Production Systems and Information Engineering
Volume 11 (1), pp. 14–26
doi: 10.32968/psaie.2023.1.2

COMPARISON OF TEXT CLASSIFICATION METHODS

Antal Kristóf Fekete
University of Miskolc, Hungary
Institute of Information Technology
fekete.antal.kristof@student.uni-miskolc.hu

Erika Baksáné Varga
University of Miskolc, Hungary
Institute of Information Technology
erika.baksane.varga@uni-miskolc.hu

Abstract. The paper presents a comparison of some text categorization methods in terms of accuracy and learning speed. These methods are selected specifically for large dataset, therefore only the Random Forest algorithm is considered from the numerous machine learning techniques. In addition to this, two LSTM models are studied as – based on our literature review – these are found best suited to the text classification task among neural networks. Our research goal is to find evidence for or against this statement. Therefore we build, train and test a classic multilayer perceptron model and show its accuracy and learning speed as compared to the other methods.

Keywords: Text classification, Random Forest, Neural networks, LSTM, MLP

1. Introduction

Text classification is a classical problem in natural language processing (NLP), which aims to assign a category label to the input text. Examples of text categorization are topic labeling, sentiment analysis, spam email detection and news classification [1]. These tasks can be performed either through manual annotation, or by automatic labeling. However, our digital society produces enormous quantity and diversity of electronic documents, which have made manual solutions to text classification tasks unaffordable.

Automatic text classification approaches can be divided into two groups: rule-based and data-driven methods. Rule-based methods classify text into different categories using a set of predefined rules, the creation of which requires deep domain knowledge. On the other hand, data-driven methods learn to classify text based on the recorded observations. These algorithms learn the associations between texts and their labels by using pre-labeled training samples and are therefore considered as supervised learning methods [2].

The work procedure of text classification using machine learning techniques is comprised of text pre-processing, feature extraction and feature selection, training, testing, and performance evaluation. Texts are usually pre-processed with tokenization, lemmatization, or stemming, followed by stop-word filtering [3]. This helps in creating a Bag-of-Words (BoW) or a Vector Space Model (VSM) for text representation [4]. As a result, texts are often represented by high-dimensional matrices, so dimensional reduction is also needed to address feature collinearity and to decrease computational costs [5]. Recent approaches propose topic modeling [6] and word embeddings [7] to be applied for reducing the size of the model, as both techniques learn a representation for text where words that have the same meaning are grouped under similar representation. Dimensional reduction can also be achieved by feature selection and feature extraction. The difference between the two methods is that feature extraction generates new variables for replacing several similar features, while feature selection removes non-defining features without creating new ones [8]. Regarding feature selection, the most common methods include Term Frequency-Inverse Document Frequency (TF-IDF), Information Gain and Mutual Information. In feature extraction, the most popular approaches are Principal Component Analysis (PCA) and Latent Semantic Indexing (LSI) [5].

In the next step, the remaining features are fed into classifiers for training, and then the trained models are used for predicting the category labels of unknown data. The most popular classifiers are Naive Bayes, K Nearest Neighbour, Decision Tree, Random Forest, and Support Vector Machine [9]. Recently, neural networks have also been utilized for text classification, and achieved significant results due to their capability to model complex non-linear or long-term relationships within the data [10].

Finally, the performance of the applied text classification technique should be evaluated. Accuracy is the simplest to calculate, but it works only for balanced data sets [11]. Other methods include F2-score, Matthews Correlation Coefficient (MCC), Receiver Operating Characteristics (ROC), and Area Under the ROC Curve (AUC) [3].

In this paper we present the results of an experiment with some machine learning and deep learning techniques when applied to a customer complaints classification task.

2. Related works

In all businesses, customer satisfaction is extremely important. For this reason, companies should pay special attention to handling customer cases, though it infers high costs on the other side, because the manual collection and analysis of complaints is ineffective and time-consuming. The solution for this paradoxical case is the introduction of a customer complaint management system, that is automatized to some extent, to achieve the business objective of customer satisfaction on handling cases and being cost effective at the same time.

