School of Information Technology and
Engineering at the
ADA University

School of Engineering and Applied Science
at the
George Washington University

SENTIMENT AND EMOTION ANALYSIS OF A TEXT IN AZERBAIJANI LANGUAGE

A Thesis
Presented to the Graduate Program of Computer Science and Data Analytics
of the School of Information Technology and Engineering
ADA University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science in Computer Science and Data Analytics
ADA University

By
Leyla E. Aliyeva

April, 2022

THESIS ACCEPTANCE

This Thesis by: Leyla E. Aliyeva
Entitled: *Sentiment and Emotion Analysis of a text in Azerbaijani Language*

has been approved as meeting the requirement for the Degree of Master of Science in Computer Science and Data Analytics of the School of Information Technology and Engineering, ADA University.

Approved:

Samir Rustamov Rustamov S 28.04.2022
(Adviser) (Date)

Abzatdin Adamov 28.04.2022
(Program Director) (Date)

Sencer Yeralan 28.04.2022
(Dean) (Date)

ABSTRACT

Since the technology evolves day by day, it is undeniable fact that the interaction between people and technology has been changed dramatically. All these changes result in the huge amount of data and information flow on the web. As it is said that data is the new oil in today's century, it has become crucial to be able to analyze and use this data appropriately. Social media is one of the crucial contributors to the issue as it is producing terabytes of data every single day.

Natural Language Processing (NLP) is known as the well-known tool to recognize and interpret the human language. Being as a branch of Artificial Intelligence, NLP helps to automate the relationship between human and machine with the help of the structure of natural language. The goal of NLP is to understand the human language and answer the questions accordingly by processing given human information.

Some of the most popular applications of NLP are virtual contacts like Siri, Alexa, and Google Assist. The simplest way to visualize how NLP work with Siri is that it transforms human commands into numbers for making it understandable for machines. Another application of NLP is chatbots of which job is to help support teams solving issues by understanding human requests and giving responses accordingly. Other applications of NLP such as spelling recommendations, automatic translations on social media, categorizing receiving emails appropriately are also on trend. To sum up, we can say that NLP aims to make the humans' life easier by creating interaction between humans and machine.

NLP applies mainly two techniques for establishing machine-human interaction: syntactic analysis and semantic analysis. To be able to apply these analysis tasks, several sub-tasks are implemented.

Syntactic analysis consists of applying grammar rules to the text for identifying the structure of the text, the organization between the words, and their relations:

- Tokenization – splitting a word or a sentence into smaller parts in order to make it more understandable.
- PoS tagging – known as Part of Speech tagging, labelling tokenized text by its parts of speech.
- Lemmatization & Stemming – splitting the words into smallest meaningful forms.
- Removal of Stop words – removing frequently occurring words which has no contributions to semantics.

Semantic analysis rather aims to identify the meaning of the text by analyzing each individual word. This can be achieved by applying following sub-tasks:

- Word sense disambiguation – identifying the sense of word within the context.
- Relationship extraction – identifying the relations between entities of the given text.

One of the main business use cases of NLP that is applied widely is Sentiment and Emotion Analysis which is also the main topic of this paper. Sentiment and Emotion Analysis identifies the sentiment and emotion values of the input text and categorizes user's opinion into 3 sentiment values (positive, negative, neutral) and 8 Plutchik's emotions (anger, anticipation, joy, trust, fear, surprise, sadness, and disgust) [28]. SEA (shortly Sentiment and Emotion Analysis) can be used for the analysis of social media comments, organization reviews, online surveys, and customer service reviews. By this way, company leaders can clearly see how their customers feel about their products and make appropriate decisions.

SEA can highly influence the organization's productivity and quality as it helps to show the strong and weak sides of the products based on customer review analysis.

Keyword List – ***Sentiment Analysis, Machine Learning, Azerbaijani language, News, ANN***

Table of Contents

Chapter

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Explanation
NLP	Natural Language Processing
SEA	Sentiment and Emotion Analysis
SEAAL	Sentiment and Emotion Analysis in Azerbaijani Language
Sentiment values	positive, negative, neutral
Emotion values	anger, anticipation, joy, trust, fear, surprise, sadness, and disgust
TF	Term Frequency
IDF	Inverse Document Frequency
BoW	Bag of Words, which is a collection of words in one statement
LD	Lexicon Dictionary, which is a dictionary of individual words
ANN	Artificial Neural Network
MLP	Multi-Layer Perceptron
ERD	Entity Relationship Diagram
GUI	Graphical User Interface
FR	Functional Requirement
Dialog	A pop-up window that allows the user to enter information or get information
CNN	Convolutional Neural Networks

1 INTRODUCTION

1.1 Definition of the Problem

NLP has been spread widely all over the world and has many applications that is serving to solve many problems in human lives. As Azerbaijan as a country is evolving in technology and digitalization, this project aims to apply NLP in Azerbaijani language. Since the digital growth is happening in Azerbaijan, the nation has begun the usage of online services in daily life more often. The need of Azerbaijani language analysis has started as the data generated from these online services consist of local people who communicate through Azerbaijani language.

To be able to see how important is to apply SEA in Azerbaijani language, the easiest way is to look at the statistics.

According to “DATAREPORTAL” (https://datareportal.com/reports/digital-2021-azerbaijan), internet users in Azerbaijan in 2021:

- There were 8.26 million internet users in Azerbaijan in January 2021.
- The number of internet users in Azerbaijan increased by 202 thousand (+2.5%) between 2020 and 2021.
- Internet penetration in Azerbaijan stood at 81.1% in January 2021.

Additionally, social media statistics in Azerbaijan in 2021 is as follows according to DATAREPORTAL:

- There were 4.30 million social media users in Azerbaijan in January 2021.
- The number of social media users in Azerbaijan increased by 600 thousand (+16%) between 2020 and 2021.
- The number of social media users in Azerbaijan was equivalent to 42.2% of the total population in January 2021.

1.2 Objective of the Study

So why Sentiment and Emotion Analysis in Azerbaijani language? As Sentiment and Emotion Analysis gives great initiative for company leaders categorize their customers based on their satisfaction levels and most significantly, there were no such Azerbaijani data labelled and categorized, this project is focused on doing Sentiment and Emotion Analysis in Azerbaijani language with the help of NLP. The fundamental concept in this project is to identify the sentiment (positive, negative, neutral) and emotion values (anger, anticipation, joy, trust, fear, surprise, sadness, and disgust) [28] in the user’s opinion.

1.3 Significance of the Problem

Sentiment and Emotion Analysis is significant in terms of extracting valuable information out of provided text in order to be able to make further decisions. The main motive here is to decrease the level of labor force and minimize the time needed to get the valuable output. As the businesses started to evolve, Sentiment and Emotion Analysis has also become on trend like other fields of technology. In other words, all the businesses care about their customers’ opinions for being able to improve their current products by overcoming their weak sides, create new product-lines, and make future based decisions. Application of NLP in businesses allows business leaders to analyze and adapt to the feedbacks quickly and act accordingly if the satisfaction level is low.

By implement Sentiment and Emotion Analysis, not only business leaders but also customers will benefit since they will get quick response to their provided reviews. The enterprise companies will not only get rid of labor force costs, but also gain customers by building bonds with each other.

Implementation of Sentiment and Emotion Analysis not only effect enterprises, but also it has crucial impact on mass media, especially news articles. It can help to estimate people’s impressions to the events happening all over the world, local news, governmental updates, political changes, neighborhood events, and etc.

1.4 Review of Significant Research

As mentioned before, Sentiment and Emotion Analysis has rarely applied in Azerbaijani language, therefore, there is a lack of research conducted. As a result, very few machine learning techniques are applied to the Azerbaijani language which triggered the research project to be SEAAL. During the project cycle, identification of sentiment and emotion values of the user's review text will be investigated.

1.4.1 Related works

In 2017, Araque et al. [1] have suggested a model to increase the accuracy and performance of existing models by applying deep learning algorithms on manually extracted features. The newly proposed model – a combination of linear machine learning algorithms and word embeddings has been implemented as a deep learning-based sentiment classifier. As a final result, it was deducted that the proposed model was performing well over the traditional models.

In 2018, Fang et al. [2] have presented a model – combination of semantic fuzziness for sentiment analysis to achieve higher accuracy, which realized the assumptions at the end.

In 2018, Mukhtar et al. [3] have implemented sentiment analysis on Urdu blogs dataset by applying Supervised machine learning algorithms and lexicon-based models. For achieving real outputs, data were collected from different sources. As machine learning algorithms, DT, KNN, SVM were implemented, at the same time, for lexicon-based models, Urdu sentiment lexicons were analyzed. As an output of analysis, it was deducted that lexicon-based models have shown more effective results than the machine learning algorithms.

In 2018, Abdi et al. [4] have suggested a model which could summarize the text (news, reviews, comments, tweets and so on). For implementing such model multiple features should have been combined into one to be able to summarize all the text. Finally, after implementing the model in different datasets, it was deducted that Information Gain (IG) method was effective in feature selection, and SVM was effective in classification approach.

In 2018, Smadi et al. [5] have suggested to build a sentiment analysis model based one supervised machine learning algorithms. The purpose of the model was to classify hotel product reviews and to detect defects on the features. The model was trained and tested on the basis of word, lexical, morphological, semantic, and syntactic features and SVM, Deep RNN machine learning algorithms were applied. As a result of comparison these two models, SVM showed better results.

In 2019, Saad and Yang [6] has implemented sentiment analysis on twitter data by applying regression with machine learning algorithms. In their application, firstly, processing of tweets, and then, features generations with the help of feature extraction models were applied. As classification of the sentiment analysis, the machine learning algorithms such as SVR, RF, SoftMax, and DTs were implemented. The same twitter dataset was used to test the implemented model of sentiment analysis, and as a result, it was identified that DTs has presented the higher accuracies.

In 2019, Afzaal et al. [7] have suggested to apply aspect (entity) level sentiment analysis, which aimed in to identify specific features in the input text, and then analyze its sentiment values. The suggested model helped to achieve higher classification accuracy since there could be more than one sentiment values in one input only. The software was applied as a mobile application, and used for tourists to identify well-known hotels, restaurants, cafes through the countries. The model was trained using the real-world data, and as a result, the outcome values were highly successful in terms of both recognition and classification.

In 2019, Feizollah et al. [8] have applied sentiment analysis on twitter data with the topic of halal products like halal meals, halal cosmetics and so on. After data was filtered, deep learning models and RNN, CNN, LSTM were applied for prediction methods. As a conclusion, it was realized that LSTM and CNN worked better together.

In 2019, Ray and Chakrabarti [9] have suggest a deep learning algorithm for entity recognition in a text and classifying its sentiment value accordingly. For feature extraction, seven-layer CNN was used as a proposal model. Additionally, the authors combined deep learning algorithms with the rule-based ones, consequently, attained higher accuracy.

In 2019, Zhao et al. [10] have suggested a program which could predict the relationship between a text and an image. To be able to implement this program, they have created a multi-modal sentiment evaluation model - SentiBank. This model worked as extracting the mid-level visual features and using them as visual theories like social, textual, and visual features. At the end, it was concluded that this multi-modal sentiment analysis model have shown better effectiveness than the other traditional models.

In 2019, Park et al. [11] have suggested a semi-supervised sentiment analysis model for overcoming issues with partial documents by also saving local structures from real data. The train data used was real-world data and the model was tested also on real-time data basis as well. The final results have shown that the model was effective.

