EdinBurg Peer Reviewed Journals and Books Publishers
Journal of Information and Technology
Vol. 5||Issue 8||pp 61-71||September||2025
Email: info@edinburgjournals.org||ISSN: 3080-9576

Enhancing Customer Service in Financial Institutions Through Automated Complaint Classification: A Machine Learning Approach

Doreen Mutesi[1*], Dr. Jonathan Ngugi[2], Dr. Sumbiri Djuma[3]
[1,2,3]Faculty of Computing and Information Sciences, University of Lay Adventists of Kigali, Rwanda
Corresponding Emails: mutesidoreen09@gmail.com; phialn1@gmail.com; sumbirdj@gmail.com

Accepted: 14 July 2025 || Published: 20 September 2025

Abstract

This paper presents a method to automatically group financial consumer complaints by the type of product they are about, using Natural Language Processing (NLP). The dataset was cleaned and prepared by removing unnecessary words, breaking it into tokens, and converting it into numbers using TF-IDF. Then, Machine learning models are trained to predict which product each complaint is related to. Among the models tested, Stochastic Gradient Descent gave good results. The findings show that NLP can help financial institutions handle complaints faster and more accurately by sorting them automatically into product categories. This approach can be helpful for banks and regulators in improving how they respond to customer issues.

Keywords: *Natural Language Processing (NLP), Term Frequency-Inverse Document Frequency (TF-IDF)*

How to Cite: Mutesi, D., Ngugi, J., & Sumbiri, D. (2025). Enhancing Customer Service in Financial Institutions Through Automated Complaint Classification: A Machine Learning Approach. *Journal of Information and Technology, 5*(8), 61-71.

1. Introduction

Today's financial world is full of many different products like credit cards, loans, mortgages, and student financing. For example, in the United States, customers share their problems by sending written complaints to the Consumer Financial Protection Bureau (CFPB). These complaints are written in free-text, which means they don't follow a fixed format. In Rwanda, customers share their queries via email or social media account. These messages contain useful information. Reading and sorting each complaint by hand takes a lot of time and can lead to mistakes.

This paper looks at how we can use Natural Language Processing (NLP) to help automatically sort these complaints based on the type of financial product involved. The goal is to build and test a machine learning system that reads the text of the complaint and correctly guesses which product it is about. To do this, start with Text cleaning and tokenization by removing stopwords, and turn the text into numbers using Term Frequency Inverse Document Frequency

(TF-IDF). After that, train and compare different models, such as Logistic Regression and Support Vector Machine (SVM), to see which one works best.

The goal is to apply and use this system in a financial institution in Rwanda. If this solution is used locally, banks and other financial institutions in Rwanda will be able to classify customer complaints related to financial products more quickly and respond to customer complaints more effectively. This will not only show that NLP can work in the financial sector, but it will also help improve how customer complaints are managed in general.

During this study, a Test of how well the models perform will be conducted and explain how this kind of system can be added to the way financial institutions currently handle complaints.

2. Related Study

Using machine learning to categorize consumer complaints is an important topic in recent research. This is because there is now a large amount of data, written complaints from customers being sent to service providers. These complaints most of the time include useful information. If those complaints get organized correctly, it can help financial institutions and government agencies respond better and improve their service delivery to their customer.

Machine learning algorithms like Logistic Regression, Support Vector Machines (SVM), Stochastic Gradient Descent, and Random Forest are commonly used for this task. These models usually work with text features extracted using a method called TF-IDF, which turns words into numbers that the model can understand.

For example, different studies in Africa have shown that traditional machine learning methods like TF-IDF, Logistic Regression, and Random Forest can work well for classifying text in different languages and topics. For instance, Adelani et al. (2023) created a news classification dataset in 16 African languages, including Kinyarwanda, and found that simple models like Logistic Regression performed very well.

In Malawi, Taylor and Amoss (2025) applied Random Forest and Logistic Regression to detect fraud in local SMS messages written in Chichewa, reaching very high accuracy. These studies show that classical machine learning models can be very effective, especially in African contexts where resources and data are limited.

Recent studies also support the strength of traditional models in this domain. For example, Kowsari et al. (2019) did a review of text classification algorithms and concluded that while deep learning models give improved performance in complex scenarios, classic algorithms like Support Vector Machine and Logistic Regression remain highly effective for short to medium length texts, such as customer complaints.

