FACULTY OF INFORMATICS
TECHNICAL UNIVERSITY OF MUNICH

Application Project

Sentiment Analysis on developers' reaction to Change in API

Project Report

Authors: Mainak Ghosh
Muhammad Zeeshan
Adviser: MSc. David Soto Setzke
Supervisor: Prof. Dr. Helmut Krcmar

31 July 2019

Contents

Abstract

In the era of digitalization and IoT, there are different APIs coming up so that developers can implement business functionalities more efficiently. So, it is worthwhile to analyze the sentiment of developers over the changes in APIs. It will help API providers to figure out which APIs, developers think are useful and why, despite complications, maintenance overhead, and risks. The following sections of this report[1] offer important aspects of the APIs and the identification of the aforementioned issue. This observation will mitigate the communication gap between API providers and API users (developers) and be helpful for API providers to learn how developers react to API evolutions.

[1]Our code-base and data-set are available at https://github.com/mzkaramat/DeveloperReactionToAPIChanges.git

1 Motivation

Monolithic infrastructures and applications that have powered the businesses over the past few decades are modularizing the business features based on independent and reusable micro-services, giving birth to API. Consequently, developers can focus on their own utility functions, which need to access other processes through APIs. Besides this, there are several reasons for APIs being popular in software development industry. A new breed of third party APIs frees developers from getting stuck to any specific platform, rather they can bring their features into the market efficiently. APIs are smarter, more generic than building own SaaS solution and reinventing the wheel again. Other than Salesforce, Amazon, Facebook, Twitter, there are promising third party APIs such as Strip, Plaid for payment connectivity, Twilio for telephony, Factual for location based data. Menlo's portfolio company Signifyd offers fraud analysis as an API. So, this horizon is interesting and revenue generator for entrepreneurs and investors. Salesforce reportedly generates 50 percent of its revenue through API, eBay nearly 60 percent *(Matt Murphy, Steve Sloane [2016])*. These motivate developers to get into enterprise-oriented technologies such as SaaS, big data, micro-services.

Besides this fruitful result, there are certain risk and challenges which developers must consider while accessing APIs. Once developer uses third party APIs for a business solution, then developer must consider security aspect of that solution, since security breach is no way acceptable. Client code developers usually evolve client application to keep the application up to date with API providers. If API is migrated, then the developers might face problem integrating the system, because API providers have little knowledge of how client code developers collaborate with the API. There are other complications as well such as if the platforms (e.g. Facebook, Twitter) on which developers build APIs, are out of service, then the business features will not work. Sometimes improper documentation misguides the client code developers and cause confusion in developers' community which leads to negative reactions to those APIs. It has been observed that adding new method causes more questions to developers *(Wang et al. [2014])*. Another aspect is that if any API provided by Facebook, Twitter gets closed, then all the developments accessing that API go in vain and developers must put extra effort to get the system working. So, there are certain complications which developers have to face while accessing API services from development platforms of Facebook, Twitter etc. We can see one-real example in Twitter API platform. Twitter launched their APIs in 2006 and initially every API used to have just basic authentication, so all the third-party development certainly would use that basic authentication. But after four or five years, Twitter brought the inevitable changes in their authentication which being OAuth protocol. This decision made the developer's life hard and hampered the smooth relation between developers and Twitter API.

So it is worthwhile to analysis the sentiment of developers over the APIs.

2 Introduction

An API can be changed across different versions such as response format can be changed, resource URL can be changed (e.g. the domain name of Twitter changed from api.twitter.com/1 to api.twitter.com/1.1.), response format is deleted (e.g. XML is not supported in Twitter API v1.1.), method name can be changed (e.g. this is a practice from Twitter, Blogger, MusicBranz, FriendFeed, Yelp and NYT Article Search) *(Wang et al. [2014])*. In this project, we addressed three research questions. By answering those questions, we figured out that for what type of changes developers give positive sentiment or negative sentiment.

3 Methodology

This section defines reserach approach, methods for developers' sentiment analysis over APIs along with some theoretical background about machine learning model.

3.1 Research Approach

In this project, we have used different tools and methods to find out answers for research questions along with logical explanations. Here we dive into details about our process for answering research questions.

3.1.1 Research Question 1

What kind of API do the developers give positive or negative sentiment for?

In this part, we found out for which type of APIs developers give positive or negative response. It was observed that TextBlob is better than the SentiWordNet approach for more accurate prediction *(Hasan et al. [2018])*. So, we used TextBlob library, which is a python-based library for text processing and it uses NLTK for natural language processing[2]. The approach behind this task was followed as:

- We pre-processed the sentences first. This step includes removal of stop words, converting upper case letters to lower case, and using Stemmer to get cleaned words out of the sentence.
- Next, we used Textblob to identify that for which sentences related to API, the sentiment is positive and for which sentences sentiment is negative. This helped in labeling the data which was used for training data and validation set.

[2]TextBlob, 2017, https://textblob.readthedocs.io/en/dev/

3.1.2 Research Question 2

What leads the developers to give positive reactions for some APIs and negative reactions for others?

It's an important factor for the project where we figured out why developer gives positive feed-backs for some APIs and negative feed-backs for others. We looked for the difference in sentences between these two sentiment levels of APIs. Then we found out different aspects of these differences. This finding was useful for training the machine learning model which was responsible to predict important aspects of the developer's sentiment for the APIs. we used Term frequency-Inverse Document frequency (TF-IDF)[3], which is an efficient approach, to understand significance of different words across multiple sentences in the dataset. This guided us to identify the set of words due to which an API is considered as positive and another set of words which causes an API to be treated as negative.

