveira_ Pic by lamecarz Tag us #bmw_world_ua @Jersey City, New Jersey https://www.instagram.com/p/B90aEdcFuL8/?
  igshid=1cqpxfzjk3j92
6 Bmw M5 competition, dreams I gotta make come true
7 It's March 1st! Do you know what that means?! : 01madgli | #BMW | #M4 | #BMW4 | #BMWRepost | #BMWRepost |
  #BMW_world_ua | #BMW_MPoweer | #BMWlife | #IG_BMW | #MPower |... https://www.instagram.com/p/B9NF63blQn3/?igshid=15ah4nistqom4
8 Bmw M235i driver A21KNH parked illegally near 222 E Broadway on March 1. This is in Manhattan Community Board 03 #CB3Man &#amp; #NYPD7.
  #VisionZero
9 But that blue BMW was the devil.
10 Clean M3 .metjahic Photo by http://ray.nyc Tag us #bmw_world_ua @New York, New York https://www.instagram.com/p/B9L7k6YFqm9/?
  igshid=6w23tiq5wykb
11 Bmw 528i uber driver T791070C drove recklessly near 197 1st Avenue on February 29 and has been reported to #nyc taxi. This is in Manhattan
  Community Board 03 #CB3Man &#amp; #NYPD9. #VisionZero
12 Stunning M8 grknation Photo by r.ego Via m8world Tag us #bmw_world_ua @New York, New York https://www.instagram.com/p/B9JShHFFXUm/?
  igshid=1gzkh9hqbxpzd
13 Bitches talking bout bmw's that I been had ..old ass years that been passed
14 Sickkkk M5 bo_buttaa Tag us #bmw_world_ua @New York, New York https://www.instagram.com/p/B9IRq0Dlr1p/?igshid=65ivighvorae
15 Hi Ammazing pic clickclacksmack Tag us #bmw_world_ua @New York, New York https://www.instagram.com/p/B9ICDCSFgq2/?igshid=17dhmy4gh8siz
16 Alfa give us 250 HP with a 6 spd manual that we can afford. I guess you don't have what it takes to face the BMW 3 series or a VW Golf.
17 Bmw X6 driver LCK3850 was speeding near 678 Madison St on February 27. This is in Brooklyn Community Board 03 &#amp; #NYPD81. #VisionZero
18 overheard on the upper west side " i also crashed my BMW into a burger king parking lot"
19 Bmw 328i driver JNZ1656 blocked the bike lane near 76 Reade St on February 27. This is in Manhattan Community Board 01 #CommunityBoard1
  &#amp; #NYPD1. #VisionZero #BlockedBikeNYC
20 " dile que ya no es fuego es lava " : vincent.pepino #bmw bmw #bmwm bmwm #m3 #igbmw #f8xgang f8xgang #f80 @kleaper619 #bmwpower #bmwrepost
  #bimmer #carlifestyle #stancenation.. https://www.instagram.com/p/B9FENUkg9NP/?igshid=mbfv6yp6n5z9
21 Save this Tax Season!!! Call/DM us today for details (718)-558-0788 http://www.HillsideAutoNY.com #HSAutoMall #Toyota #Honda #BMW #Mercedes
  #Jeep #MercedesBenz #Infiniti #Subaru #Sale #Discount #BadCredit #NoCredit #Queens #Brooklyn #LongIsland #Bronx #Canarsie #Jamaica - at
  Hillside Auto Mall
22 Bmw X5 driver HmW5127 parked illegally near 500 St Johns Pl on February 27. This is in Brooklyn Community Board 08 &#amp; #NYPD77.
  #VisionZero
23 Bmw X5 driver C00L1F blocked the bike lane near 713 E 138th St on February 27. This is in Bronx Community Board 01 &#amp; #NYPD40.
```

- NaiveBayesClassifier.py:

```
In [1]: import nltk
#training dataset is imported from NLTK
nltk.download('movie_reviews')

import re
import nltk.classify.util
from nltk.classify import NaiveBayesClassifier
from nltk.corpus import movie_reviews

[nltk_data] Downloading package movie_reviews to
[nltk_data] C:\Users\dell\AppData\Roaming\nltk_data...
[nltk_data] Package movie_reviews is already up-to-date!

