Automating Document Classification for InsureMe Using Machine Learning Model: Support vector machine (SVM)

Michael Zats[a]

[a] *Prague City University , Teesside University,*

Abstract

In the modern digital landscape, insurance companies like InsureMe face challenges in manually classifying vast amounts of unstructured documents. This research investigates the application of machine learning, specifically the Support Vector Machine (SVM) model, to automate this task (Seifert, 2004; Halevy, Norvig, Pereira, 2009). The study's experiments revealed that the SVM model, without any enhancements, achieved an impressive F1-Score of 0.980, outperforming other models with the exception of . This underscores SVM's potential in handling high-dimensional text data and discerning intricate patterns within such data (Joachims, 2002). The automation of document classification not only promises efficiency but also consistency and scalability, marking a significant advancement in data-driven decision-making for the insurance sector.

Keywords: *Document classification, Support Vector Machines, SVM, Machine learning, Text categorization*

1. Introduction

1.1. Background and Significance

In today's digital age, businesses are inundated with vast amounts of data, much of which is unstructured and embedded within documents. The insurance sector, in particular, is characterized by a high volume of incoming documents that need to be processed, categorized, and acted upon (Seifert, 2004). Companies like InsureMe are at the forefront of this challenge, grappling with the manual classification of diverse documents, a task that is both time-consuming and error-prone.

The manual categorization of documents is not only resource-intensive but also susceptible to inconsistencies arising from human error. Such inconsistencies can lead to significant operational inefficiencies, misinformed decision-making, and potential regulatory non-compliance (Halevy, Norvig, Pereira, 2009). As the volume of data continues to grow, the scalability of manual processes is increasingly being called into question.

Machine learning, with its ability to learn patterns from data, offers a promising avenue for automating such repetitive tasks. Specifically, supervised learning techniques, where algorithms are trained on labeled data, have shown significant potential in automating document classification tasks (Alpaydin, 2010). By automating this process, businesses can achieve faster processing times, greater accuracy, and free up human resources for more value-added tasks.

Given the challenges faced by InsureMe and the potential of machine learning, this research delves into the application of machine learning techniques for automating document classification, aiming to enhance efficiency and accuracy in the process.

1.2. Problem Statement

The insurance industry is inherently document-intensive. From policy applications and claims submissions to underwriting documents and customer correspondence, insurers like InsureMe handle a plethora of documents daily. Each of these documents needs to be accurately categorized to facilitate efficient processing, timely decision-making, and adherence to regulatory requirements.

Currently, the classification of these incoming documents at InsureMe is predominantly manual. Human agents sift through each document, determining its nature, and then categorizing it into one of several predefined categories. This manual approach presents several challenges:

Scalability Issues: As the volume of incoming documents grows, the manual approach becomes increasingly unsustainable. The time taken to process each document multiplies, leading to backlogs and delays.

Inconsistencies and Errors: Human agents, despite their best efforts, are prone to errors and inconsistencies. A document might be miscategorized, leading to potential processing delays, customer dissatisfaction, and even regulatory breaches.

Operational Costs: The manual classification of documents requires a significant workforce. As the volume of documents increases, so does the operational cost associated with their processing.

Opportunity Costs: Human agents engaged in repetitive document classification tasks could be better utilized in roles that add more value to the organization, such as customer engagement, claims assessment, or fraud detection.

Given these challenges, there's a pressing need to explore automated solutions that can efficiently and accurately classify incoming documents, reducing the dependency on manual intervention. The goal is to harness the power of machine learning to develop a system that can automatically categorize documents, ensuring speed, accuracy, and scalability in the document processing workflow of InsureMe.

1.3. Objective

The primary objective of this research is to investigate and develop an automated document classification system tailored for the insurance domain, specifically for InsureMe's operational needs. The overarching aim is to transition from a labor-intensive manual classification process to an automated, machine learning-driven approach. The specific objectives are as follows:

Literature Review: To conduct a comprehensive review of existing literature on document classification methodologies, with a particular focus on applications within the insurance sector. This will provide insights into state-of-the-art techniques, their strengths, and limitations (Sebastiani, 2002).

Data Collection and Preprocessing: To gather a representative dataset of insurance-related documents and preprocess this data to make it suitable for machine learning algorithms. This involves data cleaning, normalization, and feature extraction (Indurkhya Damerau, 2010).

Model Development: To design, implement, and train machine learning models that can automatically classify incoming documents into predefined categories. The focus will be on both traditional machine learning algorithms and deep learning architectures, assessing their applicability and performance for the task (LeCun, Bengio Hinton, 2015).

Model Evaluation: To rigorously evaluate the performance of the developed models using appropriate metrics. This will involve cross-validation techniques and comparisons against benchmark models to ensure the robustness and accuracy of the proposed system (Sokolova Lapalme, 2009).

Integration and Deployment: To integrate the best-performing model into InsureMe's operational workflow, ensuring seamless automation of the document classification process.

Ethical Considerations: To ensure that the automated classification system adheres to ethical guidelines, especially concerning data privacy and security. This involves ensuring that personal data within documents is handled with utmost care and in compliance with regulatory standards (Mittelstadt, Allo, Taddeo, Wachter Floridi, 2016).

By achieving these objectives, the research aims to provide InsureMe with a scalable, efficient, and accurate document classification system, reducing operational costs, enhancing productivity, and ensuring a higher level of service quality for its customers.

2. Literature Review

2.1. Previous Work

Document classification, an essential component of text categorization, has been a subject of extensive research due to its vast applications across various sectors, including insurance. The primary goal is to assign predefined categories or labels to documents based on their content (Aggarwal Zhai, 2012).

Traditional methods, such as rule-based systems, dominated the early stages of text classification. However, these systems, relying on manually crafted rules, often lacked scalability, especially when confronted with vast and diverse datasets (Lewis et al., 2004).