Motivated by this goal, researchers have recently focused on the categorization of customer feedback using machine learning (ML) algorithms to handle complaints efficiently. In most cases, categorization of customer complaints is a multiclass classification problem, where more than two possible target groups are defined, but Krishna et al. [12] simplified the task to binary classification. They performed sentiment analysis of bank customers using the respective banks' online complaints management platforms. In pre-processing the raw textual data, three different techniques were employed for structuring: document term matrix (DTM), Word2Vec embedding model and the Linguistic Inquiry and Word Count (LIWC) psycho-linguistic method. The authors experimented with Support Vector Machine (SVM) [13], Naive Bayes (NB) [14], Logistic Regression (LR) [15], Decision Tree (DT) [16], k Nearest Neighbour (kNN) [17], Random Forest (RF) [18], Extreme Gradient Boosting (XGBoost) [19] and Multi-layer Perceptron (MLP) [20] classifiers. Their results indicate that the representation form of the text has an effect on classification accuracy. Considering the same dataset, when applying DTM, Logistic Regression yielded the best accuracy score (77.13%); when working with Word2Vec embedding, SVM was the best method (74.77%); while Random Forest was the winner for the LIWC representation (77.44%).

Muaamar Nasser Saleh [21] has made investigations with the automatic assignment of customer complaints collected by a transportation company applying CART (Classification and Regression Trees, [22]) and SVM ML algorithms. As a result, the author reports 75.9% accuracy with CART, 84.4% with linear SVM, and 74% with non-linear SVM method.

Onan et al. [23] presented the categorization of service support requests using five basic ML algorithms, i.e. Naive Bayes, kNN, C4.5 Decision Tree, Random Forest and SVM algorithms on a dataset including 17831 bug reports and service support requests originating from a university information management system. The best classification accuracy (92.26%) was achieved with SVM.

In a Turkish text classification task [24], customer complaints about package food products were categorized using Logistic Regression, Naive Bayes, SVM, kNN, Random Forest and XGBoost ML algorithms. In this study, two feature representation methods were used: TF-IDF and Word2Vec. Experimental results show that the best-performing method is XGBoost with TF-IDF weighting scheme which achieved 84% F-measure score.

In this study, we have used a consumer complaints dataset with TDM text representation to compare the accuracy of the Random Forest ML method with some deep learning methods. The rest the of the paper is organized as follows. Section 3 introduces the dataset and the applied classification methods, while Section 4 presents the results obtained.

3. Research methods

3.1. Dataset

The algorithms were tested on a dataset of reports of disputes between financial institutions and consumers in the United States sent by customers to the Consumer Financial Protection Bureau, which acts as an intermediary. The dataset [25] covers a one-year period of complaints from March 2020 to March 2021. Customer inquiries and responses are divided into five categories:

1. Credit reporting
2. Debt collection
3. Mortgages and loans
4. Credit cards
5. Retail banking

The dataset can be downloaded in CSV format, and holds 162 000 records. After deleting the duplicates and records containing empty fields, the cleaned dataset is built up of more than 124 000 rows. The distribution of the records among the categories is shown in Table 1.

Product	%
Credit reporting	45.18
Debt collection	16.92
Mortgages and loans	15.04
Credit cards	12.04
Retail banking	10.82

Table 1. Distribution of records

3.2. Algorithms

Customer complaints are given as free text in English. A number of ML methods are available and can be used to learn the classification of texts in case there is a training set with predefined categories. The most important factors that affected our decision about method selection are:

- type and size of the dataset,
- computation time and accuracy.

Taking into consideration the above mentioned selection criteria, in this research we have examined the results and accuracy of the following methods on the dataset:

- Random Forest
- RNN-LSTM
- BI-LSTM
- Multilayer Perceptron (MLP)

From the ML methods that can be applied for text categorization, the Random Forest algorithm was selected because it can handle large datasets even if these are imbalanced, and it has fast computation time with high accuracy.

Random Forest (RF) [26] is a widely used supervised ML algorithm. The random forest model is made up of multiple decision trees, which iteratively split the data until reaching the final decision. At each decision node, the algorithm seeks to find the best split to subset the data. They are usually trained through the CART algorithm. Metrics, such as information gain, or mean square error (MSE) can be used to evaluate the quality of the split. When multiple decision trees form an ensemble in the RF algorithm, they predict more accurate results than individual trees, particularly when the trees are uncorrelated with each other. The RF algorithm utilizes bagging and feature randomness to create an uncorrelated forest of decision trees. In the bagging method, a random sample of data in a training set is selected with replacement. Next, these models are trained independently and the aggregation of the predictions yield a more accurate estimate than the individual results. The other method employed by the RF algorithm is feature randomness. It generates a random subset of features, unlike decision trees that consider all the possible feature splits. This method ensures low correlation among the decision trees in the collection.