3. Methodology

3.1. Study Area and Data Sources

This study focuses on the financial sector in Rwanda on handling and processing customer complaints. This study is designed for implementation within our financial sector**.** Rwanda has made good progress in digitalizing financial services by adopting digital banking, mobile money, and microfinance platforms where everyone can open an account, make a transaction, and apply for a loan online without going to a bank. However, customer complaints for the diaspora and for professionals who have limited time to go to a financial institution still have challenges of delayed responses, and the bureaucratic process of handling complaints. By

EdinBurg Peer Reviewed Journals and Books Publishers
Journal of Information and Technology
Vol. 5||Issue 8||pp 61-71||September||2025
Email: info@edinburgjournals.org||ISSN: 3080-9576

applying NLP techniques to automatically classify complaints and assign tickets to a dedicated department, Rwanda financial institutions can improve customer service, reduce response times, and enhance regulatory compliance.

The findings of this study will serve as a foundational approach that can be adapted in the operational context of banking and financial institutions in handling customer complaints.

The Data source utilized

In this study, the dataset used was sourced from a publicly available collection of consumer complaints related to various financial products and services. The data includes complaint text submissions describing issues with services such as mortgages, credit reporting, student loans, and money transfers. It provides a good foundation for analyzing and classifying complaints in the financial sector using Natural Language Processing techniques.

3.2 Data Preprocessing

The dataset content contains some noise, like punctuation, numbers, and stop words. The data cleaning was done by removing punctuation, non-alphabetic characters, and stop words. Tokenization is used to split text into individual words.

```
for chunk in reader:
    batch_count += 1
    print(f"Processing batch {batch_count}...")
    # Filter and clean data
    df1 = chunk[['Product', 'Consumer complaint narrative']].dropna()
    df1.columns = ['Product', 'Consumer_complaint']
    df1['Product'] = df1['Product'].replace(dict_y_to_replace)
```

Figure 1: Data preprocessing

The photo above shows how data processing was used in this system by removing null values from the dataset by using the function dropna, and shows how feature extraction was done by extracting product and consumer complaints.

3.3 Feature Extraction

Consumer complaints and the product s/he complained about were extracted as the main predictors for this case.

3.4 Model Selection

Five traditional classification models were trained using the TF-IDF features:

- Logistic Regression: A linear model that estimates the probability of a class based on feature weights.
- Linear SVC: A faster implementation of the Support Vector Machine classifier that finds an optimal boundary between classes in high-dimensional space.
- Random Forest: A method that uses many decision trees together to make better predictions and avoid mistakes from relying too much on one tree.
- Multinomial Naive Bayes is a machine learning algorithm that is good at classifying text. It works by looking at how often each word appears in the text to guess the right category
- Stochastic Gradient Descent Classifier is a type of linear classifier that uses the SGD optimization algorithm to train.

EdinBurg Peer Reviewed Journals and Books Publishers
Journal of Information and Technology
Vol. 5||Issue 8||pp 61-71||September||2025
Email: info@edinburgjournals.org||ISSN: 3080-9576

These models were chosen for their simplicity and strong performance in previous studies involving text classification.

```
train_models = {
    "Logistic Regression": True,
    "Linear SVC": True,
    "Random Forest": True,
    "Multinomial NB": True,
    "SGD Classifier": True
}
```

Figure 2: Model to be trained

The diagram above shows all the model algorithms used during training to check which one is going to performs well on financial complaint classification. The algorithm used for training is each algorithm that has a value set to True, which indicates that you can disable training on some algorithms and choose a specific one.

```
for model_name, model in models.items():
    # Skip models that are set to False in train_models
    if not train_models.get(model_name, True):
        print(f"Skipping {model_name} as specified in train_models")
        continue

    start_time = time.time()

    try:
        # Check if there are at least 2 unique classes in the training set
        if len(np.unique(y_train)) < 2:
            print(f"Skipping {model_name} for this batch - not enough unique classes in training set")
            continue

        # Train model
        if model_name == "SGD Classifier":
            # SGD supports partial_fit for online learning
            model.partial_fit(X_train_tfidf, y_train, classes=np.arange(len(all_classes)))
        else:
            # Other models need to be fit from scratch each time
            model.fit(X_train_tfidf, y_train)
```

Figure 3: Model training

The figure above shows all the selected models which was to be trained. for loop had to go through each model and train using the current batch of data from the dataset, which was read in chunks for resources.

