International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Investigating the Effectiveness of Word2Vec for Spam Detection Using Lazy Predict Library

Aissa Fellah*[1], Kheireddine Mekkaoui[2], Ahmed Zahaf[3], Atilla Elçi[4]

Submitted: 29/01/2024 **Revised**: 07/03/2024 **Accepted**: 15/03/2024

Abstract: The proliferation of Email for exchanging information and messages through internet coincides with a significant rise in unsolicited email (spam),making it increasingly difficult for users to manage their in-boxes and identify legitimate messages. A multitude of detection methodologies have been established and refined to address the deluge of unsolicited electronic mail messages. These approaches encompass knowledge-based techniques, clustering algorithms, learning-based models, heuristic algorithms, and potentially other methodologies. It is noteworthy that while numerous advancements have been made, none of these detection models or techniques have achieved perfect predictive accuracy. Within the domain of spam email detection, machine learning(ML) and deep learning(DL) algorithms have emerged as the most effectual methodologies amongst the plethora of models proposed. Choosing the optimal model for a ML problem can be a challenging task.To solve this problem, we start by converting the email text into vector features using word2Vec and applying various machine learning classifiers on the dataset using Lazy Predict classifiers with default parameters for ML models, We'll then evaluate our basic model's performance after fine-tuning Word2Vec hyperparameters. Here basic model means "Model without parameters", we chose the best models, then applied a hyper parameter adjustment to them. This investigation explores the efficacy of word2Vec with ML in spam email classification. The proposed approach achieved a commendable accuracy of 0.99, signifying its potential as a valuable tool for enhancing spam detection capabilities.

Keywords: *E-mail spam, Word2Vec, Machine learning technique, Lazy Predict*

1. Introduction

Emails and SMSs are the most popular tools in today's communications, and as the number of email and SMS users increases, so does the volume of messages. A recent study by the Radicati Group, Inc. [16] projects that global daily email traffic reached 347.3 billion in 2023. This represents a 4.3% increase from the 2022 figure of 333.2 billion emails sent and received daily. This number is expected to continue growing at a similar rate in 2024, reaching 361.6 billion. Attackers commonly use spam, which are unsolicited electronic messages, to entice recipients into visiting a malicious web page, replying with personal information that can lead to extra charges, or falling for false advertising offers and scams. A significant portion of email communication, exceeding one-third according to user surveys, is marred by the presence of spam. Spam, defined as unsolicited and unwanted bulk digital messages, leads to a substantial misallocation of network resources.

[1]*Computers science department, University of Moulay Tahar of Saida, Saida, Algeria*
ORCID ID : 0000-0002-5062-3041
[2]*Computers science department, University of Moulay Tahar of Saida, Saida, Algeria*
ORCID ID : 0000-0002-5994-4212
[3] *Computers science department, University of Moulay Tahar of Saida, Saida, Algeria*
ORCID ID : 0000-0002-2676-9956
[4]*Hasan Kalyoncu University, Gaziantep, Türkiye*
ORCID ID : 0000-0002-3329-0150
** Corresponding Author Email: ammfellah@gmail.com*

This occurs due to the unnecessary network traffic generated by the proliferation of such emails and text messages. Spam remains a scourge on the internet. There are various approaches to spam detection using ML[8], including “rule-based techniques”, “content-based techniques”, and “hybrid techniques” that combine both rule-based and content-based approaches[1,6,20]. Content-based approaches typically involve analyzing the text of the email and extracting features such as word frequency and length, while rule-based approaches use a set of pre-defined rules to identify spam emails. Supervised learning approaches, including "decision trees", "support vector machines (SVMs)", and "artificial neural networks (ANNs)", have proven to be highly successful in spam detection using ML [1]. The models are trained on a labeled dataset of emails, where each email is categorized as spam or legitimate. This training empowers the model to classify new emails into these categories. Word2vec, a popular technique for understanding text, creates a vector representations called "word embeddings," [2]. Machine learning models can then be trained using these word embeddings for various tasks, including spam detection. This investigation aims to comprehensively evaluate the influence exerted by factors extrinsic as message représentation and split ration to the ML model itself on the culmination of the model's performance. In order to further explor machine learning techniques for spam detection, we focused on two primary strategies. Firstly, we use Word2Vec, which is widely regarded as a widely adopted

neural network-based techniques approach for learning word embeddings. This method was first introduced by Tomas Mikolov at Google in 2013 [2]. Secondly, we used Auto-ML to automatically choose the best model, making it easier to adjust its settings. This paper explores the Efficacy of word2Vec models for spam detection using a wide range of ML techniques. The process involves importing all libraries, tuning the parameters, comparing all the models, and evaluating the model performance using different objectives. This process can take a lot of time, but Lazy Predict is a solution that addresses this issue by selecting the top-performing models.

The remainder of this paper is organized as follows. Section 2 presents the basic concepts essential for understanding this study. Section 3 reviews related works in this area. The proposed approach for spam detection is thoroughly presented in Section 4. Section 5 presents the results obtained and their analysis. Finally, we conclude by summarizing the main findings and potential future directions.

2. Basic Concepts

This section introduces the fundamental concepts that form the basis of the proposed investigation.

