

# **DAB 402 – CAPSTONE PROJECT**

## **PROJECT TOPIC**

# **CUSTOMER SENTIMENT ANALYSIS IN SHORT-TERM RENTAL INDUSTRY**

**STUDENT NAME: TRANG BUI**

**STUDENT ID: [W0753523](#)**

# TABLE OF CONTENTS

- I. Abstract
- II. Case Study Introduction
- III. Sentiment Analysis
- IV. Natural Language Processing (NLP)
- V. Data
- VI. System Architecture
- VII. Methodology
  - 1. Data Sampling
  - 2. Data Cleaning (Text Processing)
  - 3. Feature Engineering
    - i. Count based method.
      - a. BOW
      - b. TF-IDF
    - ii. Word2Vec
      - a. CBOW
      - b. Skip-Gram
  - 4. Approach 1: Supervised Machine Learning for classification problems
  - 5. Approach 2: Deep Learning Keras Model
- VIII. Evaluation Method & Results
- IX. Discussion & Conclusion
- X. References

## **I. ABSTRACT**

The goal of this study is to apply the supervised machines learning and deep learning methods to mining customer reviews in short-term rental services. First, the raw data in human natural language is extracted from the website “<http://insideairbnb.com/>”. These raw data will be pre-processed using text processing techniques such as text normalization, tokenization, remove stop-words, stemming, lemmatization. Then, the cleaned data is applied features engineering methods like Bad of words, TF-IDF (Term Frequency – Inverse Document Frequency), Word2vec to extract the outstanding features for classifier model training. Machine learning and Deep learning models are built to categorize user feedback as positive, negative, and neutral; and to summarize the key words from reviews to help the manager easily and quickly understand their business issues.

## **II. CASE STUDY INTRODUCTION**

In the recent decade, homestay is a form of short-term rental business that has been strongly developed around the world. Anyone can easily run a small business with their own apartments or houses. Airbnb (Airbed and Breakfast) appeared in 2008 by two design students in San Francisco that helped connect tenants and renters around the world through a mobile app. All reservations, payments, and feedback are done through the “Airbnb” application. In addition, this application assists to collect the customers’ reviews and feedback about their experience after staying. Besides, almost new customers look at the reviews to find out a good and suitable accommodation for their booking. As the result, the place with a lot of good feedbacks can attract more prospective customers and vice versa. Hence, reviews from clients are very important in rental business. In order to run the business well, and attract the large number of customers every year, the hosts need to analyze the customer feedbacks to understand what the problems are with their listings and services, what the customers’ expectation are. Moreover, based on the reviews, Airbnb also plays a role of a supporter to help the hosts know what are expected in the region so that the hosts can improve their listings and services to drive a profitable business.

However, the feedback data is human natural language that is form of unstructured data. Data content is not organized, and contains lots of noise such as unnecessary words, emojis, symbols, weird characters, etc. How can businesses exploit, statistic and draw evaluation results that are negative or positive feedbacks? How can they determine the criteria that need improvement to satisfy most of customers' expectations about their products and services so that they drive their products and business development?

### **III. SENTIMENT ANALYSIS**

What is Sentiment Analysis?

Sentiment analysis is the process of using algorithms to analyze positive or negative thoughts, beliefs, feelings, opinions, or trends expressed in human natural language. Emotional analysis is used in many industries to help harness customers' knowledge, emotions, and perspectives. Exploiting customers' emotions plays an important role in making decisions, making the right business strategy. For example, machine learning combined with artificial intelligence to analyze content in emails, messages, comments on social networks, etc. Accordingly, the business can determine the customer emotion. Track complaints about specific issues and better manage the customer journey experience.

Sentiment analysis is supported by NLP natural language processing technology and machine learning algorithm. Help computer systems and programs learn from big data and make predictions about emotion.

### **IV. NATURAL LANGUAGE PROCESSING (NLP)**

Natural language processing (NLP) can be a field of linguistics, computer science, and artificial intelligence involving human-language interactions; especially the way of programming computers to process and analyze large amounts of natural language data. Today, the application of Artificial Intelligence (AI) increasingly helps businesses solve many difficult problems. Natural

language processing (NLP) is a subset of AI used to analyze text to understand the meaning of human natural language. This is one of the best widely used solution to solve the customer behavior and sentiment analysis problems based on a big unstructured data.

Natural language processing is paramount in Computer Science. It has a multitude of useful applications in life as well as research. We can consider some applications of natural language processing such as:

- Voice recognition
- Speech synthesis
- Machine translation
- Retrieve information.
- Document classification
- Data mining and knowledge discovery
- Sentiment analysis

## V. DATA

Data used for this project is derived from “[Inside Airbnb](#)” website. In this project, we explore customer review data of all listings in Vancouver, Toronto, Ottawa, Montreal in Canada compiled until Feb-2021.

- The *main data* used for the client sentiment analysis in this project are the detailed review data files ([reviews.csv.gz](#)).
- The number of reviews in each raw data file:

Raw data (input files)	Nb of reviews
<a href="#">vancouver_reviews.csv</a>	147,936
<a href="#">toronto_reviews.csv</a>	424,060
<a href="#">ottawa_reviews.csv</a>	98,909
<a href="#">montreal_reviews.csv</a>	289,713

**Data term of use:**

[“Inside Airbnb”](#) is a dependent, non-commercial set of tools and data that allows everyone to explore how Airbnb is really being used in cities around the world. The data behind the Inside Airbnb is sourced from publicly available information from the Airbnb site”. This site is personally funded by Murray Cox, everyone can donate to contribute for data collection payment.

