



## Tools for Consumer Preference Analysis Based in Machine Learning

**Vitalina Babenko\*** 

\*Corresponding author, Professor, Head of Computer Systems Department, Kharkiv National Automobile and Highway University, Kharkiv, 61002, Ukraine; Daugavpils University, Daugavpils, LV – 5401, Latvia. E-mail: vitalinababenko@karazin.ua

**Sergii Pronin** 

Assistant Professor, Computer Systems Department, Kharkiv National Automobile and Highway University, Kharkiv, 61002, Ukraine. E-mail: psv59777@gmail.com

**Ludmila Aleksejeva** 

Associate professor, Head of the Department of Law, Management & Economics, Daugavpils University, Daugavpils, LV-5401, Latvia. E-mail: ludmila.aleksejeva@du.lv

**Zaiga Vītola** 

Associate professor, Department of Law, Management & Economics, Daugavpils University, Daugavpils, LV-5401, Latvia. E-mail: zaiga.vitola@du.lv

**Liudmyla Sokolova** 

Professor, Dr. Sci. (habil.) in Economics, Department of Economic Cybernetics and Economic Security Management, Kharkiv National University of Radio Electronics, Kharkiv, 61166, Ukraine. E-mail: liudmyla.sokolova@nure.ua

**Viktor Dyuzhev** 

Professor, Dr. Sci. (habil.) in Economics, Department of Business Economics and International Economic Relations, National Technical University "Kharkiv Polytechnic Institute", 61002, Kharkiv, Ukraine. E-mail: ekosistema999@gmail.com

**Nataliya Dyakova** 

Associate professor, Department of Business Economics and International Economic Relations National Technical University "Kharkiv Polytechnic Institute", 61002, Kharkiv, Ukraine. E-mail: Nataliya.Dyakova@kphi.edu.ua

---

## Abstract

Today, users generate various data increasingly using the Internet when choosing a product or service. This leads to the generation of data about the purchases and services of various consumers. In addition, consumers often leave feedback about the purchase. At the same time, consumers discuss their attitudes about goods and services on social networks, messengers, thematic sites, etc. This leads to the emergence of large volumes of data that contain useful information about various manufacturers of goods and services. Such information can be useful to both ordinary users and large companies. However, it is practically impossible to use this information due to the fact that it is located in different places, that is, it has a raw, unstructured character. At the same time, depending on the target group of users, not the entire data set is needed, but a specific target sample. To solve this problem, it is necessary to have a tool for structuring information arrays and their further analysis depending on the set goal. This can be done with the help of various frameworks that use methods of machine learning and work with data. This work is devoted to elucidating the problem of creating means for evaluating consumer preferences based on the analysis of large volumes of data for its further use by the target audience. The goal of the development of big data analysis systems is obtaining new, previously unknown information. The methodology of application of algorithms of work with large data sets and methods of machine learning is used, namely the pandas library for operations on a data set and logistic regression for information classification. As a result, a system was built that allows the analysis of lexical information, translate it into numerical format and create on this basis the necessary statistical samples. The originality of the work lies in the use of specialized libraries of data processing and machine learning to create data analysis systems. The practical value of the work lies in the possibility of creating data analysis systems built using specialized machine learning libraries.

**Keywords:** Machine learning, Data analysis, Pandas, Data set.

Journal of Information Technology Management, 2024, Vol. 16, Issue 4, pp. 1-17

Published by the University of Tehran, College of Management

doi: <https://doi.org/10.22059/jitm.2024.99048>

Article Type: Research Paper

© Authors

Received: June 21, 2024

Received in revised form: July 15, 2024

Accepted: September 04, 2024

Published online: October 16, 2024



## Introduction

Today, users increasingly use the Internet when choosing a product or service. This leads to the generation of data about the purchases and services of various consumers. In addition, consumers often leave feedback about the purchase. At the same time, consumers discuss their attitudes about goods and services on social networks, messengers, thematic sites, etc. This leads to the emergence of large volumes of data that contain useful information about

various manufacturers of goods and services. Such information can be useful to both ordinary users and large companies. However, it is practically impossible to use this information due to the fact that it is located in different places, that is, it has a raw, unstructured character. At the same time, depending on the target group of users, not the entire data set is needed, but a specific target sample. To solve this problem, it is necessary to have a tool for structuring information arrays and their further analysis depending on the set goal. This can be done with the help of various frameworks that use methods of machine learning and work with data.