2.1. Word2Vec

In natural language processing, mapping words to numerical representations (vectors) called "Word embedding". This allows computers to analyze and process natural language in a more efficient and effective way, as it captures the meaning and context of words. For instance, “'University' and 'Student' might have similar vectors because they are related words". There are several methods for generating word embeddings (Table 1). Among these models, “Bidirectional Encoder Representations from Transformers (BERT)” [4] and “Generative Pre-trained Transformer (OpenAI GPT)” [5] Have consistently achieved top performance on a variety of NLP applications.

Table 1. Word Embeddings Models

Word Embeddings Models			
No context		Context	
No ML	With ML	RNN	Transformer
-Bag of words	-Word2Vec	-ELMO	-Bert
-TF-IDF	-Glove	-COVE	-OpenAI GPT
	-FastText	-ULMFit	-RoBERTA
		Generative	

Word2Vec[2] is one of the most popular Word embedding technique is developed by Google researchers in 2013. Word2Vec is a neural network-based family of models that learns the vector representation of words by predicting their surrounding words. they are used for generating word embeddings, which are dense vector representations of words. The two main algorithms in the Word2Vec family are(Fig.1): 1.Continuous Bag-of-Words (CBOW): This model predicts the current word based on the context words (i.e., the words that surround it). 2.Skip-Gram: This model predicts the context words based on the current word. Both CBOW and Skip-Gram are trained using a neural network architecture, where the input is a one-hot encoded vector of the current word (or context words) and the output is a probability distribution over all the words in the vocabulary. There are also variations of Word2Vec, such as: GloVe (Global Vectors for Word Representation)[3]: This model is similar to Word2Vec but uses a different objective function to generate word embeddings. FastText: This model is an extension of Word2Vec that takes into account subword information (i.e., character n-grams) in addition to whole words to generate embeddings. The Word2Vec family of models are widely used in natural language processing tasks, including spam filtering, due to their ability to generate high-quality word embeddings that capture semantic and syntactic relationships between words.

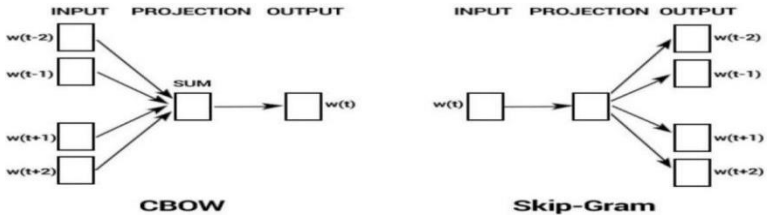

Fig. 1. Word2Vec family .**“** CBOW and skip-gram models”

2.2. Machine Learning and Spam email detection

Imagine a computer program that can improve on its own, without needing specific instructions every time. That's the core idea behind Machine Learning! It's a field of computer science that allows machines to learn from data[7], just like humans do from experience. Machine learning algorithms are like detectives, sifting through data to find hidden clues and patterns. They use statistical techniques to analyze this information and learn from it. Here's a breakdown of the different learning styles[8]: Supervised Learning: Think of it as training a student with a textbook. The data is labeled, meaning each piece of information has a clear answer attached. By studying these examples, the machine learns to recognize patterns and make predictions for new, unseen data. Unsupervised Learning: This is like exploring a new world. The data isn't labeled, so the machine has to find its own patterns and relationships within the information. It might discover hidden groups or categories in the data. Reinforcement Learning: Imagine training a pet through trial and error. The machine interacts with its environment and receives feedback (rewards or penalties) based on its actions. This feedback helps the machine learn the best course of action for different situations. Machine learning is revolutionizing many fields, from spam filtering to medical diagnosis[22]. By learning from vast amounts of data, machines can help us solve complex problems and make better decisions. The increasing availability of data, computing power, and open-source libraries have made it easier for organizations to adopt Machine Learning in their operations. Some popular Machine Learning libraries and frameworks include TensorFlow, PyTorch, Scikit-learn, Keras, and Apache Spark [8]. One of the key applications of machine learning is spam detection, ML algorithms analyze email content (text, sender info, etc.) to classify them as spam or legitimate emails.

2.3. Lazy Predict

Selecting the prediction model using auto-machine learning Automated machine learning (Auto-ML) was used to reduce the workload of data training, hyperparameter tuning, and others. Some technologies facilitate the steps of an AutoML project, such as Auto-Keras[9], Auto Sklearn, H2O , MLBox, Lazy Predict[10]. All these tools have specific configurations. For the present work, the Lazy Predict framework is more appropriate, considering that it has an open-source code, comprises steps deemed necessary for evaluating MTL, and can potentially train many models. In addition, it is applicable in the Python computational program, which presents a simple, effective, and versatile language. We leveraged the LazyPredict library (open-source Python) for efficient model training. LazyPredict provides trained models along with their performance metrics. LazyPredict (MIT License) by Shankar Rao Pandala is the Python library used for model training. Refer to the documentation for more details: https://lazypredict.readthedocs.io/. The library is designed to simplify the process of model selection and evaluation in ML and Widely used across various applications. LazyPredict gives you a clear report that shows how well each algorithm performs. Think of it as a report card, allowing you to see which algorithm gets the best grades for your specific task. The library was inspired by the AutoML concept, which aims to automate the machine learning process as much as possible. However, instead of trying to find the best model and hyperparameters automatically, Lazy Predict focuses on quickly testing multiple models and selecting the most promising ones for further optimization. Lazy Predict has been integrated into popular machine learning platforms, such as H2O.ai and DataRobot. Lazy Predict includes a variety of classifiers and regressors for both binary and multi-class classification tasks. Here is a list of the classifiers currently supported by Lazy Predict[10]:

“AdaBoost, Bagging, BernoulliNB, CalibratedClassifierCV, Decision Tree, Dummy, Extra Tree, Extra Trees, GaussianNB, GradientBoosting, KNeighbors, LabelPropagation, LabelSpreading, Linear Discriminant Analysis, Logistic Regression, Logistic RegressionCV, MLP Classifier, MultinomialNB, Nearest Centroid, NuSVC, PassiveAggressive Classifier, Perceptron, Quadratic Discriminant Analysis, RandomForest Classifier, Ridge Classifier, RidgeClassifierCV, SGD Classifier, SVC.”

3. Related Works

Spam tactics constantly evolve, demanding continuous improvement in detection methods. Machine learning and deep learning approaches have gained significant traction, with the quality of generated embeddings playing a crucial

role in achieving accurate classification. The study in [11] compared embedding techniques (count vectorizer, TF-IDF vectorizer, hashing vectorizer) with machine learning models and LSTMs. Using the UCI SMS Spam Collection dataset, LSTM combined with TF-IDF achieved the highest accuracy (98.5%). This highlights the effectiveness of TF-IDF in capturing relevant information for SMS spam classification. Paper [12] compared eight classifiers (SVM, NB, DT, LR, RF, AdaBoost, Neural Networks, CNN) for SMS spam detection. Their experiments revealed that CNN and Neural Networks outperformed others on the SMS Spam Collection v.1 dataset, achieving accuracies of 98.25% and 99.10% respectively. This suggests the potential of deep learning models for superior spam detection. Study [13] presented a deep learning model utilizing CNNs and LSTMs for SMS spam classification. While achieving 99.44% accuracy, the impact of different embedding techniques wasn't explored. Jain et al. [14] proposed an SLSTM model that extends LSTM with a semantic layer using Word2Vec embeddings. This model achieved high accuracy (99.01% on SMS Spam Collection, 95.09% on Twitter) demonstrating the effectiveness of incorporating word meaning through semantic embeddings. Building on SLSTM, Wei and Nguyen [15] proposed a Lightweight Gated Recurrent Unit (LGRU) with WordNet integration for faster training and improved word understanding. This model achieved 99.04% accuracy, highlighting the potential of combining efficient architectures with external knowledge sources. Gashti [17] explores a method combining the harmony search algorithm and decision trees for spam classification, achieving 95.25% accuracy on the Spam-base dataset and 99.80% on Ling-spam. This demonstrates the potential of non-deep learning approaches for specific datasets. The approach in [18] combines TF-IDF with an Artificial Neural Network (ANN) for spam email detection, achieving 97.58% accuracy. This highlights the viability of TF-IDF and ANNs for spam filtering. Paper [19] proposes a novel spam detection model leveraging pre-trained BERT models, achieving 97% overall accuracy across four datasets. This showcases the promise of pre-trained language models for efficient and universal spam detection. The quest for robust SMS spam detection necessitates exploration of various techniques. This analysis highlights the effectiveness of deep learning models, particularly CNNs and LSTMs, when combined with appropriate embedding techniques like TF-IDF and Word2Vec. The potential of non-deep learning approaches and pre-trained language models also warrants further investigation.

4. Methodology

The proposed approach to spam email classification has several phases: preprocessing, feature extraction, training, prediction, tuning the Word2Vec optimal combination parameters, choice the top best models, fine tuning model and final prediction résults. firstly we operate a preprocessing(removing unnecessary information or fixing typos), next extract features from the text(numecal representation). A range of ML(see 2.3 section) are trained on the extracted features. At last, the prediction step(analyzes the email's features and predicts whether it's a spam message or a real email).

4.1. Preprocessing

The main tasks involved in the Preprocessing step: -Make all characters into lowercase, -Remove stop-words, numbers, punctuations, links, white-spaces, Emojis, Tokenization: The process of splitting text data into smaller units(word) called tokens, Lemmatization: The process of reducing inflected words to their base form (lemma).

4.2. Feature extraction

Once the emails are prepared and cleaned, the system tackles another step: turning the the training set or the test set into a format that machines can understand(a Word2Vec models).

4.2.1. Creating a model with Word2Vec

We leveraged the open-source Python library Gensim [21] to generate a Word2Vec model from the preprocessed dataset. A detailed explanation of the parameters used in this model can be found in Table 2.

Let M be a message consisting of m tokens:

$$\mathrm{M} = \{\mathrm{T}_1, \mathrm{T}_2, .., \mathrm{T}_\mathrm{m}\}$$

Representation of each token $\mathrm{T_i}$ is vector:

$$\mathrm{W2V}(T_i) = \begin{pmatrix} \mathrm{x_{i_1}} \\ \mathrm{x_{i_2}} \\ \vdots \\ \mathrm{x_{i_n}} \end{pmatrix}$$

And a global representation of message:

$$\mathrm{W2V(M)} = \begin{pmatrix} \mathrm{x_1} \\ \mathrm{x_2} \\ \vdots \\ \mathrm{x_n} \end{pmatrix}$$

Where :

$$X_i = (X_{i_1} + X_{i_2} + .. + X_{i_m})/m$$

if empty message (i.e. m=0), we use a zero vector for its representation.