In [2]: #function to create dictionary of positive and negative words
def extract_features(word_list):
    return dict([(word, True) for word in word_list])

# Load positive and negative reviews using the tags mentioned in the training dataset
positive_fileids = movie_reviews.fileids('pos')
negative_fileids = movie_reviews.fileids('neg')

# Gathering Positive and negative key words in a dictionary by calling the function
features_positive = [(extract_features(movie_reviews.words(fileids=[f])), 'Positive') for f in positive_fileids]
features_negative = [(extract_features(movie_reviews.words(fileids=[f])), 'Negative') for f in negative_fileids]

# Split the data into train and test (80/20)
#threshold set to 0.8 to split the data into 80% training dataset and 20% testing dataset
threshold_factor = 0.8
threshold_positive = int(threshold_factor * len(features_positive)) #length of 80% positive words+ length of 80% of the negative
threshold_negative = int(threshold_factor * len(features_negative)) #remaining length of 20% each positive and negative words for

#80% of positive words+80% of the negative words as training dataset
features_train = features_positive[:threshold_positive] + features_negative[:threshold_negative]
#remaining 20% each positive and negative words as testing dataset
features_test = features_positive[threshold_positive:] + features_negative[threshold_negative:]

print ("\nNumber of training datapoints:", len(features_train))
print ("Number of test datapoints:", len(features_test))

# Training the model using Naive Bayes classifier
classifier = NaiveBayesClassifier.train(features_train)
print ("\nAccuracy of the classifier:", nltk.classify.util.accuracy(classifier, features_test))

# Sample input reviews for checking the sentimental analysis given by the model
input_reviews = [
    "i dont like the upholstery in this car",
]

# Saved model is run on the tweets which we got from Twitter

#for counting number of positive reviews.
p = 0
#for counting number of negative reviews.
n = 0
with open('tweets.csv', encoding="utf-8") as f:
    #checking sentimental for the first 10 tweets collected
    for i in range(10):
        Review = f.readline()
        print ("\nReview:", Review)
        #calculating probability distribution
        probdbist = classifier.prob_classify(extract_features(Review.split()))
        pred_sentiment = probdbist.max()
        print ("Predicted sentiment:", pred_sentiment )
        print ("Probability:", round(probdbist.prob(pred_sentiment), 2))
        if pred_sentiment=="Positive":
            p+=1
        elif pred_sentiment=="Negative":
            n+=1

    print("Overall response average:", (p/(p+n)))
i = 0
```

Number of training datapoints: 1600
Number of test datapoints: 400

Accuracy of the classifier: 0.735

Review: Congrats

Predicted sentiment: Positive
Probability: 0.5

Review: M3 on the lift this morning finishing up with rear sub frame replacement #bmw #bmwm3 #m3 #wedoitall #wemakeitbetter #no limits #azevedomotorsports #lovemyjob #lindenj #newjersey #portuguese @Azevedo Motorsports <https://www.instagram.com/p/B901hnxn6uQ/?igshid=16k81lqvzx16c>

Predicted sentiment: Positive
Probability: 0.53

Review: Happy Monday ... Here is the completed M coupe we delivered last week... Car came in for some minor insurance work and ended up getting a full restoration .. #Bmw #bmwmcoupe #restoration #wemakeitnewagain #nolimits... <https://www.instagram.com/p/B9OwfM0nwwF/?igshid=10mpbw0213lyi>

Predicted sentiment: Positive
Probability: 0.88

Review: ///M Power Donuts mlit2.0 yousef.zarrouk @ffreshboii96 Tag us #bmw_world_ua @New York, New York <https://www.instagram.com/p/B9OoPkwFviN/?igshid=1xdkfx69zxv35>

Predicted sentiment: Positive
Probability: 0.52

Review: Sickkkk M2 _leo.oliveira_ Pic by lamecarz Tag us #bmw_world_ua @Jersey City, New Jersey <https://www.instagram.com/p/B9OaEdcFuL8/?igshid=1cqpxf2jk3j92>

Predicted sentiment: Positive
Probability: 0.52

Review: Bmw M5 competition, dreams I gotta make come true

Predicted sentiment: Positive
Probability: 0.7

Review: It's March 1st! Do you know what that means?! : 01madgli | #BMW | #M4 | #BMWWM4 | #BMWRepost | #BMWRepost | #BMWWorldUA | #BMWMPower | #BMWLife | #IG_BMW | #MPower |... <https://www.instagram.com/p/B9NF6JblQn3/?igshid=15ah4nistqom4>

Predicted sentiment: Positive
Probability: 0.81

Review: Bmw M235i driver A21KNH parked illegally near 222 E Broadway on March 1. This is in Manhattan Community Board 03 #CB3Ma n & #NYPD7. #VisionZero

```
In [7]: # Sample input reviews
input_reviews = [
    "i dont like the upholstery in this car",
]

p = 0
n = 0

i = 0

#checking sentimental analysis for the given input
for review in input_reviews:

    print("\nReview:", review)
    probdist = classifier.proba_classify(extract_features(review.split()))
    pred_sentiment = probdist.max()

    print("Predicted sentiment:", pred_sentiment )
    print("Probability:", round(probdist.proba(pred_sentiment), 2))
```

Review: i dont like the upholstery in this car
Predicted sentiment: Negative
Probability: 0.84