This work is devoted to elucidating the problem of creating means for evaluating consumer preferences based on the analysis of large volumes of data for its further use by the target audience.

## **Literature Review**

Machine learning technologies for solving data analysis problems began almost half a century ago. During this period, many methods were developed, which are commonly referred to as machine learning, and the principle of their application was formulated, and the main obstacle was the insufficient computing capabilities of the computers at that time. The situation changed at the end of the 20th century thanks to the growth of computing power of computers, increasing the capabilities of operating systems, the development of programming languages, and the development of various specialized frameworks that represent ready-made libraries of machine learning and data analysis functions. This made it possible to develop applications for solving tasks that are solved using machine learning (Babenko et al., 2019; 2021; Lepelaar et al., 2022; Naldi et al., 2019; Ramirez et al., 2019).

In its general form, the machine learning algorithm requires the following actions (Janssens 2014; Guryanova et al., 2020, 2021; Karau et al., 2020):

- collect the necessary information from the subject area;
- create an information array that can be used in the application;
- divide this array into educational and test parts. In the first part, the model will be trained, and in the second part, we will be able to check the results of training;
- choose a problem solving algorithm that will process requests.

Further, the system can be configured in such a way that it works in the training mode, adding new data and correcting the result of its work.

In this way, by applying machine learning technologies, we somehow configure the computer so that it "learns" to extract useful knowledge from any data.

As can be seen from the description of actions to create machine learning systems, the issue of data accumulation, storage and organization is important. At the same time, the data must be organized into special structures so that convenient work with them is possible, namely, convenient access to data, formation of the necessary data set, structuring and filtering of data, and so on.

All this will make it possible to work with them further using different models, which will give the desired result at the end (Janssens, 2014; Guryanova et al., 2020).

Today, for this, you can use the technologies that have received the common name "Big data" (Big data).

Big data technologies include the following:

- Apache Spark is a complete computing system with a set of libraries for parallel data processing on computer clusters (Karau et al., 2020) Spark is currently considered to be the most actively developed open source tool for solving such tasks, making it a useful tool for any developer or specialist researcher interested in big data. Spark supports many widely used programming languages (Python, Java, Scala, and R), as well as libraries for various tasks, from SQL to streaming and machine learning, and can be run from a laptop or from a cluster, which consists of thousands of servers. Thanks to this, Apache Spark is a convenient system for initially independent work, which flows into the processing of big data on a large scale.

- MapReduce is a model of distributed calculations from Google, which is used in Big Data technologies for parallel calculations on very large (up to several petabytes) data sets in computer clusters, and a framework for calculating distributed tasks on cluster nodes (White, 2012).

MapReduce can rightfully be called the main Big Data technology, because it is primarily focused on parallel computing in distributed clusters. The essence of MapReduce consists in dividing the information array into parts, parallel processing of each part on a separate node and the final unification of all results.

Programs using MapReduce are executed on distributed nodes of the cluster, while the execution system itself takes care of the implementation details.

The technology is practically universal: it can be used for indexing web content, counting words in a large file, counters the frequency of visits to a given address, calculating the volume of all web pages from each URL address of a specific host node, creating a list of all addresses with the necessary data and other tasks of processing huge arrays of distributed information. Also, MapReduce application areas include distributed search and data sorting, web link graph retrieval, network log statistics processing, inverted index construction, document clustering, machine learning, and statistical machine translation. Also, MapReduce

is adapted for multiprocessor systems, voluntary computing, dynamic cloud and mobile environments.

But despite the advantages of the technologies discussed above, there is one drawback, namely their scale. When developing large projects, this does not matter, but for medium and small ones, their capabilities can be considered super-superfluous.

Therefore, in the future, we will consider less demanding technologies. Among them, there is a group of frameworks that work in the Python environment (Coelho, 2018; McKinne, 2022; Nelli, 2018; Plas et al., 2021; Führer et al., 2021). Such technologies include Scikit-learn (Coelho, 2018) which is built on the basis of the SciPy stack (Scientific Python), and includes (Johansson, 2018; Bressert, 2012):

- NumPy adds support for large multidimensional arrays and matrices, as well as a library of high-level mathematical functions for operations on them.
- Pandas implements various data structures and analysis

Among these two packages, we are more interested in the second one, namely Pandas, because with its help we can form information arrays and perform operations such as dividing arrays, extracting the necessary information from arrays, etc.