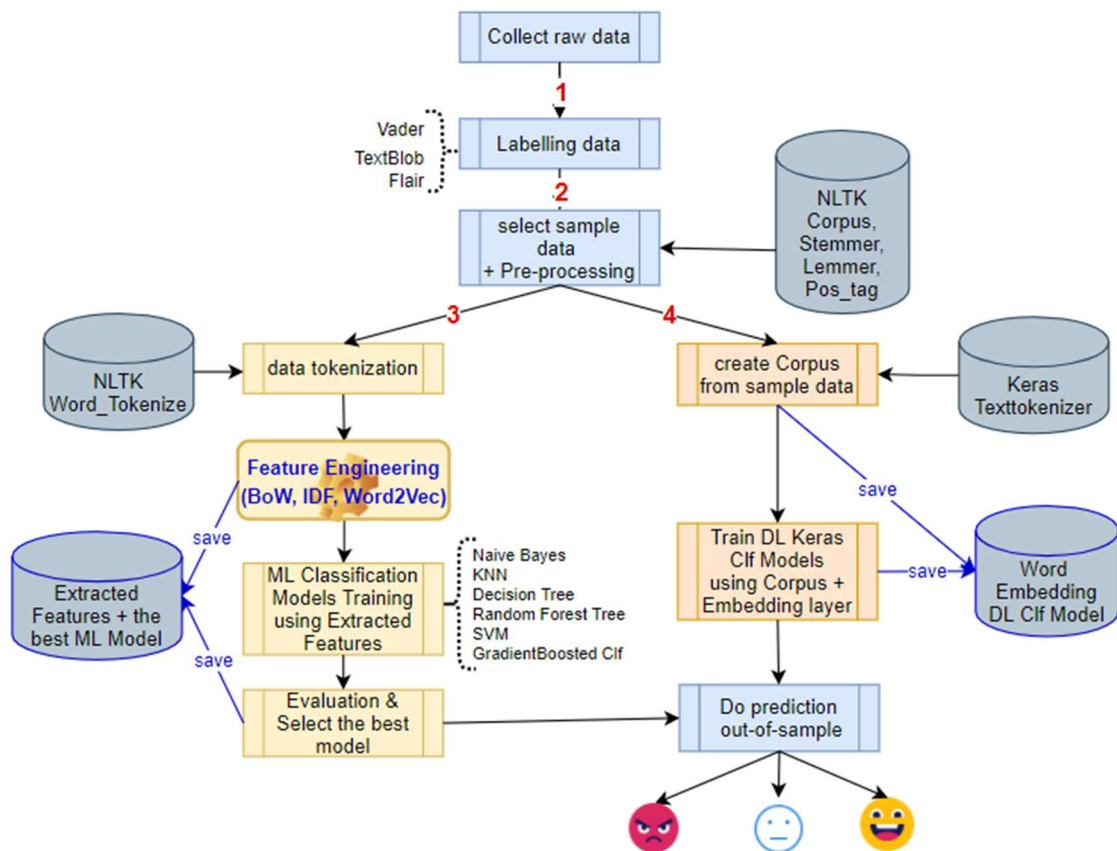
This site claimed that data contain no private information, all information are publicly display on Airbnb website. Moreover, they confirmed that data is “fair use” for non-commercial purposes and allow public analysis due to community benefit.

## VI. SYSTEM ARCHITECTURE

### 1. System Architecture

In this project, the proposed system architecture is shown in the figure:

- Input: Raw review data in human natural language
- Output: sentiment extracted from the review: POSITIVE, NEUTRAL, NEGATIVE










- **1** - collect raw data, and label sentiment state for each of review of raw data using pre-train models (Vader, TextBlob, Flair)
- **2** - select sample + text pre-processing
- **3** - **Approach 1**: train classification ML models with extracted features that are result from different feature engineering techniques such as BoW, TF-IDF, Word2Vec

(genism pre-trained model). Then, evaluate to choose the best ML model. Save the best model and corresponding extracted features.









- 4 - [Approach 2](#): train Deep Learning regression model with the first Embedding layer using the corpus created from sample data.

## 2. Processing Folders

- 4 notebooks are corresponding to 4 main state in the system architecture.

 data	File folder
 images	File folder
 1_SentimentAnalysis_Labelling_Using_Pretrained_Models	IPYNB File
 2_SentimentAnalysis_TextProcessing	IPYNB File
 3_SentimentAnalysis_FE_and_Clf_Models	IPYNB File
 4_SentimentAnalysis_KerasFE_LSTM	IPYNB File
 contractions	Python File

- The folder “data” is the place to store “csv files”, saved models, and saved word embeddings + extracted features.

 data	○ “buffer” is the place to store buffer data during text processing.
 input_output_csv	
 buffer	○ “cleaned_data” is the place to store sample data after text processing
 cleaned_data	
 raw_data	○ “raw_data” is the place to store raw review data.
 saved_embedding	
 saved_models	
 test	



## VII. METHODOLOGY

### 1. Data Sampling

- Approach 1:
  - It takes too long to train a large amount of review data, only 4105 reviews are extracted randomly from cleaned data, and used for ML classifier model training. In which, there are:

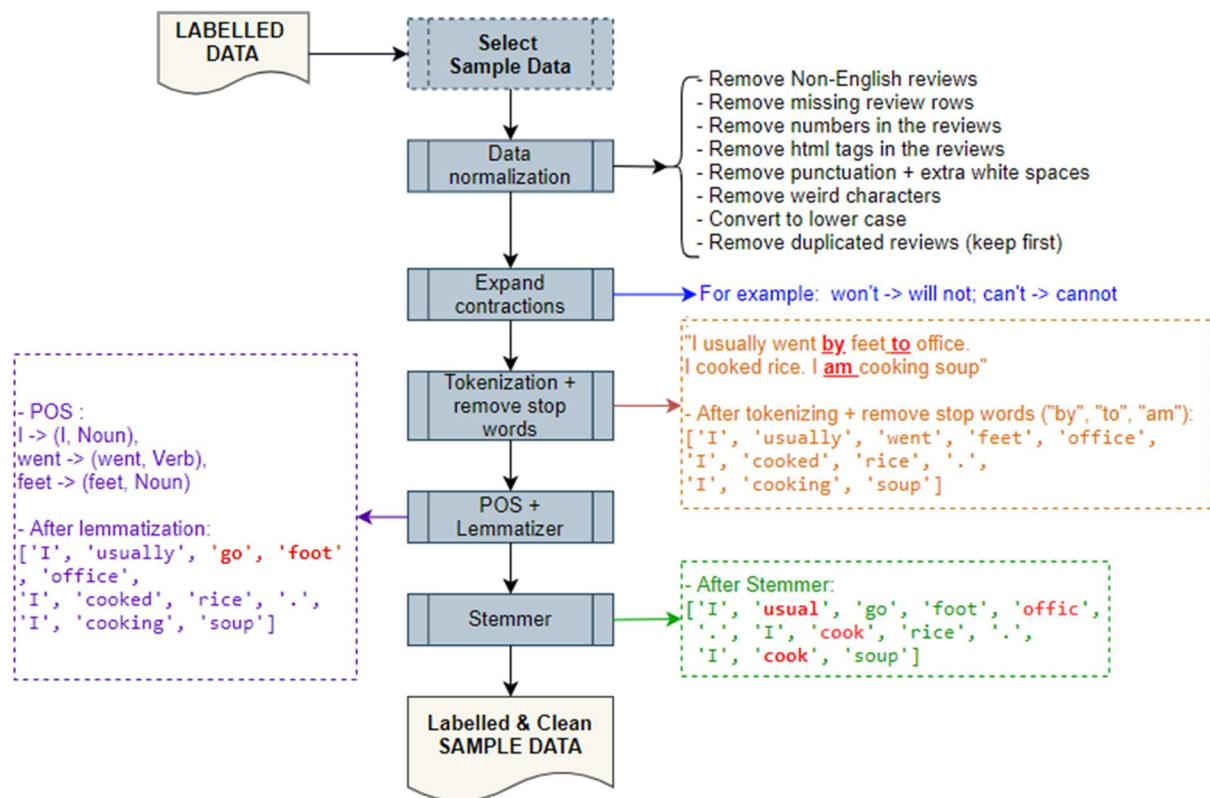
<b>sample_review.csv</b>	<b>Negative</b>	<b>Positive</b>	<b>Neutral</b>
<b>4105</b>	1744	1728	633

- Approach 2
  - To build a corpus (dictionary) with a big number of vocabularies, full cleaned data (9194 reviews) is used to create corpus and used for Deep Learning model training. In which, there are:

<b>sample_review_FULL.csv</b>	<b>Negative</b>	<b>Positive</b>	<b>Neutral</b>
<b>9194</b>	3984	3932	1278

## 2. Text Pre-Processing (data cleaning)

- Below is the text processing flow in this study:



- There are many ways to clean up as much noise as possible and pre-process textual raw data. It's depended on the data source to decide to what should be removed or normalized. For example: remove non-English reviews, remove missing data, remove number in the reviews, remove html tags or weird characters, remove duplicated reviews, convert to lower case... NLTK (Natural Language Toolkit) library provides robust text processing functions that are used mostly in this project. Besides, some other libraries such as: Beautiful Soup, Keras Text Preprocessing...
- Tokenization:** is the process of splitting a text into smaller units called tokens. Tokens can be a sentence, word, sub word, or even a character.
- Remove **stop words**: stop words are very common and often less important words. Hence, removing them is also a pre-processing step. This can be done explicitly by retaining only words in the document that are not in the stop word list. The stop word list is provided from NLTK library.

The list of stop words such as “by”, “to”, “am” are removed in below example:

```
sentence = "I usually went by feet to office. I cooked rice. I am cooking soup"

words = [word for word in nltk.word_tokenize(sentence) if word not in stopwords.words('english')]
print('remove stop words:', words)

remove stop words: ['I', 'usually', 'went', 'feet', 'office', '.', 'I', 'cooked', 'rice', '.', 'I', 'cooking', 'soup']
```

- Combination of **PoS** (part of speech) and **Lemmatization** is a better way to convert words into their basic form.

In below example, the word “went” is determined as a “verb” by POS\_TAG function. So, lemmatizing to convert it to basic form that is “go”, the same with “feet” => “foot”.

```
def get_wordnet_pos(word):
    tag = nltk.pos_tag([word])[0][1][0].upper()
    tag_dict = {"J": wordnet.ADJ,
                "N": wordnet.NOUN,
                "V": wordnet.VERB,
                "R": wordnet.ADV}
    return tag_dict.get(tag, wordnet.NOUN)

lemmatizer = WordNetLemmatizer()

sentence = "I usually went by feet to office. I cooked rice. I am cooking soup"

words = [word for word in nltk.word_tokenize(sentence) if word not in stopwords.words('english')]
print('remove stop words:', words)
lem_text = ([lemmatizer.lemmatize(w, get_wordnet_pos(w)) for w in words])
print('PoS + Lemmatize:', lem_text)

remove stop words: ['I', 'usually', 'went', 'feet', 'office', '.', 'I', 'cooked', 'rice', '.', 'I', 'cooking', 'soup']
PoS + Lemmatize: ['I', 'usually', 'go', 'foot', 'office', '.', 'I', 'cooked', 'rice', '.', 'I', 'cooking', 'soup']
```

- **Stemming** is a technique used to convert a word to its original form (root form) by extremely simply removing some characters at the end of the word.