3.1.3 Research Question 3

How Does our model perform in real-time survey?

We conducted a survey across developers' communities to gather their sentiments on some APIs. Post that, we analyzed their responses and figured out the important clauses for their sentiments. Then we compared the important aspects which we got from survey analysis with the ones we got from the machine learning model.

3.2 Methods

This section defines different methods which were used in this project.

3.2.1 TF-IDF Representation

TF-IDF stands for Term Frequency — Inverse Document Frequency.

Term Frequency (TF) calculates frequency of a word in each document of the corpus. Mathematically it is ratio of number of times the word occurs in a document and total number of words in that document.
Inverse Document Frequency(IDF) calculates log of ratio of the total number of documents and number of documents wherein the word is present.

Multiplying TF and IDF, we get TF-IDF score for a word in a document. This way we represent each document as a vector having dimension equals to number of distinct words in whole corpus. In this project, we used TfidfVectorizer from python based sklearn package[4].

[3]TFIDF, WikiPedia, https://en.wikipedia.org/wiki/Tf%E2%80%93idf
[4]TfidfVectorizer-https://scikit-learn.org/stable/modules/generated/sklearn.feature$_e$*xtraction.text.TfidfVectorizer.html*

3.2.2 Machine Learning Model

There are certain machine learning techniques such as Naïve Bayes Classifiers and Support Vector Machines(SVM) can be used for sentiment classifications. *(Gupta et al. [2017])* report that SVM has better average accuracy than Naïve Bayes Classifiers.

Support Vector Machine (SVM):

In binary classification, we find a hyperplane in N dimensional data space, where N is number of unique words in whole corpus in our case, to distinctly classify the points such as all the positive comments lie on one side of the plane and negative ones lie on the other side.

Support vector machine algorithm tries to maximize the gap between margin of these two classes so that new point lies in the proper side of the hyper-plane with higher probability.

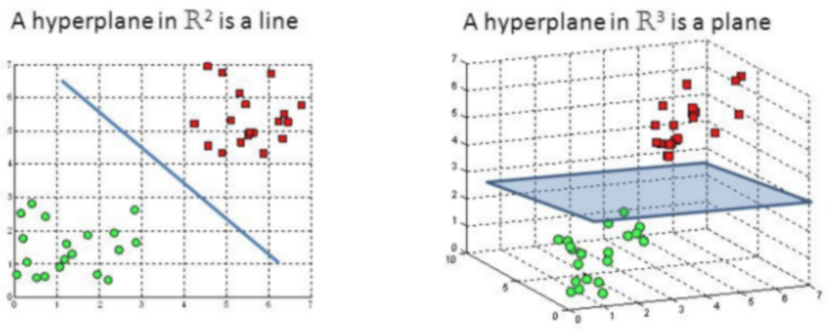


Figure 1. Hyperplanes in 2D and 3D feature space
Source: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

We used SVM with Stochastic Gradient Descent (SGD) optimizer, which is available in sklearn package[5]. This method returns weights assigned to each feature of the model which played an important role to find out important aspects out of the comments.

[5]SVM with SGD-https://scikit-learn.org/stable/modules/generated/sklearn.linear$_model.SGDClassifier.html$

API Provider	Number of Posts
facebook-graph-api	32777
winapi	31986
asp.net-web-api	29170
twitter-api	27566
google-api	23376

Table 1: Tabular representation of data-set statistics

4 Applied Work

This section talks about dataset, data analysis, and different experiments.

4.1 Data Collection

We collected data from StackOverflow, Twitter, Reddit, Ycombinator. Nowadays, developers discuss in these platforms, so these platforms are good for capturing sentiment and opinion. More over StackOverflow is very popular crowd-sourced media for developers *(Wang et al. [2014])*. Data used for this case-study mainly comprise of natural language posts regarding questions & answers being raised about performance, functionality, issues, and resolutions of APIs. Data were aggregated from different API providers from the aforementioned data sources and then we chose the APIs for further analysis based on the number of posts per API provider. Most popular API's providers are showcased above in table 1.

For these API providers, data were mined accessing a Google big query instance with data dumped from StackOverflow and stored in the comma-separated format (CSV) with the following schema (post title, post body, creation date, community score, tags, answers count, comment count) for further analysis and machine learning model generation. Further data processing and analysis are discussed in upcoming chapters,

4.2 Data Analysis and Preprocessing

After collecting data about the most discussed APIs, extensive data analysis is done for better understanding of the data distributions and underlying patterns to build model. For data exploration, R and python were used for discovering underlying patterns hidden in the aforementioned collected unstructured datasets.
Natural language processing steps were performed to convert data (which is in unstructured format) into a structured numerical format which could be further used for machine learning modelling. For that, pre-processing steps such as stemming, stop word removal, punctuation removal *(Friedman [2000])*, etc. were performed on the unstructured data-set. Stemming is used to convert

the words to their origin, stop word removal algorithm removes unnecessary commonly used words, since these words don't add any value for the sentiment analysis, punctuation removal removes punctuation as these don't add value for the natural language processing, and tokenization process splits up sentences into standard unit of text which is word in our case. The data processing pipeline is demonstrated in below diagram,