Pandas itself (Harrison, 2021; Molin and Jee, 2021) is an open-source Python package that provides efficient, easy-to-use data structures and analysis tools for observed data, simplifying problem solving.

Pandas supports all the most popular data storage formats: csv, excel, sql, clip-board, html and much more.

This simplifies the task because you do not need to focus on the form of data storage. There is a special structure for this - a dataframe. To access the data, it is enough to write one command:

```
Company = pd.read_csv (os.path.join (PATH, "Company.csv"))
```

In this post, we'll use the `pd.read` method, which allows us to read the "Company.csv" dataframe. The `pd` prefix tells us that we are using a method from the Pandas package. We get the path to the dataframe using `os.path.join`. The `join` method from the library for working with the OS operating system works here, which connects paths taking into account the features of the operating system.

Pandas adds new data structures to Python - series and dataframes.

It is best to imagine the DataFrame object as an ordinary table because a Data-Frame is a tabular data structure. There are always rows and columns in any table. Columns in the DataFrame object are Series objects, the rows of which are their immediate elements.

The purpose of the work is the development of a big data analysis system for obtaining new information for the purpose of its analysis

To achieve the set goal, it is necessary to solve the following problems:

- collect the necessary information from the subject area;
- create an information array that can be used in the application;
- carry out actions on the information array to extract new data.

## Results

Today, various social networks such as Twitter, Facebook and a variety of social, economic and other information. Directions. Analysis of this norm will allow investigators to obtain new data on the transactions and claims of corporate clients (Yigitcanlar et al., 2021; Kearney, 2018). Therefore, to generate a call model, collect data with answers about wire companies from 2015 to 2020 (Gontareva et al., 2020; Mavlutova et al., 2021).

This data set, as part of an article published at the 2020 IEEE International Conference on Big Data during the 6th Special Session on Intelligent Data Mining, was created to identify possible speculators and influencers in the stock market. This dataset contains tweets in which different users express their relationship to companies such as Amazon, Apple, Google, Microsoft, and Tesla, using the appropriate tickers for sharing (Molin and Jee, 2021; Yigitcanlar et al., 2021; Kearney, 2018).

The dataset includes more than 3 million unique tweets with information such as tweet ID, tweet author, publication date, tweet text, and the number of comments, likes, and retweets of tweets corresponding to the associated company (Malyarets et al., 2017; Ramazanov et al., 2020).

To create an application, you will need to steal the following libraries (Johansson, 2018; Bressert, 2012):

- the OS module provides many functions for working with the operating system, and their behavior, as a rule, does not depend on the OS, so programs remain portable.

- The datetime module provides classes for handling time and date in various ways. The standard way of representing time is also supported, but greater emphasis is placed on the ease of manipulation of the date, time and their parts.
- Numpy is a Python language library that adds support for large multidimensional arrays and matrices, along with a large library of high-level (and very fast) math functions for operations on these arrays.
- Pandas - a program library in the Python language for data processing and analysis. Pandas data handling is built on top of the numpy library, which is a lower-level tool. Provides special data structures and operations for manipulating numeric tables and time series.
- Seaborn - a library for creating statistical graphs in Python. It is built on Matplotlib and tightly integrates with pandas data structures. Seaborn helps to study and understand the data. Its graphing functions work with datasets and perform all necessary transformations to create informative graphs.
- The Pyplot module is a collection of command-style functions that allow Matplotlib to be used in much the same way as Matlab.
- The class pool is used to display the pool of work processes. It includes methods that allow you to offload tasks to workflows.

First, we set the path to the directory with data sets and set the pandas and sea-born settings (Listing 1).

Listing 1. Specifying the path to the directory with datasets

```
PATH = 'dataset/'
```

```
pd.set_option ('display.max_columns', 30)
```

```
sn.set_context ("paper", font_scale=2)
```

Next, we read data from a csv file with companies and convert it into a Data-Frame using the read\_csv function. Use the set\_index function to set the column index (Listing 2).