- Usually -> usual
- Office => office
- Cooked/cooking => cook

```
sentence = "I usually went by feet to office. I cooked rice. I am cooking soup"

words = [word for word in nltk.word_tokenize(sentence) if word not in stopwords.words('english')]
print('remove stop words:', words)
lem_text = ([lemmatizer.lemmatize(w, get_wordnet_pos(w)) for w in words])
print('PoS + Lemmatize:', lem_text)
stem_text = ([stemmer.stem(i) for i in lem_text])
print('stemming: ', stem_text)

remove stop words: ['I', 'usually', 'went', 'feet', 'office', '.', 'I', 'cooked', 'rice', '.', 'I', 'cooking', 'soup']
PoS + Lemmatize: ['I', 'usually', 'go', 'foot', 'office', '.', 'I', 'cooked', 'rice', '.', 'I', 'cooking', 'soup']
stemming: ['I', 'usual', 'go', 'foot', 'offic', '.', 'I', 'cook', 'rice', '.', 'I', 'cook', 'soup']
```

This is data after text pre-processing:

comments (after text processing)	comments_orig
wonder locat iman make check easi patioth best part	What a wonderful location! Iman made checking in and out so easy. And that patio...the best part!
photoshop fake pictur	photoshopped fake pictures ...
tri tow car call airbnb heat issu somehow hop would ignor heat degre temperatur place run think shadi individu extrem bia start lawsuit base occur im disappoint stay end tomorrow howev trip cut short ownersscam artist	They tried to tow our car after we called Airbnb about the heating issue, they were somehow hoping we would just ignore having no heat in -15 degree temperature. This place is run by what I think are shady individuals that are extremely bias. I am starting a lawsuit based on what occurred. I'm very disappointed in our stay that should have ended tomorrow however, our trip was cut short because of the "owners"/scam artists.
host cancel reserv day arriv autom post	The host canceled this reservation 8 days before arrival. This is an automated posting.

### 3. Feature Engineering

Feature Engineering is an extremely important technique for Machine Learning algorithms, Deep Learning to process input that is in text form because they only understand input as numbers to perform classification or regression, etc.

Word Embedding is a common name of a group of special techniques in natural language processing that map a word or phrase in a lexicon to a real number vector. From one-dimensional space for each word to continuous vector space. Word vectors are represented by the embedding method expressing the semantics of words, from which we can realize the relationship between words (same, antonym, ...).

Some embedding techniques are used in the project:

- i. **Count based method:** this embedding technique does not create a semantic relationship between words in particular contexts.

- a. Bag of Words (BOW)

- List of words in the sentence or document are collected and put into corpus regardless of the order in which they appear. Documents with many common words will be related to each other. The representation of the text as a histogram allows us to see the occurrence frequency of the words.

For instance: 1 represent the appearance of the word in the sentence.

Document		beautiful	blue	love	sky	
0	The sky is blue and beautiful.	0	1	1	0	1
1	Love this blue and beautiful sky!	1	1	1	1	1

- Disadvantage: There are too many 0 in the bag of words. The size of bag is as big as the max length of a sentence in a document. This leads to consuming memory resource.

b. Term Frequency – Inverse Document Frequency (TF-IDF)

- This technique is based on the idea that the more a common word appears, the less important it is and the low its weight is. (Example: "the", "to", "of", etc.). Therefore, in addition to counting the appearance of a word in the text, it also calculates the weight of that word in the text.

Document		beautiful	blue	love	sky	
0	The sky is blue and beautiful.	0	0.58	0.58	0.00	0.58
1	Love this blue and beautiful sky!	1	0.45	0.45	0.63	0.45

- “Love” appears 1 time => highest weight.
- “Beautiful”, “blue”, “sky” appears 2 times => same weight

ii. **Word2Vec:** Word2vec use the relationship of a word and the words around it to simulate it in a multidimensional space. Word2vec is used to produce word embeddings. This model has 2 layers neural network that are trained to reconstruct linguistic context of words. There are some pre-trained Word2Vec models such as Gensim, Spacy, Google’s Word2Vec.

a. Continuous Bag of Words (CBOW)

- This method takes input as one or more context words and tries to predict the outcome from the output (from the target) through a simple neural layer. Thanks to the output error assessment with the target magnet in the form of a hot magnet, the model can adjust the weights, learning how to represent a vector for the target magnet.

	Document	
0	I like to eat fruit and cake.	Context (X): ['like', 'fruit'] -> Target (Y): eat
1	I love to drink tea and wine.	Context (X): ['eat', 'cake'] -> Target (Y): fruit
2	Strawberry, apple are a good fruits.	Context (X): ['love', 'tea'] -> Target (Y): drink
3	Beer and wine are not really good	Context (X): ['drink', 'wine'] -> Target (Y): tea
		Context (X): ['strawberry', 'good'] -> Target (Y): apple
		Context (X): ['apple', 'fruits'] -> Target (Y): good

#### b. Skip-gram

- In contrast with CBOW, skip-gram uses target word as the input variable and predict its neighbor words. The number of neighbor words are defined by window\_size parameter while creating skip-gram model.

Example can be seen here:

<https://www.tensorflow.org/tutorials/text/word2vec>

Window Size	Text	Skip-grams
2	[ The wide road shimmered ] in the hot sun.	wide, the wide, road wide, shimmered
	The [ wide road shimmered in the ] hot sun.	shimmered, wide shimmered, road shimmered, in shimmered, the
	The wide road shimmered in [ the hot sun ].	sun, the sun, hot
3	[ The wide road shimmered in ] the hot sun.	wide, the wide, road wide, shimmered wide, in
	[ The wide road shimmered in the hot ] sun.	shimmered, the shimmered, wide shimmered, road shimmered, in shimmered, the shimmered, hot
	The wide road shimmered [ in the hot sun ].	sun, in sun, the sun, hot

## 4. Supervised Machine Learning for classification problems

Machine learning is a branch of artificial intelligence that uses knowledge of statistical probability and linear algebra to research and develop techniques and to build programs that computers can learn, analyze existing data, and generate results for new data. Machine learning has many practical applications such as sentiment analysis, market analysis, fake credit card detection, medical diagnostics, object recognition, speech and word recognition. In this study, I applied some classifier ML models to see how they work for sentiment analysis problem. Moreover, I used Grid-Search to find to hyperparameters for each of ML models. The sample data have 4105 reviews. Training set contains 70% that includes 2873 reviews. Test set contains 30% that is 1232 reviews.