3.5 Model Evaluation

The dataset was divided into training 80% and testing 20%. Model performance was evaluated using accuracy score, precision, recall, and F1-Score.

Model	Accuracy	Precision	Recall	F1-Score
Logistic Regression	0.9250	0.8656	0.9250	0.8919
Linear SVC	0.9250	0.8656	0.9250	0.8919
Random Forest	0.9250	0.8656	0.9250	0.8919
Multinomial NB	0.9250	0.8656	0.9250	0.8919
SGD Classifier	0.8917	0.8656	0.8917	0.8737

Figure 4: Model evaluation

The diagram above shows the results of the evaluation of each model's performance using accuracy score, precision, recall, and F1-Score on each batch iteration. After completing iteration through datasets, it calculates the average mean accuracy by which is used to evaluate overall model performance.

EdinBurg Peer Reviewed Journals and Books Publishers
Journal of Information and Technology
Vol. 5||Issue 8||pp 61-71||September||2025
Email: info@edinburgjournals.org||ISSN: 3080-9576

3.6 Hyperparameter tuning and validation strategy

In this project, Hyperparameter tuning was used to improve the performance of the machine learning models used for classifying financial complaints. For every algorithm like Logistic Regression, Support Vector Machine, Random Forest, and Stochastic Gradient Descent, Grid Search was used to test different combinations of hyperparameters, including regularization strength and number of decision trees by using estimators.

To ensure reliable results and avoid overfitting, a k-fold cross-validation strategy (with k=5) was implemented, where the dataset was split into five parts. The model was trained on four parts and validated on the fifth, repeating the process five times with each part serving as validation once. The average performance across all folds was used to evaluate and select the best hyperparameter settings.

Flowchart

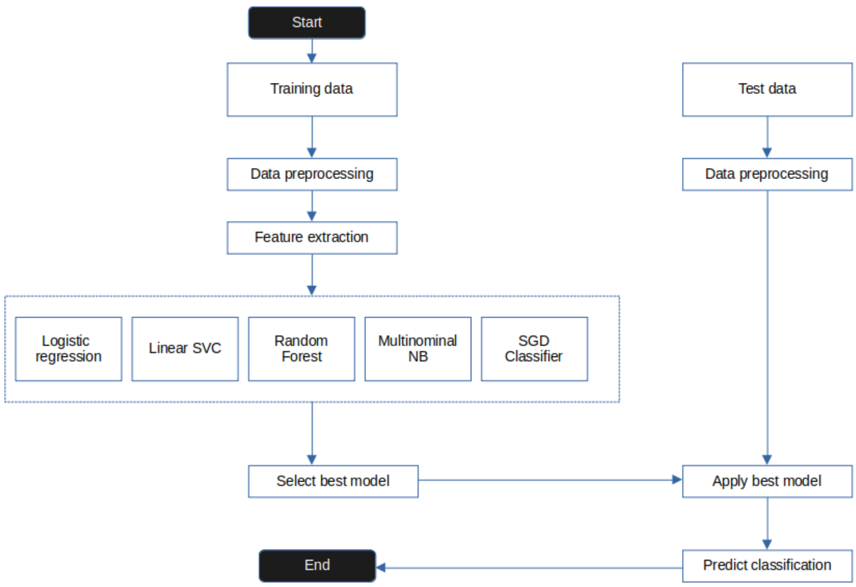


Figure 5: Model flowchart

The flowchart above shows all the processes for building a financial complaint classification system using machine learning. The process begins with getting the dataset, followed by data preprocessing, which includes removing null values, punctuation, filtering stop words, and tokenization. Next, feature extraction is performed using Term Frequency-Inverse Document Frequency (TF-IDF) vectorization. This transforms the cleaned complaint into numerical feature representations suitable for machine learning models.

The dataset is then split into training (80%) and testing (20%) subsets to evaluate model performance on unseen data. To select the right and better performing model, multiple algorithms—including Logistic Regression, Support Vector Machine (SVM), and Random Forest are trained and compared. On evaluation, different metrics used, like accuracy, precision, recall, and F1-score, are calculated. The best-performing model is selected based on mean cross-validation scores.

