Customer Voice Analysis

CS 522: Advance Data Mining

Spring 2017

SUBMITTED BY:

SAMRUDDHI NAIK

TEAMMATES: ROHAN DIGAMBAR GAWADE MANEESHA SATYANARAYANAN AMRUTA PIMPLE ABHISHEK BHARDWAJ

ILLINOIS INSTITUTE OF TECHNOLOGY

Abstract

- The project is planned to capture customer preferences and opinions, analyze them to gain new business insights, and then share them to create meaningful change.
- We intend to collect and unify customer reviews/ratings from Twitter using sentiment analysis to provide organizations to understand their brand value over the time and to help them identify investments that delight their customers and develop new products to maximize satisfaction and profits.
- We will focus on 3 categories which are important business platforms in USA. These includes: **Clothing, Food and Retail**.
- Top 5 brands from each of these categories –

Clothing – **Zara, Levis, Nike, Ralph Lauren and Tommy Hilfiger** (by Samruddhi and Amruta)

Food **– Pizza Hut, Dominos, Papa Johns, Burger King and McDonalds** (by Rohan and Maneesha)

Retail - **Walgreens, CVS Pharmacy, Target, Best buy, Costco and Walmart** (by Abhishek)

- Thus, by analyzing the tweets by various customers we can compare what can be done for the same brands with respect to different cities, so that the business unit of that brand can evaluate what the customers think about their services and products over the different cities and can think on what measures they must take to improve in the under-performing cities. This will help the businesses improve their products as well as the services to the customers.

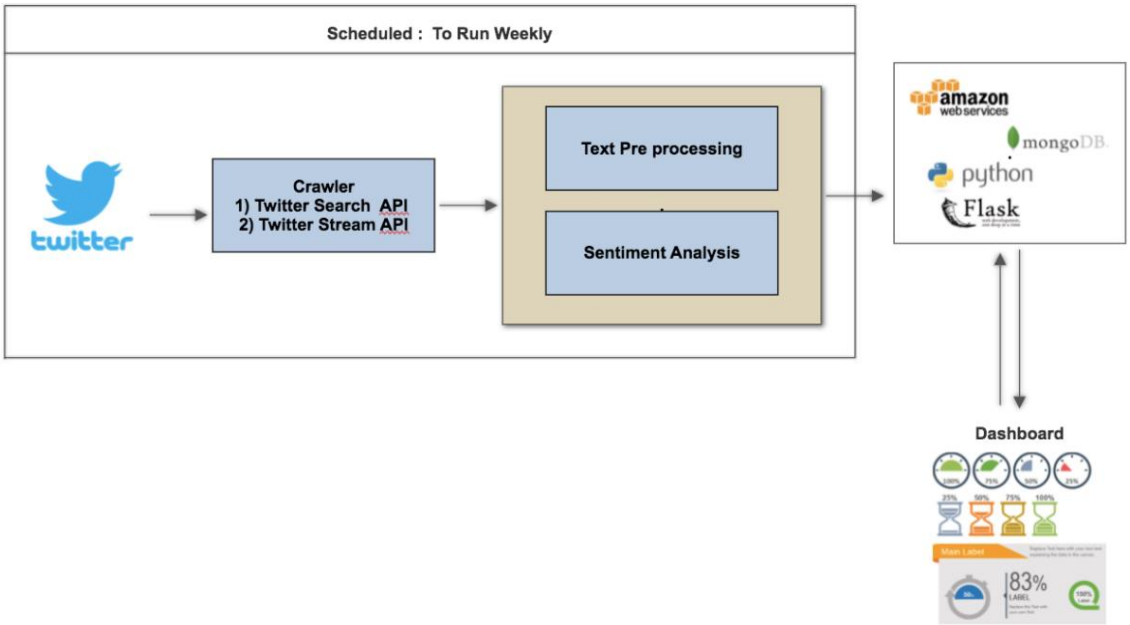


Fig: Architecture

Architecture

1. **Crawler:** We used Beautiful Soup and LXML and a GitHub repository by Jefferson Henrique for crawling
2. **Text Processing Block:** Here we plan to perform all the text processing and sentiment analysis (Textblob, or self-made classifier)
3. **Amazon Web Service**
    1. **Amazon EC2 Instance:** Free Server to host the project
    2. **MongoDB:** For simplicity of getting JSON response we plan to use MongoDb as it stores the data as a dictionary
    3. **Python Flask:** for Backend server and API hosting
4. **Dashboard:** We plan to use HTML framework bootstrap and AngularJS for fronted development