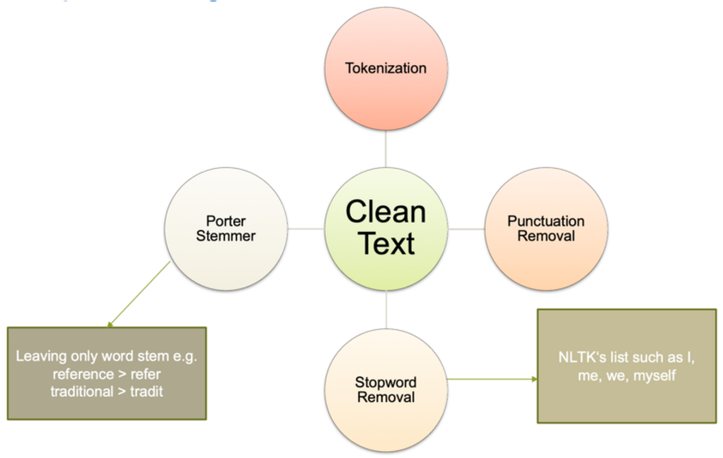

Tokenization
Porter Stemmer
Clean Text
Punctuation Removal
Leaving only word stem e.g. reference > refer traditional > tradit
Stopword Removal
NLTK's list such as I, me, we, myself

Figure 2. Data Pre-Procssing Pipeline

Since the data were unlabeled and these did not contain any meta-data about API provider, sentiment labelling of the data were performed by transfer learning and meta-data were collected using the following three approaches,

1. Using versions displayed in post URLs e.g. `https://api.twitter.com/1.1/geo/id/:place_id.json`

2. From the history log of the API being maintained by the API vendor

3. Using regular expressions to classify API endpoints e.g. `https://api.twitter.com/1.1/geo/id/:place`
   twitter Geo API

Each comment is assigned a sentiment score based on the title and body of the comment. TextBlob, python based package for sentiment classification, is used for each post to assign sentiment score. Post that, data were further classified into emotions such as trust, surprise, sadness, positive, negative, joy, fear, disgust, anticipation, anger.
After standardizing and normalizing text data from unstructured format to standard format, explorative data analysis is performed to understand data distribution and underlying patterns, starting from simple analytics including numbers of posts raised among various versions of APIs versus the timeline of the posts for analyzing underlying trends. For the selected API providers, the distribution of posts among various versions of APIs has been found to be uniformly distributed and the same goes for the number of posts raised when

analyzed against the timeline. The pattern in length of posts versus versions of API and time-line of the posts could also provide insightful information. These distributions were checked as to know the existence of uniform and identical data distribution which are required for most of the machine learning approaches *(Provost [2000])*. For Twitter API, results of the initial data modelling are shown below, with the first top left graph showing the distribution of words lengths among three versions of twitter API, while the second graph on the top right side showing the number of posts raised for twitter API versus the timeline of the posts. The third graphs on the bottom left, depicts the count of the posts concerning the version of the API while the last graph on the bottom right shows the aggregated count of the posts raised versus the month of the post creation date.

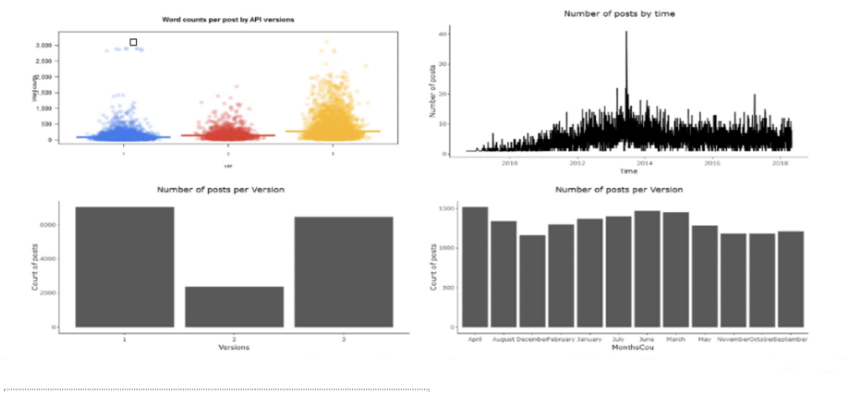


Figure 3. Explorative Data Analysis

The same initial explorative data analysis is performed for the remaining chosen API providers for confirming the assumption of uniform and identical data distribution.

4.3 Clustering Machine Learning modelling

For deeply understanding the underlying data patterns, unsupervised and supervised machine learning were applied on the data. Since the transformed data, which were generated by the previous processing steps, also contained labels of specific API version and sentiments, supervised machine learning is applied for finding the arguments for reasoning the positive and negative sentiment of the post from a developer point of view. Various supervised machine learning algorithms' performance have been bench-marked based on the Hinge loss criteria. SVM classifier performs drastically better than another state of the art supervised machine learning models in the cross-validation *(Hsu et al. [2003])*. In order to build SVM, first textual data were converted into numerical matrix using TF-IDF bag of words method, since SVM needs input data in the numerical format.

