Improving Email Efficiency: Intelligent Categorization through NLP

Tanvee Proag (2422814)

University of Mauritius

Abstract. In today's fast-paced business environment, effective email management is a high priority to respond to customer inquiries in a timely manner. Accenture Services Ltd. prides itself on quality excellence when it comes to customer satisfaction. To maintain this standard, the communication between clients and internal employees can be facilitated through the use of NLP techniques so as to maintain productivity. This paper proposes a method to make use of some NLP techniques with existing company datasets in an attempt to improve efficiency and customer service. The aim is to reduce manual sorting so that employees can turn their focus to high-priority tasks by automating email classification.

1 INTRODUCTION

Background Study - Accenture is a consultancy firm in which the technology division is dedicated to address clients' software needs and deliver insights and solutions - it involves helping organisations meet their current software requirements as well as set them up for future requirements. Accenture has always emphasized on the importance of innovation and the ability to adapt in today's rapidly evolving business landscape by having *Change* as its emblem. It assists businesses with their digital tasks by offering solutions that are suited to their specific problems. Accenture approaches each challenge like so - it tries to have a comprehensive understanding of the latest technological advancements and then provides strategic guidance that aligns with the clients' goals. Basically, Accenture succeeds in staying competitive in this business environment by encouraging businesses to embrace new technologies. Therefore, it is perfectly aligned for Accenture to implement NLP techniques which represent some of the most advanced artificial intelligence technologies available in the present day.
Problem Statement - At Accenture, the current email system presents several challenges for employees who have to manage communications with clients. The email inboxes are cluttered leading to an overwhelming volume of messages which can be difficult to navigate through. As a consequence, employees spend a lot of time manually sorting through emails which is inefficient and detracts from their productivity. The clutter makes it hard to track follow-ups in lengthy email

threads. The email management system that Accenture currently uses, Microsoft Outlook for Business, provides essential email management capabilities and it integrates well with other Microsoft tools. However, its lack of customization options and its heavy reliance on standard features limit its effectiveness for Accenture which would prefer a customized solution. It offers limited customization options, categorizing emails only into basic folders such as Inbox, Sent and Junk. This lack of flexibility does not allow employees to filter out spam or prioritize urgent communications. Both important emails such as urgent JIRA ticket notifications and non-urgent Teams notifications are mixed together in the same inbox, sorted solely by the time they were received. Employees receive different types of emails that require different levels of attention including meeting invitations, announcements about office events, client visit notifications and daily newsletters like the Accenture Good Morning News. This results in critical messages being overlooked amidst less pressing ones.
Proposed Solution - To improve efficiency and communication within the company, a more sophisticated email categorization system is needed that allows employees to classify emails by urgency and relevance. This would ensure that important tasks are prioritized effectively.

2 DATA REQUIREMENTS

Accenture collaborates with Microsoft Azure, Amazon Web Services and Google Cloud for data storage solutions that can be used as datasets for analytics and machine learning. The *Accenture Cloud First* and *Momentum Digital Transformation Platform* are cloud-based platforms that serve as centralized repositories to store and manage a wide array of data including operational reports, client information and analytics. This enables the integration of different data sources into a single platform for advanced analytics and insights for decision-making. Accenture uses its own business tools to assess and optimize data storage solutions based on strategic business needs to ensure that data is readily accessible for analysis. This data stored in the cloud can be used as a dataset for various purposes such as developing predictive models, including this proposed model.

The dataset of emails found from the cloud platform are split into the training (70%) and testing (30%) sets.

```
X_train, X_test, y_train, y_test = train_test_split(X, labels,
test_size=0.3, random_state=1)
```