In 2019, Vashishtha and Susan [12] have implemented sentiment analysis on social media dataset by applying set of fuzzy rules on lexicon dictionary. For classification of social media comments into 3 categories as positive, negative, and neutral, they have developed a model merging Word Sense Disambiguation, NLP, unsupervised fuzzy rules. The machine was trained and tested on not only social media dataset, but also sentiment lexicons, existing models, freely available datasets. The final results have shown that suggested model attained high efficiency.

In 2019, Yousif et al. [13] have suggested a model planned to use for providing citation context where feature extraction was done automatically. This multi-tasking method was built on the basis of CNN and RNN and was tested based on two freely available dataset. The results showed that the model achieved higher results than the traditional models.

In 2019, Abdi et al. [14] have proposed a sentiment analysis model to classify the users' opinions and their reviews. They developed a unified feature set as a deep learning model on the basis of shifting rules, word embedding, sentimental knowledge, linguistic and statistical knowledge. Furthermore, RRN consisting of LSTM was implemented to see the effect of sequential processing which overperformed than the other traditional models.

In 2019, Bardhan et al. [15] have implemented a model for analyzing the effects of different gender orientations on Sexual and Reproductive Health (SRH). For this purpose, they have used datasets on the basis of semi-structured interviews, and group discussions. As a machine learning algorithm, Natural Language Processing (NLP) was used for the sentiment analysis classification.

In 2020, Kumar et al. [16] have presented a hybrid model for sentiment analysis and prediction that has combined Con, VNet, SVM, BoVW models in one. Data was trained by using SVM, however, it was deducted that conventional models performed better than the hybrid deep learning approach.

In 2020, Hassonah et al. [17] have suggested to use hybrid models of machine learning, specifically, SVM for sentiment classification and MVO, Relief models for feature extractions. As train and test datasets, tweets were used, and the experiments have shown that the recommended hybrid algorithms attained well-performing results.

In 2020, Xu et al. [18] have proposed a model to be used in e-commerce programs for classifying the product reviews. The model was implemented with Naïve Bayes (NB) method for continuous learning process and fine-tuning was applied on already learnt classifications. The outcomes have shown that the suggested model was performing well.

In 2020, Maqsood et al. [19] proposed sentiment analysis classification as a result of different events happening through 2012 and 2016. Machine learning algorithms were applied on twitter dataset to analyze the effects of those events on stock markets.

In 2020, Park et al. [20] have done research how to improve accuracy of existing sentiment classification models. For this purpose, they have developed content attention method for combining multiple attention outputs. As a result of this implementation, they could achieve higher results.

Additionally, two Azerbaijani papers were investigated for research purposes.

According to Rustamov S. [25], Linear regression, Naïve Bayes and SVM models were applied to the twitter dataset in Azerbaijani language. The highest results were achieved with SVM model, the others showed approximately the same results.

According to Rustamov S. [26], different structures such as FCS, ANFIS, HMM and their combinations, Hybrid-I and Hybrid II, were applied to classify reviews on the document level in Azerbaijani language. Among all, FCS is the fastest, but ANFIS gives higher results (83%). As a combination of classifiers, Hybrid II model has increased accuracy from 83% to 83.95%.

1.4.2 Performance measures

Table 1 below represents evaluation metrics for the sentiment analysis applications that are explained earlier [29].

Citations	Accuracy	Precision	Recall	F-Measure
[1]	+	+	+	+
[2]	+	+	-	+
[3]	+	+	+	+
[4]	-	-	-	-
[5]	-	-	-	-
[6]	+	+	+	+
[7]	+	+	+	+
[8]	+	+	+	+
[9]	+	+	+	-
[10]	+	+	+	+
[11]	+	-	-	-
[12]	+	+	+	+
[13]	-	+	+	+
[14]	+	+	+	+
[15]	+	-	-	-
[16]	+	+	+	-
[17]	+	+	+	+
[18]	+	-	-	-
[19]	-	-	-	-
[20]	+	-	-	+
[25]	+	+	+	+
[26]	+	-	-	-

Table 1: Performance measures of each research applications

1.4.3 Data types

Table 2 represents the data types of each research on sentiment analysis above by using machine learning algorithms [29].

Citations	Data Type
[1]	microblogging and movie reviews domain
[2]	Review of consumer products and services on hotel
[3]	Urdu blogs in multiple domains
[4]	DUC 2002, and Movie Review Data
[5]	Arabic hotel's review
[6]	Twitter data
[7]	Restaurant and hotel data
[8]	Twitter keywords related to halal tourism and halal cosmetics
[9]	Nikon Camera Data, and laptop domain data
[10]	Social Media data
[11]	Amazon reviews and Yelp reviews
[12]	multiple public twitter data
[13]	citation sentiment, and citation purpose
[14]	Movie Review
[15]	Text data from corpous
[16]	Textual and visual semiotic modalities of social data
[17]	Twitter Social Network data
[18]	Amazon product and Movie review data
[19]	stock markets review
[20]	laptop and restaurant reviews from SesmEval 2014
[25]	Twitter dataset in Azerbaijani
[26]	"Rotten Tomatoes" movie reviews

Table 2: Data types of each research application above

1.5 Assumptions and Limitations

Sentiment and Emotion Analysis in Azerbaijani language has many benefits for the society, at the same time, it has some limitations for the project to be fully implement. Here are the limitations:

- Lack of labelled data in Azerbaijani language
- Incorrectly trained machine
- Inconsistency between trained and test data

Lack of labelled data in Azerbaijani language – as few machine learning techniques are applied in Azerbaijani language, there is no valid and consistent set of data that can be used for Sentiment and Emotion Analysis. Therefore, all the data is built from zero to provide the machine with training data.

Incorrectly trained machine – since the trained data is labelled by human force, there can be mislabeling, or different perspectives for specific data.

Inconsistency between trained and test data – since the local organizations are not willing to provide their customer feedbacks, news data is used as a training data for this research project. In other words, project is planned to be used for entrepreneurs, however, it is trained with news data, therefore, concepts of trained and test data are different.

During the project cycle, it is assumed that each sentence expresses only one sentiment value which can have more than one entity. Additionally, each sentence can express more than one emotion value.

2 RESEARCH APPROACH OR METHODOLOGY

2.1 Levels of SEAAL

Mainly, analysis is implemented at three levels, providing different outputs:

- Document level
- Sentence level
- Entity level

The document level analysis - classifies the entire document opinion into different sentiment and emotion values. Here, document does not necessarily mean word or pdf document, but rather, it indicates the whole input that the user entered as feedback.

The sentence level analysis - determines the sentiment and emotion values of each sentence throughout all the documents. This type of analysis is useful when reviews or feedbacks have more than one sentence.

The entity level analysis **-** is extracting entities from the document and classify sentiment and emotion values based on each entity. This type of analysis is useful when reviews or feedbacks should be classified based on desired feature/aspect of the product.

This project focuses on Entity-level (or named as aspect-level as well) analysis for the Sentiment and Emotion values extraction

2.2 Vectorization method

Vectorization in NLP is used to convert input data from its raw text into vectors of real numbers that the machine learning models can understand. It helps to extract features from the text for training the machine learning model. Here are the main vectorization techniques used depending on the context of the problem:

- CountVectorizer
- TF-IDF Vectorizer
- Word2Vec
- GloVe

In this current project, only CountVectorizer and TF-IDF Vectorizer are used for feature extraction, therefore, their explanations are written below.

CountVectorizer – is a vectorization method used to convert a given text into a vector depending on the frequency of each word occurring through the entire text. The operations needed to apply CountVectorizer are explained step by step below:

1. Tokenization – the input text is tokenized into sentences, and then tokenized into words.
2. Vocabulary creation – after tokenizing input text into words, a vocabulary is created based on the unique words only.
3. Vector creation – depending on the frequency of the vocabulary (unique) words, a matrix is created where each row represents the sentence vector with the length of vocabulary.

Implementation of CountVectorizer below:

Importing required CountVectorizer package:

```
from sklearn.feature_extraction.text import CountVectorizer
```

Reading the data and tokenizing it into sentences:

```
data = ['this is a dual program',
   'ADA and GWU provides master program']
```

Creating instance of CountVectorizer:

```
cv = CountVectorizer()
```

Converting data into vectors and then to a NumPy array for visualization:

```
X = cv.fit_transform(data)
X = X.toarray()
```

Sorting vocabulary (unique words in data) to see most frequent and rare words:

```
sorted(cv.vocabulary_.keys())
```

In CountVectorizer, every row represents the vector of sentences in data (the input text). Accordingly, the size of each vector is associated with the size of vocabulary. Each value of the vector represents the frequency of the word occurring in the data. In the example above, only single words are considered as a feature for extraction, which is called unigrams. Additionally, n-gram features can be implemented such as bigrams, trigrams, and so on. The number of n-grams depends on the dataset type and can be manipulated accordingly.

Here, for tokenization – NLTK, and vectorization – SKLEARN libraries are used in Python, however, there are multiple other ways as well to do the same operations, totally up to the programmer.

Advantages of CountVectorizer:

- Simple to calculate

Disadvantages of CountVectorizer:

- Inability in identifying more important and less important words for analysis
- Considers all the frequent words as the most significant words
- Does not identify the relationships between words

TF-IDFVectorizer – Term Frequency - Inverse Document Frequency is used to identify how significant a word is with the input document. CountVectorizer, where only the frequency of a word mattered, has a drawback since it also evaluated prepositions, conjunctions, and etc. which do not add any value to the meaning of the document. However, as they appeared often, they are considered as significant words through the document. Here, TF-IDF solves the issue by not overestimating the words that get repeated too often and does not contribute any value.

TF – Term Frequency, which represents the proportion of frequency of a word happening through the document. The formula to calculate Term Frequency is:

$$TF = \frac{\text{Frequency of the word in a document}}{\text{Total number of words in the document}}$$

From the formula above it can be deducted that the value of TF is always less than 1.

IDF – Inverse Document Frequency, identifies how common a word through the whole document.

$$DF = \frac{\text{Documents containing word W}}{\text{Total number documents}}$$

DF represents how frequent a word W is within all the documents.

$$IDF = \log\left(\frac{\text{Total number documents}}{\text{Documents containing word W}}\right)$$

IDF is calculated as logarithm of reverse of DF since the more frequent a word is through all the documents, the less significant it is for that certain document.

Finally, TF-IDF is calculated as shown below:

$$TF - IDF = TF * IDF$$

TF-IDF value is straightly associated with the significance of the word, the higher the value of TF-IDF, the more significant the specific word is.

Implementation of TF-IDFVectorizer below:

Importing required TF-IDFVectorizer package:

```
from sklearn.feature_extraction.text import TfidfVectorizer
```

Reading the data and tokenizing it into sentences:

```
data = ['this is a dual program',
   'ADA and GWU provides master program']
```

Creating instance of TF-IDFVectorizer:

```
tfidf = TfidfVectorizer()
```

Converting data into vectors:

```
transformed = tfidf.fit_transform(data)
```

Importing required pandas package:

```
import pandas as pd
```

Creating data frame with the words and their TF-IDF values:

```
df = pd.DataFrame(transformed[0].T.todense(),
      index=tfidf.get_feature_names(), columns=["TF-IDF"])
df = df.sort_values('TF-IDF', ascending=False)
```

The idea explained above about n-grams (number of features) can be implemented in TF-IDF as well. Addition to n-grams, TF-IDF has parameters like min_df, max_df, max_features, sublinear_tf, and etc. to be able to modify model's serving opportunities.