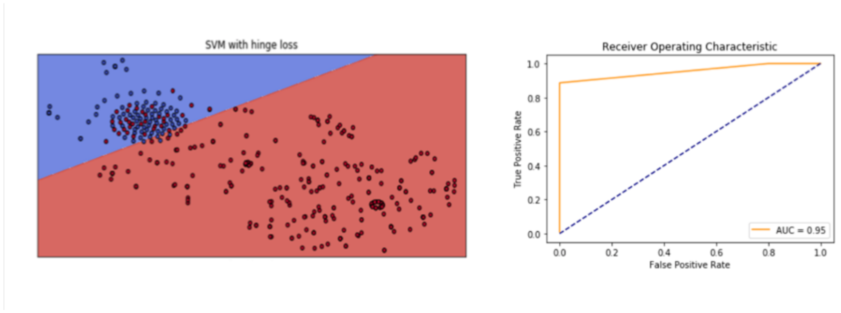


Figure 4. Machine Learning Model Results

To avoid over-fitting and check the performance of the model, overall data were split into Training Data: 60%, Validation Data: 20%, and Test Data: 20% ratio. Stochastic gradient descent is used as optimization algorithm.
In figure 4, model performance is shown in terms of hyper-plane and ROC curve. Furthermore, for finding the interpretation of the model's parameters for positive and negative sentiment, weight matrix of the SVM is scanned. These weights are associated with the clauses for positive and negative sentiments. Following are the results for the model interpretation of positive sentiment in diagram below,

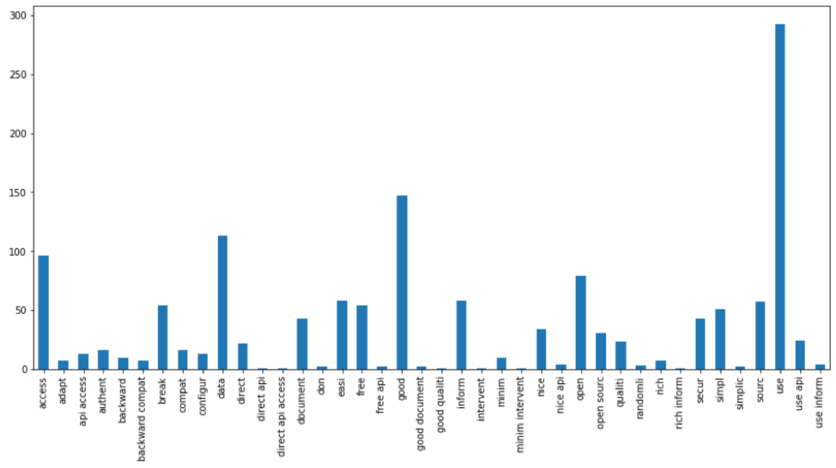


Figure 5. Positive Sentiments Factors

While the model interpretation results for negative sentiment for developer's opinion are shown below,

For finding the further insights governing the developer's opinion about the API connotation, unsupervised machine learning modelling is also applied including such as dimensionalality reduction and k-means clustering *(Hartigan and Wong [1979])*. The objective is to pull the similar APIs closer and push

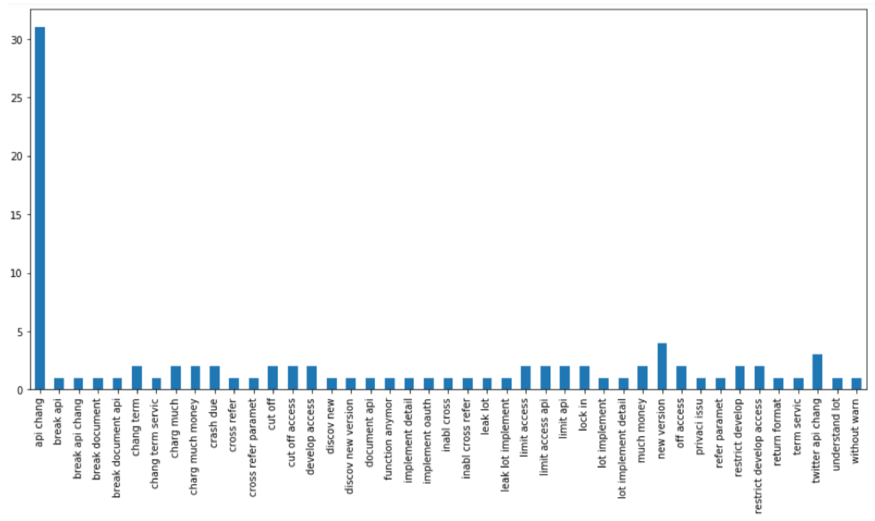


Figure 6. Negative Sentiments Factors

away the opposite sentiment based on the polarity score. The results of K-means clustering algorithm and t-distributed Stochastic Neighbor Embedding (t-SNE) *(Maaten and Hinton [2008])* for visualization are shown below:

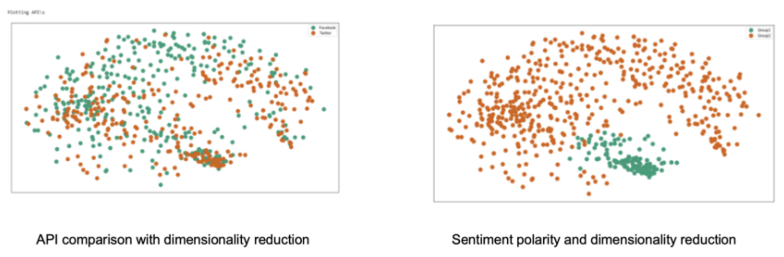


Figure 7. Unsupervised Machine Learning Results

Furthermore, post with API version labelled from the previous step has also been assigned a sentiment score for following dimensions such as trust, surprise, sadness, positive, negative, joy, fear, disgust, anticipation, and anger. various versions of API are visualized on spider chart for ranking API on several sentiment scores. Most variance is observed in positive and negative sentiment while for other sentiments, there is not enough variance observed. That's why opinion is only considered for positive and negative sentiments. Results of the spider chart visualization along various sentiments are visualized in diagram below alongside the most influential words affecting that sentiment on the right side:

After performing these analysis and machine learning modelling steps, these steps have been made reproducible with the help of self-service user-centric web

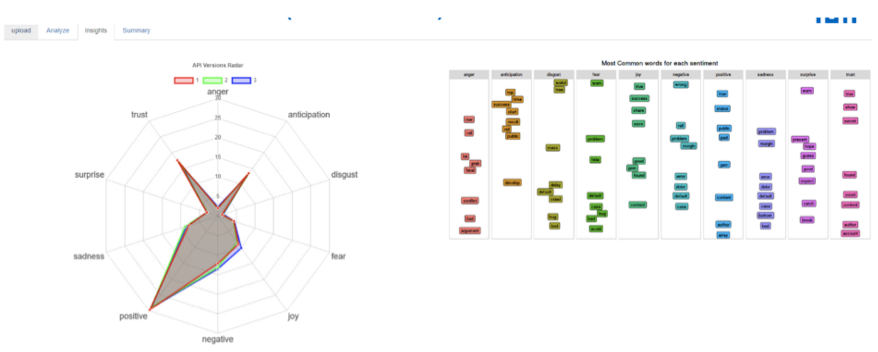


Figure 8. Sentiment Classification Spider Chart Visualization

application which takes input as the structured comma-separated formatted data form, then applies data transformation, builds model and depicts results which could be exported into PDF for record purposes and further analysis. This self-service user interface is further discussed in the upcoming chapter.

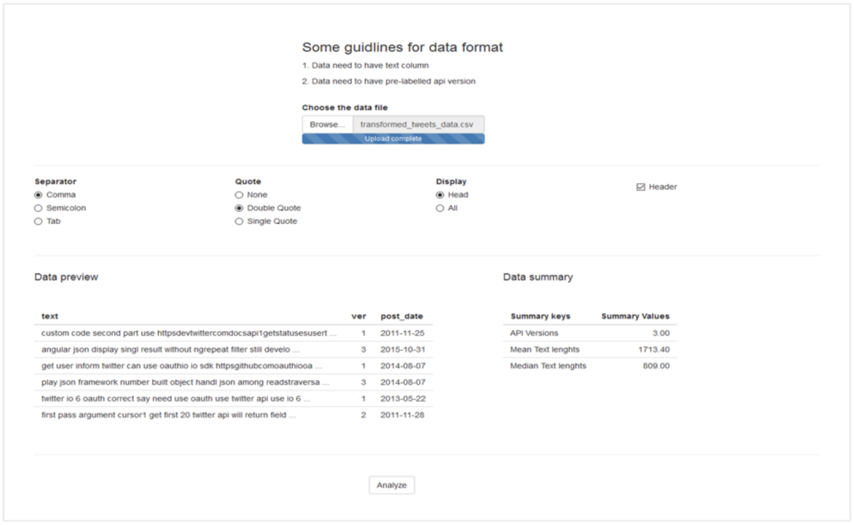
4.4 Self-Service UI/UX

To make the results as reproducible as possible, the data transformations including natural language processing algorithm i.e. stemming, stop words removal, punctuations removal, tokenization, bag of words are streamlined into a data pipeline which expects input to be a standardized unstructured format in the schema(post title, post body, creation date, community score, tags, answers count, comment count, API-version). User journey begins by uploading data into the portal as shown in diagram below,

Upon uploading the data, the user interface shows basic statistics about data including the types of columns and their frequency. The developed platform is flexible enough to handle variations in the input data including the separator (comma, semicolon, tab), quotes, display and headers in the uploaded data. After the user is satisfied with the Data options, user can proceed with the Analyze option. Data transformations discussed in the previous chapter are applied on the uploaded data, and the results of the explorative data analysis are visualized as discussed in previous sections. Figure in the below diagram is the glimpse of the visualization.

Besides these functionalities, the user can look into the insights about the sentiment score of anger, trust, surprise, sadness, positive, negative, joy, fear, disgust, and anticipation along with the versions of the API in the format of the spider chart as shown in diagram below,

Finally, machine learning model interpreted most influential words affecting the sentiment are also visualized on explorative data analysis page on the top right for observing more insights about the developer's opinion about the factors affecting the API ranking. On the bottom right, the aggregated sentiment, grouped by API version, is also visualized to show the improving or downgrad-


Some guidlines for data format

1. Data need to have text column
2. Data need to have pre-labelled api version

Choose the data file

transformed_tweets_data.csv

Separator: [✓ Comma] Semicolon Tab

Quote: None [✓ Double Quote] Single Quote

Display: [✓ Head] All

[✓ Header]

Data preview

text	ver	post_date
custom code second part use httpsdevtwittercomdocsapi1getstatusesusert ...	1	2011-11-25
angular json display singl result without ngrepeat filter still develo ...	3	2015-10-31
get user inform twitter can use oauthio io sdk httpsgithubcomoauthiooa ...	1	2014-08-07
play json framework number built object handl json among readstraversa ...	3	2014-08-07
twitter io 6 oauth correct say need use oauth use twitter api use io 6 ...	1	2013-05-22
first pass argument cursor1 get first 20 twitter api will return field ...	2	2011-11-28

Data summary

Summary keys	Summary Values
API Versions	3.00
Mean Text lenghts	1713.40
Median Text lenghts	809.00

Figure 9. Self-Service Data Upload UI

ing trend of the performance of the API with new releases of the API. R shiny dashboard [6] is used for building up the user interface, while R ggplot [7] and plotly [8] packages were used for explorative data analysis. For data transformation, dplyr [9], plyr [10] libraries were used for R and TextBlob [11] combined with NLTK were used for natural language processing algorithms in the back-end. The web app is deployed on the shiny-apps server while the machine learning models are deployed on the Azure virtual machine as a restful endpoint and is communicating with the shiny app over the HTTP protocol.