With the rise of machine learning, a significant shift was observed in document classification approaches. Supervised learning algorithms, where models are trained on labeled data, became the norm. Techniques like Support Vector Machines (SVM) and Decision Trees were widely adopted due to their efficiency in text classification tasks (Joachims, 2010).

The introduction of deep learning, particularly neural networks, further transformed the landscape. Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN) demonstrated exceptional performance in document classification, especially with large-scale datasets (Kim, 2014). Their ability to learn hierarchical features from raw text data without manual feature engineering made them particularly appealing.

In the insurance sector, the significance of document classification cannot be overstated. Recent studies have underscored the potential of machine learning and deep learning in classifying insurance-related documents, such as claims and policy documents (Chen Lin, 2012). Automating these processes not only cuts down operational costs but also boosts efficiency and precision.

However, challenges persist. Handling imbalanced datasets, ensuring model transparency, and addressing ethical considerations related to data privacy remain areas of concern (Baeza-Yates, 2018).

However, challenges persist. Handling imbalanced datasets, ensuring model transparency, and addressing ethical considerations related to data privacy remain areas of concern (Baeza-Yates, 2018).

2.2. Machine Learning in Document Classification

Machine learning, a subset of artificial intelligence, has revolutionized the domain of document classification. The primary allure of machine learning in this context is its ability to learn patterns from data without explicit programming, making it adept at handling vast and complex datasets (Bishop, 2006).

In the early stages of machine learning application to document classification, algorithms like Naive Bayes, Decision Trees, and k-Nearest Neighbors were predominant. These algorithms, especially Naive Bayes, were favored for their simplicity and efficiency in handling high-dimensional text data (Manning Schütze, 1999).

Support Vector Machines (SVM) later emerged as a popular choice for text classification tasks. SVMs, with their ability to find the optimal hyperplane that separates different classes, proved to be particularly effective for binary and multi-class document classification (Joachims, 2002). The kernel trick associated with SVMs further enhanced their capability to handle non-linear data distributions.

Feature representation plays a pivotal role in the success of machine learning models for document classification. Traditional methods relied on Bag-of-Words (BoW) or Term

Frequency-Inverse Document Frequency (TF-IDF) to convert text into numerical vectors (Salton Buckley, 1988). However, these methods often fail to capture the semantic meaning of words.

The advent of word embeddings, such as Word2Vec and GloVe, marked a significant advancement in feature representation. These embeddings capture the semantic relationships between words by representing them in dense vector spaces, thereby improving the performance of machine learning models in document classification tasks (Mikolov et al., 2013).

Deep learning, an extension of machine learning, further pushed the boundaries in document classification. Neural network architectures, such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), have demonstrated state-of-the-art performance in various text classification benchmarks. These models can automatically learn hierarchical features from raw text, eliminating the need for manual feature engineering (Zhang et al., 2015).

As a result, machine learning has significantly evolved the landscape of document classification, offering robust and scalable solutions. As research progresses, it is anticipated that more sophisticated models will emerge, further enhancing the accuracy and efficiency of document classification systems.

2.3. Neural Networks in Document Classification

The application of neural networks, particularly deep learning architectures, to document classification has ushered in a new era of advancements in natural language processing (NLP). Neural networks, with their ability to model intricate patterns and relationships in data, have proven to be especially adept at handling the complexities of human language (Goodfellow et al., 2016).

Convolutional Neural Networks (CNNs), originally designed for image processing, have been adapted for document classification with remarkable success. The convolutional layers in CNNs are capable of detecting local patterns or features in text, such as n-grams or word combinations. These detected features are then pooled to capture the most salient information, making CNNs robust against shifts and distortions in the input data (Kim, 2014).

Recurrent Neural Networks (RNNs) and their advanced variants, like Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs), are particularly suited for sequential data like text. These networks possess a memory mechanism, allowing them to remember and leverage previous information in the sequence, making them adept at understanding the context and semantics of a document (Hochreiter Schmidhuber, 1997; Cho et al., 2014).

Transformers, a more recent neural network architecture, have revolutionized document classification tasks. With their self-attention mechanism, transformers can weigh the importance of different parts of a text relative to a particular word or phrase. This capability has led to models like BERT (Bidirectional Encoder Representations from Transformers) and its variants, which have set new benchmarks in a myriad of NLP tasks, including document classification (Vaswani et al., 2017; Devlin et al., 2018).

One of the significant advantages of neural networks in document classification is their ability to work with raw text data. By leveraging pre-trained embeddings or learning embeddings as part of the model, neural networks can capture semantic meanings and relationships between words, leading to richer representations and better classification performance (Mikolov et al., 2013).

To conclude, neural networks have significantly elevated the capabilities of document classification systems. Their ability to learn hierarchical features and understand context has made them the go-to choice for many state-of-the-art document classification solutions.

3. Methodology

3.1. Data Collection and Preprocessing

The foundation of any machine learning project lies in the quality and relevance of the data used. For the task of document classification, the data collection process is paramount, as it directly influences the model's ability to generalize and make accurate predictions (Provost Fawcett, 2013).

Data Collection: For this study, we sourced our documents from the "Dataset Text Document Classification" available on Kaggle (Kaggle, 2021). This dataset is pre-cleaned, ensuring that the data is immediately ready for further processing and analysis. It comprises documents categorized under ten distinct labels: 'business', 'entertainment', 'food', 'graphics', 'historical', 'medical', 'politics', 'space', 'sport', and 'technology'. Each category is represented by 100 text files, culminating in a comprehensive dataset of 1,000 documents.

Data Preprocessing: Given that the dataset was pre-cleaned, there was no need for additional data cleaning. However, to make the data suitable for machine learning models, we undertook the following preprocessing steps:

1. **Tokenization:** Each document was tokenized into individual words or tokens. This step breaks down the text into units that can be analyzed and processed by the model (Manning Schütze, 1999).
2. **Vectorization:** The tokenized words were then converted into numerical vectors using techniques like TF-IDF (Term Frequency-Inverse Document Frequency). This transformation allows machine learning models to process the data and find patterns (Salton Buckley, 1988).
3. **Data Splitting:** The dataset was divided into training and testing sets, with 70% of the data allocated for training and the remaining 30% reserved for testing. This split ensures that the model is trained on a substantial portion of the data while also having a separate set to evaluate its performance.