- i. Naïve Bayes
  - This is an algorithm to predict what is the probability of a data element belonging to a class. The Naïve Bayes algorithm is based on the Bayes theory.
- ii. KNN Classifier
  - This is a classification algorithm based on the k-nearest neighbors.
- iii. Decision Tree Classifier
  - This ML algorithm create a tree of questions to make a prediction. The tree will start at its foundation with a first Yes/No question. Based on the answer the tree will continue to ask new Yes/No questions following a specific branch.
- iv. Random Forest Tree
  - RFT are a large number of trees (same with decision tree), combined using averages at the end of the process. This is based on the idea that a single decision tree is a weak predictor.
- v. SVM
  - This is a supervised learning method used for classification problem. Each of sample data will be represented as a vector and mapped to a point in a space. The main idea is to find a hyperplane that divides partial data into n layers and the distance from the nearest data points to the plane is the farthest.
- vi. Gradient Boosted Classifier
  - This is a set of decision trees where trees are built successfully. This algorithm is quite similar to Random Forest Tree, however, the average results are combined along the way instead of at the end of the process like RFT.

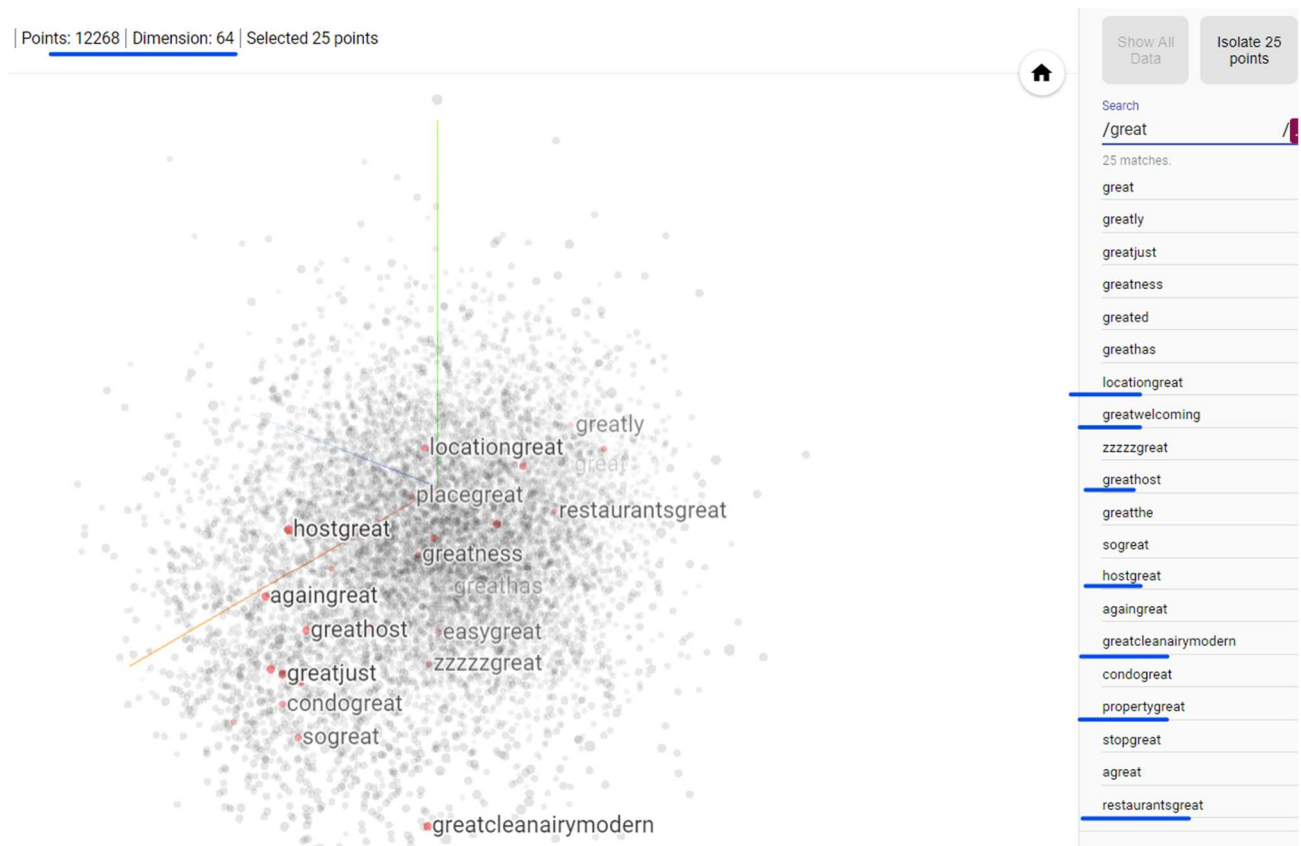
## 5. Deep Learning Keras Model

With this approach, I train word embedding using a simple Keras model for a sentiment classification task. The full sample data that contains 9194 reviews, in which 70% (6435 reviews) are used for training phase, and 30% (2759 reviews) are used for validation phase. I tried with some models with the embedding layer that is always the first layers. Moreover, I did many combinations of optimizers such as “adam”, “rmsprop”, “sgd” and activation

functions like “relu”, “sigmoid”, and “tanh”. The activation function “tanh” is always used in the last layer because the negative values are expected to return for negative score in sentiment problems. During model training, I also tried with different batch\_size and number of epochs. However, all models’ accuracy could not reach 60%. The best model is at 57.1%.