EdinBurg Peer Reviewed Journals and Books Publishers
Journal of Information and Technology
Vol. 5||Issue 8||pp 61-71||September||2025
Email: info@edinburgjournals.org||ISSN: 3080-9576

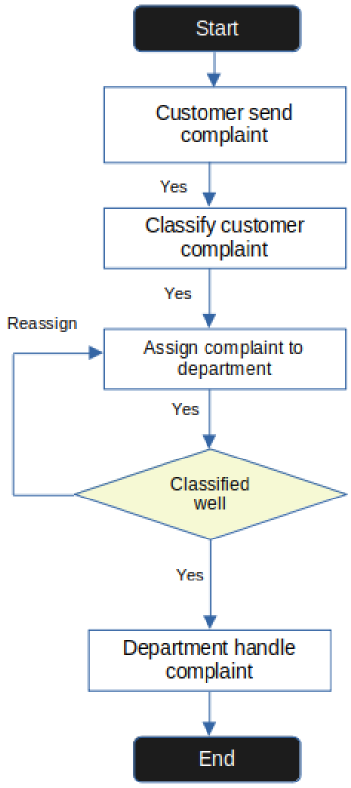


Figure 6: System flowchart

The diagram above is a flowchart that shows how the system will handle customer complaints by receiving them and using a trained model to get their classification category.

4. Tools and Technologies

Python: A general-purpose programming language, used in this project to write model code

Flask: a lightweight micro web framework written in Python, was used to design and render these user interfaces for the users to interact with the model

NumPy: a library for the Python programming language that gives the ability to use large multi-dimensional arrays and matrices with more mathematical functions

Pandas: a software library written in Python, was used in data manipulation, especially on datasets, to read them through and select the data of our interest

Scikit-Learn: a machine learning library written in Python, which helps in building models and has been used in this project by giving us all those model algorithms to test and use, like Logistic regression and Stochastic Gradient Descent

Joblib: a Python library for running computationally intensive tasks in parallel. It was used in this project to save the model during the training process when the model was not getting a good percentage of prediction or when the user canceled the training, save the model at that progress.

EdinBurg Peer Reviewed Journals and Books Publishers
Journal of Information and Technology
Vol. 5||Issue 8||pp 61-71||September||2025
Email: info@edinburgjournals.org||ISSN: 3080-9576

5. System Results Screenshots

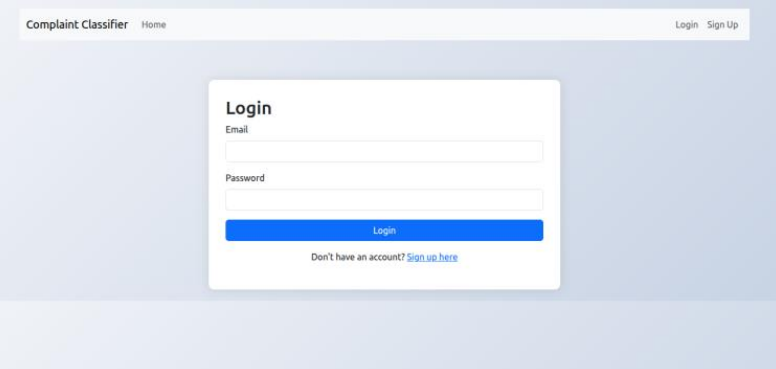
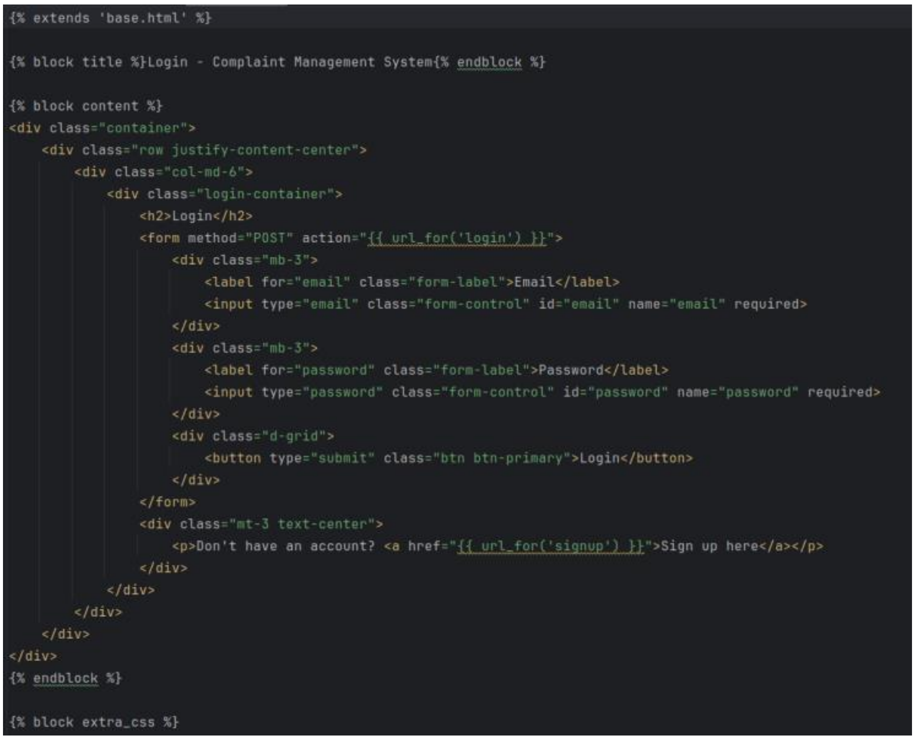