Listing 2. Reading data from a csv file

```
Company=pd.read_csv (os.path.join(PATH, "Company.csv"))
```

```
Company=company.set_index ("ticker_symbol").to_dict()["company_name"]
```

At the next stage, use the drop function to remove unnecessary columns. Thanks to the inplace argument, we edit the dataset in which we call this function

Listing 3. Removing unnecessary columns

```
tweet.drop(['comment_num', 'retweet_num', 'like_num'], axis=1, inplace=True)
```

Next, we create new columns so that it is convenient to work with the data frame in the future

Listing 4. Creating new columns in a date frame

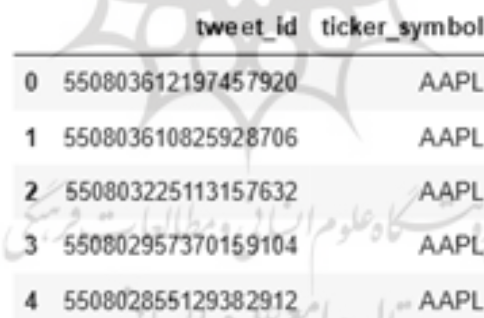
```
Tweet['datetime'] = pd.to_datetime(tweet['post_date'], unit='s')
```

```
Tweet['date'] = tweet['datetime'].dt.date
```

```
Tweet["week"] = ((tweet.post_date.values - tweet.post_date.values[0]) / 604800).astype(int)
#604800=7*24*60*60
```

Next, we read and edit the dataset with indexes of tweets and companies addressed.

Accordingly, the dataset will be edited with tweet indexes and companies addressed (Fig. 1):



	tweet_id	ticker_symbol
0	550803612197457920	AAPL
1	550803610825928706	AAPL
2	550803225113157632	AAPL
3	550802957370159104	AAPL
4	550802855129382912	AAPL

Figure 1. Edited dataset

We find out the number of records, of which they are unique, the number of authors and the percentage of duplicate tweets (listing 5).

Listing 5. Getting the number of records, authors and their uniqueness

```
print("Total Entries: {} | Unique Tweet Indexes: {}")
```

```
format(len(company_tweet), company_tweet['tweet_id'].nunique()))
```

```
Total entries: 4336445 | Unique index of tweets: 3717964
```



```
tweet = tweet.dropna()
```

```
print(f"Number of authors: {tweet['writer'].nunique()}")
```

```
Number of authors: 140131
```

```
print("Percentage of duplicate tweets: {:.2f}
%".format(sum(tweet['body'].duplicated())/len(tweet) * 100))
```

```
Percentage of duplicate tweets: 10.35 %
```

After that, using the seaborn module, we create a histogram of the number of tweets (Listing 6).

Listing 6. Histogram of the number of tweets

```
Stats = tweet[['writer', 'tweet_id']].groupby('writer').agg("count").rename(columns={'tweet_id'
: 'tweet_count'})
```

```
sn.histplot(data=stats, x='tweet_count', bins=50, log_scale=True)
```

```
plt.figure(figsize=(18, 6))
```

```
plt.yscale('log')
```

```
plt.title("Histogram of number of tweets")
```

```
plt.xlabel("Number of users")
```

```
plt.ylabel("Number of tweets")
```

When starting the program, we will get the following diagram (Fig. 2)