The idea of this approach is creating a specific word embedding by using the review data from rental industry to build a sentiment prediction model in this industry. The neuron network can learn from each of word that is labeled before. The advantage of this approach is that the corpus created contains amount of vocabulary in rental industry.

“TensorFlow” provides a projector to visualize the multiple-dimensional space, so I tried to load the word embedding created with Airbnb dataset. For example, with the word “great”, there are some other similar words such as: “locationgreat”, “greatwelcoming”, “greathost”, “greatcleanairymodern”, “propertygreat”, “restaurantgreat”.



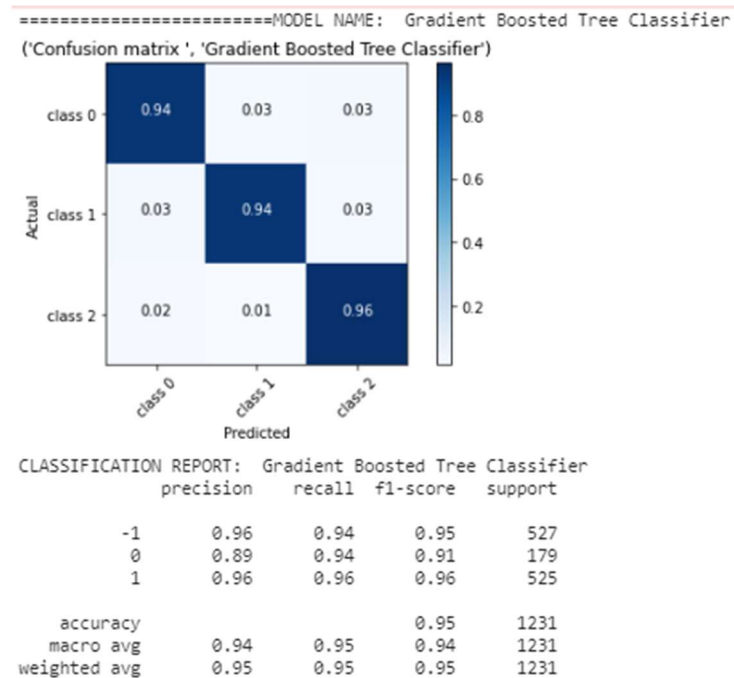


## VIII. EVALUATION METHOD & RESULTS

### 1. Evaluation Method

For multiple classifier Machine Learning model, I use Confusion Matrix and Classification Report as estimators to evaluate the model performance. There are three classes such as “POSITIVE”, “NEGATIVE”, and “NEUTRAL” in this problem. Therefore, the confusion matrix has 3 rows and 3 columns.

For instance:



- The first line with the name “class 0” of the confusion matrix corresponding to the first line of classification report indicates “NEGATIVE” class with 527 reviews.
- The second line with the name “class 1” of the confusion matrix corresponding to the second line of classification report is the “NEUTRAL” class with 179 reviews.
- The third line with the name “class 2” of the confusion matrix corresponding to the third line of classification report is the “POSITIVE” class with 525 reviews.
- The overall accuracy of the model algorithm reaches 95% that is the ratio of the predicted results and the actual values.

- The precision is computed as the number of predictions made correctly among all predictions based on the positive class.
- The recall shows the number of predictions made correctly based on all positive reviews.
- F\_score: The F1 score is the harmonic mean of accuracy and recall. It helps us optimize the classifier to balance accuracy and recall. In some cases of unbalancing data, the accuracy is very high, but the F1-score is too low. Therefore, F1-score is also very important to use for evaluation.

## 2. Results

### i. **Approach 1:** ML models with feature Engineering

- This is the summary of results of all classification ML models trained with different feature engineering techniques in this study.

	Model	Feature Engineering	Rsquare Train	R-square Test	MAE Train	MAE Test	Accuracy	Fscore	Training_duration
2	Support Vector Machine Model	BoW	0.988862	0.938312	36.1	238.6	0.938312	0.924648	52.573105
5	<u>Gradient Boosted Tree Classifier</u>	<u>BoW</u>	0.994779	0.931006	16.6	243.2	<u>0.931006</u>	<u>0.908380</u>	<u>8.153292</u>
2	Support Vector Machine Model	TF-IDF	0.994779	0.926948	18.5	265.9	0.926948	0.899414	127.451061
5	<u>Gradient Boosted Tree Classifier</u>	<u>TF-IDF</u>	0.997912	0.908279	5.8	352.3	<u>0.908279</u>	<u>0.884303</u>	<u>8.035508</u>
5	<u>Gradient Boosted Tree Classifier</u>	<u>Skip_Gram</u>	0.999304	0.895292	1.9	397.7	<u>0.895292</u>	<u>0.857892</u>	<u>20.004974</u>
2	Support Vector Machine Model	Skip_Gram	0.913679	0.893669	311.9	384.1	0.893669	0.844238	2.533225
4	Random Forest Classifier	Skip_Gram	0.999304	0.887175	1.9	409.1	0.887175	0.837050	7.828330
3	Decision Tree Classification Model	TF-IDF	0.895231	0.841721	462.0	727.3	0.841721	0.834632	4.105088
3	Decision Tree Classification Model	BoW	0.922381	0.840909	336.2	706.8	0.840909	0.829151	2.109359
1	KNN	Skip_Gram	0.914027	0.879870	308.0	456.8	0.879870	0.824223	0.004961
0	NAIVE BAYES	Skip_Gram	0.841281	0.850649	541.9	522.7	0.850649	0.800046	0.026928
0	NAIVE BAYES	TF-IDF	0.983641	0.847403	84.8	643.2	0.847403	0.791396	0.618345
3	Decision Tree Classification Model	Skip_Gram	0.928646	0.838474	244.6	570.5	0.838474	0.782748	2.174206
0	NAIVE BAYES	BoW	0.940480	0.787338	309.9	975.0	0.787338	0.761284	0.493833
4	Random Forest Classifier	TF-IDF	0.830143	0.814123	520.4	661.4	0.814123	0.597511	2.340768
4	Random Forest Classifier	BoW	0.833623	0.811688	514.6	675.0	0.811688	0.596255	2.120331
1	KNN	BoW	0.785590	0.577110	739.7	1475.0	0.577110	0.566373	0.000998
1	KNN	TF-IDF	0.201880	0.165584	2235.7	2338.6	0.165584	0.122932	0.041888