Advantages of TF-IDFVectorizer:

- Simple to calculate
- Computationally cheap
- Start point for similarity calculations

Disadvantages of TF-IDFVectorizer:

- Does not carry semantic meanings
- Considers importance of the words based on frequencies/weights
- Ignores word order
- Memory inefficiency

2.3 Machine learning

The scope of the project is to provide a user-friendly system that extracts people’s sentiment values (positive, negative, neutral) and emotion values (anger, anticipation, disgust, fear, joy, sadness, surprise, trust) based on their feedbacks towards the particular product of that organization in the form of text. This section describes the internal architecture of the Sentiment and Emotion Analysis application, and how the software incorporates with bag of words.

a. Architectural Strategies

Below are described any design strategies that affect the overall organization of the system and its higher-level structures. These strategies provide insight into the key abstractions and mechanisms used in the system architecture. Three possible strategies exist for sentiment and emotion analysis, including:

- Machine Learning – employs a machine-learning technique and diverse features to construct a classifier that can identify text that expresses sentiment and emotion.
- Lexicon-Based – uses a variety of words annotated by polarity score to decide the general assessment score of a given content. It does not require any training data; however, a large number of words and expressions are needed to be included in lexicons.
- Hybrid – combines machine learning and lexicon-based approaches.

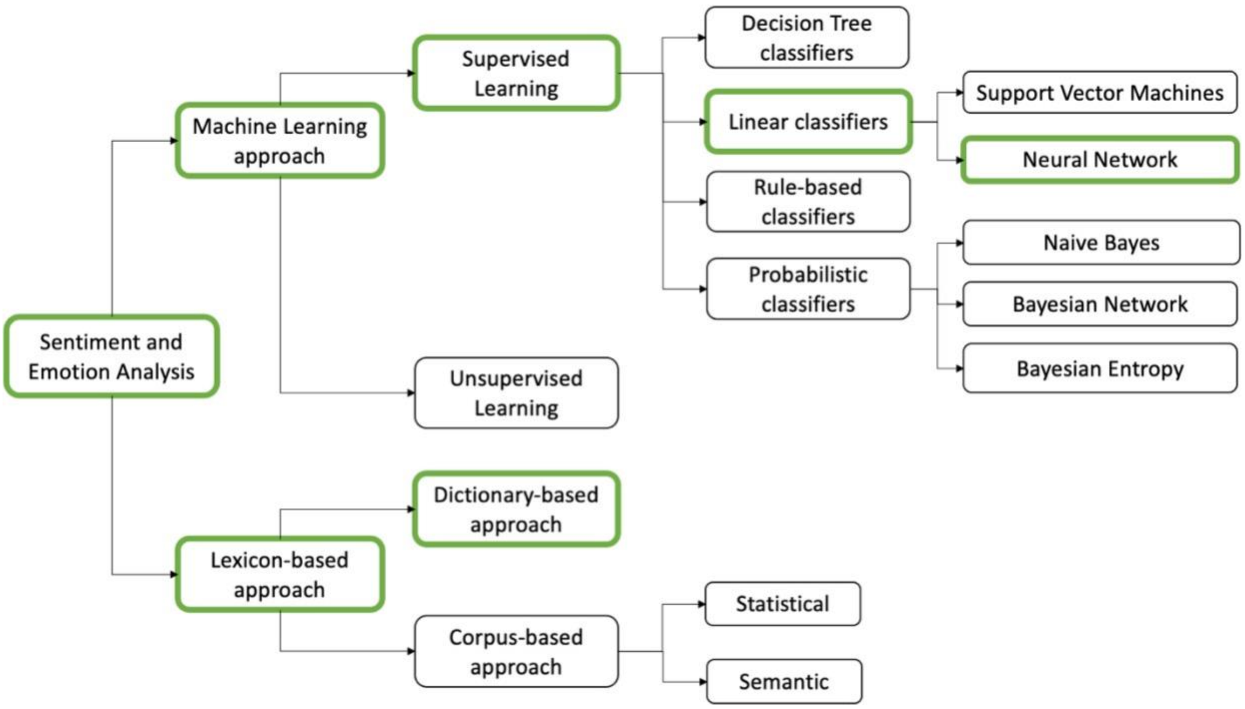


Figure 1: Sentiment Classification Techniques

This project focuses on researching the comparison of two approaches for the Sentiment and Emotion Analysis in Azerbaijan Language, particularly Machine Learning approach and Lexicon-based approach as shown in Figure 1. Therefore, Machine Learning is used in this project.

Machine Learning splits into two major sections based on the types of data, including:

- **Unsupervised Learning** – is a technique that uses machine learning algorithms to analyze and cluster unlabeled datasets. This learning includes clustering model.
- **Supervised Learning** – is a technique for training provided data by splitting it into two parts that are train data and test data. This learning includes regression and classification models.

This project belongs to Supervised Learning which operates with labeled results. Labeled data essentially applies descriptive features to unlabeled data for predictive modeling. Regularly, classification algorithms take this labeled dataset, process it, and produce labels that show which text content corresponds to which category.

Several levels of Sentiment and Emotion Analysis exist regardless of whether Machine Learning tools are used, including:

- **Document Level Analysis** – concerns with processing and defining the sentiment and emotion(s) of the whole documentation. Each text has a distinct polarity, sentiment and emotion(s) are extracted as a result of whole review and whole opinion.
- **Sentence Level Analysis** – determines the polarity of each sentence in a text in case that the sentence expresses an opinion.
- **Word/Phrase Level Analysis** – takes and analyzes the sentiment and emotion(s) of each individual word/phrase.

- **Aspect/Entity Level Analysis** – processes and analyzes the features of each object in order to define polarity from several aspects.
- **Comparative Level Analysis** – determines the arrangement of many entities in the sentence rather than a single entity.

This project concerns with examining the polarity of given text documents; therefore, this research is devoted to Entity Level Analysis.

A number of techniques and complex algorithms are used to command and train machines to perform sentiment and emotion analysis. Each has its own advantages and disadvantages, and some of them are explained below.

- **Decision Tree** – is a machine learning algorithm that deals on both discrete and continuous data to perform classification or prediction. This classification algorithm is a recursive, tree-structured model that determines the class for every given dataset. Prediction can be accomplished by dividing the root training set into subsets as nodes, with each node containing the contribution of the decision, label, or state. Each node is recursively separated after sequentially selecting alternate decisions.

  Why not Decision Tree? – It is difficult to estimate values for continuous variables in regression. After all, Decision Tree is known as a splitable model. It takes some complications with it. If the number of groups and datasets grows, the tree begins to fragment recursively. It causes equations to get more difficult, and the cost of training to rise. With a limited number of datasets, the likelihood of obtaining high precision is poor. Other machine learning methods usually provide better outcomes than decision trees [21].

- **Naive Bayes** – is a probabilistic model based on the Bayes Theorem. It determines the likelihood of hypothesis operation in relation to the given input.

  Why not Naive Bayes? – It assumes that all predictors are independent, rarely happening in real life. This limits the applicability of this algorithm in real-world use cases. Its estimations can be wrong in some cases [22].

- **BERT** – Bidirectional Encoder Representations from Transformers. It determines the meaning of a word with the help of the near text.

  Why not BERT? – It cannot deal with the long input documents. In our case, the data is trained with the model, and the accuracy was 82% which is not a high result.

- **ANN (Artificial Neural Network) –** is made up from following components:
  - Input Layer
  - Arbitrary of hidden layers
  - Output Layer
  - A set of weights and biases between each layer

- A choice of activation function for each hidden layer

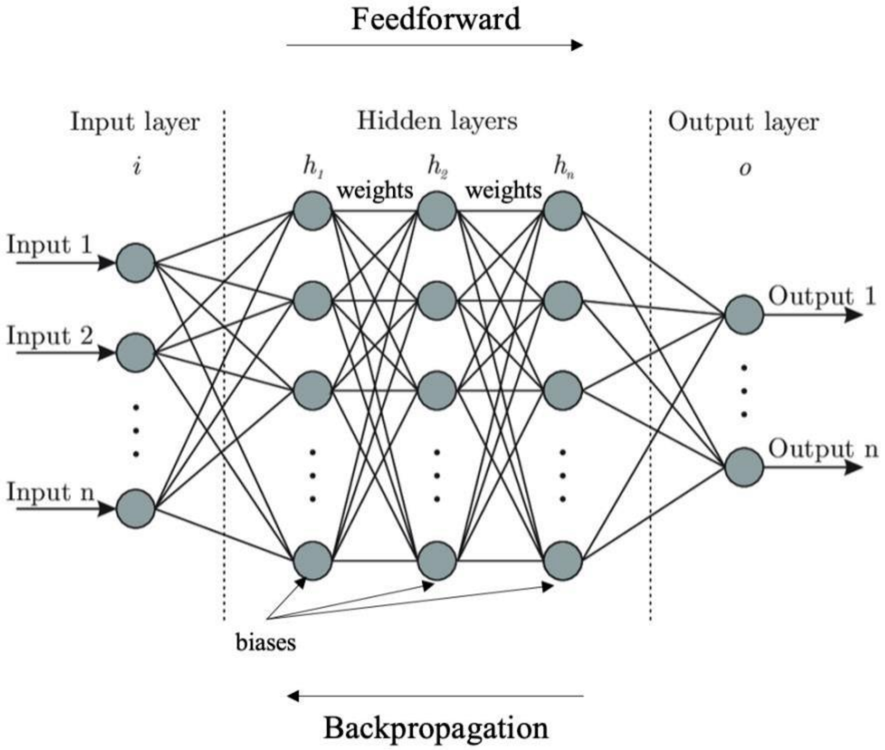


Figure 2: ANN Architecture

A multilayer perceptron (MLP) is a class of feedforward artificial neural network (ANN). An MLP consists of at least three layers of nodes: an input layer, a hidden layer, and an output layer. Except for the input nodes, each node is a neuron that uses a nonlinear activation function. MLP utilizes a supervised learning technique called backpropagation for training. In general, at each training iteration, two procedures will be performed: estimating the predicted result – feedforwards and updating the weights and biases – backpropagation [15].

Loss Function is used to check the effectiveness of forecasts. The loss function's derivative with respect to weights and biases is also determined to understand the way that is needed to achieve the minima, since the weights will be coordinated at the minima of the function that is considered for reducing the loss. To find the derivative of the equation, the chain law is used.

Backpropagation is applied in the method after achieving the derivative with the aid of the chain law. Following completion of these steps, neural networks is introduced to the dataset and an optimal set of weights is learned by the neural network. The loss is reduced towards zero as the neural network will be trained iteratively. The predictions are marginally different from the real values at the end of the training, indicating that the neural network generalizes better outcomes for unseen input avoiding overfitting [11].