1. Data

1.1 Data Source

i. The data is extracted by crawling on www.twitter.com.
ii. As new tweets are updated everyday data size will increase proportionately.
iii. The tweets which will be extracted are stored into csv file along with username, date, text, geo location.
iv. I have generated CSV file generated containing data for November 2016 to January 2017 for twitter handle @ZARA. The file has 11001 tweets and is of size 2.5Mbs.
v. As we have a csv file of manageable size, processing of this file won't be an issue.

1.2 Data Format

- The get the tweets in text format.
- We then crawl and store the customer tweets into csv file each for distinct brands.
- The csv file consists of tweets;favorites;text;geo;mentions;hashtags;id;permalink.
- We will preprocess this data and store it into the database.

Column	Data Type	Discreption
Date	YYYY-MM-DD HH:MM	Time of the Tweet
Retweet Count	NUMBER	Number of times tweet has been retweeted
Mention	NUMBER	@keywords' mentions
Favorites	NUMBER	"Number of Likes"
Hashtag	ALPHANUMERIC	#tag present in the tweet
Text	STRING	Actual Tweet
Link	URL	Link of the Tweet
Geo	STRING	Location of tweet/User

1.3 Programming Language, packages

- **Programming Language:** Python
- **Packages:** TwitterSearch, Tweepy, BeautifulSoup, TextBlob, urllib, PyQuery, Flask,NLTK, Scikit, Scilearn and Pandas.
- We are using Twitter's search and stream API like TwitterSearch and Tweepy to get the recent data of past one week.
- Twitter API's have a limitation that we can scrape the data of only past one week. Hence, we used an open source API GetOldTweets-python.
  The link is given below:
  https://github.com/Jefferson-Henrique/GetOldTweets-python

2. Preliminary Experiments

We have divided the project into three categories and we will be working in teams of two for food and clothing, while one of our project mate will be working on retail. I will be contributing in Clothing section and working towards extracting data for three brands. I have used twitter handles for these brands to search queries for the three months(Nov-Jan) and create a csv file containing tweets.

2.1 Text Preprocessing Steps

- Convert tweets to lower case.
- Removing URL for the tweets.
- Removing Punctuations.
- Remove Hashtags.

Sample:
Original - @ZARA is my Fav Store for men
After preprocessing – is my fav store for men

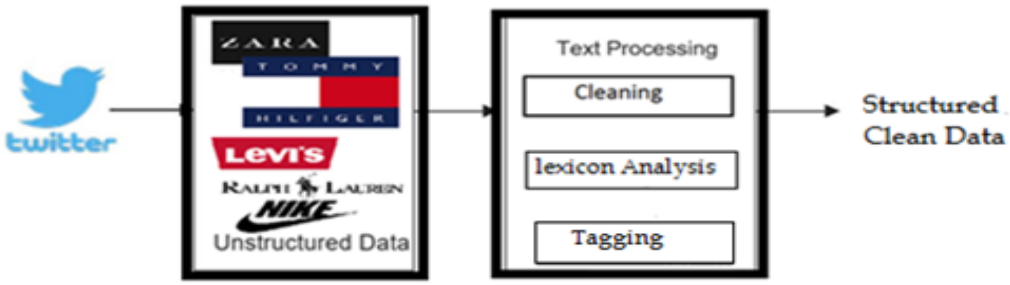


Fig: Text Preprocessing

3. Data Cleaning:

The very first task after obtaining data is to clean it by extraction of keywords and symbols. For instance – Emoticons are the smiley used in textual form to represent emotions e.g. ":-)", ":)", "=)",":D", ":-(", ":(", "=(", ";(", etc.. Correcting the all uppercase and all lowercase to a common case, removing the non-English (or proffered language texts), removing un-necessary white spaces and tabs, etc. So, the data of all three clothing brands was cleaned by me.