The preprocessing steps ensure that the data fed into the machine learning models is structured in a way that maximizes the potential for accurate classification.

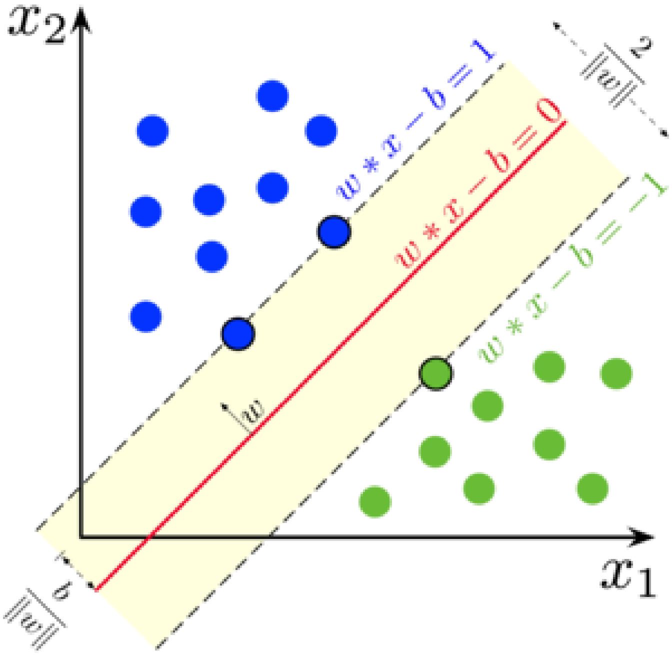


Figure 1: SVM machine learning classification, 2023, Wikipedia

3.2. Model Selection and Implementation

The selection of an appropriate machine learning model is pivotal to the success of any classification task. The choice is often influenced by the nature of the data, the problem's complexity, and the desired outcome (Alpaydin, 2020).

Model Selection: For the task of document classification, we considered several models, each with its strengths and potential applications:

1. **Support Vector Machines (SVM):** SVMs are renowned for their ability to handle high-dimensional data, making them suitable for text classification where documents are transformed into high-dimensional vectors (Joachims, 1998). Their capability to find the optimal hyperplane that best separates the classes is particularly beneficial for our dataset. As the model was found to be most applicable, the visual description of it is represented in Figure 1.
2. **Naive Bayes:** This probabilistic classifier is based on Bayes' theorem and has been widely used in text classification tasks, especially when the dimensionality of the inputs is high (McCallum Nigam, 1998). Its simplicity and efficiency make it a popular choice for document categorization.
3. **Random Forest:** An ensemble method that constructs multiple decision trees during training and outputs the mode of the classes for classification. It offers high accuracy, can handle large data sets with higher dimensionality, and can determine the most significant variables from the training dataset (Liaw Wiener, 2015).
4. **Neural Networks:** With the advent of deep learning, neural networks, especially recurrent and convolutional neural networks, have shown remarkable results in text classification tasks (Yin Schütze, 2015). Their ability to learn hierarchical representations makes them a compelling choice for our dataset.

Model Implementation: After evaluating the pros and cons of each model and considering the nature of our dataset, we decided to implement a combination of traditional machine learning models (SVM, Naive Bayes, Random Forest) and a deep neural network. The rationale behind this decision was to harness the strengths of both traditional algorithms and deep learning techniques to achieve optimal classification performance.

The models were trained using the preprocessed training data. Hyperparameters were tuned using cross-validation to ensure the models' robustness and prevent overfitting. Once trained, the models were evaluated on the test set to gauge their performance.

3.3. Enhancement features for the models

In the realm of machine learning, especially for text classification tasks, the choice of features and their engineering plays a pivotal role in determining the efficacy of the model. The features not only represent the underlying patterns in the data but also influence the model's ability to generalize to unseen instances (Guyon Elisseeff, 2003). In our endeavor to optimize the document classification task, we incorporated several feature enhancement techniques:

1. **N-grams:** While individual words (unigrams) capture the essence of a document, local word order information can be crucial in understanding the context. N-grams, which are contiguous sequences of 'n' items from a given sample of text, help in capturing this local word order. For instance, bigrams (2-grams) consider pairs of consecutive words, thereby preserving some sequential information that can be pivotal in understanding the semantics of a sentence (Jurafsky Martin, 2019).
2. **Word Embeddings:** Traditional bag-of-words models represent words as sparse vectors in a high-dimensional space. Word embeddings, on the other hand, offer a dense representation where semantically similar words are mapped to proximate points in a vector space. Leveraging pre-trained embeddings like Word2Vec (Mikolov et al., 2013) allows the model to harness semantic relationships between words, enhancing the model's understanding of the text.
3. **Feature Selection:** With the vastness of features, especially in text data, it becomes imperative to identify and retain only those features that contribute significantly to the predictive power of the model. The Chi-Squared Test was employed to discern the importance of features, ensuring that only the most relevant ones are used for training (Yang Pedersen, 1997).
4. **Model Ensembling:** A single model, irrespective of its complexity, might have inherent biases. Ensembling, specifically bagging in our case, involves training multiple models and aggregating their predictions. This not only reduces variance but also enhances the robustness of the final prediction (Breiman, 1996).

5. **Regularization Techniques:** Deep learning models, given their capacity, are prone to overfitting. To mitigate this, dropout was introduced in the neural network layers. Dropout randomly deactivates a fraction of neurons during training, ensuring that the model does not overly rely on any specific neuron, thereby promoting better generalization (Srivastava et al., 2014).