4.3. Training

We leveraged the open-source Python library LazyPredict [10] to streamline the machine learning model training process. This library automates the pipeline, saving us valuable time. It offers a variety of algorithms suitable for multivariate classification task. One of LazyPredict's key strengths is that it returns both the trained models and their

performance metrics. This enables us to perform a comparative analysis of various models and identify the one with the highest performance in terms of accuracy. After an initial evaluation of approximately thirty algorithms, we selected the top performers based on their accuracy. This paves the way for further hyperparameter tuning to potentially refine the chosen model and achieve even better results.

4.3.1. Tuning Word2Vec parameters

Tuning Word2Vec parameters is crucial for squeezing the most performance out of the model for specific task. We employed the open-source Python library Gensim [21] to generate a Word2Vec model from the preprocessed dataset. Word2Vec has several hyperparameters that can be tuned to achieve better performance for a specific task. Here are some of the commonly used hyperparameters: “sentences, corpus_file, vector_size, alpha, window, min_count, max_vocab_size, sample, seed, workers, min_alpha, sg, hs”

Table 2 details the Word2Vec parameters that significantly influence the results. To find the optimal settings, experiment with different values and evaluate the model's performance on our task using relevant metrics. The optimal configuration is highly influenced by factors such as the dimensionality of the feature space, the presence of class imbalances, and the available computational resources, as well as the problem to solve.

Table 2. The Word2Vec commonly used hyperparameters

Parameters	Explanations	type	Best value
vector_size	“Size of the word embedding”	(Facultative int)	300
window	“The maximum context window size for word prediction.”	(Facultative int)	5
min_coun	“Ignores all words with total frequency lower than this.”	(Facultative int)	1
workers	“Use these many worker threads to train the model (=faster training with multicore machines).”	(Facultative int)	4
sg	“Training algorithm: 1 for skip-gram; otherwise CBOW.”	({0, 1}, Facultative)	0
hs	“If 1, hierarchical softmax will be used for model training. If 0, and negative is non-zero, negative sampling will be used.”	({0, 1}, Facultative)	0

5. Experiments

5.1. Datasets

We employed two publicly available datasets for this project:

5.1.1. Ling-spam dataset

This dataset on Kaggle[24] offers 2,893 emails categorized as spam (481) or legitimate (2,412). The emails, likely related to job postings, software discussions, and research opportunities, were derived from the Linguist List archives.

5.1.2. SMS Spam dataset

This dataset, sourced from the UCI Machine Learning Repository and accessed through Kaggle [23], contains 5,574 messages categorized into two classes: spam (724 messages) and legitimate messages (ham, 4,850 messages).

The data originates from the mobile phone spam research domain.

5.2. Performance Metrics

To see how effective our proposed spam filter is, we use a metric called "Accuracy." This essentially tells us how good the filter is at correctly identifying both real emails and spam. The system considers four categories: True Positives (TP): Legitimate emails correctly identified as "good." False Positives (FP): Spam emails incorrectly classified as "good" (mistakes!). True Negatives (TN): Spam emails correctly identified as "spam." False Negatives (FN): Legitimate emails incorrectly classified as "spam". Calculating Accuracy: We take the total number of correctly classified emails (TP + TN) and divide it by the total number of emails analyzed (TP + TN + FP + FN). This gives us a percentage that reflects the overall accuracy of the filter. The closer the accuracy gets to 100%, the better the filter performs.

5.3. Results

This section presents the results obtained from the finalized model by varying train sizes from 90 to 80, different train-test distributions like 90:10. 85:15, 80:20 and vector size set to 300. Accuracy metrics is considered while evaluating our proposed model and is explained as below(Table 3):

Table 3. Summary of our best results

Dataset	Word Embeddings Models	Split Ratio 90:10 ML Model	Acc	85:15 ML Model	Acc	80:20 ML Model	Acc
Ling-spam	TF-ID	Linear SVC	99.31	LinearSVC	99.31	XGBClassifier	99.31
	Word2Vec	Ridge Classifier CV	99.99	LogisticRegression	99.99	PassiveAggressiveClassifier	99.41
SMS Spam	TF-ID	RandomForest Classifier	99.61	ExtraTreesClassifier	98.58	ExtraTreesClassifier	98.26
	Word2Vec	SVC	99.14	SVC	99.24	SVC	99.28

Our initial evaluation identified five promising models: SVC, ExtraTreesClassifier, LogisticRegression, XGBClassifier, and RidgeClassifierCV. All achieved accuracy exceeding 0.98, which is a strong starting point. We can likely improve performance even further through hyperparameter tuning. The complete experimental results are presented in the appendix, and the next section will discuss these findings in more detail. We wanted to see how our spam filter performed compared to other approaches. Since previous studies on this dataset all reported accuracy, we used this metric as a common ground for comparison. Tables 4 and 5 show a comparison of our model's accuracy with different models used in past research on the same dataset.

Table 4. Comparative analysis of our model's accuracy against prior research on the SMS spam dataset.