Text preprocessing prepares the email text for analysis. Firstly, tokenization can be applied to the email content - to lay the groundwork for further analysis, it is broken down into individual words (or, tokens). The Natural Language Toolkit (NLTK) library provides strong support for tokenization as it has word and sentence tokenizers: `nltk.word_tokenize()` is used to tokenize text into individual words and `nltk.sent_tokenize()` into sentences. TreebankWordTokenizer follows the Penn Treebank conventions to handle contractions, possessives and punctuation accurately. Then, normalization is applied to standardize the text by converting all characters to lowercase (A $\rightarrow$

a), removing punctuation (full stops, commas, semi-colons etc.) and eliminating special characters (dashes, hyphens, hashtags etc.). The `re` module is used for removing punctuation and special characters efficiently using regular expressions. `re.sub()` can be used to substitute unwanted characters like punctuation and special symbols with spaces like so -

```
cleaned_email = re.sub(r'[^\w\s]', '', text)
```

where the regular expression matches any character that is not a word character (w) or a white space character (s) and substitutes the matched pattern with an empty string. It is flexible and allows for complex text manipulations. This ensures consistency across the dataset. Stop words removal can be performed to eliminate common words found in documentation reports such as "and" and "the" as they do not contribute significant meaning to the text. NLTK provides a built-in list of stop words (a stopwords corpus) that contains common stop words for multiple languages. It is customizable - it allows adding and removing words, which is going to be useful in different projects.

```
stop_words = set(stopwords.words('english'))
custom_stopwords = {'email', 'meeting'}
stop_words.update(custom_stopwords)
stop_words.remove('not')
```

At Accenture, each project involves a team that uses a specific vocabulary or jargon depending on client needs which justifies the need for customization. Stemming and lemmatization techniques can be used to refine the text - Stemming reduces words to their root forms by cutting off prefixes or suffixes while lemmatization considers the context of words and converts them to their dictionary forms. NLTK includes various stemming algorithms such as the Porter, Snowball and Lancaster stemmers that are widely used in industry projects like this one. SnowballStemmer is a language-specific stemming algorithm that provides high accuracy. PorterStemmer is the most widely used algorithm for stemming in English. LancasterStemmer would be a more aggressive alternative. In this case study, SnowballStemmer might prove more useful as it supports multiple languages and is more efficient, given that Accenture is an international organization that engages in multilingual communication to accommodate its diverse global audience despite English being the most commonly used language.

```
english_stemmer = SnowballStemmer("english")
french_stemmer = SnowballStemmer("french")
english_stemmed = [english_stemmer.stem(w) for w in english_words]
french_stemmed = [french_stemmer.stem(w) for w in french_words]
```

NLTK provides a WordNet lemmatizer that uses the WordNet lexical database to take into account the context of words allowing for more accurate lemmatization. The WordNetLemmatizer takes into account the part of speech (POS) of a word to determine the correct lemma which helps improve accuracy. For instance, the word ”meeting” will be lemmatized as ”meet” if recognized as a verb but ”meeting” will remain unchanged if is identified as a noun. That is because WordNetLemmatizer is customizable as the POS tag can be specified to improve context-awareness.

```
def get_wordnet_pos(tag):
    if tag.startswith('J'):
        return wordnet.ADJ
    elif tag.startswith('V'):
        return wordnet.VERB
    elif tag.startswith('N'):
        return wordnet.NOUN
    elif tag.startswith('R'):
        return wordnet.ADV
    else:
        return wordnet.NOUN
pos_tags = nltk.pos_tag(words)
lemmatized_sent = [lemmatizer.lemmatize(word, get_wordnet_pos(tag))
                  for word, tag in pos_tags]
```

The next phase is feature extraction. Bag of Words (BoW) model can be used to represent text data as a collection of words, disregarding grammar and word order but retaining word frequency. Scikit-learn offers a simple implementation of the BoW model with the CountVectorizer which is easy to use and integrate with machine learning models. CountVectorizer breaks down text data into individual words (or tokens), counts the frequency of each word and creates a sparse matrix where each row represents a document and each column represents a word from the corpus.

```
X = CountVectorizer().fit_transform(documents)
feature_names = CountVectorizer().get_feature_names_out()
dense_matrix = X.toarray()
```

This can integrate with Scikit-learn's machine learning models which makes it easy to add a text representation step to a machine learning pipeline. Term Frequency-Inverse Document Frequency (TF-IDF) can prove helpful in evaluating the importance of a word in an email relative to the collection of emails. Scikit-learn's TfidfVectorizer computes TF-IDF scores and enables easy integration with machine learning pipelines. It calculates the Term Frequency (TF) and Inverse Document Frequency (IDF) scores for words in the corpus. TF measures how often a word appears in an email; IDF measures how important a word is across multiple email contents. Words that are common across emails (like “dear”, “regards”) get lower scores while less common words get higher scores.

```
vectorizer = TfidfVectorizer()
X = TfidfVectorizer().fit_transform(corpus)
feature_names = TfidfVectorizer().get_feature_names_out()
```

```
tfidf_matrix = X.toarray()
```