Although Random Forest is computationally less expensive than neural networks and does not require a GPU for training, we have implemented three neural networks to compare their classification accuracy on the given dataset.

Deep learning based techniques are specifically designed for processing large volumes of data, and recurrent neural networks (RNNs) were the first to be widely used for processing sequential data. RNN [27] is a generalization of a feed-forward neural network that has internal memory. It performs the same function on each input, and the output of the current input depends on the previous computation. The output learnt from the previous input is stored in the RNN's internal memory, which makes it capable of processing sequences of inputs. An RNN, however, is not able to memorize data for long time and begin to forget its previous inputs. This is called the vanishing or exploding gradient problem, which is solved by the introduction of gate functions into the memory structure.

The neural network, that can handle the problem of long-term dependencies, and therefore best suited for text processing, is called LSTM [28]. LSTM stands for long short-term memory network and has long-term memory in the form of weights which can change slowly during the training. It also has short-term memory in the form of ephemeral activations that pass from each node to the successive nodes. In the LSTM model, each recurrent node is replaced by a memory cell which is a composite unit built up from simpler nodes and a number of multiplicative gates that determine whether (i) a given input should impact the internal state (the input gate), (ii) the internal state should be flushed to (the forget gate), or (iii) the internal state should have an impact on the output (the output gate) [29].

The bidirectional LSTM (Bi-LSTM) network is more efficient because its output depends on the previous and also on the next segments. Unlike the LSTM network, the Bi-LSTM network has two parallel layers, composed of LSTM units, that propagate in two directions with forward and backward passes to capture dependencies in two contexts. This way, Bi-LSTM can learn long-term dependencies without retaining duplicate context information. Therefore, it has demonstrated excellent performance for sequential modeling problems and is widely used for text classification [30].

The implementation of the first three methods for customer complaints categorization is available at kaggle.com ([31], [32], [33]). In our experiment, we intended to implement a Multilayer Perceptron (MLP) from scratch to see its performance in comparison with the others.

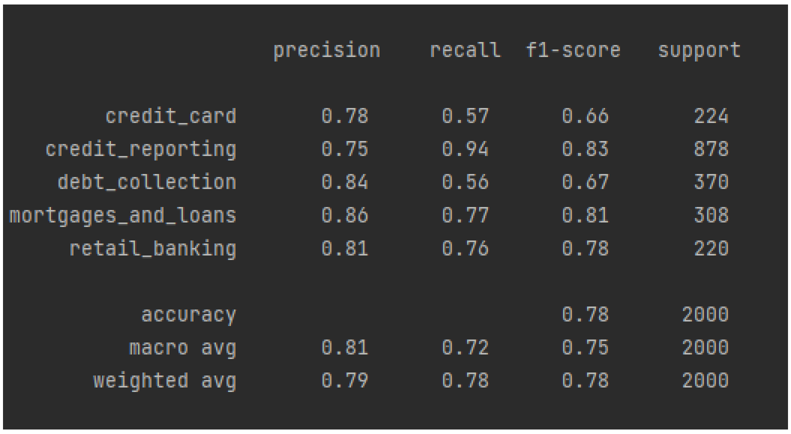
MLP is the classical neural network that is comprised of an input layer, one or more hidden layers and an output layer. This architecture is suitable for classification prediction problems where inputs are assigned a label prediction on the output layer [34], [35].

The Python implementation of each of the methods mentioned above was run in PyCharm IDE 2022.3.1 (Professional Edition) with Python 3.10 on the following PC architecture:

- OS: Windows 10 Pro 64-bit 22H2
- CPU: Intel Core i7-10700K CPU @ 3.80GHz
- GPU: NVIDIA GeForce GTX 1660 Ti
- RAM: 32 GB 3600 MHz

4. Results

In the Random Forest implementation [31], a Tf-Idf vectorizer is used to transform the data, then the data is split into a training set and a testing set, and the RandomForestClassifier is invoked to create the model based on the training set. Finally, the prediction is made using the test set, the results of which are listed in Fig. 1.