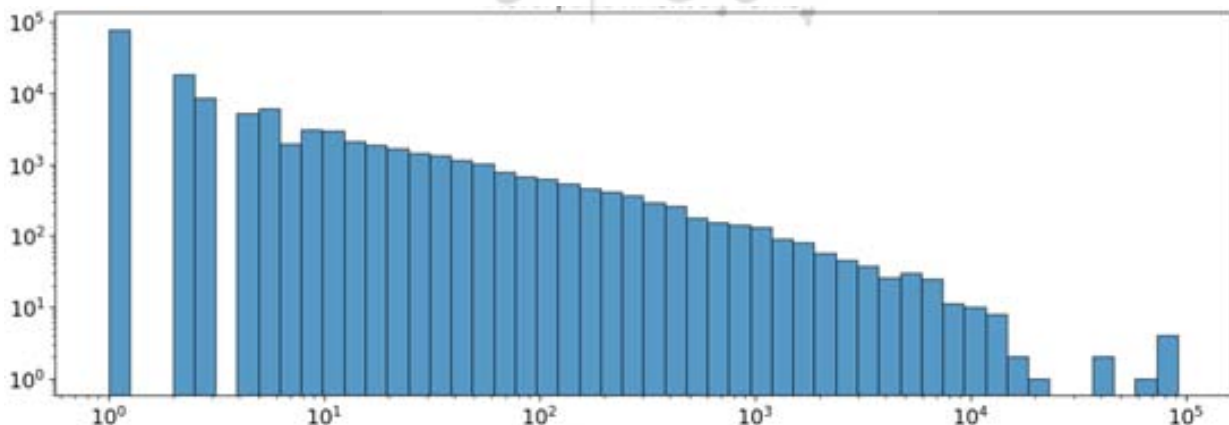


Figure 2. Number of tweets

Next, we merge the tweet and company\_tweet datasets (listing 7) and display the result (Fig.3). To do this, we will use the merge function from the panda's library.

Listing 7. Merging the tweet and company\_tweet datasets

```
dataframe = pd.merge(tweet, company_tweet, on='tweet_id', how='inner')
dataframe.head ()
```

	tweet_id	writer
0	550441509175443456	VisualStockRSRC
1	550441672312512512	KeralaGuy77
2	550441732014223360	DozenStocks
3	550442977802207232	ShowDreamCar
4	550443807834402816	i_Know_First

**Figure 3. Merger of tweets**

To analyze user evaluations of these companies using the NumPy framework, we create a Negative module with negative words that will be used to determine the negativity of tweets (Listing 8).

Listing 8. Creating the Negative module

```
Import numpy as np
egs = np.array(['hate', 'flip', 'don\'t understand', 'doesn\'t understand', 'upset', 'plummet',
'misinformation', 'fraud', 'idiot', 'stupid', 'losing focus', 'angry'])
def wordpresent (s):
Return [negs[i] in s.lower () for i in range (len(negs))]
```

Let's connect our module, and with the help of the Pool class, we speed up the search for tweets with negative words (Listing 9).

Listing 9. Searching for negative statements using the Negative module

```
import Negative
%%time
with Pool(processes=8) as pool:
outcome=pool.map(Negative.wordpresent, dataframe.body.values)
```

Next, we display a list of negative words and the number of tweets broken down by words in the dataset (Listing 10).

Listing 10. List of negative words

For w, cnt in

```
Zip(Negative.negs,np.sum(outcome, axis=0)):
```

```
Print(w, " ", cnt)
```

When writing the method, we will get the following entries:

```
hate 11847
```

```
flip 5115
```

```
don't understand 922
```

```
doesn't understand 194
```

```
upset 789
```

```
plummet 1114
```

```
misinformation 317
```

```
.....
```

```
fraud 18085
```

```
idiot 4023
```

```
stupid 5809
```

```
losing focus 16
```

```
angry 940
```

Using the sum function of the NumPy module, we will count the number of negative tweets (Listing 11)

Listing 11. Number of tweets classified as negative

```
np.sum (np.max(outcome, axis=1))
```

```
135299
```

In the dataframe, we create columns that will determine whether a tweet is negative or neutral (Listing 12)

Listing 12. Method for selecting negative and neutral tweets from a dataset

```
dataframe["NegativeTweet"] = np.max(outcome, axis=1)
```

```
dataframe["NeutralTweet"] = ~dataframe.NegativeTweet.values
```

```
dataframe.head()
```

When starting the program, we will get the following result (Fig. 4):

	tweet_id	writer	post_date
0	550441509175443456	VisualStockRSRC	1420070457
1	550441672312512512	KeralaGuy77	1420070496
2	550441732014223360	DozenStocks	1420070510