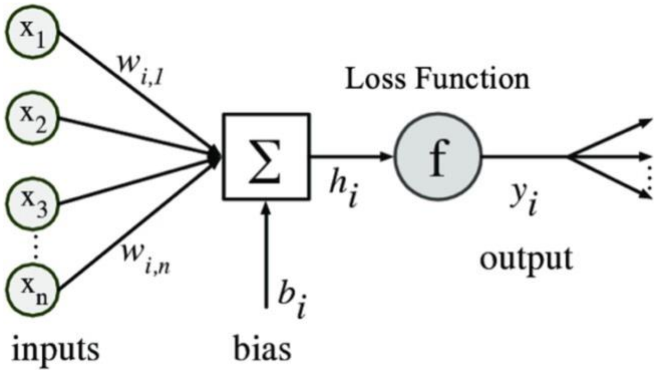


Figure 3. Loss Function Architecture

Why ANN? – It can learn and model non-linear and complex relationships, which is a significant benefit since the implementation of ANN. This method is more efficient because they can store information across the whole network and losing data across many regions of the network does not preclude the system from running. Storing data across the whole network makes ANNs more fault tolerant, as the system will continue to run even after losing many cells. ANNs succeed at working with incomplete information, and data can also yield output with incomplete knowledge. Artificial Neural Networks can process in parallel, allowing them to execute several tasks at the same time [23].

For splitting the dataset into train and test datasets CountVectorizer class of scikit-learn library is used. Scikit-learn is a Python library for machine learning and provides the CountVectorizer class, which can be used to call the fit() function to construct vocabulary and transform() function to build every matrix. Both words were converted into vectors before beginning Machine Learning Methods and feature extraction.

Here is precise sequence of how our program works and why ANN is chosen for this particular project:

- User will input a text to the text box to get the sentiment and emotion values result (*see Figure 8).*
- The program will be trained by two types of data (bag of words and lexicon dictionary) which will be stored in CSV files manually beforehand.
- These datasets will be split into train and test datasets through CountVectorizer class of scikit-learn Python library.
- After user presses submit button on the screen (*see Figure 8*), the inputted text will be split into words by separating them through spaces.
- Each word will be an input for the ANN and these words will be checked in both datasets for the corresponding sentiment and emotion values, separately. As our Bag-of-words dataset is in the form of paragraphs, each word occurs several times in various contexts. The machine learning algorithm will search for each word in the whole database and will calculate the average sentiment

and emotion values for that specific word. However, as one word can occur in different contexts, the output result is not biased based on one specific labeling.

- The results will be reported to the user through Python GUI (*see 3.6 User Interface section*).
- Users will be able to see both overall sentiment and emotion values of the inputted text, and also sentiment and emotion values of each entity (noun) in that inputted text with their percentages (*see Figure 9*).
- The inputted text will be inserted to the existing bag of words dataset for learning purposes whenever at least one word in the text is not found in the dataset. By this way, the error conditions are handled which can happen when the inputted words are not found in the datasets.
- The end result will be an MLP that learns how to automatically classify sentences and documents with sentiment and emotions.

Implementation of ANN MLP Classifier below:

Importing required package:

```
from sklearn.neural_network import MLPClassifier
```

Assigning model and setting number alpha, hidden layer size, and random state:

```
clf = MLPClassifier(solver='lbfgs', alpha=1e-5,
...                     hidden_layer_sizes=(5, 2), random_state=1)
```

Fitting (training) model:

```
clf.fit(X, y)
MLPClassifier(alpha=1e-05, hidden_layer_sizes=(5, 2), random_state=1,
              solver='lbfgs')
```

Predicting data based on trainings:

```
clf.predict([[2., 2.], [-1., -2.]])
array([1, 0])
```

2.4 System Architecture

The application has three main components. The first is a sequence of modules that does all the processing work, fetching the bag of words (news data) and lexicon dictionary, and calculating sentiment and emotion value(s). The second part is the data itself, which contains news data and individual words and their associated keywords. The final part of the application is the user interface, which allows the user to enter a text as an input and view calculated results.

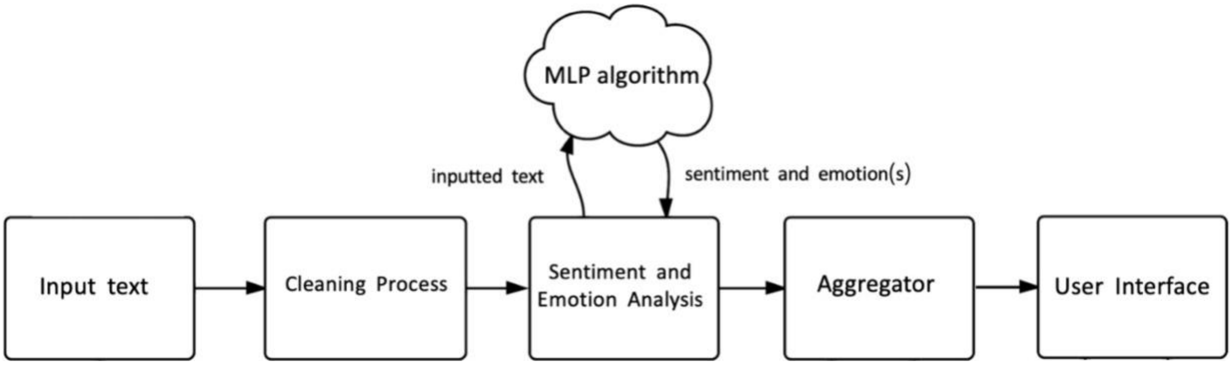


Figure 4. General Architecture Design

The module sequence in Figure 2 is used in each analysis session. Each module runs as its own thread, communicating with the next through a safe queue data structure. There are three distinct modules: news & dictionary dataset, sentiment and emotion analysis, and the aggregator module.

Input text

This module is the process of user inputting the text into a box which can indented to be feedbacks towards particular product of that organization.

Cleaning process

Data usually comes from a variety of different sources that is often in a variety of different formats. For this reason, cleaning given raw data is an essential part of preparing the dataset. However, cleaning is not a simple process, as text data often contain redundant and/or repetitive words.

This phase involves the deletion of words or characters that do not add value to the meaning of the text. Some of the standard cleaning steps are below:

- Lowering case
- Removal of special characters
- Removal of stop-words

Lowering the case of text is essential for the following reasons: The words, ‘News’, ‘NEWS’, and ‘news’ all add the same value to a sentence. Lowering the case of all the words helps to reduce the dimensions by decreasing the size of the vocabulary.

Removal of special characters will help to treat words like ‘hurray’ and ‘hurray!’ in the same way. At this stage, all punctuation marks are removed.

The equivalences of stop-words such as ‘the’, ‘a’, ‘an’, ‘is’ etc. in Azerbaijani is removed because they do not provide any valuable information.

Sentiment and Emotion Analysis

The sentiment and emotion analysis application will send user's inputted text to the MLP algorithm which is a class of feedforward ANN combined with the CountVectorizer. CountVectorizer's feature collection is founded on a predefined lexicon provided by the programmer. On all such instances, the number of features is determined by the length of the vocabulary. 'Scikit-learn' library is an MLP algorithm that is used to apply an artificial neural network to SEAAL project.

Once the sentiment and emotion value(s) are returned, it is added to the outgoing queue to go to the next thread/module, the aggregator.

For comparison purposes, TF-IDF Vectorizer is also used to identify the feature extraction.

Aggregator

To drive the output of the application, the result values will be aggregated as they are received from the sentiment and emotion analysis module. To get a numeric aggregate, different result values for each sentiment and emotion values over the number of sentences in the text should be summed, and a percentage which is rounded to the nearest integer should be calculated. For example, to get the positive percentage, the aggregator uses the equation below:

$$positive\ \% = \frac{\sum(positive\ values) * weight}{\sum(all\ values) * weight}$$

As the aggregator calculates new percentages, it sends results to the main thread via another queue for demonstrating the output.

User interface

The user interface provides sentiment and emotion analysis results of the inputted text through a GUI application which belongs to Python.

The following ERD (Entity Relationship Diagram) diagram explains the flow process of the system:

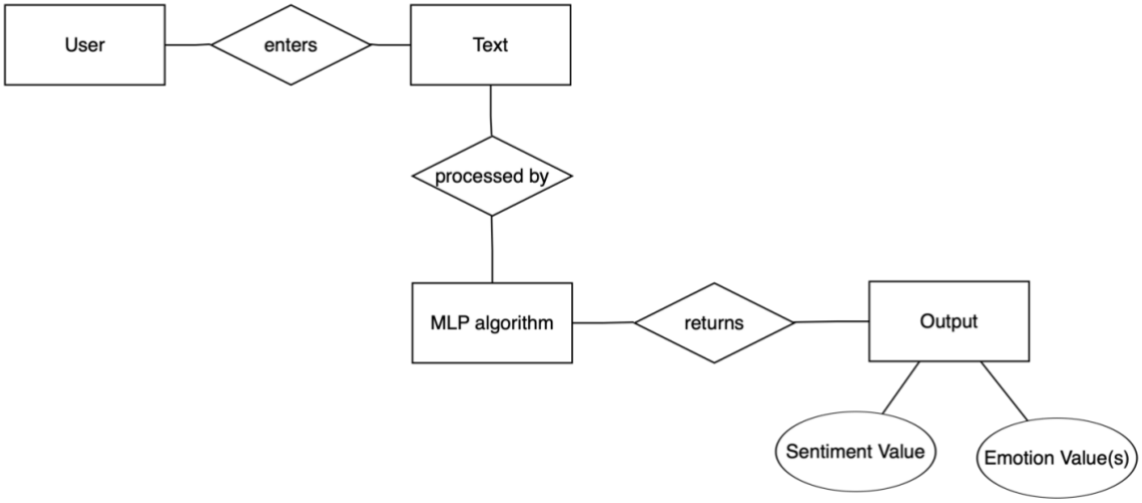


Figure 5: ERD diagram – flow of the system

Following is the example of the expected form of the data flowing between modules:

Module 1. Input text:

Azerbaijani: ‘Həmin cinayət vəhşicəsinə törədilmişdir, və cinayətkar qaçmışdır.’
English (for better visualization): ‘That crime was committed brutally, and the criminal run away.’

After user inputs all the text and pushes the submit button, all the text is sent to cleaning process at once.

Module 2. Cleaning process:

Converted text in Azerbaijani: ‘cinayət vəhşicəsinə törədilmişdir cinayətkar qaçmışdır’

Converted text in English: ‘crime committed brutally criminal run away’

For the cleaning purposes the followings are implemented:

- the stopwords ‘həmin’, ‘və’ (‘that’,‘and’, ‘the’) are removed
- the words are converted to lowercase
- the punctuations are removed

Module 3. Sentiment and Emotion analysis:

	Sentiment value	**Emotion value(s)**
cinayət (crime)	negative	anger, fear
vəhşicəsinə (brutally)	negative	anger, disgust, fear

törədilmişdir (committed)	neutral	-
cinayətkar (criminal)	negative	anger, fear
qaçmışdır (run away)	neutral	-

Table 3: Sentiment and Emotion values of the given sentence

Module 4. Aggregator:

Sentiment analysis:

$$negative\ \% = \frac{\sum(negative\ values)}{\sum(all\ values)} = \frac{3}{5} = 0.6$$

$$neutral\ \% = \frac{\sum(neutral\ values)}{\sum(all\ values)} = \frac{2}{5} = 0.4$$

Emotion analysis:

$$anger\ \% = \frac{\sum(anger\ values)}{\sum(all\ values)} = \frac{3}{5} = 0.6$$

$$disgust\ \% = \frac{\sum(disgust\ values)}{\sum(all\ values)} = \frac{1}{5} = 0.2$$

$$fear\ \% = \frac{\sum(fear\ values)}{\sum(all\ values)} = \frac{3}{5} = 0.6$$

The aggregator sends the outputs – negative for sentiment analysis, and anger, fear for emotion analysis as they have higher percentages.

Module 5. User interface:

Success Case:

When the individual words in the user inputs are all available in the dataset, a pop-up window is shown to the user informing the analysis outputs of the text (see Figure 9).