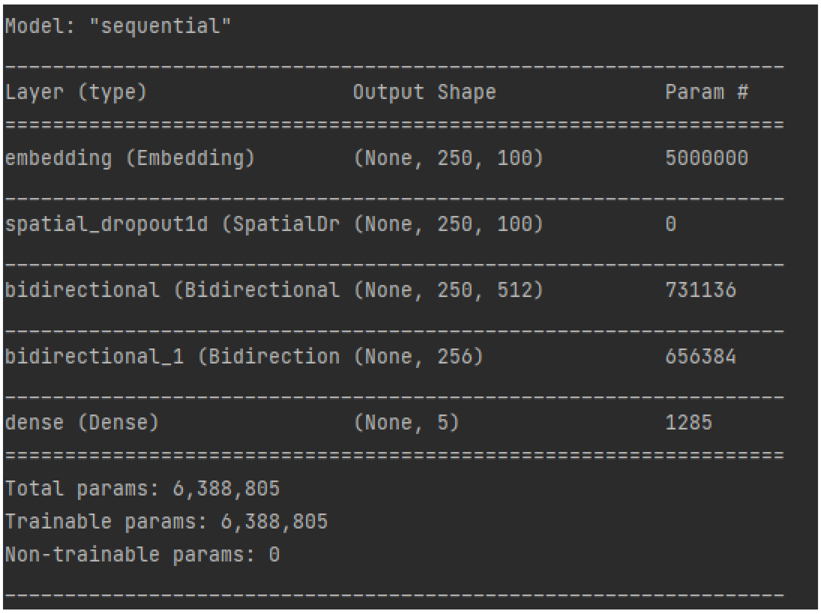
```
                    precision    recall  f1-score   support

       credit_card       0.78      0.57      0.66       224
  credit_reporting       0.75      0.94      0.83       878
   debt_collection       0.84      0.56      0.67       370
mortgages_and_loans      0.86      0.77      0.81       308
    retail_banking       0.81      0.76      0.78       220

          accuracy                           0.78      2000
         macro avg       0.81      0.72      0.75      2000
      weighted avg       0.79      0.78      0.78      2000
```

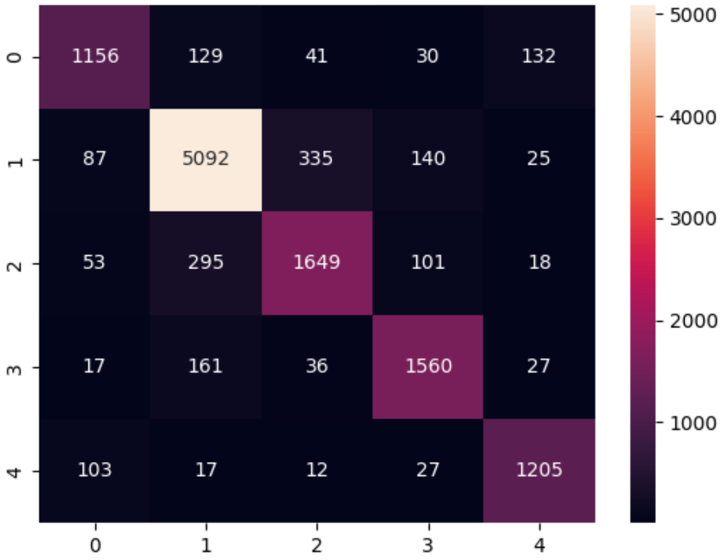
Figure 1. Classification results for the RF implementation

In the case of the RNN-LSTM [32] and Bi-LSTM [33] implementations, after splitting the data into training and testing set and tokenizing the narratives, the sequential model is built and evaluated using the tensorflow.keras package. The architecture of the RNN-LSTM neural network is demonstrated in Fig. 2, and in Fig. 3 the confusion matrix summarizes the results of the test predictions. In the confusion matrix diagram, the numbers on the axes denote the predefined classes. Namely, credit card (0), credit reporting (1), dept collection (2), shortgages and loans (3), and retail banking (4).