4.5 Results and Insights

Based on the methodological research described in the previous sections, data were analyzed, transformed, and visualized to find out the important factors affecting the sentiment of API and version-ed API both. Following are the results in diagram below,

Combining it with the aggregated sentiment results as depicted in the diagram below,

following patterns are observed for developer's opinions about the API's ranking and factors impacting the developer's opinion.

[6]https://shiny.rstudio.com
[7]https://www.statmethods.net/advgraphs/ggplot2.html
[8]https://plot.ly
[9]https://dplyr.tidyverse.org
[10]https://github.com/sampotts/plyr
[11]https://textblob.readthedocs.io/en/dev/

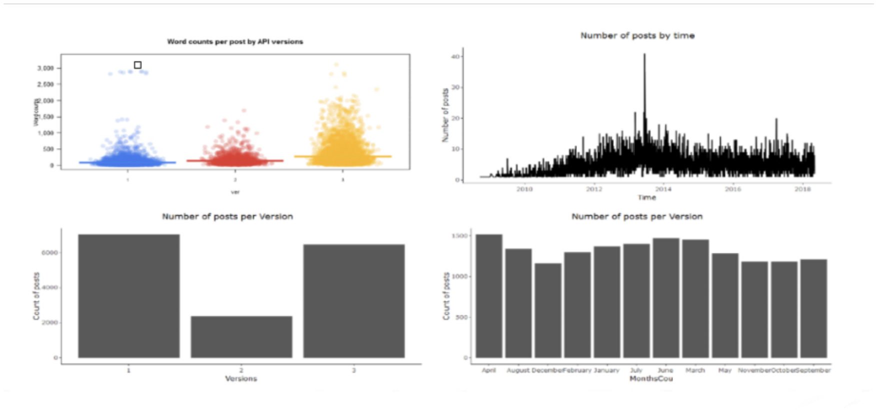


Figure 10. Self-Service Data Exploration UI

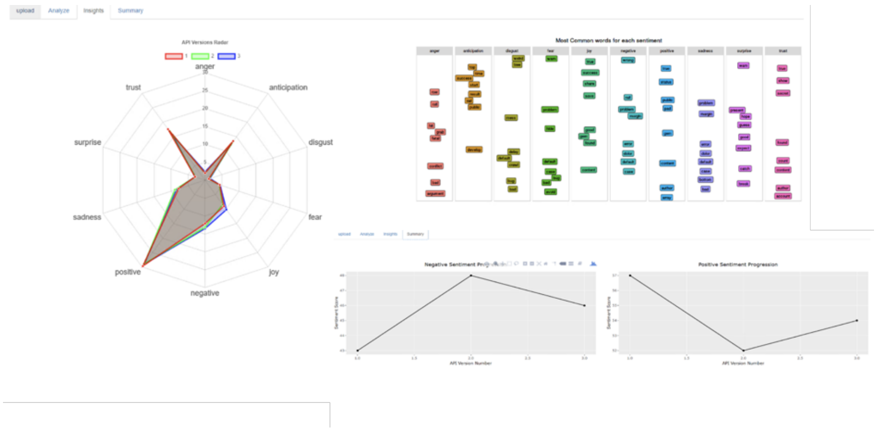

API Versions Radar
anger
anticipation
disgust
fear
joy
negative
positive
sadness
surprise
trust
Most Common words for each sentiment
Negative Sentiment Progression
Positive Sentiment Progression

Figure 11. Self-Service Sentiment Visualization UI

Positive Opinion Patterns:

1. Deleted methods in new API versions
2. Less changes in API endpoints parameters
3. Longer discussions in the posts
4. Large lengths of the posts

Negative Opinion Patterns:

1. Less deleted endpoints or functionality
2. More changes in API endpoints parameters
3. Smaller discussion or no answers

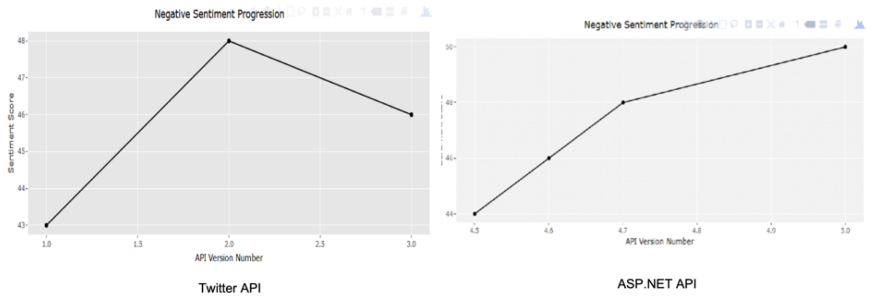


Figure 12. Sentiment Score across API Versions.