Paper	Model	ACC
Our work	Word2Vec+SVC	99,28
[11]	LSTM	98.50
[12]	CNN	99.10
[13]	CNN	99.44
[14]	Word2Vec + LSTM	99.01
[15]	WordNet + LGRU	99.04

Table 5. Comparative analysis of our model's accuracy against prior research on the Ling dataset

Paper	Model	ACC
Our work	RidgeClassifier CV	99,99
[18]	ANN	97.50
[17]	Hamonic search algorithm + DT	99.80
[19]	BERT	98.00

5.4. Discussion

This study explores how well machine learning models can identify spam emails. We compared two techniques for representing words in emails: Word2Vec (a new method) and TF-IDF (a common method). We also investigated how splitting the data for training and testing the models affects their performance. Tried different techniques: We tested various machine learning approaches to classify emails as spam or legitimate. Splitting the data: We divided the email data into different training and testing sets (e.g., 80% for training, 20% for testing) to see how the models perform on unseen data. This is the first time a study has looked at how these data splits impact spam detection accuracy. Evaluating performance: We used a metric called "Accuracy" to measure how well each model classified the emails. The goal was to find the most accurate model for predicting spam. You can find detailed results in Fig. 2-5, which compare the performance of Word2Vec and TF-IDF across different data splits. Tables 6-9 show the top 10 most accurate models for each data split scenario. These findings highlight the importance of considering data splitting strategies when training machine learning models for spam detection. There might not be a single "perfect" model, and the best approach might depend on how the data is divided.

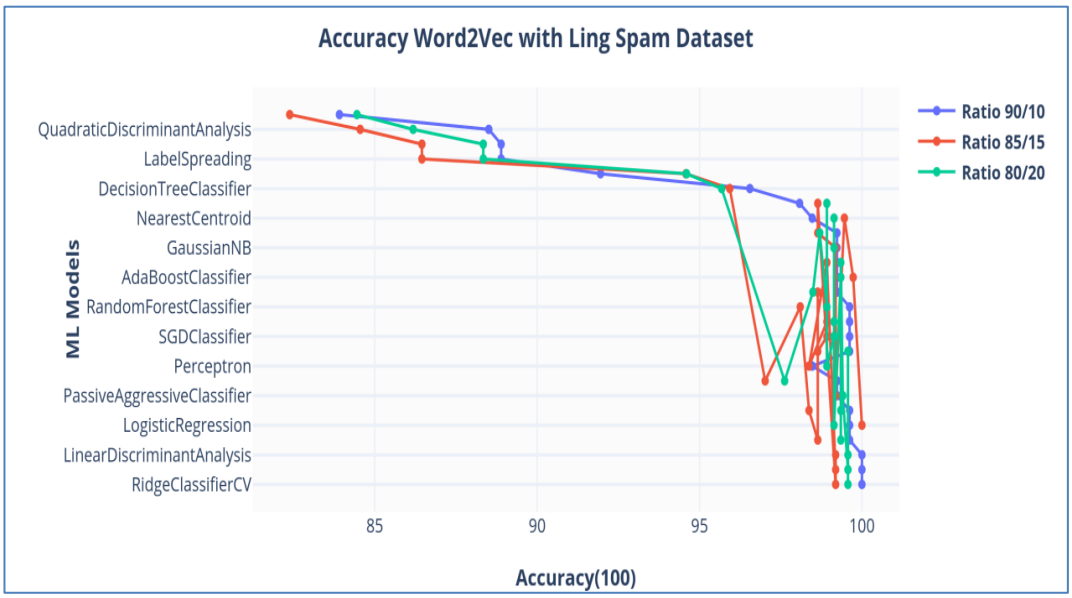

Fig. 2. AccuracyLing Spam dataset with (ML+Word2Vec)

Table 6. Top 10 results: Dataset Ling Spam (ML+Word2Vec)

Ratio 90/10		Ratio 85/15		Ratio 80/20	
Model	Acc	Model	Acc	Model	Acc
RidgeClassifierCV	99.99	LogisticRegression	99.99	LinearDiscriminantAnalysis	99.57
RidgeClassifier	99.99	AdaBoostClassifier	99.73	CalibratedClassifierCV	99.57
LinearDiscriminantAnalysis	99.99	NearestCentroid	99.46	RidgeClassifierCV	99.57
LGBMClassifier	99.62	PassiveAggressiveClassifier	99.19	RidgeClassifier	99.57
LogisticRegression	99.62	GaussianNB	99.19	PassiveAggressiveClassifier	99.41
XGBClassifier	99.62	LinearSVC	98.64	KNeighborsClassifier	99.35
PassiveAggressiveClassifier	99.23	BernoulliNB	98.64	XGBClassifier	99.35
BaggingClassifier	99.23	RidgeClassifierCV	99.19	LGBMClassifier	99.35
Perceptron	98.47	LinearDiscriminantAnalysis	99.19	AdaBoostClassifier	99.35
CalibratedClassifierCV	99.62	RidgeClassifier	99.19	LogisticRegression	99.14

- Ling-Spam Dataset :

With Word2Vec usage, the RidgeClassifierCV, achieving the strong accuracy of 99.99% if the training dataset is of 90%, LogisticRegression, achieving the strong accuracy of 99.99% if the training dataset is of 85% and PassiveAggressiveClassifier, achieving the strong accuracy of 99.14% if the training dataset is of 80%