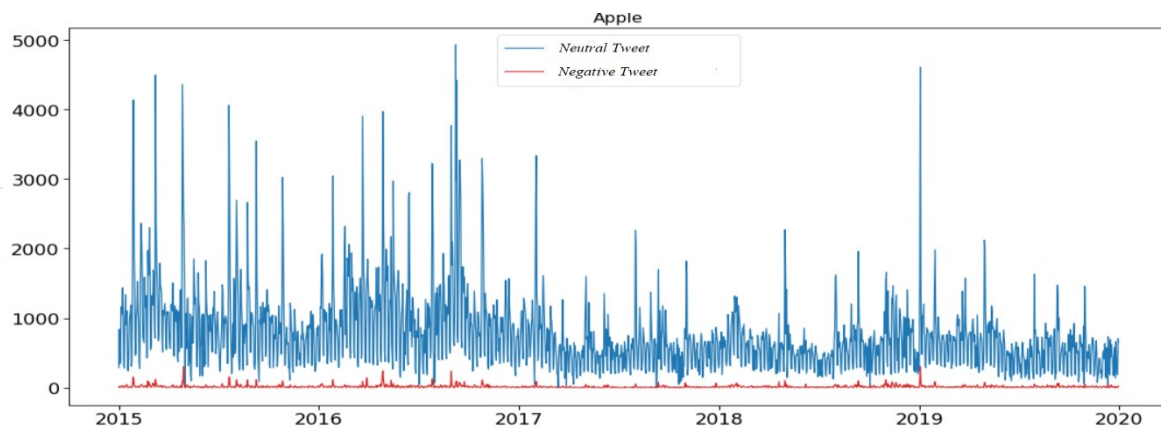
**Figure 4. An example of displaying negative and neutral tweets**

To output information, we will create a DrawLinePlot method that will select negative and neutral tweets from the dataset (Listing 13).

Listing 13. DrawLinePlot method for displaying information as a graph

```
def DrawLineplot(dataframe, companyName, title, drawNegative=False, drawNeutral=False):
    if companyName is None:
        company = dataframe
    else:
        company = dataframe.iloc[dataframe.ticker_symbol.values == companyName]
    plt.figure(figsize=(18, 8))
    if drawNeutral:
        companyNeutralTweetCount
        = company[["date", "NeutralTweet"]].groupby(["date"]).sum()
        sn.lineplot(x=companyNeutralTweetCount.index, y=companyNeutralTweetCount.NeutralTwe
        et.values,
        color='tab:blue', label='Neutral Tweets')
    if drawNegative:
        companyNegativeTweetCount
        = company[["date", "NegativeTweet"]].groupby(["date"]).sum()
        sn.lineplot(x=companyNegativeTweetCount.index,
        y=companyNegativeTweetCount.NegativeTweet.values,
        color='tab:red', label='Negative Tweets')
```

After that, with the help of the Sea-born library, we will build a graph for each company. For example, here is a graph of negative and neutral tweets by week for the Apple Company (Fig. 5).



**Figure 5. Graph of statements about the Apple Company**

At the next stage, we will analyze and classify statements using machine learning methods. We will build a model based on logistic regression, the input of which will be a statement, and the model must estimate whether this statement is neutral or negative.

If the model returns array ([0]), then the text is neutral, if array ([1]) is negative.

To do this, we create two dataframes for training and testing, the size of which is 150,000 for training and 15,000 for testing (Listing 14).

Listing 14. Creating a training and test data set

```
df = dataframe[["body", "NegativeTweet"]]
df["NegativeTweet"] = df["NegativeTweet"].astype(int)
df.head()
test_df["NegativeTweet"].value_counts()
0  14515
1   485
Name: NegativeTweet, dtype: int64
train_df["NegativeTweet"].value_counts()
0  145258
1   4742
Name: NegativeTweet, dtype: int64
```

Next, we create a function for the sentence lexeme (Listing 15).

Listing 15. Function for thing tokens

```
snowball=SnowballStemmer(language="english")
```

```

stop_words = stopwords.words("english")
def tokenize_sentence(sentence: str, remove_stop_words: bool = True):
tokens = word_tokenize(sentence, language="english")
tokens = [i for i in tokens if i not in string.punctuation]

```

In order to continue further, it is necessary to carry out the so-called cleaning of the dataset, namely the removal of various punctuation marks, stop words, and so on. In order for the program to be able to separate one part of information from another, we set file formats, that is, an agreement about how the text is written inside the file. The simplest format - each unit of this information is on a separate line. Such a file almost does not require additional processing - it is enough to consider it as means of the used programming language and break it into lines. Most languages allow you to split a file into lines with one or two commands. Unfortunately, most of the files that need to be processed have a slightly more complex format.

Therefore, for text files, the structure of which is more complex than a list of lines, the method of splitting into tokens using regular expressions has been used successfully for a long time. The word "token" usually means a small part of the text that is located in a certain place of this text and has a certain meaning.

At the final stage, we revise the accuracy of guessing the tweet. For which we construct a logistic regression model, by connecting a different module, we create a pipeline for them to help the pipeline method, we start that robimo reverification (listing 16).