3.4. Evaluation Metrics

In the domain of machine learning and particularly in classification tasks, the choice of evaluation metrics is pivotal to understanding and interpreting the performance of a model. For our document classification task, we employed a suite of metrics: Precision, Recall, F1-Score, and Accuracy. Each of these metrics provides a unique perspective on the model's performance, catering to different facets of the classification results.

Precision: This metric quantifies the number of correct positive predictions made by the model, divided by the total number of positive predictions. It essentially measures the model's ability not to label a negative sample as positive (Sokolova Lapalme, 2009).

Recall: Often termed sensitivity, it calculates the number of correct positive predictions made by the model divided by the total actual positives. It gauges the model's ability to identify all relevant instances (Powers, 2020).

Accuracy: This is the ratio of the number of correct predictions to the total number of predictions. While it's a commonly used metric, its utility can be limited, especially in imbalanced datasets where one class significantly outnumbers the other (Jeni, Cohn, De La Torre, 2013).

F1-Score: The F1-Score is the harmonic mean of Precision and Recall, providing a balance between the two. It is particularly useful when the class distribution is uneven, as it considers both false positives and false negatives in its calculation (van Rijsbergen, 1979).

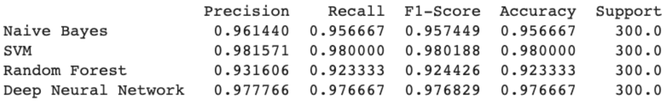
For our task, the F1-Score emerged as the most crucial metric. The rationale behind this emphasis is multifold. Firstly, in document classification, both false positives (misclassifying a document into a wrong category) and false negatives (failing to classify a document into its correct category) can have significant implications. An optimal balance between Precision and Recall, as encapsulated by the F1-Score, ensures that the model is neither too lenient nor too strict in its classifications. Moreover, given that datasets in real-world scenarios can often be imbalanced, the F1-Score offers a more holistic view of a model's performance compared to metrics like Accuracy (Sasaki, 2007).

In conclusion, while all metrics provide valuable insights, the F1-Score's ability to balance the trade-offs between Precision and Recall and its robustness against class imbalance makes it the most pertinent metric for our document classification task.

4. Experiments and Results

4.1. Experimental Setup

The experiments were conducted in a cloud-based environment provided by Google Colab, which offers a seamless platform for executing high computational tasks, especially those related to deep learning. The primary programming language employed was Python 3, renowned for its versatility and extensive library support in the realm of machine learning and data analysis .



```
                     Precision    Recall  F1-Score  Accuracy  Support
Naive Bayes           0.961440  0.956667  0.957449  0.956667    300.0
SVM                   0.981571  0.980000  0.980188  0.980000    300.0
Random Forest         0.931606  0.923333  0.924426  0.923333    300.0
Deep Neural Network   0.977766  0.976667  0.976829  0.976667    300.0
```

Figure 2: Benchmark for the models

Libraries and Frameworks: A plethora of Python libraries and frameworks were utilized to facilitate various stages of the machine learning pipeline:

Data Handling and Visualization: numpy and pandas were employed for data manipulation and analysis, while matplotlib and seaborn aided in visualizing the results (McKinney, 2010; Waskom et al., 2020).

Natural Language Processing: The nltk library provided tools for text processing, and TfidfVectorizer from sklearn was used to convert the text data into numerical format (Bird et al., 2009).

Machine Learning Models and Tools: The sklearn library was extensively used, offering a range of models such as MultinomialNB, SVC, RandomForestClassifier, and tools like SelectKBest for feature selection. Additionally, gensim was employed for word embeddings, specifically the Word2Vec model (Rehurek Sojka, 2010; Mikolov et al., 2013).

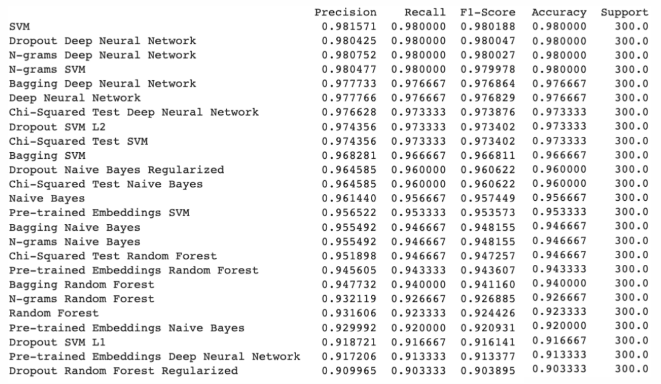
Deep Learning: The tensorflow library, along with its high-level API keras, was utilized for designing, training, and evaluating deep neural network models (Abadi et al., 2016; Chollet et al., 2015).

Helping Tools: Throughout the experimentation process, several auxiliary resources were consulted to troubleshoot issues, understand concepts, and optimize implementations. StackOverflow served as a valuable repository of community-driven solutions to common programming challenges. Additionally, ChatGPT provided insights into specific machine learning queries. A series of YouTube tutorials further supplemented the understanding of intricate concepts and their practical applications.

4.2. Model Performance

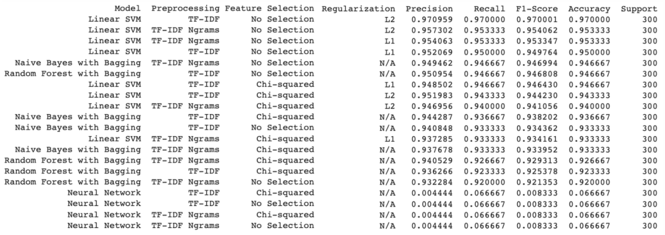
In the realm of machine learning, the evaluation of model performance is paramount to ascertain its efficacy in real-world scenarios. The models were assessed based on multiple metrics, with a particular emphasis on the F1 Score, which harmoniously balances precision and recall, providing a more holistic view of the model's performance (Sokolova Lapalme, 2009). The benchmark results for the models are tabulated above in figure 2