2.5 Data Structure Design

This data is retrieved from the news dataset and dictionary dataset separately. The news dataset is gained from local news agency, and the dictionary dataset is gained from an open-source website in English and translated to Azerbaijani language. In case English words cannot be translated into Azerbaijani directly as a word, they are expressed as word phrases. Additionally, if a particular English word is not found in Azerbaijani language, that word is removed from the dictionary as the input text should consist of the words existing in Azerbaijani language.

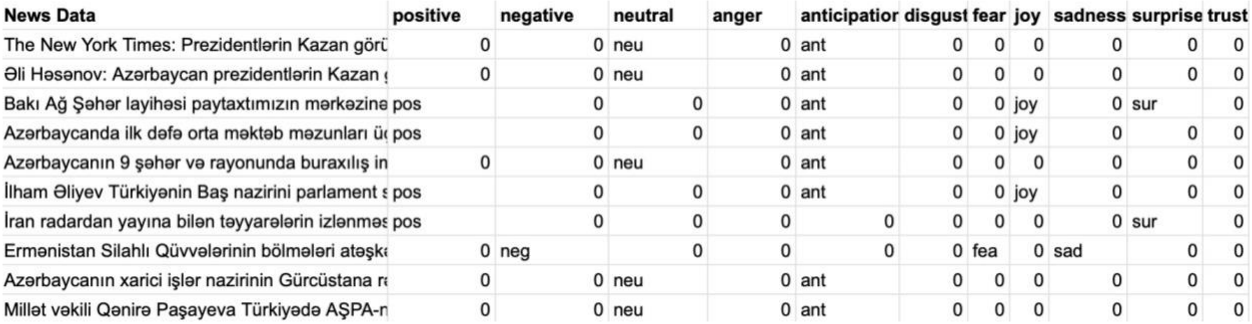
Dataset consists of two types of data, more specifically, bag of words and lexicon dictionary being stored as csv files. They are loaded into the program manually.

The structure of both data types has the following columns:

content	the data demonstrated for training and testing
positive	has value "pos" if sentiment of content has positivity
negative	has value "neg" if sentiment of content has negativity
neutral	has value "neu" if sentiment of content has neutrality
anger	has value "ang" if emotion of content is anger
anticipation	has value "ant" if emotion of content is anticipation
disgust	has value "dis" if emotion of content has disgust
fear	has value "fea" if emotion of content has fear
joy	has value "joy" if emotion of content has joy
sadness	has value "sad" if emotion of content has sadness
surprise	has value "sur" if emotion of content has surprise
trust	has value "tru" if emotion of content has trust

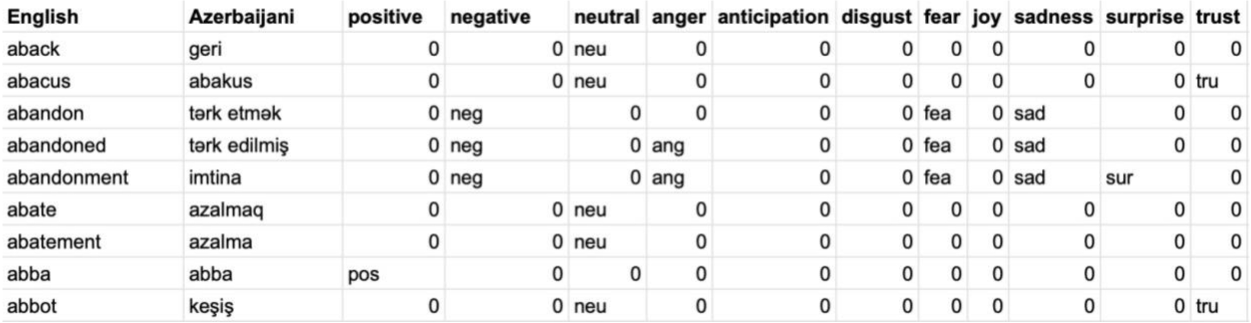
Table 4: Data types of structure

Below are the examples of datasets of bag of words and lexicon dictionary, respectively.



News Data	positive	negative	neutral	anger	anticipatior	disgust	fear	joy	sadness	surprise	trust
The New York Times: Prezidentlərin Kazan görü	0	0	neu	0	ant	0	0	0	0	0	0
Əli Həsənov: Azərbaycan prezidentlərin Kazan	0	0	neu	0	ant	0	0	0	0	0	0
Bakı Ağ Şəhər layihəsi paytaxtımızın mərkəzinə	pos	0	0	0	ant	0	0	joy	0	sur	0
Azərbaycanda ilk dəfə orta məktəb məzunları ü	pos	0	0	0	ant	0	0	joy	0	0	0
Azərbaycanın 9 şəhər və rayonunda buraxılış in	0	0	neu	0	ant	0	0	0	0	0	0
İlham Əliyev Türkiyənin Baş nazirini parlament	pos	0	0	0	ant	0	0	joy	0	0	0
İran radardan yayına bilən təyyarələrin izlənməs	pos	0	0	0	0	0	0	0	0	sur	0
Ermənistan Silahlı Qüvvələrinin bölmələri atəşk	0	neg	0	0	0	0	fea	0	sad	0	0
Azərbaycanın xarici işlər nazirinin Gürcüstana r	0	0	neu	0	ant	0	0	0	0	0	0
Millət vəkili Qənirə Paşayeva Türkiyədə AŞPA-n	0	0	neu	0	ant	0	0	0	0	0	0

Figure 6: Bag of Words Dataset example



English	Azerbaijani	positive	negative	neutral	anger	anticipation	disgust	fear	joy	sadness	surprise	trust
aback	geri	0	0	neu	0	0	0	0	0	0	0	0
abacus	abakus	0	0	neu	0	0	0	0	0	0	0	tru
abandon	tərk etmək	0	neg	0	0	0	0	fea	0	sad	0	0
abandoned	tərk edilmiş	0	neg	0	ang	0	0	fea	0	sad	0	0
abandonment	imtina	0	neg	0	ang	0	0	fea	0	sad	sur	0
abate	azalmaq	0	0	neu	0	0	0	0	0	0	0	0
abatement	azalma	0	0	neu	0	0	0	0	0	0	0	0
abba	abba	pos	0	0	0	0	0	0	0	0	0	0
abbot	keşiş	0	0	neu	0	0	0	0	0	0	0	tru

Figure 7: Lexicon Dictionary Dataset example

2.6 Overview of User Interface

The user accesses the GUI application of sentiment and emotion analysis in Azerbaijani language. The user provides a document by typing the text into an input window. The user pushes the submit button to see the analysis result. The system demonstrates two types of results, more specifically, the sentiment value and the emotion values. The system demonstrates one sentiment value out of three options, specifically positive, negative, or neutral. The system demonstrates one or more appropriate emotion values out of eight options, specifically anger, anticipation, disgust, fear, joy, sadness, surprise, and trust. The system gives an approximate prediction based on the machine learning if a word in the inputted text is not included in the trained dataset.

The SEAAL user interface will be composed of one main page and two types of output dialogs. The sketch of main page interface is as follows:

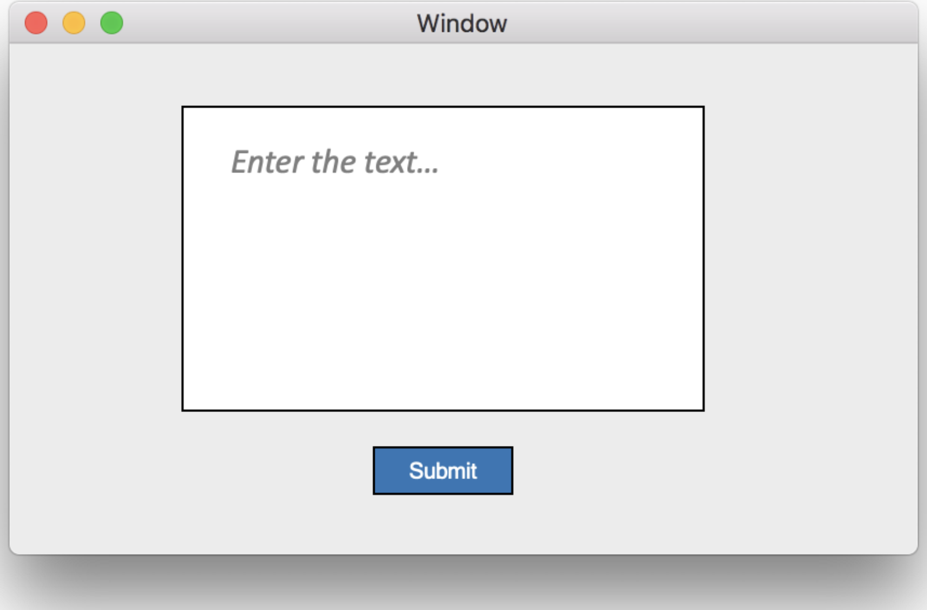
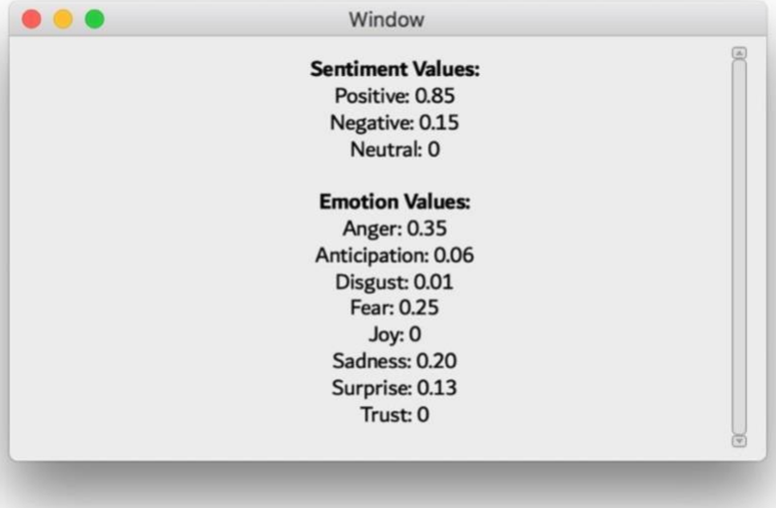


Figure 8: Main page interface

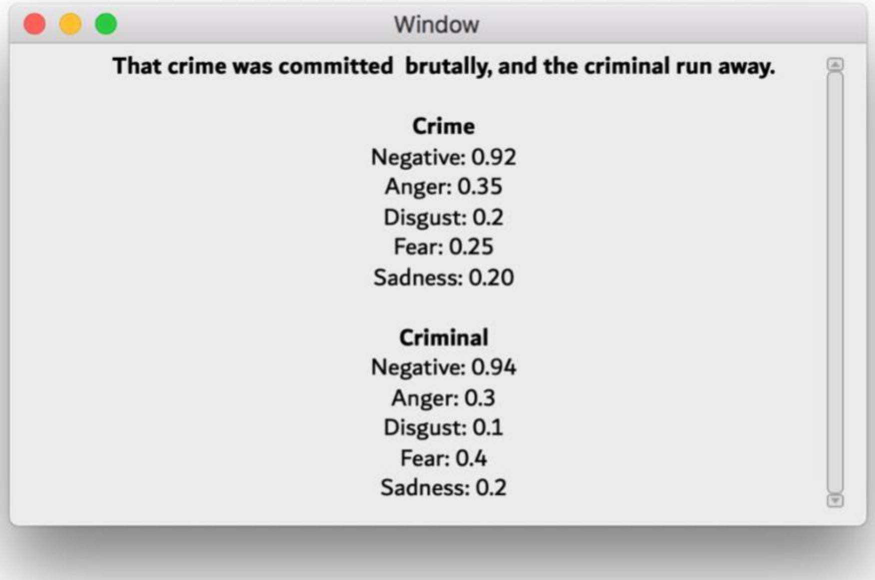
The sketch of successful output dialog interface is as follows which shows all the sentiment and emotion values with their percentages next to them:



Sentiment Values:
Positive: 0.85
Negative: 0.15
Neutral: 0

Emotion Values:
Anger: 0.35
Anticipation: 0.06
Disgust: 0.01
Fear: 0.25
Joy: 0
Sadness: 0.20
Surprise: 0.13
Trust: 0

Figure 9: Output dialog interface: Upper part of window



That crime was committed brutally, and the criminal run away.