Listing 16. Training the model

```

model=LogisticRegression(random_state=0)
model.fit(features, train_df["NegativeTweet"])
LogisticRegression(random_state=0)
# If array([0]) returns neutral text, if array([1]) - negative
model.predict(features[203])
array([0])
# We create a data pipeline pipeline
model_pipeline = Pipeline([
("vectorizer", TfidfVectorizer(tokenizer=lambda x: tokenize_sentence(x,
remove_stop_words=True))),
("model", LogisticRegression(random_state=0))
])

```

```
# Training
model_pipeline.fit(train_df["body"],train_df["NegativeTweet"])
Pipeline(steps=[('vectorizer',
TfidfVectorizer(tokenizer=<function <lambda> at 0x0000020592919670>)),
('model', LogisticRegression(random_state=0))])
# Example of negative text
model_pipeline.predict(["You are stupid monkey"])
array([1])
# Example of neutral text
model_pipeline.predict(["Man, how are you?"])
array([0])
```

The accuracy of recognizing a negative comment is 98%, which meets the requirements.

## Conclusion

An approach to creating relevant content and analyzing large data sets using machine learning methods is considered. In the work, a software model was created that allows you to evaluate the attitude of users to the activities of various companies. The developed model analyzes the statements of users in social networks, can divide them into positive, negative or neutral (in the work, the example includes only a robot with negative and neutral statements) and, based on these statements, can form time summaries of users' attitudes towards companies whose activities are analyzed. The model is based on the use of dictionaries with keywords and special functions for working with datasets from the NumPy, Pandas and Scikit-learn libraries.

## Conflict of interest

The authors declare no potential conflict of interest regarding the publication of this work. In addition, the ethical issues including plagiarism, informed consent, misconduct, data fabrication and, or falsification, double publication and, or submission, and redundancy have been completely witnessed by the authors.

## Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

## References

- Babenco, V., Kulczyk, Z., Perevozova, I., Syniavska, O., Davydova, O. (2019). Factors of Development of International e-Commerce in the Context of Globalization. *CEUR Workshop Proceedings*, vol. 2422, pp. 345-356. <http://ceur-ws.org/Vol-2422/paper28.pdf>
- Babenco V., Panchyshyn A., Zomchak L., Nehrey M., Artym-Drohomyretska Z., Lahotskyi T. Classical Machine Learning Methods in Economics Research. Macro and Micro Level Example. *WSEAS Transactions on Business and Economics* 2021, 18, 209-217; <https://doi.org/10.37394/23207.2021.18.22>
- Bressert E. *SciPy and NumPy* 1st Edition; Publisher: O'Reilly Media, USA, 2012; 57.
- Coelho L. P. *Building Machine Learning Systems with Python*; Publisher: Packt Publishing, UK, 2018; 406.
- Führer C., Solem J. E., Verdier O. *Scientific Computing with Python: High-performance scientific computing with NumPy, SciPy, and pandas*, 2nd Edition 2nd ed. Edition; Publisher: Packt Publishing, UK, 2021; 392 p.
- Gontareva, I., Babenco, V., Shmatko, N., Litvinov, O., Hanna, O. (2020). The Model of Network Consulting Communication at the Early Stages of Entrepreneurship. *WSEAS Transactions on Environment and Development*, Vol. 16, pp. 390-396. <https://doi.org/10.37394/232015.2020.16.39>
- Guryanova L., Yatsenko R., Dubrovina N., Babenco V. (2020). Machine learning methods and models, predictive analytics and applications. *CEUR Workshop Proceedings*, 2020, Available online: <http://ceur-ws.org/Vol-2649/>
- Guryanova L., Yatsenko R., Dubrovina N., Babenco V., Gvozditskyi V. Machine Learning Methods and Models, Predictive Analytics and Applications: Development Trends in the Post-crisis Syndrome Caused by COVID-19. *CEUR Workshop Proceedings*, 2021, Available online: <http://ceur-ws.org/Vol-2927/paper1.pdf>
- Janssens J. *Data Science at the Command Line: Facing the Future with Time-Tested Tools* 1st Edition; Publisher: O'Reilly Media, USA, 2014; 212 p.
- Johansson R. *Numerical Python: Scientific Computing and Data Science Applications with Numpy, SciPy and Matplotlib* 2nd ed. Edition; Publisher: Apress, USA, 2018; 723 p.
- Harrison M. *Effective Pandas: Patterns for Data Manipulation (Treading on Python)*; Publisher: Independently published, USA, 2021; 497 p.
- Karau H., Konwinski A., Wendell P., Zaharia M. *Learning Spark: Lightning-Fast Big Data Analysis* 1st Edition; Publisher: O'Reilly Media, USA, 2020; 276 p.
- Kearney, M.W. R: Collecting and Analyzing Twitter Data. 2018. Available online: [https://mkearney.github.io/nicar\\_tworkshop](https://mkearney.github.io/nicar_tworkshop) (accessed on 10 November 2021).
- Marianna Lepelaar, Adam Wahby, Martha Rossouw, Linda Nikitin, Kanewa Tibble, Peter J. Ryan, Richard B. Watson Sentiment Analysis of Social Survey Data for Local City Councils. *J. Sens. Actuator Netw.* 2022, 11(1), 7; <https://doi.org/10.3390/jsan11010007>
- Mavlutova, I., Babenco, V., Dykan, V., Prokopenko, N., Kalinichenko, S., Tokmakova, I. (2021). Business Restructuring as a Method of Strengthening Company's Financial Position. *Journal of Optimization in Industrial Engineering*, 14(1), 129-139. <http://dx.doi.org/10.22094/JOIE.2020.677839>
- Malyarets, L., Draskovic, M., Babenco, V., Kochuyeva, Z., Dorokhov, O. (2017). Theory and practice of controlling at enterprises in international business. *Economic Annals-XXI*, Vol. 165, Iss. 5-6, 90-96. <https://doi.org/10.21003/ea.V165-19>