- Based on above figures, I prefer to select the “Gradient Boosted Tree Classifier” is the best model with all feature engineering techniques because of the high accuracy and F-score as well as the short training time. The accuracy is more than 89% and F-score is more than 85%.

While although the “SVM” models reach high accuracy and F-score, but they take much time for training with 2873 reviews.

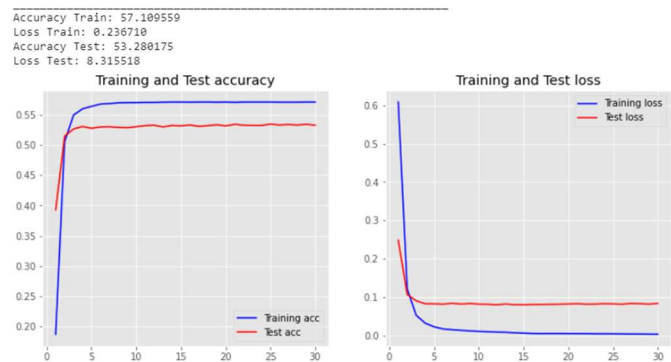
- As before expectation, the best model would work well with extracted features from Word2Vec (skip-gram) feature engineering. However, the results’ figure of Gradient Boosted Tree Classifier using skip-gram are lower among the “Gradient Boosted Tree Classifier” with other feature techniques. This might be investigated about text processing and input parameters configuration.
- KNN and Forest Random Models are worse model, they don’t well learn from all extracted features in Airbnb reviews dataset.

## ii. **Approach 2:** Deep Learning models with word embedding.

	Accuracy Test	Accuracy Train	Activation	Batch size	Loss Func	Loss Test	Loss Train	Model Name	Nb Epochs	Optimizer	Training Duration	input_dim	input_length	output_dim
2	53.28	57.11	relu	64.00	mse	8.32	0.24	model_13	30.00	adam	29.55	12135.00	80.00	64.00
4	53.03	57.11	relu	1028.00	mse	10.52	0.02	model_22	100.00	adam	182.62	12135.00	80.00	64.00
1	52.23	57.03	sigmoid	128.00	mse	8.34	0.51	model_12	30.00	adam	19.33	12135.00	80.00	64.00
3	52.19	56.83	sigmoid	1028.00	mse	15.09	2.18	model_21	100.00	adam	185.36	12135.00	80.00	64.00
0	51.76	57.02	sigmoid	1028.00	mse	8.55	0.38	model_11	100.00	adam	35.44	12135.00	80.00	64.00
5	42.55	42.86	relu	128.00	mse	190.54	185.81	model_3	50.00	adam	198.83	12135.00	80.00	64.00

- Below are results of all Deep Learning models trained with the same word embedding. The model “model\_13” is the best with the highest accuracy and the lowest loss in train and test phases. The model reaches 57.1% accuracy after 5 epochs. Then, the accuracy and lost lines shows that the model cannot learn more. Moreover, the train accuracy is always higher than test accuracy, the train loss is always lower the test loss. This indicated

that the model tends to overfitting. To improve this situation, more training data might be required.



### 3. Results from out-of-sample prediction using the best model.

❖ Below is the prediction out-of-sample using the selected Gradient Boosted Tree Model

comments	labelled_by_Human	IDF	BOW	Skip-Gram
We spent 4 nights at this beautiful cottage an...	1	0	1	1
We loved staying here! Having coffee and watch...	1	1	1	1
Bob was an owner who was easily reachable if I...	1	0	1	1
This location is amazing. It is close enough ...	1	0	1	1
A very roomy and comfortable apartment - it co...	1	-1	1	1
Excellent service, lovely home with the extras...	1	0	1	1
Gerry made our first Airbnb experience a fanta...	0	0	0	-1
Great spot	1	0	0	1
Host is great but the place was unfortunately ...	-1	0	1	1
Very spacious house with friendly owners. 3 se...	0	-1	1	1
Great space!	1	0	1	1
It is convenient, comfortable place for stayin...	1	0	1	1
They were great host, we extended our stay and...	1	0	1	1
The cottage is ok.\n	0	0	0	0
The street noise at night was loud and disrupt...	-1	0	-1	-1
Beautiful quiet place to stay	0	0	1	1
Quick response and great location	0	0	1	1
We a nice size, the landscaping beautiful as w...	-1	0	-1	-1
Unfortunately, I did not have the same positiv...	-1	-1	-1	-1
Had a great time at this cottage.	0	0	0	1
They run a bed and breakfast in the house. Thi...	-1	0	-1	-1
Hosts were excellent hosts and the room in whi...	1	0	1	1
This little cottage truly is paradise! A beaut...	1	-1	1	1
You can expect a filthy yard filled with dog ...	-1	0	-1	-1
Decent place, Noah was a little rude. No coffe...	-1	0	-1	-1