4. Lexicon-based sentiment analysis:

Application of a lexicon is to do sentiment analysis and it involves calculating the sentiment from the semantic orientation of word or phrases that occur in a text. With this approach, a dictionary of positive and negative words is required, with a positive or negative sentiment value assigned to each of the words. In lexicon-based approaches a piece of text message is represented as a bag of words.

In our work, we have decided to apply a lexicon-based approach to avoid the need to generate a labelled training set. After performing on around **1000 tweets** for each brand, I got some good candidates which further were examined.

VaderSentiment handles negation on its own.

4.1 Tagging

In our work, I used VaderSentiment (lexicon based technique) and TextBlob technique for constructing the model. TextBlob uses pattern sentiment analysis which has tools for data mining (Google, Twitter and Wikipedia API, a web crawler, a HTML DOM parser), natural language processing (part-of-speech taggers, n-gram search, sentiment analysis, WordNet), machine learning (vector space model, clustering, SVM), network analysis and canvas visualization.

I have executed on positive, negative, neutral for training data to do lexicon analysis and give us the polarity of sentiment as either positive, negative or neutral. we have now 1000 tweets each of positive, negative, neutral sentiment for three clothing brands each.

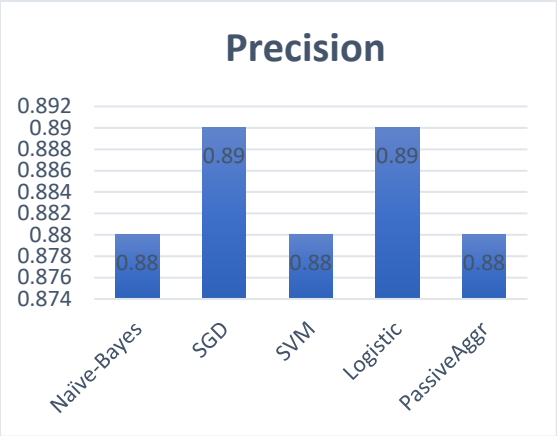
1) Count Vectorizer: Convert a collection of text documents/Tweets to a matrix of token counts so that they can be used as features in predictive modelling

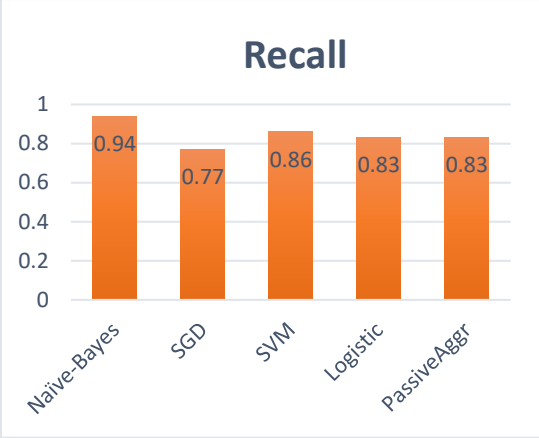
2) Tf-idf Transformer: Transform a count matrix to a normalized term frequency or tf-idf representation

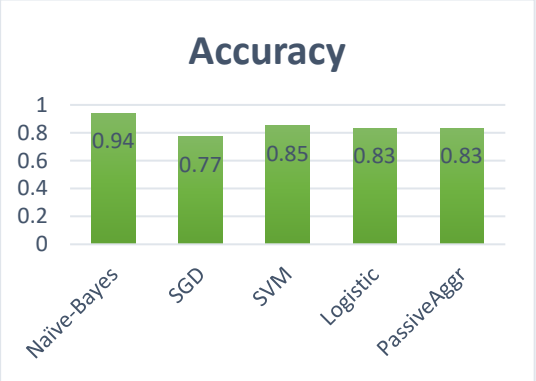
We used the following 5 classifiers:

1) Naïve Bayes
2) Stochastic Gradient Decent Classifier
3) Support Vector Classifier
4) Logistic Regression
5) Passive Aggressive Classifier

I got the following output after I calculated precision recall and accuracy of tweets using **confusion matrix for the brands** which I have considered – Zara, Tommy Hilfiger and Ralph Lauren.