TF-IDF highlights the words that are informative and relevant to each email, making it more powerful for this text-based machine learning task compared to simple word counts. N-grams (Unigrams, Bigrams and Trigrams) can be used to capture common phrases e.g. "urgent response needed".

```
tfidf_vectorizer = TfidfVectorizer(ngram_range=(1, 3))
X_ngrams = tfidf_vectorizer.fit_transform(email_texts)
```

sentiment.SentimentIntensityAnalyzer from NLTK uses the VADER (Valence Aware Dictionary and Sentiment Reasoner) lexicon which assigns sentiment scores to words based on their emotional intensity. It can help identify emails with strong emotional content which may indicate urgency. `compound` is a normalized score that summarizes the overall sentiment ranging from -1 (very negative) to +1 (very positive).

```
def get_sentiment_score(email):
    sentiment = SentimentIntensityAnalyzer().polarity_scores(email)
    return sentiment['compound']
X_sentiment = [get_sentiment_score(email) for email in email_texts]
```

Metadata can be extracted from the email. This way, emails from executives or clients can be flagged for quicker responses; emails with attachments might need quicker action; longer subjects may imply more detailed discussions -

```
def extract_metadata(email):
    important_senders = ['CEO', 'client']
    is_important_sender = any(sender in email['from'] for sender
                          in important_senders)
    has_attachment = email['has_attachment']
    subject_length = len(email['subject'].split())
    return [is_important_sender, has_attachment, subject_length]
X_metadata = [extract_metadata(email) for email in email_data]
```

Next, they are combined into a single feature matrix using hstack from scipy.sparse.

```
X_combined = hstack([X_ngrams, np.array(X_sentiment).reshape(-1, 1),
np.array(X_metadata)])
```

This combination of different feature matrices into a single structure helps in understanding the context and urgency better.

3 METHODOLOGY

To categorize emails into different levels of urgency, keyword matching, time-sensitive phrases and response requirements are considered.

```
high_urgency_keywords = ["ASAP", "urgent", "immediate"]
medium_urgency_keywords = ["end of day", "this week", "priority"]
low_urgency_keywords = ["FYI", "no rush", "whenever"]
response_required_keywords = ["action", "respond by", "follow-up"]
```

With features extracted, classification algorithms are employed to categorize emails effectively. The Naive Bayes classifier can be used for spam detection as it is efficient in handling large datasets. However, Support Vector Machines (SVM) might be better suited for high-dimensional spaces as they find the optimal hyperplane that separates different classes of emails.

```
svm_classifier = SVC(kernel='linear')
svm_classifier.fit(X_train, y_train)
svm_predictions = svm_classifier.predict(X_test)
```

To improve the accuracy even further, Random Forests can be used - this ensemble learning method builds multiple decision trees and merges their results.

```
rf_classifier = RandomForestClassifier(n_estimators=100, random_state=1)
rf_classifier.fit(X_train, y_train)
rf_predictions = rf_classifier.predict(X_test)
```

For performance analysis, metrics like accuracy, precision, recall and the F1-score are employed.

```
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
```

Implementing a confusion matrix using `seaborn.heatmap()` can provide a visual representation of performance across different categories to easily identifiy misclassifications.

```
confusion_matrix(y_test, y_pred)
```