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, 250, 100)          5000000
_________________________________________________________________
spatial_dropout1d (SpatialDr (None, 250, 100)          0
_________________________________________________________________
bidirectional (Bidirectional (None, 250, 512)          731136
_________________________________________________________________
bidirectional_1 (Bidirection (None, 256)               656384
_________________________________________________________________
dense (Dense)                (None, 5)                 1285
=================================================================
Total params: 6,388,805
Trainable params: 6,388,805
Non-trainable params: 0
_________________________________________________________________
```

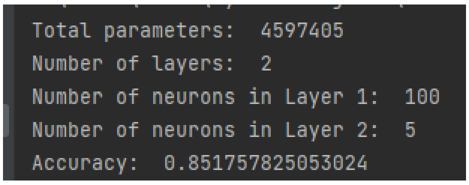
Figure 2. RNN-LSTM model architecture



	0	1	2	3	4
0	1156	129	41	30	132
1	87	5092	335	140	25
2	53	295	1649	101	18
3	17	161	36	1560	27
4	103	17	12	27	1205

Figure 3. Confusion matrix for RNN-LSTM

In our unique MLP solution, we use the Tf-Idf vectorizer first, then we split the data into training and testing parts with 25% test size. After that we create the MLPClassifier for 10 iterations, the architecture of which is shown in Fig. 4.

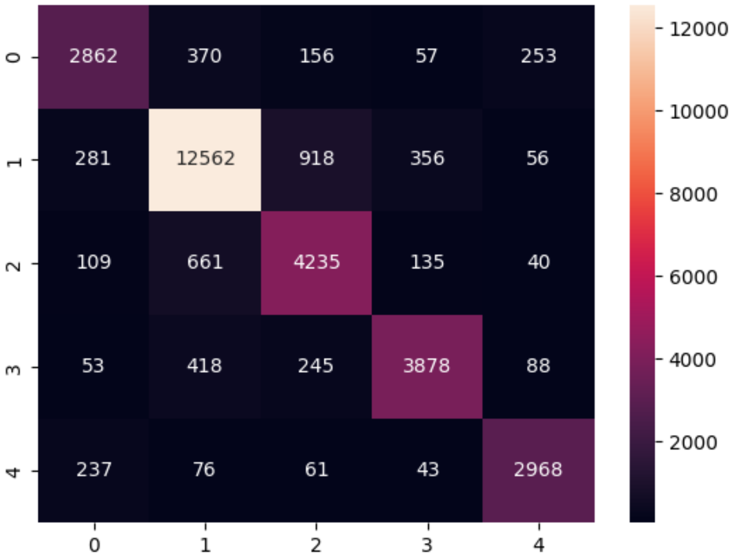