```
In [4]: # Sample input reviews
input_reviews = [
    "This car is amazing",
]

p = 0
n = 0

i = 0

for review in input_reviews:

    print("\nReview:", review)
    probdist = classifier.proba_classify(extract_features(review.split()))
    pred_sentiment = probdist.max()

    print("Predicted sentiment:", pred_sentiment )
    print("Probability:", round(probdist.proba(pred_sentiment), 2))
```

Review: This car is amazing
Predicted sentiment: Positive
Probability: 0.52

```

probability: 0.69

In [5]: # Sample input reviews
input_reviews = [
    "the car has a gay color",
]

p = 0
n = 0

i = 0

for review in input_reviews:
    print ("\nReview:", review)
    probdist = classifier.proba_classify(extract_features(review.split()))
    pred_sentiment = probdist.max()

    print ("Predicted sentiment:", pred_sentiment )
    print ("Probability:", round(probdist.proba(pred_sentiment), 2))

```

```

Review: the car has a gay color
Predicted sentiment: Positive
Probability: 0.69

```

```

In [6]: # Sample input reviews
input_reviews = [
    "This car is not good",
]

p = 0
n = 0

i = 0

for review in input_reviews:
    print ("\nReview:", review)
    probdist = classifier.proba_classify(extract_features(review.split()))
    pred_sentiment = probdist.max()

    print ("Predicted sentiment:", pred_sentiment )
    print ("Probability:", round(probdist.proba(pred_sentiment), 2))

```

```

Review: This car is not good
Predicted sentiment: Negative
Probability: 0.61

```

6. LIST OF CHALLENGES FACED:

The challenges faced while doing the sentimental analysis of the tweets for the brands mentioned was:

- For a word that is positive, when negative words are added before it then the overall polarity of the word still remains positive.
- In the movie review dataset that we imported from NLTK, each word has a polarity as positive or negative. Whenever a negative word is detected, all the words from a negation cue to the next punctuation token are negated. So, the positive word following it, gets the negative polarity.

7. SNAPSHOTS OF ERRORS AND FIXES:

Error: Whenever there was a negation word like ‘not’, the sentiment was not predicted correctly.

```
In [4]: # Sample input reviews
input_reviews = [
    "This car is not bad",
]

p = 0
n = 0

i = 0

for review in input_reviews:

    print ("\nReview:", review)
    probdist = classifier.proba_classify(extract_features(review.split()))
    pred_sentiment = probdist.max()

    print ("Predicted sentiment:", pred_sentiment )
    print ("Probability:", round(probdist.proba(pred_sentiment), 2))
```

```
Review: This car is not bad
Predicted sentiment: Negative
Probability: 0.75
```

Fixing the error: We set the polarity of each word as positive or negative. When we come across a negation word, negate the polarity of all the words encountered as that till the next punctuation.

```
In [6]: # Sample input reviews
input_reviews = [
    "This car is not good",
]

p = 0
n = 0

i = 0

for review in input_reviews:

    print ("\nReview:", review)
    probdist = classifier.proba_classify(extract_features(review.split()))
    pred_sentiment = probdist.max()

    print ("Predicted sentiment:", pred_sentiment )
    print ("Probability:", round(probdist.proba(pred_sentiment), 2))
```

```
Review: This car is not good
Predicted sentiment: Negative
Probability: 0.61
```

8. **REFERENCES:**

- [1] Songbo Tan, Xueqi Cheng, Yuefen Wang, Hongbo Xu (2009). Adapting Naive Bayes to Domain
- [2] Adaptation for Sentiment Analysis. Lecture notes in Computer Science, 5478, pp. 337-349.
- [3] M. Venugopalan, D. Gupta (2015). Exploring sentiment analysis on twitter data. Eighth International Conference on Contemporary Computing (IC3), pp. 241-247.
- [4] Narayanan V., A. I. (2013). Fast and Accurate Sentiment Classification Using an Enhanced Naive Bayes Model. Intelligent Data Engineering and Automated Learning vol. 8206.
- [5] . Batool, A. M. Khattak, J. Maqbool, S. Lee (2013). Precise tweet classification and sentiment analysis. *IEEE/ACIS 12th International Conference on Computer and Information Science (ICIS)*), pp. 461-466.
- [6] C. Troussas, M. V. (2013). Sentiment analysis of Facebook statuses using Naive Bayes classifier for language learning. *IISA Piraeus*, 1-6.
- [7] Lopamudra Dey, S. C. (2016). Sentiment Analysis of Review Datasets Using Naive Bayes and K-NN Classifier. *arXiv:1610.09982*.
- [8] Yustinus Eko Soelistio, M. R. (2015). Simple Text Mining for Sentiment Analysis of Political Figure Using Naive Bayes Classifier Method. *arXiv, arXiv:1508.05163*.