Crime
Negative: 0.92
Anger: 0.35
Disgust: 0.2
Fear: 0.25
Sadness: 0.20

Criminal
Negative: 0.94
Anger: 0.3
Disgust: 0.1
Fear: 0.4
Sadness: 0.2

Figure 10: Output dialog interface: Lower part of window

2.7 Entity recognition

Named as entity recognition or entity detection, means extracting the entities out of all the inputted document, which can refer to people, places, entities, or the main character of the text. Mainly, there exist four techniques to detect entities from the text, including:

- The statistical-based recognition method
- The rule-based recognition method
- Combination of statistical and rule-based methods
- Machine learning-based recognition method

The statistical-based recognition method – is using statistics from the given examples to recognize the entities from the given text. To be more specific, it includes gathering observations, learning, and analyzing them in order to deduct standard rules or concepts that can be used to predict unseen inputs. The statistical-based recognition methods include hidden Markov model, CoreNLP, decision tree model, CNN, support vector machine (SVM) model, maximum entropy model, and conditional random fields model.

The rule-based recognition method – is setting certain rules for the machine to detect the entities of the text. It mainly consists of four steps:

- Topic-related data pre-processing
- Text tokenizing and tagging dataset
- Sentiment and emotion tagging of each word in the text
- Aspect level recognition

Combination of statistical and rule-based methods – is using both statistical and rule-based methods to construct a mutual model. The table below shows the combined model and their accuracies on aspect level analysis [24]:

Domain	Classifier	% of Accurate aspects	Precision	Recall	F-Score
cellphone	CoreNLP + Rule-based	65.3	72.4	75.55	74.86
cellphone	CNN	71.3	75.68	85.15	80.56
cellphone	CNN + Rule-based	75	79.24	88.4	82.34
camera	CoreNLP + Rule-based	59.8	73.6	79.57	75.5
camera	CNN	68.7	76.6	88.87	78.5
camera	CNN + Rule-based	72.4	78.79	89.9	80.5
laptop	CoreNLP + Rule-based	64.8	73.9	81.53	79.45
laptop	CNN	71.4	76.9	85.23	82.35
laptop	CNN + Rule-based	77.4	79.25	88.45	83.24
restaurant	CoreNLP + Rule-based	59.4	74.46	80.8	79.55
restaurant	CNN	67.4	77.56	84.8	81.45
restaurant	CNN + Rule-based	74.4	79.67	86.2	83.34
movie review	CoreNLP + Rule-based	63.7	74.26	78.8	75.55
movie review	CNN	69.4	75.36	79.8	78.45
movie review	CNN + Rule-based	75.6	78.67	82.2	80.34

Table 5: Comparison of CNN, CoreNLP, and their combinations with rule-based

Machine learning-based recognition method – is applying neurons of parallel structures to retrieve aspects of the given text. One of the well-known applications of it is Feed-Forward Backpropagation neural network (FFBPNN).

In this project, rule-based recognition methods are used to detect aspects of the document. To be more specific, grammar rules of Azerbaijani language are followed by considering also semantic meanings. The rules used for the current project are listed below:

Rule 1: If a word in a sentence ends with suffixes ["nı", "ni", "nu", "nü", "ı", "i", "u", "ü"] and, if they are not the part of the core of the word, and if the stemmed version of a word is noun, and if the word is not a stop-word, label it as an entity.

Rule 2: If no entity found from the previous rule, and, if the word included in the pronouns, and, if the word is not included in the stop-words, label it as an entity.

Rule 3: If no entity found from the previous rules, and, if the word does not have any suffixes except plurality suffixes, and, if the word is not included in the stop-words, and, if the word is noun, and, if the word is not the last one of the sentence, and, if the word's first letter is uppercase and as well as the next word, then label them as an entity. Else, if the word's first letter is uppercase and as well as the previous word, then label them as an entity. Else, if the next word is a verb, label it as an entity.

Rule 4: If no entity found from the previous rules, and, if the word starts with uppercase letter and it is not the first word of the sentence, and, if the word is not included in the stop-words, label it as an entity.

Rule 5: If no entity found from the previous rules, and, if the word is not included in the stop-words, and, if the word is a noun, label it as an entity.

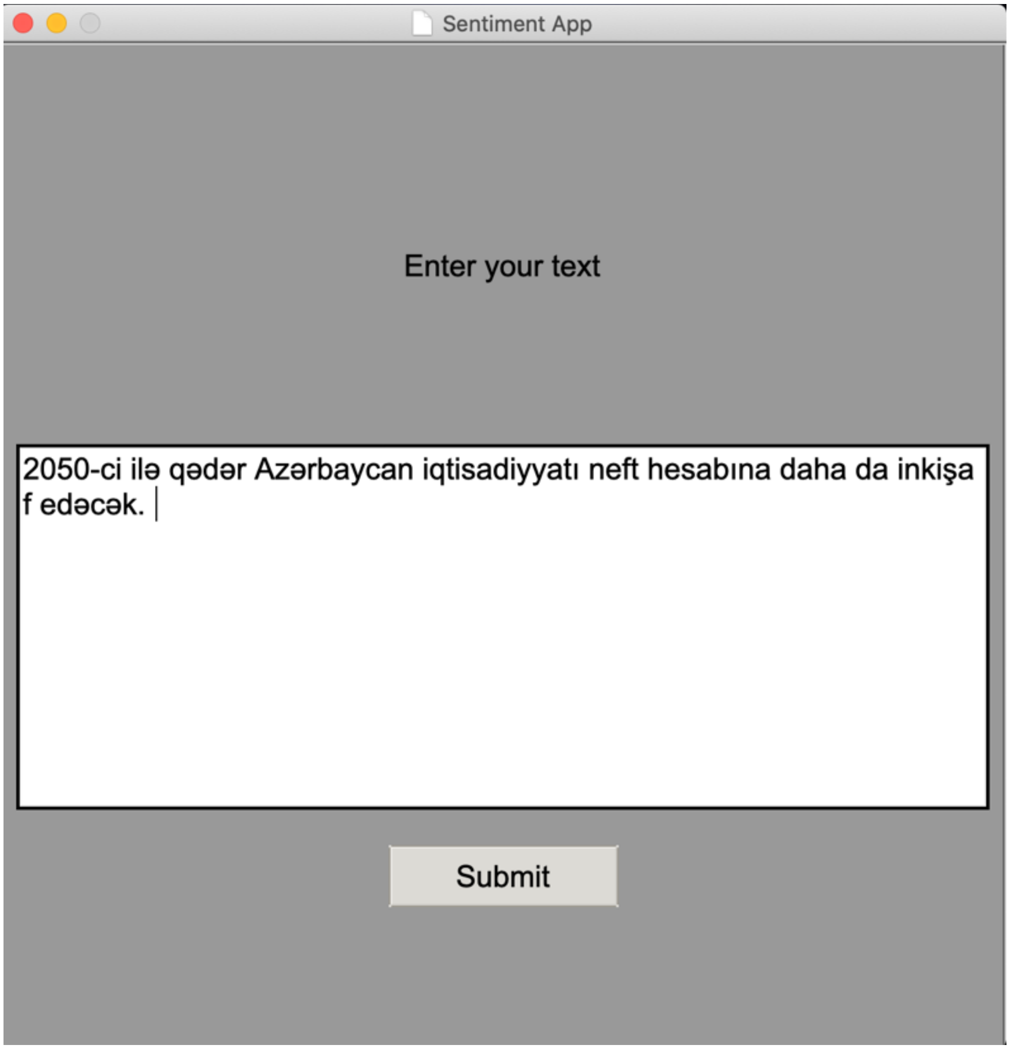
The table below shows some of the real examples and the outputs based on the applied rule-based entity recognition:

Input	Entity recognition output
Şərq küləyi əsəcək.	Şərq küləyi
Azərbaycan Respublikasının Prezidenti İlham Əliyevin sərəncamı ilə 2017/2018-ci tədris ilində Prezident təqaüdünə layiq görülən 102 nəfərdən 24-ü Bakı Ali Neft Məktəbinin (BANM) tələbəsi oldu.	BANM tələbəsi

Beləliklə, BANM I ixtisas qrupu üzrə ölkə lideri oldu.	ölkə lideri
Belə ki, buna qədər böyük rəqabətin getdiyi I qrupda heç bir universitet maksimal sayda Prezident təqaüdçüsü ilə təmsil olunmamışdır.	Prezident təqaüdçüsü
Son məlumata görə, dağıntılar altında 2 nəfər qalmaqdadır.	dağıntılar
Saatlı rayonu ərazisindəki fermalardan birindən 2 baş qoyun oğurlanması barədə Rayon Polis Şöbəsinə müraciət daxil olub.	qoyun oğurlanması
Dünyasını dəyişən ananın ölüm səbəbi isə tibbi ekspertizanın rəyindən sonra bəlli olacaq.	ölüm səbəbi
Azərbaycanın müvafiq qurumları valideynləri İŞİD terror təşkilatına qoşulan və hazırda İraqda qalan azərbaycanlı uşaqların ölkəyə gətirilməsi istiqamətində işlər aparır.	ölkəyə gətirilməsi
Uzun müddətdir Faytonçu Nazim səhhətindəki problemlərlə bağlı efirdən uzaqlaşmışdı.	Faytonçu Nazim
O, Azərbaycanda həkim səhvi ucbatından xəstə olduğunu və yalnış müalicə ucbatından səhhətində problemlərin daha da kəskinləşdiyindən gileylənib.	həkim səhvi
O, Ülvinin hazırlığa gəlmədiyini bildirib.	O
“Hazırda səhhəti normaldır, amma hələlik narkozdan ayılmayıb.	səhhəti