5. Naïve Bayes

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. The parameters θ_y is estimated by a smoothed version of maximum likelihood, i.e. relative frequency counting: $\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n}$ where $N_{yi} = \sum_{x \in T} x_i$ is the number of times feature i appears in a sample of class y in the training set T, and $N_y = \sum_{i=1}^{|T|} N_{yi}$ is the total count of all features for class y.(Wikipedia)

Thus, I identified that **Naïve Bayes Classifier** gives the best possible precision, recall and accuracy for the brands Zara, Tommy and Ralph Lauren which I have selected for my analysis and highlighted in below summary table.

Parameters	Overall Clothing brands	Levis, Nike	**Zara, Tommy, Ralph**
Precision	SVM	SVM	**SGD**
Recall	Naïve-Bayes	SVM	**Naïve-Bayes**
Accuracy	Naïve-Bayes	SVM, Naïve-Bayes	**Naïve-Bayes**

Naive Bayes is a simple technique for constructing classifiers: models that assign class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. An advantage of naive Bayes is that it only requires a small number of training data to estimate the parameters necessary for classification.
Given a class variable y and a dependent feature vector x_1 through x_n, Bayes' theorem states the following relationship:

$$P(y \mid x_1, \dots, x_n) = \frac{P(y)P(x_1, \dots x_n \mid y)}{P(x_1, \dots, x_n)}$$

Using the naive independence assumption that

$$P(x_i|y, x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_n) = P(x_i|y),$$

for all i, this relationship is simplified to

$$P(y \mid x_1, \dots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \dots, x_n)}$$

Since $P(x_1, \dots, x_n)$ is constant given the input, we can use the following classification rule:

$$P(y \mid x_1, \dots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y)$$

$$\Downarrow$$

$$\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y),$$

and we can use Maximum A Posteriori (MAP) estimation to estimate $P(y)$ and $P(x_i \mid y)$; the former is then the relative frequency of class y in the training set. (Wikipedia)

(Cite:http://scikitlearn.org/stable/modules/generated/sklearn.naive_bayes.MultinomialNB.html#examples-using-sklearn-naive-bayes-multinomialnb)

Parameters:	alpha: float, optional (default=1.0) Additive (Laplace/Lidstone) smoothing parameter (0 for no smoothing). fit_prior: boolean, optional (default=True) Whether to learn class prior probabilities or not. If false, a uniform prior will be used. class_prior: array-like, size (n_classes,), optional (default=None) Prior probabilities of the classes. If specified the priors are not adjusted according to the data.

Attributes:	class_log_prior_: array, shape (n_classes,) Smoothed empirical log probability for each class. intercept_: property Mirrors class_log_prior_ for interpreting MultinomialNB as a linear model. feature_log_prob_: array, shape (n_classes, n_features) Empirical log probability of features given a class, P(x_i|y). coef_: property Mirrors feature_log_prob_ for interpreting MultinomialNB as a linear model. class_count: array, shape (n_classes,) Number of samples encountered for each class during fitting. This value is weighted by the sample weight when provided.