Figure 7: Login interface



```
{% extends 'base.html' %}

{% block title %}Login - Complaint Management System{% endblock %}

{% block content %}
<div class="container">
    <div class="row justify-content-center">
        <div class="col-md-6">
            <div class="login-container">
                <h2>Login</h2>
                <form method="POST" action="{{ url_for('login') }}">
                    <div class="mb-3">
                        <label for="email" class="form-label">Email</label>
                        <input type="email" class="form-control" id="email" name="email" required>
                    </div>
                    <div class="mb-3">
                        <label for="password" class="form-label">Password</label>
                        <input type="password" class="form-control" id="password" name="password" required>
                    </div>
                    <div class="d-grid">
                        <button type="submit" class="btn btn-primary">Login</button>
                    </div>
                </form>
                <div class="mt-3 text-center">
                    <p>Don't have an account? <a href="{{ url_for('signup') }}">Sign up here</a></p>
                </div>
            </div>
        </div>
    </div>
</div>
{% endblock %}

{% block extra_css %}
```

Figure 8: Login interface code

The photo above shows a user interface of how a customer logs in to their account on the complaints classifier system, and the codes that make up that user interface, which enables customers to enter credentials and get verified on the backend.

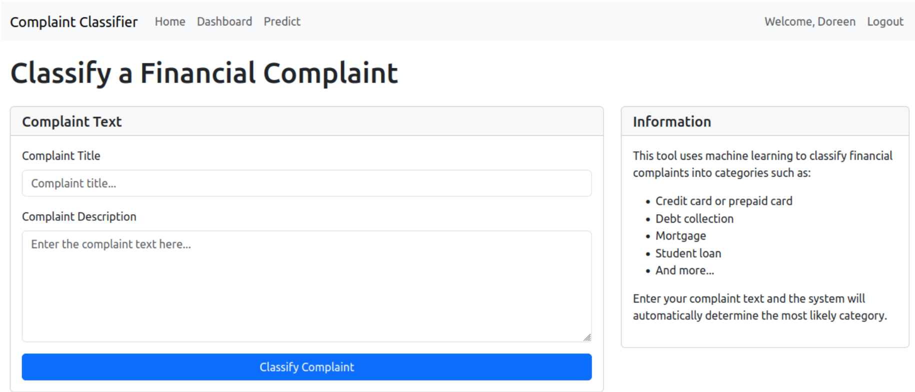
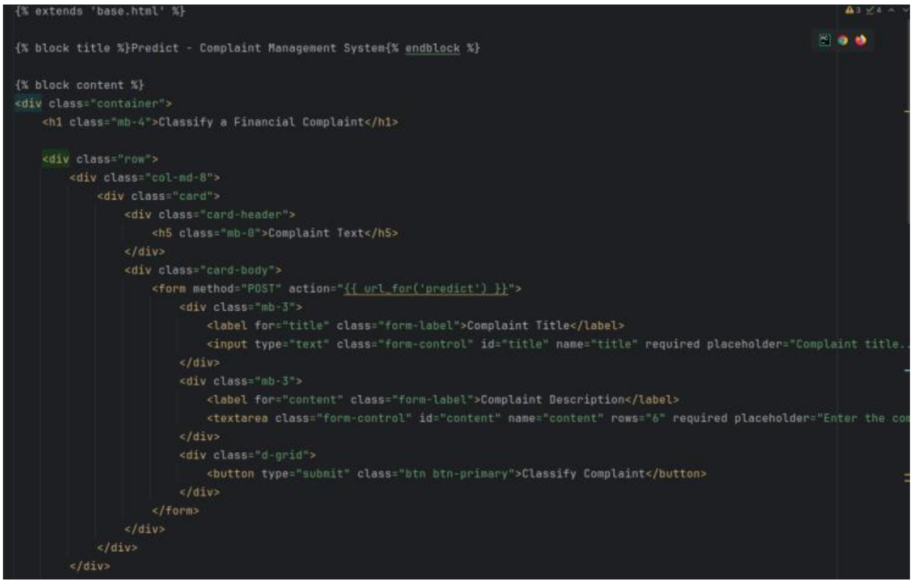