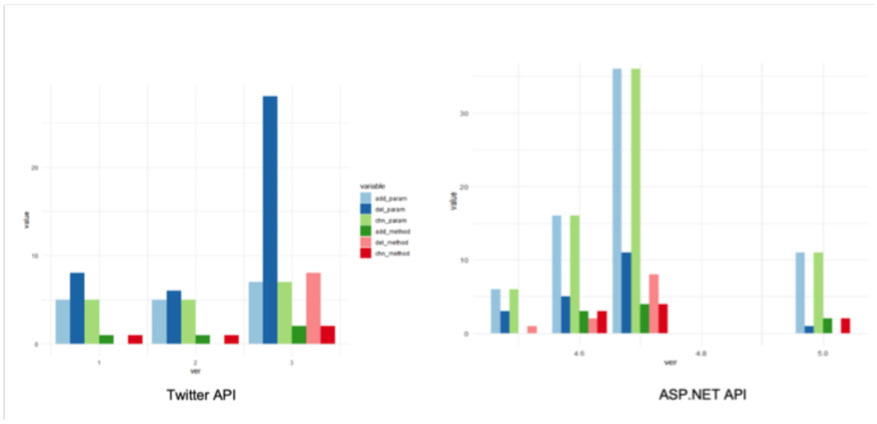


Figure 13. Methods Variation Aggregation across API Versions.

Interpretation of the results after applying supervised machine learning approach for developer opinion mining depicts that following are the factors affecting the sentiment of developer opinion about API performance:

Reason for positive sentiment:

1. Simplicity
2. Easier to use
3. Provide rich information while invoking the API
4. Quality
5. Good documentation about API
6. Free access to API
7. Backward compatibility across versions of API

8. Rich security

Reason for negative sentiment:

1. Change in response format
2. Lack of documentation or no documentation about what the API does
3. Limited access to API
4. Not functional anymore
5. Privacy issue
6. Change in terms of service

In the upcoming section, analysis of the applied work will be compared with the analysis of the results gathered from the survey.

5 Evaluation of Theoretical Insights

For comparing our applied results within the developers' communities, we have conducted a survey, in which we asked developers opinion about their experience with the APIs.

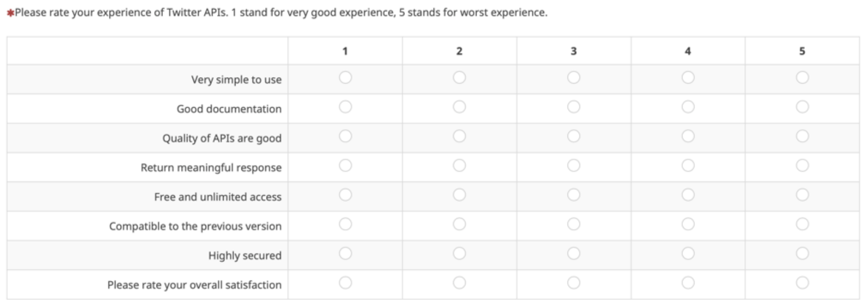
5.1 Survey Approach

Our survey questions were pragmatic, and we tried asking developers' about their experience with software development and then their rating for various factors affecting opinion about the APIs they've worked with. Our focus is on following API's while asking for the developer's opinion,

1. Google API
2. Twitter API
3. Facebook API
4. WinAPI

Based on the developers' experiences with the above mentioned API, we asked them to rate the APIs based on various factors from simplicity to security of API. One sample question which we asked for each of aforementioned API from the developer is below,

50 responses were obtained after sharing the survey in the developers' community from LinkedIn to Facebook groups. In the upcoming chapter, results of the surveys are shown, and insights will be drawn from it.



*Please rate your experience of Twitter APIs. 1 stand for very good experience, 5 stands for worst experience.

	1	2	3	4	5
Very simple to use	○	○	○	○	○
Good documentation	○	○	○	○	○
Quality of APIs are good	○	○	○	○	○
Return meaningful response	○	○	○	○	○
Free and unlimited access	○	○	○	○	○
Compatible to the previous version	○	○	○	○	○
Highly secured	○	○	○	○	○
Please rate your overall satisfaction	○	○	○	○	○

Figure 14. Sample survey Questions.

5.2 Analysis of results

Out of the 50 surveys responses, overall feedback for all the mentioned APIs except WinAPI is quite representative. So, the analysis for Google and twitter API is further performed in detail. For Google API, the comparison of developer's year of experience with the API rating is visualized in below diagram,

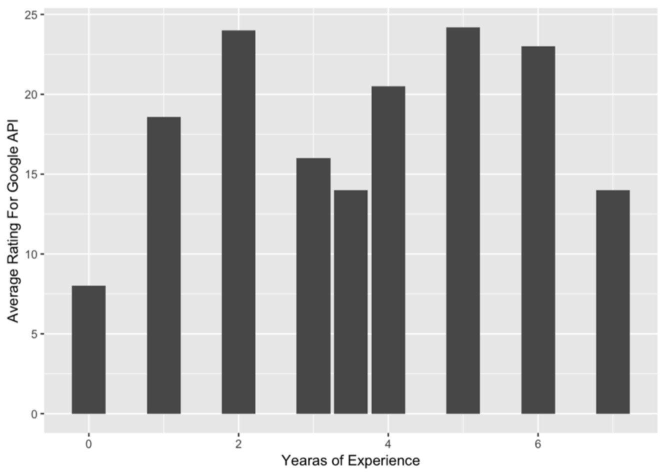


Figure 15. Google API usage

While for the twitter API, the developers' years of experience along with developers' API rating for APIs is visualized below. This shows that developers with a higher year of experience tend to rate higher their experience with API.