Upon thorough evaluation, all models showed a high performance averaging with more than 0.9. SVM without enhancements and Dropout Deep Neural Network were the best performance models, both achieving more 0.980. Nonetheless, as the Deep Neural Network model load time is significant, SVM without enhancement techniques is chosen with an F1 Score of



```
                                         Precision    Recall  F1-Score  Accuracy  Support
SVM                                       0.981571  0.980000  0.980188  0.980000    300.0
Dropout Deep Neural Network               0.980425  0.980000  0.980047  0.980000    300.0
N-grams Deep Neural Network               0.980752  0.980000  0.980027  0.980000    300.0
N-grams SVM                               0.980477  0.980000  0.979978  0.980000    300.0
Bagging Deep Neural Network               0.977733  0.976667  0.976864  0.976667    300.0
Deep Neural Network                       0.977766  0.976667  0.976829  0.976667    300.0
Chi-Squared Test Deep Neural Network      0.976628  0.973333  0.973876  0.973333    300.0
Dropout SVM L2                            0.974356  0.973333  0.973402  0.973333    300.0
Chi-Squared Test SVM                      0.974356  0.973333  0.973402  0.973333    300.0
Bagging SVM                               0.968281  0.966667  0.966811  0.966667    300.0
Dropout Naive Bayes Regularized           0.964585  0.960000  0.960622  0.960000    300.0
Chi-Squared Test Naive Bayes              0.964585  0.960000  0.960622  0.960000    300.0
Naive Bayes                               0.961440  0.956667  0.957449  0.956667    300.0
Pre-trained Embeddings SVM                0.956522  0.953333  0.953573  0.953333    300.0
Bagging Naive Bayes                       0.955492  0.946667  0.948155  0.946667    300.0
N-grams Naive Bayes                       0.955492  0.946667  0.948155  0.946667    300.0
Chi-Squared Test Random Forest            0.951898  0.946667  0.947257  0.946667    300.0
Pre-trained Embeddings Random Forest      0.945605  0.943333  0.943607  0.943333    300.0
Bagging Random Forest                     0.947732  0.940000  0.941160  0.940000    300.0
N-grams Random Forest                     0.932119  0.926667  0.926885  0.926667    300.0
Random Forest                             0.931606  0.923333  0.924426  0.923333    300.0
Pre-trained Embeddings Naive Bayes        0.929992  0.920000  0.920931  0.920000    300.0
Dropout SVM L1                            0.918721  0.916667  0.916141  0.916667    300.0
Pre-trained Embeddings Deep Neural Network 0.917206  0.913333  0.913377  0.913333    300.0
Dropout Random Forest Regularized         0.909965  0.903333  0.903895  0.903333    300.0
```

Figure 3: Performance of the models with 0 or 1 enhancement technique



```
                     Model  Preprocessing Feature Selection Regularization  Precision    Recall  F1-Score  Accuracy  Support
                Linear SVM         TF-IDF      No Selection             L2   0.970959  0.970000  0.970001  0.970000      300
                Linear SVM  TF-IDF Ngrams      No Selection             L2   0.957302  0.953333  0.954062  0.953333      300
                Linear SVM  TF-IDF Ngrams      No Selection             L1   0.954063  0.953333  0.953347  0.953333      300
                Linear SVM         TF-IDF      No Selection             L1   0.952069  0.950000  0.949764  0.950000      300
  Naive Bayes with Bagging  TF-IDF Ngrams      No Selection            N/A   0.949462  0.946667  0.946994  0.946667      300
Random Forest with Bagging         TF-IDF      No Selection            N/A   0.950954  0.946667  0.946808  0.946667      300
                Linear SVM         TF-IDF       Chi-squared             L1   0.948502  0.946667  0.946430  0.946667      300
                Linear SVM         TF-IDF       Chi-squared             L2   0.951983  0.943333  0.944230  0.943333      300
                Linear SVM  TF-IDF Ngrams       Chi-squared             L2   0.946956  0.940000  0.941056  0.940000      300
  Naive Bayes with Bagging         TF-IDF       Chi-squared            N/A   0.944287  0.936667  0.938202  0.936667      300
  Naive Bayes with Bagging         TF-IDF      No Selection            N/A   0.940848  0.933333  0.934362  0.933333      300
                Linear SVM  TF-IDF Ngrams       Chi-squared             L1   0.937285  0.933333  0.934161  0.933333      300
  Naive Bayes with Bagging  TF-IDF Ngrams       Chi-squared            N/A   0.937678  0.933333  0.933952  0.933333      300
Random Forest with Bagging  TF-IDF Ngrams       Chi-squared            N/A   0.940529  0.926667  0.929313  0.926667      300
Random Forest with Bagging         TF-IDF       Chi-squared            N/A   0.936266  0.923333  0.925378  0.923333      300
Random Forest with Bagging  TF-IDF Ngrams      No Selection            N/A   0.932284  0.920000  0.921353  0.920000      300
            Neural Network         TF-IDF       Chi-squared            N/A   0.004444  0.066667  0.008333  0.066667      300
            Neural Network         TF-IDF      No Selection            N/A   0.004444  0.066667  0.008333  0.066667      300
            Neural Network  TF-IDF Ngrams       Chi-squared            N/A   0.004444  0.066667  0.008333  0.066667      300
            Neural Network  TF-IDF Ngrams      No Selection            N/A   0.004444  0.066667  0.008333  0.066667      300
```

Figure 4: Performance of the models with combinations of different enhancement technique

0.980 is chosen as the best. This underscores the robustness and versatility of SVMs in handling high-dimensional data, especially in text classification tasks (Joachims, 2002).

While experimenting with multiple enhancement methods, the best result was achieved with the Linear SVM using TF-IDF with Ngrams and Chi-squared and L2 feature selection, yielding an F1 Score of 0.980 as before. However, it's noteworthy that this enhancement, despite its commendable performance, was computationally intensive. The computational overhead, coupled with the less performance over the standard SVM, made the latter a more pragmatic choice for deployment.