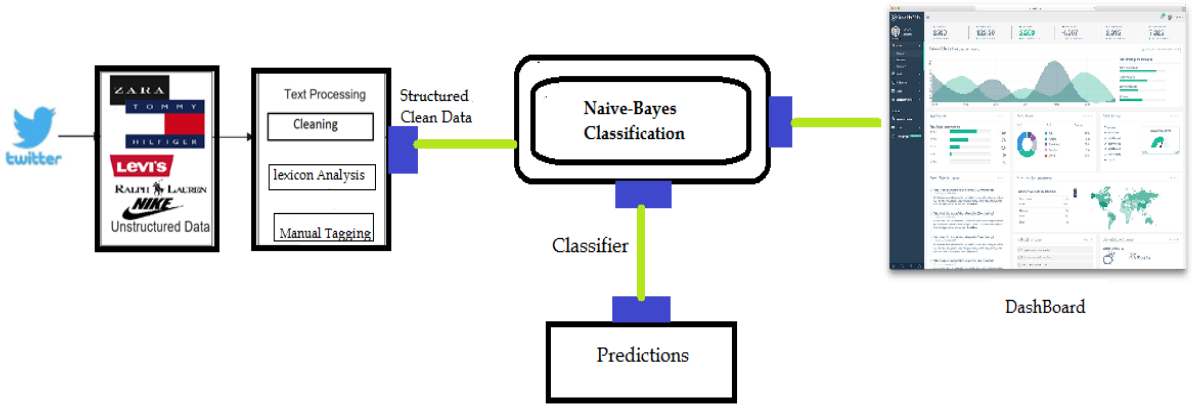


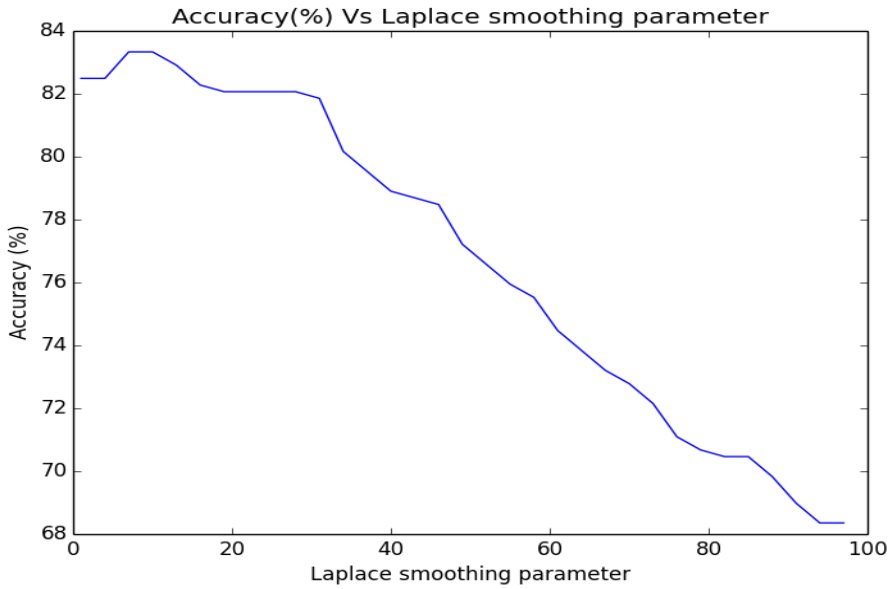
Fig: Classification of Dataset

6. Parameter Tuning

So, for calculating the accuracy of the Naïve Bayes, I used **Laplace smoothing**.

Given an observation $\mathbf{x} = (x_1, \ldots, x_d)$ from a multinomial distribution with N trials and parameter vector $\boldsymbol{\theta} = (\vartheta_1, \ldots, \vartheta_d)$, a "smoothed" version of the data gives the estimator:where the pseudocount $\alpha > 0$ is the smoothing parameter ($\alpha = 0$ corresponds to no smoothing).Laplace smoothing is a type of shrinkage estimator, as the resulting estimate will be between the empirical estimate x_i / N, and the uniform probability $1/d$. Using Laplace's rule of succession, in practice a smaller value is typically chosen.