Figure 9: Complaint form



```
{% extends 'base.html' %}

{% block title %}Predict - Complaint Management System{% endblock %}

{% block content %}
<div class="container">
    <h1 class="mb-4">Classify a Financial Complaint</h1>

    <div class="row">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header">
                    <h5 class="mb-0">Complaint Text</h5>
                </div>
                <div class="card-body">
                    <form method="POST" action="{{ url_for('predict') }}">
                        <div class="mb-3">
                            <label for="title" class="form-label">Complaint Title</label>
                            <input type="text" class="form-control" id="title" name="title" required placeholder="Complaint title..
                        </div>
                        <div class="mb-3">
                            <label for="content" class="form-label">Complaint Description</label>
                            <textarea class="form-control" id="content" name="content" rows="6" required placeholder="Enter the com
                        </div>
                        <div class="d-grid">
                            <button type="submit" class="btn btn-primary">Classify Complaint</button>
                        </div>
                    </form>
                </div>
            </div>
        </div>
```

Figure 10: Complaint form code

After entering the email and password and logging in, the user will see a form to enter his/her complaint. S/he can fill in the details about the issue he is facing. Once submitted, the system will automatically categorize the complaint using a machine learning model trained to returns a product or service based on the user complaint.

EdinBurg Peer Reviewed Journals and Books Publishers
Journal of Information and Technology
Vol. 5||Issue 8||pp 61-71||September||2025
Email: info@edinburgjournals.org||ISSN: 3080-9576

```
def predict_complaint(complaint_text, model_name="SGD Classifier"):   3 usages   Roger
        # Sort by modification time (newest first)
        model_filename = max(model_files, key=os.path.getmtime)

        # Find the most recent vectorizer file
        vectorizer_files = glob.glob("models/tfidf_vectorizer.pkl")
        if not vectorizer_files:
            return "No vectorizer files found"

        # Sort by modification time (newest first)
        vectorizer_filename = max(vectorizer_files, key=os.path.getmtime)

        # Find the most recent classes file
        classes_files = glob.glob("models/unique_predictable_class.pkl")
        if not classes_files:
            return "No classes files found"

        # Sort by modification time (newest first)
        classes_filename = max(classes_files, key=os.path.getmtime)

        # Load model, vectorizer and classes
        model = joblib.load(model_filename)
        tfidf = joblib.load(vectorizer_filename)
        all_classes = joblib.load(classes_filename)

        # Transform text
        complaint_tfidf = tfidf.transform([complaint_text])

        # Predict
        predicted_label = model.predict(complaint_tfidf)[0]

        return all_classes[predicted_label]
    except Exception as e:
```

Figure 11: Complaint prediction code

When customer submit his/her complaint, the photo of codes above is the one who does the prediction of the category of the complaint by loading saved pre-trained model which uses Stochastic Gradient Descent algorithm and load vector database saved for resource optimization during prediction, Transform complaint given from customer into vector to obtain the relevance of the word and after that it predict the complaint category and return the predicted label.