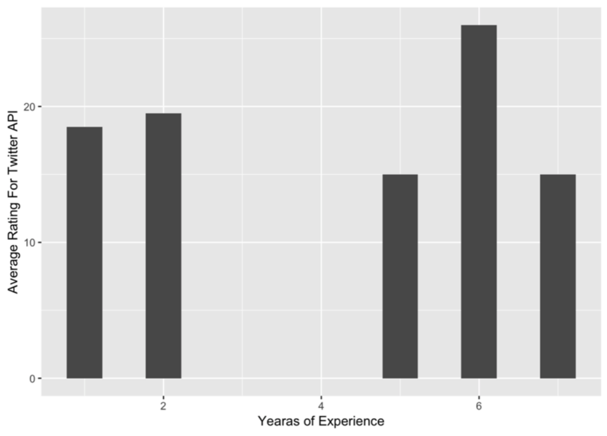


Figure 16. Twitter API usage

Further analyzing the results have shown that the important factors for the positive feedback for **Google API** are,

- Simple to use
- Highly Secured
- Free Access
- Backwards Compatible
- Return meaningful response
- Good quality

While the same for **Twitter API** are,

- Good documentation
- Simple to use
- Backward compatible
- Highly secured
- Free access

And the same for **Facebook API** are,

- Simple to use
- Good document
- Highly secured

So, combining them, we get an unified list of importance factors from the surveys:

- Simple to use
- Good documentation
- Highly secured
- Free access
- Backward compatible

In the next section, empirical results from the survey will be correlated with the applied results obtained from the data analysis of the developer's posts analysis.

5.3 Correlations with Theoretical Insights

Our machine learning model has found the important aspects which cause positive sentiment classification and negative sentiment classification from the insights (Red stands for negative classification and blue stands for positive classification). They're visualized in below diagram:

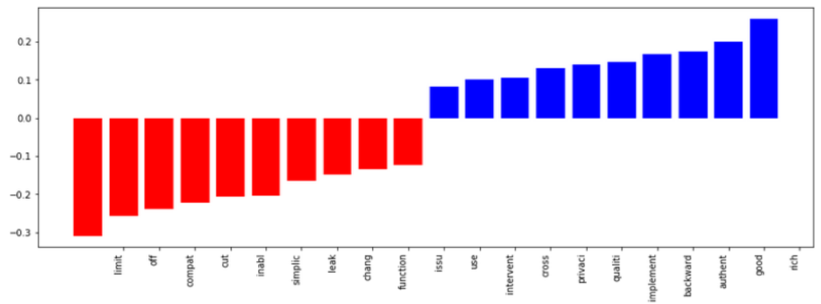


Figure 17. Model Interpretation Results.

This diagram yields the following important factors sorted by their importance affecting the developer's opinion.

<u>Positive factors:</u>

- Backward compatibility
- Quality of the APIs
- Privacy and Authentication (same as security)
- Good documentation about APIs

- Rich information provided by APIs

Negative factors:

- Limited access to API
- Not simple
- Change in API such as response, argument parameter etc.
- Not functioning
- Less compatible with the previous version

Comparing them with the unified results obtained from our empirical surveys results, they are found to be quite similar.

6 Summary and Outlook

Comparing empirical surveys results with the applied data analysis results, it confirms the similarity between these two results. This application project analyzed developers' sentiment about their experience with third party APIs. Backward compatibility, quality, privacy, authentication, quality documentation, and intuitive interface yield positive sentiment in the developer's opinion about APIs. While limited access, complexity, changes in response or calling parameters across versions, broken functionality, and non compatibility with previous versions yield negative sentiment in the developer's opinion about API. This work could be further integrated into API release cycles by various vendors for providing better client side development experience with API changes.

7 Acknowledgement

It was indeed a pleasure to work on such an interesting topic. We are very much thankful to Information System chair to provide us this idea, especially grateful to MSc. David Soto Setzke for guiding and motivating us throughout the project. Lastly, we would like to thank all the participants who took part in the survey and helped us to draw some meaningful insights.

References

Carol Friedman. A broad-coverage natural language processing system. In *Proceedings of the AMIA Symposium*, page 270. American Medical Informatics Association, 2000.

Bhumika Gupta, Monika Negi, Kanika Vishwakarma, Goldi Rawat, and Priyanka Badhani. Study of twitter sentiment analysis using machine learning algorithms on python. *International Journal of Computer Applications*, 165(9):29–34, May 2017. ISSN 0975-8887. doi: 10.5120/ijca2017914022. URL `http://www.ijcaonline.org/archives/volume165/number9/27604-2017914022`.

John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

Ali Hasan, Sana Moin, Ahmad Karim, and Shahaboddin Shamshirband. Machine learning-based sentimental analysis for twitter accounts. *Mathematical and Computational Applications*, 23:11, 02 2018. doi: 10.3390/mca23010011.

Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification. 2003.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Matt Murphy, Steve Sloane. The Rise of APIs. `https://techcrunch.com/2016/05/21/the-rise-of-apis/`, 2016.

Foster Provost. Machine learning from imbalanced data sets 101. In *Proceedings of the AAAI'2000 workshop on imbalanced data sets*, volume 68, pages 1–3. AAAI Press, 2000.

Shaohua Wang, Iman Keivanloo, and Ying Zou. How do developers react to restful api evolution? In Xavier Franch, Aditya K. Ghose, Grace A. Lewis, and Sami Bhiri, editors, *Service-Oriented Computing*, pages 245–259, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg. ISBN 978-3-662-45391-9.