```
Total parameters:  4597405
Number of layers:  2
Number of neurons in Layer 1:  100
Number of neurons in Layer 2:  5
Accuracy:  0.851757825053024
```

Figure 4. MLP model architecture

It is worth noting, that the accuracy of the MLP solution improved steadily up to the 10th iteration. As opposed to this, in the case of the previous LSTM models this improvement was not significantly observed after the 4th iteration.

After training and testing the MLP model, the best accuracy gained can be seen in Fig. 5, where the numbers on the axes denote the categories: credit card (0), credit reporting (1), dept collection (2), shortgages and loans (3), and retail banking (4).



	0	1	2	3	4
0	2862	370	156	57	253
1	281	12562	918	356	56
2	109	661	4235	135	40
3	53	418	245	3878	88
4	237	76	61	43	2968

Figure 5. Confusion matrix of MLP

Table 2 summarizes the accuracy and execution time of the text classification methods we have studied on the given dataset.

Method	**Accuracy**	**Time**
Random Forest	78.00%	5.93 s
RNN-LSTM	85.62%	405.33 s
BI-LSTM	85.60%	720.50 s
Multilayer perceptron	85.18%	632.47 s

Table 2. Results of the examined methods

These results give evidence that the Random Forest algorithm learns significantly faster than neural networks, while achieving quite fair accuracy in predicting the categories of customer complaints. Considering the implemented neural network models, there is no significant difference between their accuracy, but in learning speed the RNN-based LSTM model beats the others. The interesting consequence of this test is that the use of the classic MLP network for text classification yields no worse performance metrics than LSTM networks.

Lastly, we have made experiments to validate the trained model. We have created 10 new complaint texts, transformed them using the Tf-Idf vectorizer, and then tried to classify these narratives into one of the five categories using the model we have trained earlier. Our solution correctly classified the given complaints in all test cases, which confirms that the created model is correct and suitable for the given task.

5. Conclusion

The aim of this research was to compare the accuracy and learning speed of some classic text categorization models and to search for proof of the applicability of MLP networks in text categorization. For this purpose, we took a consumer complaints dataset from kaggle.com and implemented the ML-based Random Forest method and two LSTM neural networks with different architectures. Their comparison showed that neural networks achieve higher accuracy, but Random Forest wins the speed race in learning to classify text into more than two categories.

Among neural networks, LSTMs are said to be best suited to sequence prediction tasks, including text classification. Our goal was to find evidence for or against this statement. For this reason, we have created a classic MLP network and tested on the given text dataset. As the results in Section 4. indicate, we could not find firm support for the statement. Yet indeed, we must conclude that a multilayer perceptron is just as appropriate for text categorization as LSTMs.

References

[1] Aggarwal, C. C. and Zhai, C.: A survey of text classification algorithms. In C. C. Aggarwal and C. Zhai (eds.), *Mining Text Data*, pp. 163–222, Springer US, Boston, MA, ISBN 978-1-4614-3223-4, 2012, URL https://doi.org/10.1007/978-1-4614-3223-4_6.

[2] Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., and Gao, J.: Deep learning based text classification: A comprehensive review. *ACM Computing Surveys*, **54**(3), (2022), 1–40, URL https://doi.org/10.1145/3439726.

[3] Kowsari, K., Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L., and Brown, D.: Text classification algorithms: A survey. *Information*, **10**(4), URL https://doi.org/10.3390/info10040150.

[4] Santos, F., Domingues, M., Sundermann, C., Carvalho, V. D., Moura, M., and Rezende, S.: Latent association rule cluster based model to extract topics for classification and recommendation applications. *Expert Systems with Applications*, **112**, (2018), 34–60, URL https://doi.org/10.1016/j.eswa.2018.06.021.

[5] Shah, F. and Patel, V.: A review on feature selection and feature extraction for text classification. In *International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, 2016, pp. 2264–2268, URL https://doi.org/10.1109/WiSPNET.2016.7566545.

[6] Pavlinek, M. and Podgorelec, V.: Text classification method based on self-training and lda topic models. *Expert Systems with Applications*, **80**, (2017), 83–93, URL https://doi.org/10.1016/j.eswa.2017.03.020.

[7] Stein, R., Jaques, P., and Valiati, J.: An analysis of hierarchical text classification using word embeddings. *Information Sciences*, **471**, (2019), 216–232, URL https://doi.org/10.1016/j.ins.2018.09.001.

[8] Seyyedi, S. and Minaeibidgoli, B.: Estimator learning automata for feature subset selection in high-dimensional spaces, case study: Email spam detection. *International Journal of Communication Systems*, **31**(7), URL https://doi.org/10.1002/dac.3541.