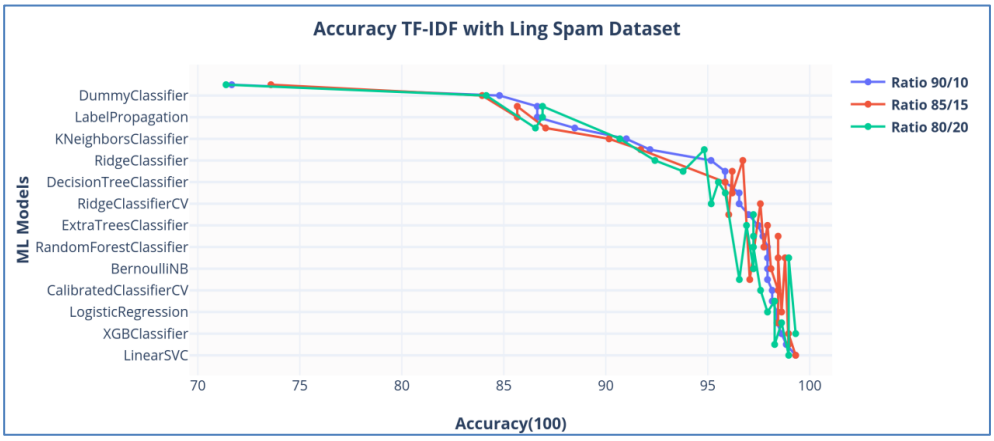

Fig. 3. AccuracyLing Spam dataset with (ML+TF-IDF)

Table 7. Top 10 results: Dataset Ling Spam (ML+ TF-IDF)

Ratio 90/10		Ratio 85/15		Ratio 80/20	
Model	Acc	Model	Acc	Model	Acc
LinearSVC	99.31	LinearSVC	99.31	XGBClassifier	99.31
PassiveAggressiveClassifier	98.85	XGBClassifier	98.96	Perceptron	98.97
XGBClassifier	98.62	PassiveAggressiveClassifier	98.96	LinearSVC	98.97
LGBMClassifier	98.39	Perceptron	98.79	LGBMClassifier	98.62
LogisticRegression	98.39	LogisticRegression	98.62	PassiveAggressiveClassifier	98.28
AdaBoostClassifier	98.16	Perceptron	98.45	AdaBoostClassifier	98.28
CalibratedClassifierCV	98.16	NearestCentroid	98.45	LogisticRegression	97.93
BaggingClassifier	97.93	LGBMClassifier	98.45	CalibratedClassifierCV	97.59
BernoulliNB	97.93	CalibratedClassifierCV	98.45	RandomForestClassifier	97.24
Perceptron	97.93	BernoulliNB	98.10	NearestCentroid	97.24

- Ling-Spam Dataset :

With TF-IDF usage LinearSVC, achieving the strong accuracy if the training dataset is of 90% or 85% and XGBClassifier, achieving the strong accuracy if the training dataset is of 80%

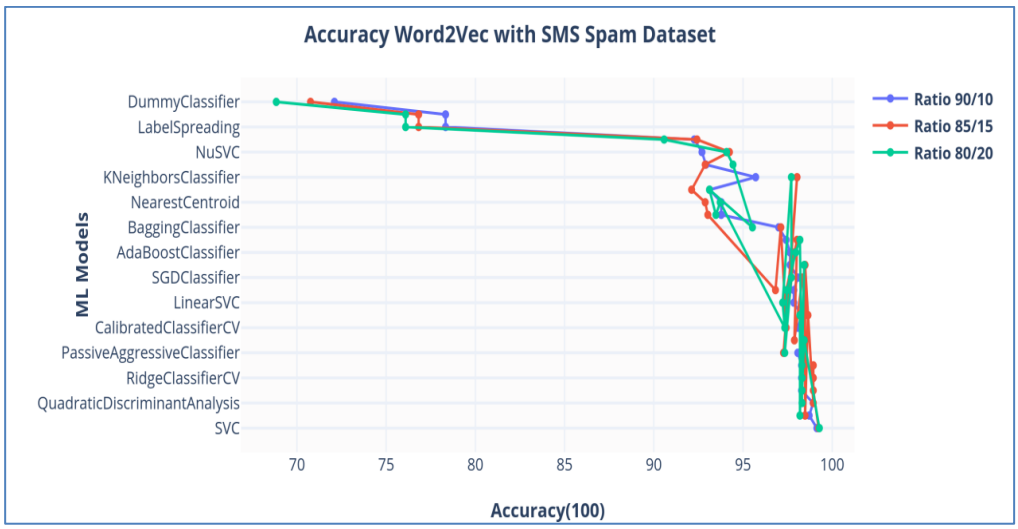

Fig. 4. Accuracy SMS Spam dataset with (ML+Word2Vec)

Table 8. Experiments results: Dataset SMS Spam (ML+Word2Vec)

Ratio 90/10		Ratio 85/15		Ratio 80/20	
Model-10	Acc-10	Model-15	Acc-15	Model-20	Acc-20
SVC	99.14	SVC	99.24	SVC	99.28
LogisticRegression	98.71	LinearDiscriminantAnalysis	98.94	LGBMClassifier	98.43
QuadraticDiscriminantAnalysis	98.93	RidgeClassifierCV	98.94	LinearDiscriminantAnalysis	98.31
LinearDiscriminantAnalysis	98.28	RidgeClassifier	98.94	RidgeClassifierCV	98.31
RidgeClassifierCV	98.28	QuadraticDiscriminantAnalysis	98.94	RidgeClassifier	98.31
RidgeClassifier	98.28	ExtraTreesClassifier	98.48	ExtraTreesClassifier	98.43
PassiveAggressiveClassifier	98.07	LogisticRegression	98.48	LogisticRegression	98.19
LGBMClassifier	98.50	XGBClassifier	98.64	XGBClassifier	98.19
CalibratedClassifierCV	98.07	SGDClassifier	98.33	QuadraticDiscriminantAnalysis	98.31
XGBClassifier	98.28	LGBMClassifier	97.88	RandomForestClassifier	98.19