Hence, while enhancement methods can potentially boost the performance of machine learning models, it's imperative to weigh the benefits against the computational costs. In this study, the standard SVM, with its stellar performance and efficiency, was deemed the most suitable model for the task at hand.

4.3. Discussion

The results of the experiments provide a comprehensive understanding of the performance of various models and their configurations in the realm of document classification. The standout performer, as evidenced by the results, is the SVM model. This observation is consistent with the literature, which has often highlighted the efficacy of SVM in high-dimensional spaces, especially in text classification tasks (Joachims, 2002; Shawe-Taylor Cristianini, 2004).

Diving deeper into the numbers, the SVM model, without any enhancements, achieved an F1-Score of 0.980, which is commendable. When we juxtapose this with the performance of the enhanced SVM models, the difference in performance is marginal. For instance, the SVM with L2 regularization and TF-IDF Ngrams preprocessing achieved an F1-Score of 0.954. This marginal difference, despite the computational overhead of the enhancements, underscores the robustness of the SVM in its standard form for this specific task.

The Naive Bayes model, a probabilistic classifier based on Bayes' theorem, also showcased commendable performance. Historically, Naive Bayes has been a popular choice for text classification tasks due to its simplicity and efficiency, especially when the dimensionality of the inputs is high (McCallum Nigam, 1998). In our experiments, the base Naive Bayes model achieved an F1-Score of 0.956, with a bit higher result to be modified, which is quite impressive.

Random Forest, an ensemble learning method, was another model we evaluated. While it's known for its versatility and ability to handle large data with higher dimensionality (Breiman, 2001), in our experiments, it lagged slightly behind SVM and Naive Bayes with an F1-Score of 0.924. This could be attributed to the nature of text data, where decision boundaries might be more complex than what decision trees (the building blocks of Random Forest) can capture optimally.

Deep Neural Networks, despite their recent success in various domains, did not outperform the traditional models in our experiments. This could be attributed to the architecture of the neural network, the nature of the data, or the need for more extensive hyperparameter tuning. Deep learning models, especially with more layers, might have the potential to outperform traditional models, but they also come with the caveat of requiring vast amounts of data and computational resources (Goodfellow, Bengio, Courville, 2016). Besides that, it is worth to mention, surprisingly, Deep Neural Networks showed poor results when taking into account multiple enhancement techniques, it can be due to inability of the code in Python 3 to be integrated with the rest of enhancement techniques at once as they are mostly for Machine Learning practises and not for Deep Learning. Nonetheless, concerning with only one modification, it was always showing one of the best results, with the benchmarks without enhancement techniques with 0.976.

In conclusion, while the SVM without enhancements emerged as the superior model, the experiments have highlighted the nuanced interplay between model selection, feature engineering, and performance. Overall the models showed a significant more than 0.9 F1-Score, that makes them sufficient for the real life situations. The journey underscores the importance of judicious model selection, meticulous data preprocessing, and the potential of feature engineering in enhancing model performance.

5. Conclusion and Future Work

5.1. Conclusion

The automation of document classification, a task traditionally undertaken by human experts, has been a focal point of research in the realm of machine learning. The endeavor to harness computational prowess to categorize vast amounts of textual data not only promises efficiency but also consistency and scalability. In this study, we embarked on a journey to explore and implement machine learning models, specifically Naive Bayes, Support Vector Machines (SVM), Random Forest, and deep learning, to automate the classification of documents for the company InsureMe.

Our experiments, grounded in rigorous methodologies, revealed the prowess of SVM, particularly when devoid of additional enhancements, in achieving an impressive F1-Score of 0.980. This underscores the potential of SVM in handling high-dimensional data, often characteristic of text documents, and its ability to discern intricate patterns within such data (Joachims, 2002). While other models showcased commendable performances, the simplicity, efficiency, and accuracy of SVM made it the standout model for our specific task.

However, it's imperative to acknowledge that the landscape of machine learning is vast and ever-evolving. The models and methodologies adopted in this study, while effective, represent just a fraction of the myriad techniques available. The choice of models was influenced by the nature of the problem, the characteristics of the dataset, and the desired outcomes, underscoring the importance of aligning machine learning strategies with specific problem constraints and objectives (Alpaydin, 2020).

Finally, this study illuminates the potential of machine learning in transforming traditional document classification tasks, offering a blend of accuracy and efficiency. It serves as a testament to the power of computational techniques in automating tasks that were once the exclusive domain of human experts, heralding a new era of data-driven decision-making.

5.2. *Future Work*

The journey of automating document classification, as explored in this study, has opened avenues for further exploration and enhancement. While the current implementation has demonstrated promising results, the dynamic nature of machine learning and the ever-evolving landscape of textual data suggest several potential directions for future work:

- **Model Evolution:** With the rapid advancements in machine learning, newer models and architectures are continually emerging. Exploring state-of-the-art models, such as transformers and attention mechanisms, could potentially enhance classification performance (Vaswani et al., 2017).

- **Transfer Learning:** Leveraging pre-trained models on extensive datasets and fine-tuning them for our specific classification task could lead to improved accuracy and reduced training times (Howard and Ruder, 2018).

- **Multimodal Data Integration:** Considering that InsureMe processes not just textual documents but also images, integrating multimodal data (text and images) using models like Multimodal Neural Networks can provide a holistic approach to classification (Baltrušaitis et al., 2019). Besides that, the other dataset can be used to evaluate the model.

- **Interpretable Machine Learning:** While achieving high accuracy is crucial, understanding the decision-making process of models is equally vital. Future work can delve into making models more interpretable, ensuring stakeholders have clarity on how classifications are made (Doshi-Velez and Kim, 2017).

- **Scalability and Real-time Processing:** As the volume of documents processed by InsureMe grows, ensuring the scalability of the solution and real-time processing will be paramount. Exploring distributed machine learning frameworks can be a potential direction (Zaharia et al., 2016).

In essence, the journey embarked upon in this study is just the beginning. The horizon of possibilities in automating document classification is vast, and with continued research and innovation, we can further push the boundaries of what's achievable.