[9] Aggarwal, A., Singh, J., and Gupta, K.: A review of different text categorization techniques. *International Journal of Engineering and Technology*, **7**(3), (2018), 11–15, URL https://doi.org/10.14419/ijet.v7i3.8.15210.

[10] LeCun, Y., Bengio, Y., and Hinton, G.: Deep learning. *Nature*, **521**, (2015), 436–444, URL https://doi.org/10.1038/nature14539.

[11] Huang, J. and Ling, C.: Using auc and accuracy in evaluating learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, **17**(3), (2005), 299–310, URL https://doi.org/10.1109/TKDE.2005.50.

[12] Krishna, D. G. J., Ravi, V., Reddy, B. V., Zaheeruddin, M., Jaiswal, H., Teja, P. S. R., and Gavval, R.: Sentiment classification of indian banks' customer complaints. In *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*, 2019, pp. 429–434, URL https://doi.org/10.1109/TENCON.2019.8929703.

[13] Cortes, C. and Vapnik, V.: Support-vector networks. *Machine Learning*, **20**(3), (1995), 273–297, URL https://doi.org/10.1007/BF00994018.

[14] Berrar, D.: Bayes' theorem and naive bayes classifier. In S. Ranganathan, M. Gribskov, K. Nakai, and C. Schönbach (eds.), *Encyclopedia of Bioinformatics and Computational Biology*, pp. 403–412, Academic Press, Oxford, ISBN 978-0-12-811432-2, 2019, URL https://www.sciencedirect.com/science/article/pii/B9780128096338204731.

[15] Wright, R.: Logistic regression. In L. Grimm and P. Yarnold (eds.), *Reading and Understanding Multivariate Statistics*, pp. 217–244, American Psychological Association, Washington DC, 1995.

[16] Rokach, L. and Maimon, O.: Decision trees. In O. Maimon and L. Rokach (eds.), *Data Mining and Knowledge Discovery Handbook*, pp. 165–192, Springer US, Boston, MA, ISBN 978-0-387-25465-4, 2005, URL https://doi.org/10.1007/0-387-25465-X_9.

[17] Peterson, L.: K-nearest neighbor. *Scholarpedia*, **4**(2), (2009), 1883.

[18] Biau, G. and Scornet, E.: A random forest guided tour. *TEST*, **25**, (2016), 197–227, URL https://doi.org/10.1007/s11749-016-0481-7.

[19] Chen, T. and Guestrin, C.: Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, Association for Computing Machinery, New York, NY, USA, ISBN 9781450342322, 2016, pp. 785–794, URL https://doi.org/10.1145/2939672.2939785.

[20] Murtagh, F.: Multilayer perceptrons for classification and regression. *Neurocomputing*, **2**(5), (1991), 183–197, URL https://doi.org/10.1016/0925-2312(91)90023-5.

[21] Saleh, M. M. N.: *Customer Complaints Auto-assignment using Machine Learning Algorithms*. Master's thesis, Rochester Institute of Technology, 2020.

[22] Breiman, L., Friedman, J., Stone, C., and Olshen, R.: *Classification and Regression Trees*. Taylor & Francis, 1984, ISBN 9780412048418.

[23] Onan, A., Atik, E., and Yalçin, A.: *Machine Learning Approach for Automatic Categorization of Service Support Requests on University Information Management System*, pp. 1133–1139. Springer, ISBN 978-3-030-51155-5, 2021.

[24] Bozyiğit, F., Dogan, O., and Kilinç, D.: Categorization of customer complaints in food industry using machine learning approaches. *Journal of Intelligent Systems Theory and Applications*, **5**, (2022), 85–91.

[25] Tiwari, S.: Consumer complaints dataset for NLP. 2021, URL www.kaggle.com/datasets/shashwatwork/consume-complaints-dataset-fo-nlp. Accessed: 2022-11-30.

[26] Breiman, L.: Random forests. *Machine Learning*, **45**, (2001), 5–32, URL https://doi.org/10.1023/A:1010933404324.

[27] Marhon, S. A., Cameron, C. J. F., and Kremer, S. C.: Recurrent neural networks. In M. Bianchini, M. Maggini, and L. C. Jain (eds.), *Handbook on Neural Information Processing*, pp. 29–65, Springer Berlin Heidelberg, Berlin, Heidelberg, ISBN 978-3-642-36657-4, 2013, URL https://doi.org/10.1007/978-3-642-36657-4_2.

[28] Hochreiter, S. and Schmidhuber, J.: Long short-term memory. *Neural computation*, **9**(8), (1997), 1735–1780.

[29] Zhang, A., Lipton, Z. C., Li, M., and Smola, A. J.: Dive into deep learning, 10.1. lstm. 2019, URL https://d2l.ai/chapter_recurrent-modern/lstm.html. Accessed: 2023-01-24.

[30] Jang, B., Kim, M., Harerimana, G., Kang, S.-u., and Kim, J. W.: Bi-lstm model to increase accuracy in text classification: Combining word2vec cnn and attention mechanism. *Applied Sciences*, **10**(17), URL https://www.mdpi.com/2076-3417/10/17/5841.

[31] Product classification using Random Forest. 2021, URL www.kaggle.com/code/sushantghorpade/product-classification-using-randomforest. Accessed: 2023-01-27.

[32] Text classification with RNN LSTM. 2022, URL www.kaggle.com/code/thinkstudio21/text-classification-with-rnn-lstm. Accessed: 2023-01-27.

[33] Multiclass complaints classification using Bi-LSTM. 2021, URL www.kaggle.com/code/vikram92/multiclass-complaints-classification-using-bi-lstm. Accessed: 2023-01-27.

[34] Popescu, M.-C., Balas, V., Perescu-Popescu, L., and Mastorakis, N.: Multilayer perceptron and neural networks. *WSEAS Transactions on Circuits and Systems*, **8**(7), (2009), 579–588.

[35] Brownlee, J., Tam, A., and Chng, Z. M.: *Deep Learning with Python*. Private publishing, 2022. 2nd Edition.