- Spam text messages dataset:

With Word2Vec usage, SVC, achieving the strong accuracy with tall raining dataset ratio

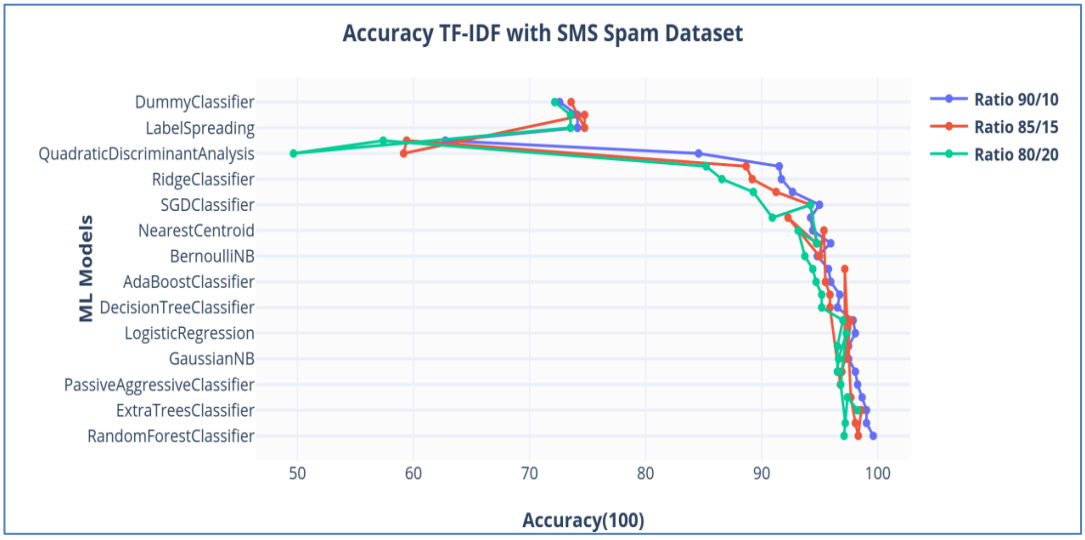

Fig. 5. Accuracy SMS Spam dataset with (ML+TF-IDF)

Table 9. Top 10 results: Dataset SMS Spam (ML+TF-IDF)

Ratio 90/10		Ratio 85/15		Ratio 80/20	
Model	Acc	Model	Acc	Model	Acc
RandomForestClassifier	99.61	ExtraTreesClassifier	98.58	ExtraTreesClassifier	98.26
LGBMClassifier	99.03	RandomForestClassifier	98.32	XGBClassifier	97.39
ExtraTreesClassifier	99.03	LGBMClassifier	98.07	RandomForestClassifier	97.10
XGBClassifier	98.65	XGBClassifier	97.68	LGBMClassifier	97.20
PassiveAggressiveClassifier	98.26	Perceptron	97.16	PassiveAggressiveClassifier	96.81
LinearSVC	98.07	GaussianNB	97.16	LogisticRegression	97.29
GaussianNB	97.49	CalibratedClassifierCV	97.68	GaussianNB	96.62
SVC	97.49	SVC	97.42	LinearSVC	96.52
LogisticRegression	98.07	LinearSVC	96.91	SVC	96.52
CalibratedClassifierCV	97.88	LogisticRegression	97.29	CalibratedClassifierCV	97.00

- Spam text messages dataset:

With TF-IDF usage, RandomForestClassifier, achieving the strong accuracy if the training dataset is of 90% and ExtraTreesClassifier, achieving the strong accuracy if the training dataset is of 85% or 80%.

Finaly: It can be observed from the above that, SVC, RidgeClassifierCV, LogisticRegression, and PassiveAggressiveClassifier achieved an enriched accuracy. The results obtained from Word2Vec are better than those with TF-IDF. We can conclude that Word2Vec is a potential model for use in spam filtering.

6. Conclusion

In today's email-flooded world, effective spam detection is crucial for businesses. A lot of great work has been done on spam filtering in the past. However, many traditional methods struggle to keep up with the ever-changing world of spam.Word2Vec have emerged as a promising solution for representing text data, combined with a relevant selected of machine learning model. The task of choosing the appropriate model has been greatly simplified with lazy predict. This research is a first step in understanding how data splitting and word representation techniques influence spam detection with machine learning. By analyzing these results, we can identify the best model and data split combination for accurate spam filtering. we demonstrated the Effectiveness of Word2Vec for Spam Detection Using Lazy Predict Library help achieve better results in review spam detection.

References

[1] Ahmed, N. ,Amin, R., Aldabbas, H. ,Kounda, D. ,Alouff, B., Shah, T. , "Machine Learning Techniques for Spam Detection in Email and IoT Platforms: Analysis and Research Challenges", Security and Communication Networks, vol. 2022, pp. 1–19, 2022.

https://doi.org/10.1155/2022/1862888.