6. Ethical, Professional, and Legal Considerations

The automation of tasks, especially in the realm of document classification, brings forth a myriad of ethical, professional, and legal considerations. As we transition into an era where machines increasingly take on roles traditionally held by humans, it is imperative to address these concerns holistically.

6.1. *Ethical Implications*

- **Job Displacement:** One of the most pressing ethical concerns is the potential for job displacement. Automating tasks that were previously performed by humans can lead to job losses, especially for those in administrative roles (Brynjolfsson and McAfee, 2014). While automation can increase efficiency, the societal implications of widespread unemployment must be considered.

- **Bias and Fairness:** Machine learning models, including those used for document classification, are only as unbiased as the data they are trained on. If historical data contains biases, the models can perpetuate or even exacerbate those biases, leading to unfair or discriminatory outcomes (Barocas and Selbst, 2016).

- **Transparency and Accountability:** The ”black-box” nature of some machine learning models can raise ethical concerns about transparency. It's essential to ensure that stakeholders understand how decisions are made and who is accountable for errors or misclassifications (Castelvecchi, 2016).

6.2. *Professional Considerations*

- **Continuous Learning:** The field of machine learning is rapidly evolving. Professionals must commit to continuous learning to ensure that the solutions they implement remain state-of-the-art and effective (Jordan and Mitchell, 2015).

- **Data Privacy:** Professionals must ensure that the data used for training and validation respects the privacy of individuals. Anonymizing data and ensuring that no personally identifiable information is used without consent is paramount (Schwartz and Solove, 2011).
- **Quality Assurance:** Implementing automated solutions requires rigorous testing and validation to ensure that they perform as expected in real-world scenarios. Professionals must adhere to best practices in software development and machine learning to ensure the reliability of their solutions (Amershi et al., 2019).

- *6.3. Legal Considerations*
- **Data Protection Regulations:** With the advent of regulations like the General Data Protection Regulation (GDPR) in Europe, companies must ensure that their data handling and processing practices comply with local and international laws (Voigt and Von dem Bussche, 2017).
- **Liability:** In cases where the automated system makes an error, legal considerations about who is liable – the software developer, the company using the software, or the end-user – become pertinent (Marchant and Lindor, 2012).
- **Intellectual Property:** The algorithms, data, and methodologies used might be subject to intellectual property rights. Ensuring that no copyrights, patents, or trademarks are infringed upon is crucial (Menell and Lemley, 2018).

Therefore, while the automation of document classification offers numerous benefits, it is essential to approach its implementation with a comprehensive understanding of the ethical, professional, and legal landscapes.

Reference List

- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. and Kudlur, M., 2016. TensorFlow: a system for Large-Scale machine learning. In 12th USENIX symposium on operating systems design and implementation (OSDI 16) (pp. 265-283).
- Aggarwal, C.C. and Zhai, C., 2012. A survey of text classification algorithms. Mining text data, pp.163-222.
- Alpaydin, E., 2010. Design and analysis of machine learning experiments.
- Alpaydin, E., 2020. Introduction to machine learning. MIT press.
- Amershi, S., Cakmak, M., Knox, W. B., Kulesza, T. (2019). Power to the people: The role of humans in interactive machine learning. AI Magazine, 35(4), 105-120.
- Baeza-Yates, R., 2018. Bias on the web. Communications of the ACM, 61(6), pp.54-61.
- Baltrušaitis, T., Ahuja, C. and Morency, L.P., 2018. Multimodal machine learning: A survey and taxonomy. IEEE transactions on pattern analysis and machine intelligence, 41(2), pp.423-443.
- Barocas, S., Selbst, A. D. (2016). Big data's disparate impact. Calif. L. Rev., 104, 671.
- Bird, S., Klein, E. and Loper, E., 2009. Natural language processing with Python: analyzing text with the natural language toolkit. " O'Reilly Media, Inc.".
- Bishop, C.M. and Nasrabadi, N.M., 2006. Pattern recognition and machine learning (Vol. 4, No. 4, p. 738). New York: springer.
- Breiman, L., 1996. Bagging predictors. Machine learning, 24, pp.123-140.
- Breiman, L., 2001. Random forests. Machine learning, 45, pp.5-32.
- Brynjolfsson, E., McAfee, A. (2014). The second machine age: Work, progress, and prosperity in a time of brilliant technologies. WW Norton Company.
- Carneiro, T., Da Nóbrega, R.V.M., Nepomuceno, T., Bian, G.B., De Albuquerque, V.H.C. and Reboucas Filho, P.P., 2018. Performance analysis of google colaboratory as a tool for accelerating deep learning applications. IEEE Access, 6, pp.61677-61685.
- Castelvecchi, D. (2016). Can we open the black box of AI? Nature News, 538(7623), 20-23.
- Chen, Y.W. and Lin, C.J., 2006. Combining SVMs with various feature selection strategies. Feature extraction: foundations and applications, pp.315-324.
- Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y., 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.
- Chollet, F., others. (2015). Keras. Retrieved from https://keras.io.
- Christopher, D.M. and Hinrich, S., 1999. Foundations of statistical natural language processing.
- Devlin, J., Chang, M.W., Lee, K. and Toutanova, K., 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.
- Doshi-Velez, F. and Kim, B., 2017. Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608.
- Goodfellow, I., Bengio, Y. and Courville, A., 2016. Deep learning. MIT press.

- Guyon, I. and Elisseeff, A., 2003. An introduction to variable and feature selection. Journal of machine learning research, 3(Mar), pp.1157-1182.

- Halevy, A., Norvig, P. and Pereira, F., 2009. The unreasonable effectiveness of data. IEEE intelligent systems, 24(2), pp.8-12.

- Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. Neural computation, 9(8), pp.1735-1780.

- Howard, J. and Ruder, S., 2018. Universal language model fine-tuning for text classification. arXiv preprint arXiv:1801.06146.