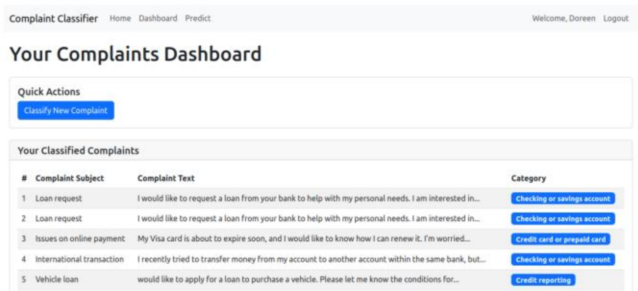


Figure 12: Complaint classification dashboard

The photo above shows the classification result of the complaint, where the predicted category has been added in the blue background

6. Conclusion

This project explored how Natural Language Processing and machine learning can be used to automatically classify financial complaints based on the type of product mentioned. Instead of having staff manually go through complaints and categorize each complaint, the system uses a machine learning model trained using the Stochastic Gradient Descent algorithm to automatically classify complaints into the right category.

EdinBurg Peer Reviewed Journals and Books Publishers
Journal of Information and Technology
Vol. 5||Issue 8||pp 61-71||September||2025
Email: info@edinburgjournals.org||ISSN: 3080-9576

The results showed that the system can predict well the type of financial product in most cases. This makes the complaint reviewing and handling process faster, helping institutions respond to customer issues in a timely way.

However, the system has limitations. The dataset may have imbalances, where some product categories appear more often than others, which may affect to make a good prediction. In the future, the system can be improved by using deep learning and neural networks, which are better at understanding advanced language patterns and get the full context.

7. Recommendations

This study recommends that banks and financial institutions in Rwanda should consider and implement automated systems to classify customer complaints. These types of systems can help to sort and understand complaints more quickly, which saves time and improves how they respond to customers. Using simple machine learning models like Stochastic Gradient Descent or Logistic Regression can already make a huge difference.

For researchers and developers working in Rwanda, it is important to start building local datasets using complaints written in Kinyarwanda and English. Having local data will help create systems that better understand how customer send their issues in our context. In the future, using more advanced methods like deep learning and neural networks could make the application of these systems smarter.

8. References

Adebayo, S. M., & Nwakanma, C. I. (2020). A comparative study of text classification algorithms for financial data. International Journal of Advanced Computer Science and Applications,11(5),370–377.

Adelani, D., Ruiter, D., Alabi, J., & Others. (2023). MasakhaNEWS: News Topic Classification in African Languages. arXiv. https://arxiv.org/abs/2304.10412

Aggarwal, C. C., & Zhai, C. (2012). A survey of text classification algorithms. In C. C. Aggarwal & C. Zhai (Eds.), Mining text data (pp. 163–222)

Grunewald, E., Seitz, H., & Fuchs, L. (2019). Automatic classification of financial consumer complaints: A text mining approach. *International Journal of Information Management*, *47*, 218–225.

Joachims, T. (1998). Text categorization with Support Vector Machines: Learning with many relevant features. In Machine Learning: ECML-98 (pp. 137–142)

Kowsari, K., Jafari Meimandi, K., Heidarysafa, M., Mendu, S., Barnes, L. E., & Brown, D. E. (2019). Text classification algorithms: A survey. *Information*, *10*(4), 150. https://doi.org/10.3390/info10040150

Nguyen, T. T., Nguyen, N. D., & Nguyen, T. V. (2015). Random forest classifier combined with feature selection for financial text classification. *Journal of Computer Science and Cybernetics*, *31*(4), 343–356.

Rennie, J. D., Shih, L., Teevan, J., & Karger, D. R. (2003). Tackling the poor assumptions of naive Bayes text classifiers. In ICML (Vol. 3, pp. 616–623)

Sandoval, L., Nunez, J. C., & Gomez-Adorno, H. (2020). Consumer complaint classification using TF-IDF and machine learning models. *Proceedings of the International Conference on Artificial Intelligence (ICAI)*, 84–90.

Taylor, J., & Amoss, T. (2025). SMS Fraud Detection in Chichewa using Classical Machine Learning Models. African Journal of AI Research, 9(1), 45–52.

Zhang, Y., & Jin, R. (2010). Understanding the bag-of-words model: A statistical framework. International Journal of Machine Learning and Cybernetics, 1(1), 43–52. https://doi.org/10.1007/s13042-010-0001-0