Table 6: Entity recognition examples based on rule-based model

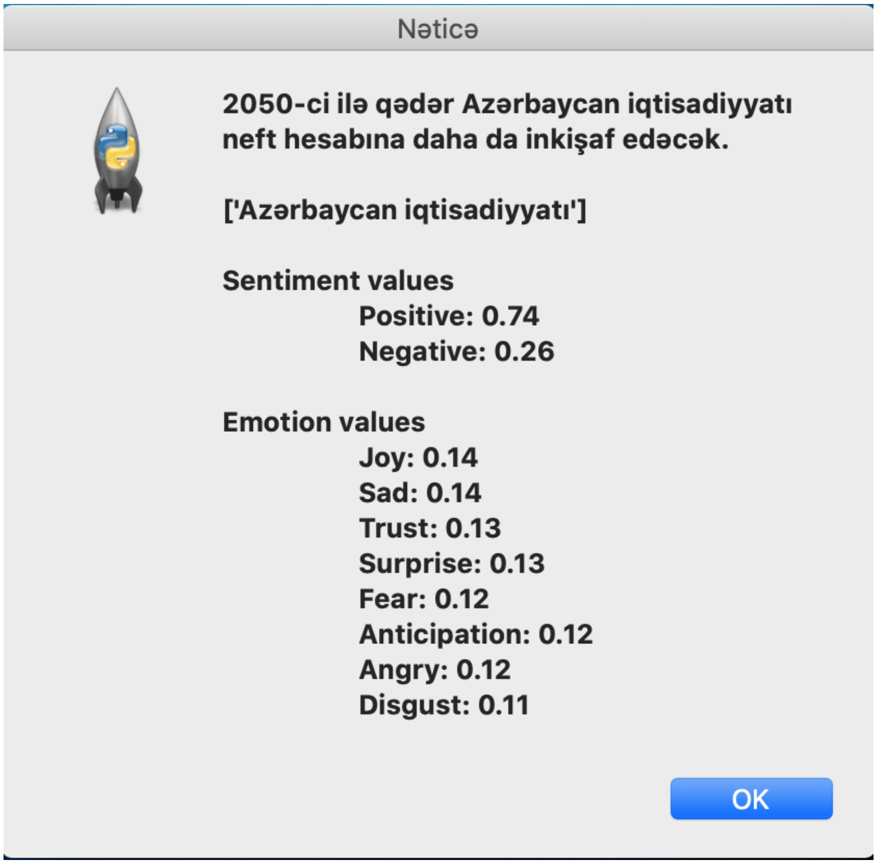
The figures below show the real input and output from the program:



2050-ci ilə qədər Azərbaycan iqtisadiyyatı neft hesabına daha da inkişa
f edəcək.

Figure 11: Input to the SEAAL program

Input sentence – “2050-ci ilə qədər Azərbaycan iqtisadiyyatı neft hesabına daha da inkişaf edəcək.”



```
2050-ci ilə qədər Azərbaycan iqtisadiyyatı
neft hesabına daha da inkişaf edəcək.

['Azərbaycan iqtisadiyyatı']

Sentiment values
            Positive: 0.74
            Negative: 0.26

Emotion values
            Joy: 0.14
            Sad: 0.14
            Trust: 0.13
            Surprise: 0.13
            Fear: 0.12
            Anticipation: 0.12
            Angry: 0.12
            Disgust: 0.11
```

Figure 12: Output of the SEAAL program

Entity – “Azərbaycan iqtisadiyyatı”
Sentiment value – Positive
Emotion values – Joy, Trust, Surprise

3 RESEARCH RESULTS AND ANALYSIS OF RESULTS

Despite of the fact thar four machine learning algorithms have been applied to the Sentiment and Emotion Analysis in Azerbaijani language, this project focuses on the highest achieved accuracy model – Artificial Neural Network (ANN).

3.1 Implementation

Below the sequential lines of codes are represented for the application of the SEALL program:

Firstly, we import the needed packages to call afterwards:

```
import pandas as pd
import numpy as np

from nltk.tokenize import sent_tokenize
from nltk.tokenize import  word_tokenize
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.neural_network import MLPClassifier
from soz_analizi import *
from sklearn import metrics

import pickle
import tkinter as tk
from tkinter import ttk
from tkinter import messagebox
```

Reading our data:

```
news = pd.read_csv("news.csv", low_memory=False)
```

Initializing arrays that will be used in the next lines:

```
objectSuffix1 = ["ı", "i", "u", "ü"]
objectSuffix2 = ["nı", "ni", "nu", "nü"]
pluralSuffix = ["lar", "lər"]
pronouns = ["mən", "sən", "biz", "siz", "onlar"]
stopWords = ["olan", "saat", "zamanı", "görə", "üçün", "onu", "ona", "rayonu", "gün", "üzrə", "ilə", "hazırda",
"edilən", "edən", "olaraq", "bir", "də", "və", "ki", "vaxt", "kimi"]
stopVerbs = ["xəbər", "baş", "qeyd", "dünyasını", "təhvil", "diqqət", "təqdim", "təhlil", "təsvir", "təsir", "müdafiə",
"səbəb"]
```

```
emotionList = ["Angry", "Anticipation", "Disgust", "Fear", "Joy", "Sad", "Surprise", "Trust"]
```

Parsing our data as sentiment and emotion values:

```
posNews = news[news.positive == 'pos']
negNews = news[news.negative == 'neg']
neuNews = news[news.neutral == 'neu']
angNews = news[news.anger == 'ang']
antNews = news[news.anticipation == 'ant']
disNews = news[news.disgust == 'dis']
feaNews = news[news.fear == 'fea']
joyNews = news[news.joy == 'joy']
sadNews = news[news.sadness == 'sad']
surNews = news[news.surprise == 'sur']
truNews = news[news.trust == 'tru']
```

Using TF-IDF Vectorizer for feature extraction:

```
sentVector = TfidfVectorizer(norm = None)
emotVector = TfidfVectorizer(norm = None)
```

Combining all the sentiment, emotion data and labels into one variable:

```
sentData = posList + negList + neuList
sentLabels = pos + neg + neu
emotData = angList + antList + disList + feaList + joyList + sadList + surList + truList
emotLabels = ang + ant + dis + fea + joy + sad + sur + tru
```

Converting those variables into numpy array:

```
sentData = np.array(sentData)
sentLabels = np.array(sentLabels)
emotData = np.array(emotData)
emotLabels = np.array(emotLabels)
```

Splitting sentiment data into train and test corpus:

```
sentTrainCorpus, sentTestCorpus, sentTrainLabels, sentTestLabels = train_test_split(sentData, sentLabels,
test_size=0.2)
```

Splitting emotion data into train and test corpus:

```
emotTrainCorpus, emotTestCorpus, emotTrainLabels, emotTestLabels = train_test_split(emotData, emotLabels,
test_size=0.2)
```

Fitting and transforming our sentiment train data:

```
sentTrain = sentVector.fit_transform(sentTrainCorpus)
sentTest = sentVector.transform(sentTestCorpus)
sentTest.shape
```

Fitting and transforming our emotion train data:

```
emotTrain = emotVector.fit_transform(emotTrainCorpus)
emotTest = emotVector.transform(emotTestCorpus)
emotTest.shape
```

Assigning MLP as a classifier to our model for sentiment data:

```
sentMLP = MLPClassifier()
sentMLP.fit(sentTrain, sentTrainLabels)
```

Assigning MLP as a classifier to our model for emotion data:

```
emotMLP = MLPClassifier()
emotMLP.fit(emotTrain, emotTrainLabels)
```

Saving our sentiment and emotion model for future running:

```
with open('model.pickle', 'wb') as f:
  pickle.dump(sentMLP, f)
  pickle.dump(emotMLP, f)
```

Saving our sentiment and emotion vector for future running:

```
with open('vect_ld.pickle', 'wb') as f:
  pickle.dump(sentVector, f)
  pickle.dump(emotVector, f)
```

Predicting sentiment and emotion values from test data:

```
sentPredictions = sentMLP.predict(sentTest)
emotPredictions = emotMLP.predict(emotTest)
```

Predicting accuracy for sentiment data:

```
sentAccuracy = np.sum(sentPredictions == sentTestLabels) / len(sentTestLabels)
print('--------\nSentiment Accuracy: ', sentAccuracy)
```

Predicting accuracy for emotion data:

```
emotAccuracy = np.sum(emotPredictions == emotTestLabels) / len(emotTestLabels)
print('Emotion Accuracy: ', emotAccuracy)
```

Printing Confusion matrix and Classification Report for Sentiment values:

```
print('Confusion Matrix and Classification Report for Sentiment values')
print(metrics.classification_report(sentTestLabels, sentPredictions, target_names=sorted(set(sentLabels))))
print(metrics.confusion_matrix(sentTestLabels, sentPredictions, labels=sorted(set(sentLabels))))
```

Printing Confusion matrix and Classification Report for Emotion values:

```
print('Confusion Matrix and Classification Report for Emotion values')
print(metrics.classification_report(emotTestLabels, emotPredictions, target_names=sorted(set(emotLabels))))
print(metrics.confusion_matrix(emotTestLabels, emotPredictions, labels=sorted(set(emotLabels))))
```

Reading saved pickle files for not training data every time:

```
with open('model.pickle', 'rb') as f:
    sentMLP = pickle.load(f)
    emotMLP = pickle.load(f)


with open('model.pickle', 'rb') as f:
    sentVector = pickle.load(f)
    emotVector = pickle.load(f)
```

Clearing the user input from punctuations:

```
punctuation= '''!()[]{};:,'"”\<>/?@#$%^&*_~'''
```

```
        userInput = userInputWithPunc.translate(str.maketrans("","",punctuation))
```

Tokenizing input into sentences:

```
sentences = sent_tokenize(userInput)
```

Tokenizing those sentences into words, creating two dimensional array:

```
words = []
        for i in sentences:
            words.append(word_tokenize(i))
```

Stemming to get the core of the words:

```
for j in range(len(words)):
            for i in range(len(words[j])):
                if(baslangic(words[j][i], 'u') != ''):
                    stemmedWords[j][i] = baslangic(words[j][i], 'u')
```

Checking if the stemmed is noun, if yes, appending them into array:

```
for i in range(len(sentences)):
            for j in range(len(stemmedWords[i])):
                if(isNoun(stemmedWords[i][j].lower())):
                    nouns[i][j] = stemmedWords[i][j]
```

Entity recognition, *rule 1*:

```
#Object Suffixes
        for j in range(len(words)):
            for i in range(1,len(words[j])):
                if (((words[j][-i][-2:] in objectSuffix2) or (words[j][-i][-1:] in objectSuffix1)) and  #if last letters are an
object suffix
                    (words[j][-i].lower() != stemmedWords[j][-i].lower()) and   #if object suffixes are not a part of root
                    (words[j][-i].lower() not in stopWords) and #if word is not a stop-word
                    (isNoun(words[j][-i].lower()))  ):  #if word is noun
                    if words[j][-i-1].lower() not in stopWords: #if previous word is not a stop-word print a phrase
                        entities[j][i] = words[j][-i-1] + " " + words[j][-i]
            else: #if previous word is a stop-word print a word
                entities[j][i] = words[j][-i]
print("Object Suffixes: ", entities)
```

Entity recognition, *rule 2:*

```
#Pronouns
for j in range(len(words)):
    for i in range(len(words[j])):
        if(all(l == '0' for l in entities[j])): #if no entity found yet
            if ((words[j][i].lower() in pronouns) and  #if word is pronoun
               (words[j][i].lower() not in stopWords)):  #if word is not a stop-word
                entities[j][i] = words[j][i]
print("Pronouns: ", entities)
```