- McKinne W. Python for Data Analysis: Data Wrangling with pandas, NumPy, and Jupyter 3rd Edition; Publisher: O'Reilly Media, USA, 2022; 579 p.
- Molin S., Jee K. Hands-On Data Analysis with Pandas: A Python data science handbook for data collection, wrangling, analysis, and visualization, 2nd Edition; Publisher: Packt Publishing, UK, 2021; 788 p.
- Naldi, M. A review of sentiment computation methods with R packages. arXiv Prepr. 2019, arXiv:1901.08319 2019. Available online: <https://arxiv.org/pdf/1901.08319.pdf> (accessed on 2 June 2021).
- Nelli F. Python Data Analytics: With Pandas, NumPy, and Matplotlib 2nd ed. Edition; Publisher: Apress, USA, 2018; 588 p.
- Plas J., Vander P. J. Python for Complex Tasks: Data Science and Machine Learning; Publisher: O'Reilly Bestsellers, USA, 2021; 576 p.
- Ramazanov, S., Babenko, V., Honcharenko, O., Moisieieva, N., Dykan, V. (2020). Integrated intelligent information and analytical system of management of a life cycle of products of transport companies. *Journal of Information Technology Management*, 2020, 12(3), 26-33. <https://doi.org/10.22059/jitm.2020.76291>
- Ramirez, C.M.; Abrajano, M.A.; Alvarez, R.M. Using Machine Learning to Uncover Hidden Heterogeneities in Survey Data. *Sci. Rep.* 2019, 9, 16061.
- White T. Hadoop: The Definitive Guide, Third Edition; Publisher: Yahoo Press, USA, 2012; 688 p.
- Yigitcanlar, T.; Kankanamge, N.; Vella, K. How Are Smart City Concepts and Technologies Perceived and Utilized? A Systematic Geo-Twitter Analysis of Smart Cities in Australia? *J. Urban Technol.* 2021, 28, 135–154.

---

**Bibliographic information of this paper for citing:**

Babenko, Vitalina; Pronin, Sergii; Aleksejeva, Ludmila; Vītola, Zaiga; Sokolova, Liudmyla; Zaiga; Dyuzhev, Viktor & Dyakova, Nataliya (2024). Tools for Consumer Preference Analysis Based in Machine Learning. *Journal of Information Technology Management*, 16 (4), 1-17. <https://doi.org/10.22059/jitm.2024.99048>

---

Copyright © 2024, Vitalina Babenko, Sergii Pronin, Ludmila Aleksejeva, Zaiga Vītola, Liudmyla Sokolova, Viktor Dyuzhev and Nataliya Dyakova