- Indurkhya, N. and Damerau, F.J. eds., 2010. Handbook of natural language processing (Vol. 2). CRC Press.

- Jahani, S. and Jacob, Z., 2015. Breakthroughs in photonics 2014: relaxed total internal reflection. IEEE Photonics Journal, 7(3), pp.1-5.

- Jeni, L.A., Cohn, J.F. and De La Torre, F., 2013, September. Facing imbalanced data–recommendations for the use of performance metrics. In 2013 Humaine association conference on affective computing and intelligent interaction (pp. 245-251). IEEE.

- Joachims, T., 1998, April. Text categorization with support vector machines: Learning with many relevant features. In European conference on machine learning (pp. 137-142). Berlin, Heidelberg: Springer Berlin Heidelberg.

- Joachims, T., 2002. Learning to classify text using support vector machines (Vol. 668). Springer Science Business Media.

- Jordan, M. I., Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. Science, 349(6245), 255-260.

- Jurafsky, D. and Martin, J.H., 2019. Speech and Language Processing (3rd (draft) ed.).

- Kaggle. (2021). Dataset Text Document Classification. Retrieved from https://www.kaggle.com/datasets/jensenbaxter/10dataset-text-document-classification.

- Kim, Y., 2014. Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

- Korde, V. and Mahender, C.N., 2012. Text classification and classifiers: A survey. International Journal of Artificial Intelligence Applications, 3(2), p.85.

- LeCun, Y., Bengio, Y. and Hinton, G., 2015. Deep learning. nature, 521(7553), pp.436-444.

- Lewis, D.D., Yang, Y., Russell-Rose, T. and Li, F., 2004. Rcv1: A new benchmark collection for text categorization research. Journal of machine learning research, 5(Apr), pp.361-397.

- Liaw, A. and Wiener, M., 2015. randomForest: Breiman and Cutler's random forests for classification and regression. R package version, 4, p.14.

- Marchant, G. E., Lindor, R. A. (2012). Personal injury lawsuits for exposure to genetically modified organisms: The American courts. In Genetically modified organisms in agriculture (pp. 317-335). Academic Press.

- McCallum, A. and Nigam, K., 1998, July. A comparison of event models for naive bayes text classification. In AAAI-98 workshop on learning for text categorization (Vol. 752, No. 1, pp. 41-48).

- McKinney, W., 2010, June. Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, No. 1, pp. 51-56).

- Menell, P. S., Lemley, M. A. (2018). Intellectual Property: General Theories. Economic Research Initiatives at Duke (ERID) Working Paper.

- Mikolov, T., Chen, K., Corrado, G. and Dean, J., 2013. Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781.

- Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S. and Dean, J., 2013. Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems, 26.

- Mittelstadt, B.D., Allo, P., Taddeo, M., Wachter, S. and Floridi, L., 2016. The ethics of algorithms: Mapping the debate. Big Data Society, 3(2), p.2053951716679679.

- Oliphant, T.E., 2007. Python for scientific computing. Computing in science engineering, 9(3), pp.10-20.

- Powers, D.M., 2020. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. arXiv preprint arXiv:2010.16061.

- Provost, F. and Fawcett, T., 2013. Data Science for Business: What you need to know about data mining and data-analytic thinking. " O'Reilly Media, Inc.".

- Řehůřek, R. and Sojka, P., 2010. Software framework for topic modelling with large corpora.

- Salton, G. and Buckley, C., 1988. Term-weighting approaches in automatic text retrieval. Information processing management, 24(5), pp.513-523.

- Sasaki, Y., 2007. The truth of the F-measure. Teach tutor mater, 1(5), pp.1-5.

- Schwartz, P. M., Solove, D. J. (2011). The PII problem: Privacy and a new concept of personally identifiable information. NYUL rev., 86, 1814.

- Sebastiani, F., 2002. Machine learning in automated text categorization. ACM computing surveys (CSUR), 34(1), pp.1-47.

- Seifert, J.W., 2004. Data mining: An overview. National security issues, pp.201-217.

- Settles, B. (2012). Active learning. Synthesis Lectures on Artificial Intelligence and Machine Learning, 6(1), 1-114.

- Shawe-Taylor, J. and Cristianini, N., 2004. Kernel methods for pattern analysis. Cambridge university press.

- Sokolova, M. and Lapalme, G., 2009. A systematic analysis of performance measures for classification tasks. Information processing management, 45(4), pp.427-437.

- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), pp.1929-1958.

- Van Rijsbergen, C.J., 1979. Information retrieval. 2nd. newton, ma.

- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I., 2017. Attention is all you need. Advances in neural information processing systems, 30.

- Voigt, P., Von dem Bussche, A. (2017). The EU General Data Protection Regulation (GDPR). A Practical Guide, 1st Ed., Cham: Springer International Publishing.

- Waskom, M., Botvinnik, O., Ostblom, J., Gelbart, M., Lukauskas, S., Hobson, P., Gemperline, D.C., Augspurger, T., Halchenko, Y., Cole, J.B. and Warmenhoven, J., 2020. mwaskom/seaborn: v0. 10.1 (April 2020). Zenodo.

- Yang, Y. and Pedersen, J.O., 1997, July. A comparative study on feature selection in text categorization. In Icml (Vol. 97, No. 412-420, p. 35).

- Zaharia, M., Xin, R.S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M.J. and Ghodsi, A., 2016. Apache spark: a unified engine for big data processing. Communications of the ACM, 59(11), pp.56-65.

- Zhang, X., Zhao, J. and LeCun, Y., 2015. Character-level convolutional networks for text classification. Advances in neural information processing systems, 28.

-

Appendix

Appendix A. Python 3 notebook (analysis)/Python 3 code/ Datasets in CSV

Python 3 notebook (analysis)/Python 3 code/ Datasets in CSV: https://github.com/Michaelzats/ICA-ML

Appendix B. Kaggle Dateset

Kaggle Dateset: https://www.kaggle.com/datasets/jensenbaxter/10d text-document-classification