VRIJE
UNIVERSITEIT
BRUSSEL

Graduation thesis submitted in partial fulfilment of the requirements for the degree of Master of Science in Applied Sciences and Engineering: Computer Science

INBOX HARMONY

A Recommendation System to Manage and Organise Emails Based on PIM Principles

Anisha Sachdeva

Academic year 2022–2023

Promotor: Prof. Dr. Beat Signer
Supervisor: Prof. Dr. Beat Signer

Science and Bio-Engineering Sciences

Abstract

In today's digital era, email communication has become a fundamental aspect of personal and professional interactions. However, the increasing volume of emails causes significant challenges in organising and managing the mailbox efficiently. Existing email clients provide automation such as rule-based filtering and spam detection. However, Park et al.'s [19] investigation highlights that users express the need for better automation within email clients. This thesis investigates the reasons why users refrain from utilising rule-based filtering and folders while managing and organising emails. When conducting the literature study on the state of the art of email overload and the challenges related to it, we observed that no recent study has been performed to identify the gaps between the automation provided by email clients and the actual usability of this functionality by users. Much research has taken place to optimise spam filters and provide users with the platform to filter emails by defining their own rules using richer data models, but there is less recent exploration to identify the reasons behind a user's cluttered inbox and their reluctance to utilise the folder functionality already available for organising emails. Therefore, we first conducted an exploratory survey to gain insights about arbitrary users' email management practices. We observed how participants have a large number of emails in their inbox and do not create and manage folders to organise their mailbox. Additionally, we noticed various reasons why the participants did not prefer the rule-based filtering offered by email clients. Based on our findings, we designed *Inbox Harmony*, a recommendation system to manage and organise emails. The recommendation system consists of two parts. The first part is the selection, optimisation and deployment of the machine learning model to perform natural language processing and text classification. We carefully chose Classifier Chains, a multi-label classification algorithm and Fuzzy C-Means, a soft-clustering algorithm based on the principles of fuzzy logic. Further, we optimally deployed both these models to be trained on an individual's Gmail account and provide personalised yet accurate machine learning-based recommendations to users to move emails from their inbox to other folders. These suggested folders could either be already existing in the user's mailbox or might be a newly suggested folder by the system. The second part consists of the development of the add-on for Gmail, a popular email client. Using Google Apps Script, we display the add-on in an individual's Gmail interface. Using the add-on, the users are displayed various machine learning based recommendations to move different emails from the inbox to the suggested folders. The users can easily move, delete or reject the selections with a click of a button. New folders can also be created while moving the email from the inbox to the newly suggested label. The machine learning model learns from the actions taken by the user in order to provide better recommendations each time. The add-on was evaluated with ten participants. The feedback received was positive and users showed interest to use this add-on on a regular basis to manage their mailbox. Overall, we conclude that we have tried to solve a real-life problem faced by many of us and we have succeeded at presenting the prototype, Inbox Harmony, a recommendation system for managing emails.

Acknowledgements

Without the guidance and support I have received during my master's journey, I would not have been able to successfully complete this thesis. Therefore, I would like to take the time to thank some amazing people in my life who made this journey possible.

First and foremost, I would like to thank Professor Dr. Beat Signer. He gave me my first introduction to Personal Information Management (PIM) during courses I took at the Vrije Universiteit Brussel. Those courses sparked my interest and I am incredibly grateful for the amazing opportunity to work in the same field, guided by the expertise of Prof. Signer. His support and guidance during our weekly progress meetings mentored me to move in the right direction. This research would not have reached its fullest potential without his supervision and encouragement.

I also want to thank my friends and fellow students for their support throughout the year. A special thanks to Melba Raj and Robin De Haes for always inspiring and motivating me during the challenging moments throughout my master's journey. Additionally, I want to thank my boyfriend, Shubhang Khattar for his ultimate encouragement during the whole journey. Further, I am very grateful to my parents, brother, sister-in-law and cute little nephew. Their incredible support has kept me striving for excellence. A heartfelt thank you is extended especially to my mother, whose love, dedication and belief in me have been the driving force behind all my accomplishments. A special shoutout to my incredible best friends, Parth Malhotra, Karishma KS, and Surbhi Bansal, who consistently made the effort to call, check up on me and provide consistent support and cheer throughout.

Contents

Chapter 1

Introduction

As technology has evolved in the past few years, techniques for storing information have also developed. In the past, data was stored in physical form on paper, whereas nowadays most of the information is available in digital form. This digital information is stored on electronic devices and other digital storage media using file and folder systems. *Email* has emerged to be one of the primary forms of communication and sharing information over the internet. The increasing dependency on emails as a primary communication medium has led to a growing requirement for efficient management strategies.