[2] Mikolov, T., Chen, K., Corrado, G., Dean, J. , "Efficient Estimation of Word Representations in Vector Space", arXiv.org, Sep. 07, 2013.

https://arxiv.org/abs/1301.3781

[3] Pennington, J.,Socher, R., Manning, C., "GloVe: Global Vectors for Word Representation," Association for Computational Linguistics, 2014. Available:

https://aclanthology.org/D14-1162.pdf

[4] Devlin, J., Chang, M.-W., Lee, K., Toutanova, K., "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," arXiv.org, Oct. 11, 2018. https://arxiv.org/abs/1810.04805

[5] Kaplan, N. R. M. S. J., Shyam, P. D. A. N. P., Agarwal, G. S. A. A. S., Tom, A. H. V. G. K., Daniel, H. R. C. A. R., Winter, M. Z. J. W. C., ... & Mann, B., "Language models are few shot learners", ar5iv.labs.arxiv.org, (2020).

https://ar5iv.labs.arxiv.org/html/2206.10498

[6] Nallamothu T., Shais Khan, M., "Machine Learning for SPAM Detection", Asian Journal of Advances in Research, vol. 6(1), pp. 167–179, 2023.

http://eprint.subtopublish.com/id/eprint/3333/

[7] Alpaydin, E., "Introduction to machine learning", The Mit Press, 2014. ISBN: 9780262043793 https://mitpress.mit.edu/9780262043793/introduction-to-machine-learning/

[8] Jordan, M. I., Mitchell, T. M., "Machine learning: Trends, perspectives, and prospects," Science, vol. 349, no. 6245, pp. 255–260, Jul. 2020.

https://doi.org/10.1126/science.aaa8415.

[9] Raschka, S., Patterson, J., Nolet, C., "Machine Learning in Python: Main Developments and Technology Trends in Data Science, Machine Learning, and Artificial Intelligence," Information, vol.11(4), p. 193, 2020. https://doi.org/10.3390/info11040193.

[10] Pandala, S. R., "shankarpandala/lazypredict," GitHub, Jan. 10, 2024.

https://github.com/shankarpandala/lazypredict

[11] Gadde, S., Lakshmanarao, A., Satyanarayana, S., "SMS Spam Detection using Machine Learning and Deep Learning Techniques," 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS), Mar. 2021. https://doi.org/10.1109/icaccs51430.2021.9441783.

[12] Gupta, V., Mehta, A., Goel, A., Dixit, U., Pandey, A. C., "Spam Detection Using Ensemble Learning", Harmony Search and Nature Inspired Optimization Algorithms, pp. 661–668, Aug. 2018. https://doi.org/10.1007/978-981-13-0761-4_63.

[13] Roy, P. K., Singh, J. P., Banerjee, S., "Deep learning to filter SMS Spam," Future Generation Computer Systems, vol. 102, pp. 524–533, Jan. 2020.

https://doi.org/10.1016/j.future.2019.09.001.

[14] Jain, G., Sharma, M., Agarwal, B., "Optimizing semantic LSTM for spam detection," International Journal of Information Technology, vol. 11(2), pp. 239–250, 2018. https://doi.org/10.1007/s41870-018-0157-5

[15] Wei, F., Nguyen, T., "A Lightweight Deep Neural Model for SMS Spam Detection," 2020 International Symposium on Networks, Computers and Communications (ISNCC), Oct. 2020. https://doi.org/10.1109/isncc49221.2020.9297350.

[16] "The Radicati Group, Inc." https://www.radicati.com/

[17] Gashti, M. Z., "Detection of Spam Email by Combining Harmony Search Algorithm and Decision Tree," Engineering, Technology & Applied Science Research, vol. 7(3), pp. 1713–1718, 2017. https://doi.org/10.48084/etasr.1171.

[18] Bansal, C. , Sidhu, B., "Machine Learning based Hybrid Approach for Email Spam Detection," IEEE Xplore, Sep. 01, 2021. https://ieeexplore.ieee.org/document/9596149.

[19] Tida, V. S., Hsu, S., "Universal Spam Detection using Transfer Learning of BERT Model," arxiv.org, Feb. 2022. https://doi.org/10.48550/arXiv.2202.03480.

[20] Dada, E. G., Bassi, J. S., Chiroma, H., Abdulhamid, S. M., Adetunmbi, A. O., Ajibuwa, O. E., "Machine learning for email spam filtering: review, approaches and open research problems," Heliyon, vol. 5(6), p. e01802, Jun. 2019. https://doi.org/10.1016/j.heliyon.2019.e01802.

[21] "gensim: topic modelling for humans," radimrehurek.com. accessed January 2024

https://radimrehurek.com/gensim/models/word2vec.html,

[22] Donoho, D., "50 Years of Data Science," Journal of Computational and Graphical Statistics, vol. 26, no. 4, pp. 745–766, Oct. 2017. https://doi.org/10.1080/10618600.2017.1384734.

[23] "UCI Machine Learning Repository," archive.ics.uci.edu. (accessed Mar. 10, 2024).

https://archive.ics.uci.edu/ml/datasets/SMS%2BSpam%2BCollecti on.

[24] "Metatext," metatext.io.

https://metatext.io/datasets/ling-spam-dataset