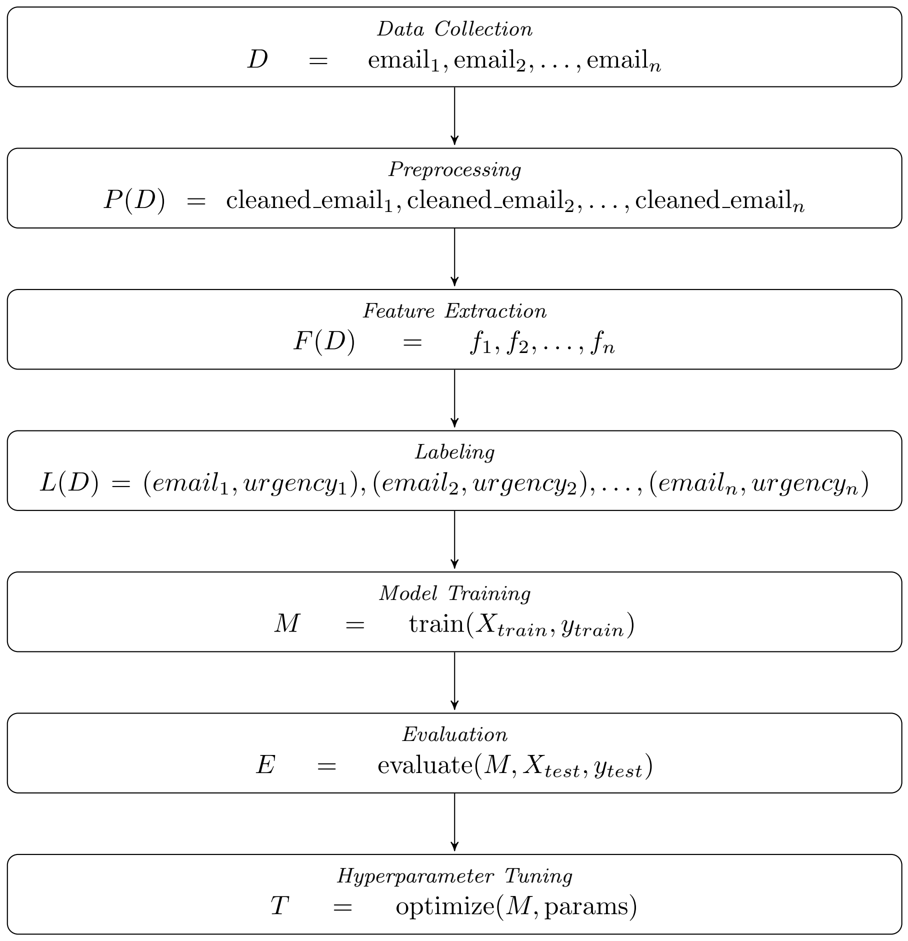
To improve the performance, hyperparameter tuning can be done - in SVM, the parameters C and kernel can be tuned using Grid Search with GridSearchCV from sklearn.model_selection.

```
param_grid = {'C': [0.1, 1, 10, 100], 'kernel': ['linear', 'rbf']}
grid_search = GridSearchCV(SVC(), param_grid, cv=5, scoring='accuracy')
grid_search.fit(X_train, y_train)
```

To ensure the model generalizes well to new unseen data, it needs to be trained and tested on different subsets of the dataset. cross_val_score is used for k-fold cross-validation. This reduces overfitting as the model is tested on different data splits and it provides a more accurate evaluation.

```
cross_val_scores = cross_val_score(svm_classifier, X_train, y_train, cv=5)
```

The proposed email classification model can be illustrated as


Data Collection
$D = \text{email}_1, \text{email}_2, \ldots, \text{email}_n$
Preprocessing
$P(D) = \text{cleaned\_email}_1, \text{cleaned\_email}_2, \ldots, \text{cleaned\_email}_n$
Feature Extraction
$F(D) = f_1, f_2, \ldots, f_n$
Labeling
$L(D) = (email_1, urgency_1), (email_2, urgency_2), \ldots, (email_n, urgency_n)$
Model Training
$M = \text{train}(X_{train}, y_{train})$
Evaluation
$E = \text{evaluate}(M, X_{test}, y_{test})$
Hyperparameter Tuning
$T = \text{optimize}(M, \text{params})$

4 EXPECTED RESULTS

These NLP techniques can be used for automated email routing where emails are automatically categorized and routed based on their content. It significantly reduces manual sorting and response time. The implementation of intent recognition helps understand the purpose behind emails making it easy to categorize by both content and by required actions. Including sentiment analysis into the system prioritizes responses based on urgency. This is a valuable feature in customer service contexts because understanding client emotions results in better engagement and satisfaction. Through these techniques, Accenture can create an efficient email categorization system that increases productivity and improves communication management.