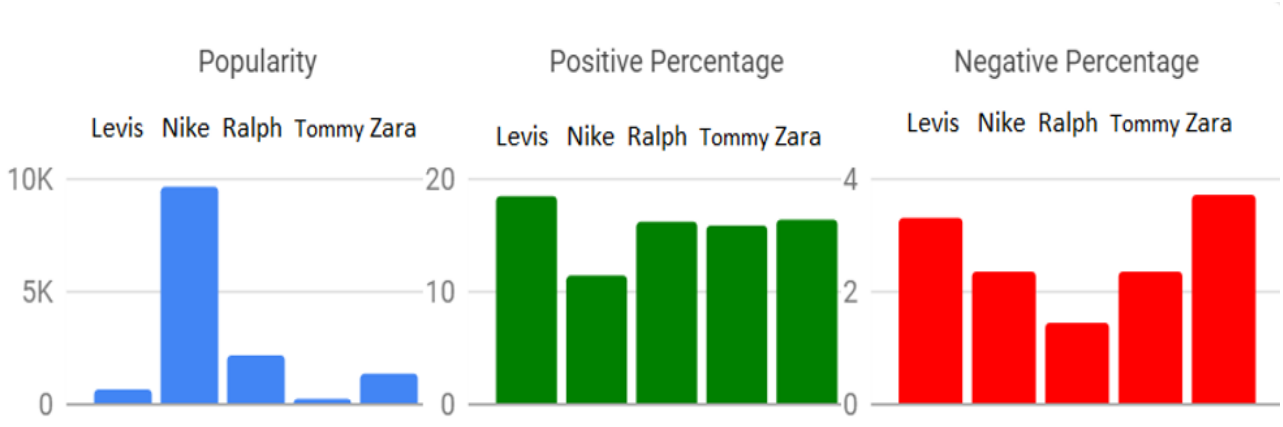
For my data of **three clothing brands wiz Ralph Lauren, Zara and Tommy Hilfiger** after varying the Laplace smoothing parameter following graph was obtained.



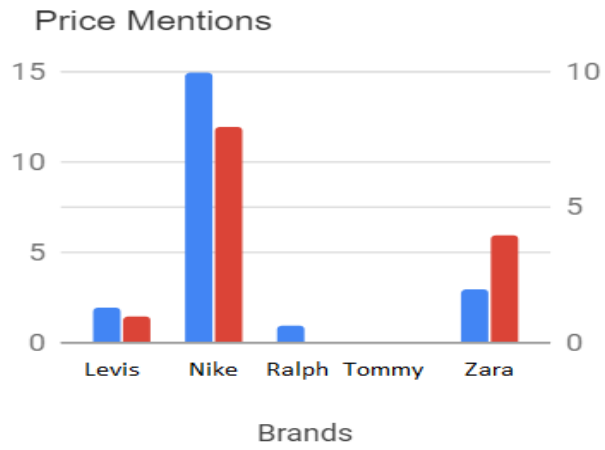
As seen from the graph, Laplace smoothing gives the best accuracy when the parameter is between 10 to 20. As the value of the parameter is increased the accuracy falls rapidly. Thus, the parameter should be set to a minimum value to achieve better results and higher accuracy.

Conclusion:

Finally, we created a dashboard to observe the performance of clothing, food and retail companies. The following bar graph was obtained for the data set of food category with all the five brands. As seen, the most popular brand is Nike and our analysis is true with the fact as Nike is the most widely used brand in the USA. Secondly, we mostly customers were happy with the service given by Levis and most were unhappy by Zara.



Also, the customers talked mostly about Nike and its price and were pretty satisfied on an average.



Thus, we can say that there is a great benefit of listening to the voice of the customer which is the ability to tailor your products to your customers' needs. When you hear exactly what's being said, you can use that information to make improvements. Companies invest a lot in customer delegation, and sometimes it may feel like one critical review could bring your company to its knees.

Appendix:

Code for finding accuracy:

```
for i in range(1,100,3):
    text_clf = Pipeline([('vect', CountVectorizer()),('tfidf', TfidfTransformer()),('clf', MultinomialNB(alpha=i)) ])
    text_clf = text_clf.fit(df['Text'][:int(len(df)*train_split)], df['Tag'][:int(len(df)*train_split)])
    predicted = text_clf.predict(df['Text'][int(len(df)*train_split)+1:])
    print i
    result.append({"parameter":i,"accuracy":metrics.accuracy_score(df['Tag'][int(len(df)*train_split)+1:], predicted)*100 })
```

Dashboard