- ❖ Below is the prediction out-of-sample using the Keras neuron network model with created word-embedding using Airbnb dataset.

comments	labelled_by_Human	pred_label
We spent 4 nights at this beautiful cottage an...	1	1.00
We loved staying here! Having coffee and watch...	1	0.83
Bob was an owner who was easily reachable if I...	1	0.99
This location is amazing. It is close enough ...	1	-0.42
A very roomy and comfortable apartment - it co...	1	0.97
Excellent service, lovely home with the extras...	1	1.00
Gerry made our first Airbnb experience a fanta...	0	0.87
Great spot	1	0.98
Host is great but the place was unfortunately ...	-1	-0.05
Very spacious house with friendly owners. 3 se...	0	0.84
Great space!	1	0.98
It is convenient, comfortable place for stayin...	1	0.99
They were great host, we extended our stay and...	1	0.99
The cottage is ok.\n	0	0.56
The street noise at night was loud and disrupt...	-1	-0.73
Beautiful quiet place to stay	0	0.98
Quick response and great location	0	0.99
We a nice size, the landscaping beautiful as w...	-1	-0.69
Unfortunately, I did not have the same positiv...	-1	-0.87
Had a great time at this cottage.	0	0.96
They run a bed and breakfast in the house. Thi...	-1	-1.00
Hosts were excellent hosts and the room in whi...	1	0.99
This little cottage truly is paradise! A beaut...	1	0.99
You can expect a filthy yard filled with dog ...	-1	-1.00
Decent place, Noah was a little rude. No coffe...	-1	-1.00

The results of out-of-sample prediction are interesting. I found that the results from the neuron network model are good. The sentiment scores returned from prediction are nearly like the label made manually by myself.

Moreover, I found that the weight of sentiment score from neuron model changes depending on the semantic of the text. For instance:

```
test_sen=["There is not enough light in the bathroom"]
pred_label = do_prediction(test_sen,p_max_length)
print(pred_label)

[[-0.4459612]]
```

```
test_sen=["There is not enough light in the bathroom, it made me uncomfortable"]
pred_label = do_prediction(test_sen,p_max_length)
print(pred_label)

[[-0.9866566]]
```

- The sentiment score in the first sentence is nearer zero (Neutral) but gets a bit negative.
- The second sentence is added 1 more clause to show the bad experience about the light in the bathroom, the sentiment score returns -0.98 => This clearly shows the negative attitude.

## IX. DISCUSSION & CONCLUSION

In this project, I do sentiment analysis by creating sentiment classification models using Machine Learning algorithms as well as Deep Learning (neuron network). I realize that making machines understand the semantics of human natural language is a very complex task, and I admit that feature extraction technique is the most important part of sentiment analysis problem. During project, I experienced the importance of extracted features (or word embedding) created from FE and used for model training.

- Regarding to count-based method, the model cannot learn the real emotion in the context because the sentiment score based on the frequency of appearance of a word. Moreover, this technique require a lot of available memory resource to store too many value 0. It clearly doesn't work for a big document.
- Regarding to word2vec (CBOW and skip-gram) that is the result of a neuron network, the model can learn better about the context of a sentence because the target and input variables are created in a context. In this study, the good result cannot be worked out for Word2Vec, but I believe the model can be learn better if there are more training data. However, to create a Word2Vec lexicon, training time is a big challenge. Moreover, to train a large text data, the hardware resource is required.

- I really prefer the approach to create word embedding by a neuron network model from Airbnb dataset. In this project, although the model accuracy is not much high (it's only 57%), the model still worked quite well. I think this approach can be much better if we have a large, labeled training set.

## X. REFERENCES

<https://www.tensorflow.org/tutorials/text/word2vec>

[https://www.tensorflow.org/tutorials/text/word\\_embeddings](https://www.tensorflow.org/tutorials/text/word_embeddings)

[Airbnb Data Collection: Methodology and Accuracy – Tom Slee](#)

[Digging into Airbnb data: reviews sentiments, superhosts, and prices prediction \(part1\) | by Dmytro Iakubovskiy | Towards Data Science](#)

[Towards Data Science](#)

[Basic text classification | TensorFlow Core](#)

[NLTK Tokenize: Words and Sentences Tokenizer with Example \(guru99.com\)](#)

[Sentiment Analysis in Python: TextBlob vs Vader Sentiment vs Flair vs Building It From Scratch - neptune.ai](#)

[Analyzing the AirBnB Dataset for trends using Data Visualizations and Modeling | by Ravish Chawla | ML 2 Vec | Medium](#)

	Terms	Description
1	NLP	Natural Language Processing
2	Tokenization	is essentially splitting a phrase, sentence, paragraph, or an entire text document into smaller units such as individual words or terms. Each of these smaller units are called tokens. ( <a href="https://www.webstep.se/an-introduction-for-natural-language-processing-nlp-for-beginners/">https://www.webstep.se/an-introduction-for-natural-language-processing-nlp-for-beginners/</a> )
3	Vader	Valence Aware Dictionary for Sentiment Reasoning is a model used for text sentiment analysis that is sensitive to both polarity (positive/negative) and intensity (strength) of emotion. It is available in the NLTK package and can be applied directly to unlabeled text data <a href="https://medium.com/ro-data-team-blog/nlp-how-does-nltk-vader-calculate-sentiment-6c32d0f5046b">https://medium.com/ro-data-team-blog/nlp-how-does-nltk-vader-calculate-sentiment-6c32d0f5046b</a>
4	Textblob	TextBlob is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more. <a href="https://textblob.readthedocs.io/en/dev/">https://textblob.readthedocs.io/en/dev/</a>