It is expected that there will be a quicker email turnaround, leading to shorter response times for client inquiries and internal communications. This is beneficial as timely responses can greatly impact client satisfaction and project timelines. In addition, provided that the manual workload associated with sorting and managing emails is reduced, a boost in overall productivity is to be expected. Employees can concentrate on more strategic tasks that benefit the company like creating client solutions and growing relationships given that repetitive tasks have been automated through NLP. Teams will be able to allocate resources effectively with the help of the proposed email classification. Employees will be better equipped to prioritize their work and deal with critical issues involving more pressing matters quickly if they have a better understanding of which emails are urgent and which are non-urgent. This implementation creates a better customer experience at Accenture. Finally, the quick response time is expected to strengthen client satisfaction and lead to client loyalty which is key to reduce churn in the current competitive marketplace.

5 DISCUSSION

The proposed model relies heavily on having historical email data which if not readily available, might have required a lot of effort to gather but since Accenture's Momentum Digital Transformation Platform is very reliable, this should not cause issue. The Accenture Cloud First platform provides a scalable and flexible cloud-based environment to develop, deploy and manage machine learning models and NLP pipelines. The cloud resources it provides allows scaling up the infrastructure as it can accommodate increasing demands when the email data eventually grows over time. Cloud First supports CI/CD pipelines which can be used to integrate model updates and improvements. Implementing Support Vector Machines and Random Forests require appropriate computational infrastructure for large scale data classification - this should not be a problem since Scikit-learn library provides well-optimized algorithms. The proposed features for determining urgency (keywords, metadata and N-grams) should be flexible - this customization might need some domain expertise to fine-tune. Given that most likely an API will have to be called to use this model in the existing Outlook email management system, it will have to be trained and retrained periodically since the nature of emails evolve over time. This reduces bias in feature engineering as the model might inadvertently focus on specific features that do not apply across all emails. This model was designed under the assumption that employees will readily embrace this new technology - but if they feel the urgency of their communications is not being reflected, they might lose trust in it. As for data privacy concerns, Accenture's global Client Data Protection (CDP) backed by security processes, policies and governance develops a plan for each client project and provides end-to-end security risk management and data security.

References

1. Microsoft.com. (2024). Microsoft Outlook for business - Microsoft. [online] Available at: `https://www.microsoft.com/en-us/microsoft-365/outlook/outlook-for-business?msockid=3ec6102fd19e6a0a2a7e03f0d0096b77` [Accessed 8 Oct. 2024].
2. Narain, K. and Guan, L. (2022). A New Dawn for Dormant Data. [online] Accenture Cloud Data Value: A New Dawn for Dormant Data. Available at: `https://www.accenture.com/content/dam/accenture/final/capabilities/technology/cloud/document/Accenture-Cloud-Data-Value-A-New-Dawn-for-Dormant-Data-vF.pdf` [Accessed 11 Oct. 2024].
3. www.accenture.com. (n.d.). Accenture and Amazon Web Services take you further, faster. [online] Available at: `https://www.accenture.com/us-en/services/cloud/aws-business-group` [Accessed 8 Oct. 2024].
4. www.accenture.com. (n.d.). Accenture Momentum. [online] Available at: `https://www.accenture.com/us-en/services/consulting/accenture-momentum` [Accessed 11 Oct. 2024].
5. Information Security at Accenture. (n.d.). Available at: `https://www.accenture.com/content/dam/accenture/final/capabilities/technology/technology-innovation/document/Accenture-Information-Security-at-Accenture.pdf` [Accessed 8 Oct. 2024].
6. Scikit-learn (2018). sklearn.feature_extraction.text.TfidfVectorizer — scikit-learn documentation. [online] Available at: `https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html` [Accessed 9 Oct. 2024].
7. Scikit-learn (2018). sklearn.feature_extraction.text.CountVectorizer — scikit-learn 0.20.3 Documentation. [online] Scikit-learn.org. Available at: `https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html` [Accessed 9 Oct. 2024].
8. NLTK (2009). Natural Language Toolkit — NLTK 3.4.4 documentation. [online] Nltk.org. Available at: `https://www.nltk.org/` [Accessed 11 Oct. 2024].