Entity recognition, *rule 3:*

```
#Subject
for j in range(len(words)):
    for i in range(len(words[j])):
        if(all(l == '0' for l in entities[j])): #if no entity found yet
            if(words[j][0][0] == "i"): #speacial "i" case, beacuse lower() function does not lower properly this
letter
                temp = list(words[j][i])
                temp[0] = "i"
                temp = "".join(temp)
                words[j][0] = temp

            if((words[j][i].lower() == stemmedWords[j][i].lower() #if there is no suffixes
                    or words[j][i][-3:] in pluralSuffix) and #or if there is only plural suffixes
                (isNoun(words[j][i].lower())) and #if word is noun
                (i != len(words[j])) and #if word is not last word of the sentence
                (words[j][i].lower() not in stopWords)): #if word is not a stop-word
                if(words[j][i][0].isupper() and words[j][i-1][0].isupper()): #if previous word is uppercase special
noun
```

```
              entities[j][i] = words[j][i-1] + " " + words[j][i]
            elif(words[j][i][0].isupper() and words[j][i+1][0].isupper()): #if next word is uppercase special noun
              entities[j][i] = words[j][i] + " " + words[j][i+1]
            elif((words[j][i].lower() in stopVerbs)):  #if word is a stop-verb
              if(not feildi(words[j][i+1])): #if next word is verb
                entities[j][i] = words[j][i]
            else: #if word is not a stop-verb
             entities[j][i] = words[j][i]
    print("Subject: ", entities)
```

Entity recognition, *rule 4:*

```
#Special Nouns
    for j in range(len(words)):
      for i in range(len(words[j])):
        if(all(l == '0' for l in entities[j])): #if no entity found yet
          if ((words[j][i][0].isupper()) and #if words starts with an uppercase letter
            i != 0 and  #if word is not the first word of sentence
            (words[j][i] not in stopWords)): #if word is not a stop-word
            entities[j][i] = words[j][i]
    print("Special Nouns: ", entities)
```

Entity recognition, *rule 5:*

```
    #Last Else - first noun in the sentence
    for j in range(len(words)):
      for i in range(len(words[j])):
        if(all(l == '0' for l in entities[j])):
          if (isNoun(words[j][i].lower()) and #if word is noun
            (words[j][i] not in stopWords)): #if word is not a stopword
            entities[j][i] = words[j][i]
    print("Last else: ", entities)
```

Predicting the sentiment and emotion values of user input:

```
sentProba = sentMLP.predict_proba(sentVector.transform([existingWordsJoined[i]]))
```

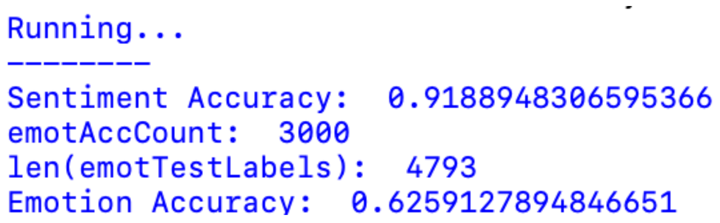
```
emotProba = emotMLP.predict_proba(emotVector.transform([existingWordsJoined[i]]))
```

Printing the output in the Python GUI:

```
if(len(entities[i]) == 0):
            messagebox.showinfo("Nəticə",
            f'''{sentences[i]}
            \nSentiment values
            Positive: {round(posProba, 2)}
            Negative: {round(negProba, 2)}
            \nEmotion values
           {emotionList[emotProbaMap[7][0]]}: {round(emotProbaMap[7][1], 2)}
           {emotionList[emotProbaMap[6][0]]}: {round(emotProbaMap[6][1], 2)}
           {emotionList[emotProbaMap[5][0]]}: {round(emotProbaMap[5][1], 2)}
           {emotionList[emotProbaMap[4][0]]}: {round(emotProbaMap[4][1], 2)}
           {emotionList[emotProbaMap[3][0]]}: {round(emotProbaMap[3][1], 2)}
           {emotionList[emotProbaMap[2][0]]}: {round(emotProbaMap[2][1], 2)}
           {emotionList[emotProbaMap[1][0]]}: {round(emotProbaMap[1][1], 2)}
           {emotionList[emotProbaMap[0][0]]}: {round(emotProbaMap[0][1], 2)}
            ''')
```

3.2 Output analysis

Here is the printed accuracy for sentiment and emotion values:



```
Running...
---------
Sentiment Accuracy:  0.9188948306595366
emotAccCount:  3000
len(emotTestLabels):  4793
Emotion Accuracy:  0.6259127894846651
```

4 SUMMARY AND CONCLUSIONS

The product is designed for marketing executors to see how their customers feel about different areas of their business in Azerbaijan. By using this system, the analysis process can be automated, and customer’s reviews can be evaluated without reading them manually. In this phase of the project which aims at developing a documented software application, emphasis shall be put on Azerbaijani data as a bag of words.

The project aims to provide accurate sentiment and emotion analysis results through a GUI (Graphical User Interface) application. The key goal of this project is to analyze and interpret the results of a sentiment and emotion analysis in Azerbaijani language using ML (Machine Learning) algorithms.

Unlike English, machine learning algorithms have not been applied to Azerbaijani language. The key goal of this project is to initiate sentiment and emotion analysis in Azerbaijani language. To be more specific, the goal is to create reliable software that will work with the input texts in Azerbaijani to which the concepts of machine learning methods are applied.

5 BIBLIOGRAPHY (ACM/IEEE STANDARD)

[1] OscarAraque, IgnacioCorcuera-Platas, J. FernandoSánchez-Rada, and Carlos A.Iglesias, "Enhancing deep learning sentiment analysis with ensemble techniques in social applications", Expert Systems with Applications, vol. 77, pp. 236-246, 1 July 2017.

[2] Y. Fang, H. Tan and J. Zhang, "Multi-Strategy Sentiment Analysis of Consumer Reviews Based on Semantic Fuzziness," IEEE Access, vol. 6, pp. 20625-20631, 2018.

[3] NeelamMukhtar, Mohammad AbidKhan, and NadiaChiragh, "Lexicon-based approach outperforms Supervised Machine Learning approach for Urdu Sentiment Analysis in multiple domains", Telematics and Informatics, vol. 35, no. 8, pp. 2173-2183, December 2018.

[4] AsadAbdi, Siti MariyamShamsuddin, ShafaatunnurHasan, and JalilPiranMD, "Machine learning-based multi-documents sentiment-oriented summarization using linguistic treatment", Expert Systems with Applications, vol. 109, pp. 66-85, 1 November 2018.

[5] MohammadAl-Smadi, OmarQawasmeh, MahmoudAl-Ayyoub, YaserJararweh, and BrijGupta, "Deep Recurrent neural network vs. support vector machine for aspect-based sentiment analysis of Arabic hotels' reviews", Journal of Computational Science, vol. 27, pp. 386-393, July 2018.

[6] S. E. Saad and J. Yang, "Twitter Sentiment Analysis Based on Ordinal Regression," IEEE Access, vol. 7, pp. 163677-163685, 2019.

[7] M. Afzaal, M. Usman and A. Fong, "Tourism Mobile App With Aspect-Based Sentiment Classification Framework for Tourist Reviews," IEEE Transactions on Consumer Electronics, vol. 65, no. 2, pp. 233-242, May 2019.

[8] A. Feizollah, S. Ainin, N. B. Anuar, N. A. B. Abdullah and M. Hazim, "Halal Products on Twitter: Data Extraction and Sentiment Analysis Using Stack of Deep Learning Algorithms," IEEE Access, vol. 7, pp. 83354-83362, 2019.

[9] ParamitaRay, and AmlanChakrabarti, "A Mixed approach of Deep Learning method and Rule- Based method to improve Aspect Level Sentiment Analysis", Applied Computing and Informatics, Available online 4 March 2019.

[10] ZiyuanZhao, HuiyingZhu, ZehaoXue, ZhaoLiu, JingTian, Matthew Chin HengChua, and MaofuLiu, "An image-text consistency driven multimodal sentiment analysis approach for social media", Information Processing & Management, vol. 56, no. 6, November 2019.

[11] SaeromPark, JaewookLee, and KyoungokKim, "Semi-supervised distributed representations of documents for sentiment analysis", Neural Networks, vol. 119, pp. 139-150, November 2019.

[12] SrishtiVashishtha, and SebaSusan, "Fuzzy rule based unsupervised sentiment analysis from social media posts", Expert Systems with Applications, vol. 138, 30 December 2019.

[13] AbdallahYousif, ZhendongNiu, JamesChambua, and Zahid YounasKhan, "Multi-task learning model based on recurrent convolutional neural networks for citation sentiment and purpose classification", Neurocomputing, vol. 335, pp. 195-205, 28 March 2019.

[14] AsadAbdi, Siti MariyamShamsuddin, ShafaatunnurHasan, and JalilPiran, "Deep learning- based sentiment classification of evaluative text based on Multi-feature fusion", Information Processing & Management, vol. 56, no. 4, pp. 1245-1259, July 2019.

[15] RonitaBardhan, MinnaSunikka-Blank, and Anika NasraHaque, "Sentiment analysis as tool for gender mainstreaming in slum rehabilitation housing management in Mumbai, India", Habitat International, vol. 92, October 2019.

[16] AkshiKumar, KathiravanSrinivasan, ChengWen-Huang, and Albert Y.Zomaya, "Hybrid context enriched deep learning model for fine-grained sentiment analysis in textual and visual semiotic modality social data", Information Processing & Management, vol. 57, no. 1, January 2020.

[17] Mohammad A.Hassonah, RizikAl-Sayyed, AliRodan, Ala' M.Al-Zoubi, IbrahimAljarah, and HossamFaris, "An efficient hybrid filter and evolutionary wrapper approach for sentiment analysis of various topics on Twitter", Knowledge-Based Systems, vol. 192, 15 March 2020.

[18] FengXu, ZhenchunPan, and RuiXia, "E-commerce product review sentiment classification based on a naïve Bayes continuous learning framework", Information Processing & Management, Available online 13 February 2020.

[19] HaiderMaqsood, IrfanMehmood, MuazzamMaqsood, MuhammadYasir, SitaraAfzal, FarhanAadil, Mahmoud MohamedSelim, and KhanMuhammad, "A local and global event sentiment based efficient stock exchange forecasting using deep learning", International Journal of Information Management, vol. 50, pp. 432-451, February 2020.

[20] Hyun-jungPark, MinchaeSong, and Kyung-ShikShin, "Deep learning models and datasets for aspect term sentiment classification: Implementing holistic recurrent attention on target- dependent memories", Knowledge-Based Systems, vol. 187, January 2020.

[21] Quinlan, J. R. (1986). "Induction of decision trees" (PDF). Machine Learning. 1: 81– 106. doi:10.1007/BF00116251. S2CID 189902138.

[22] Zhang, Harry. The Optimality of Naive Bayes (PDF). FLAIRS2004 conference.

[23] Sharma, Anuj, and Shubhamoy Dey. "A document-level sentiment analysis approach using artificial neural network and sentiment lexicons." ACM SIGAPP Applied Computing Review 12.4 (2012): 67-75.

[25] Rustamov S., Hasanli H. “Sentiment Analysis of Azerbaijani twits Using Logistic Regression, Naive Bayes and SVM” 2019

[26] Rustamov S., Mustafayev E. “Sentiment analysis using Neuro-Fuzzy and Hidden Markov models of text” 2013

[27] Ray, Chakrabarti. “A Mixed approach of Deep Learning method and Rule-Based method to improve Aspect Level Sentiment Analysis.” 2018-2019. P 174

[28] Ross, Buck & Oatley, Keith. (2007). Robert Plutchik (1927–2006). American Psychologist - AMER PSYCHOL. 62. 142-142. 10.1037/0003-066X.62.2.142.

[29] G, Devi & Somasundaram, Kamalakkannan. (2020). Literature Review on Sentiment Analysis in Social Media: Open Challenges toward Applications. 29. 1462-1471.