The underlying architecture of an email management system is typically the folder system where the inbox is the primary folder which receives all information. However, a user can create more folders in their mailbox[1] to group related emails together based on certain criteria like the sender's name or subject. By utilising the folder functionality, a user can manage the ever-increasing number of emails in their inbox. They can create folders to organise their mailbox and reduce clutter in their inbox. As simple as it may sound, it requires significant effort by a user to manage their mailbox. Despite the availability of various productivity tools and techniques, users often find themselves overwhelmed by the amount of emails they receive. Today, email clients provide some automation like spam detection and rule-based filtering. Using various machine learning-driven filters and metadata such as the domain and characteristics of the IP address, email clients automatically filter out unsolicited emails (spam). Rule-based filtering allows users to define specific actions that can be applied to various incoming emails based on pre-defined rules by the user. However, studies show that most of the users do not manage their emails by creating folders [14] and they feel the need for more automation [19] in the email clients. The consequences of inadequate email management can range from information overload to a loss of important information and missed opportunities for collaboration or networking. Hence, this thesis aims to contribute towards email management by developing a tool to organise emails.

The objective of this thesis is to analyse current email management practices, identify the challenges faced by users while managing emails in their mailbox, research state-of-the-art to automate email management and how Personal Information Management (PIM) principles can be applied to solve the issues related to the organisation of emails.

[1]We consider the mailbox as a whole container containing all the folders (including the inbox) and respective emails available in a user's email account. The inbox is considered to be a folder which gets all the incoming emails.

1.1 Research Question

In some research conducted by McKinsey[2], it was found that a working professional spends 28% of their work time in a day reading and answering emails. Some users are aiming for an *inbox-zero* approach by cleaning their inbox and only keeping the emails in the inbox which require some action. However, with the high volume of incoming emails, many have given up on filtering their inbox. Once an email enters the inbox, it remains there forever. There could be numerous reasons why a user might not organise emails in folders as they might feel creating folders to be a tedious task, they might not consider using rule-based filtering as they would want to get the emails in their inbox first and then sort or maybe they feel difficult to categorise an email into a single folder. Additionally, the Harvard Business Review[3] observed that these users prefer piling up emails as they believe that the search functionality provided by the email clients is powerful enough to retrieve emails at a later point in time and they do not need to necessarily organise the emails. It is true that search is pretty fast and powerful. However, a crowded inbox costs users time as they tend to re-read emails when they check their completely filled inbox.

Even though email clients provide automatic rule-based filtering along with spam filtering, the issue of efficiently managing the inbox still prevails. This brings us to the two research questions which we try to address in this thesis.

1. What is lacking in today's email client UIs, restricting users from easily managing their mailbox?

2. Can we use machine learning algorithms to recommend actions on similar emails, helping users to better manage their mailbox automatically but also with the human in the loop?

1.2 Contribution

Our contributions in this thesis are twofold. First, we conducted a survey to better understand current email management practices. We examined how often users check their emails, how many emails they receive in a day, how they manage emails in their inbox, whether they create and maintain folders, whether users are aware of the rule-based filtering available in email clients, whether they prefer using rule-based filtering and if not then why and whether users would be interested in a machine learning-based component in their email client which could recommend them to move certain emails to folders based on their previous actions.

Based on the detailed analysis of the survey and the resulting findings, we then developed and evaluated a recommendation system called as *Inbox Harmony* for email management. As 'harmony' means an orderly and pleasing arrangement of parts, our recommendation system intends to bring harmony to a user's inbox. The system aims at reducing clutter and better organising a user's mailbox. As highlighted in Figure 1.1, this system is an add-on available for a Gmail account's interface which displays dynamic and personalised suggestions to a user to move certain emails from their inbox to other folders. Unlike spam filters which are generic for all users, this recommendation system is personalised for each user and recommends moving emails based on the previous actions taken by a user in their inbox. Our solution not only displays the suggestion to move emails to already existing folders but also finds similar emails present in an inbox, groups them and suggests a new folder for those emails. Inbox Harmony enables users to move emails to existing or new folders from the inbox with just a click of a button.

[2] https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/the-social-economy

[3] https://hbr.org/2019/01/how-to-spend-way-less-time-on-email-every-day

In contrast to the existing rule-based filtering where an email can only be assigned to a single folder, Inbox Harmony might suggest multiple labels for a given email. Another advantage of using our solution is that it does not only classify incoming emails, but can also help users to declutter the already existing email chaos in their inbox.

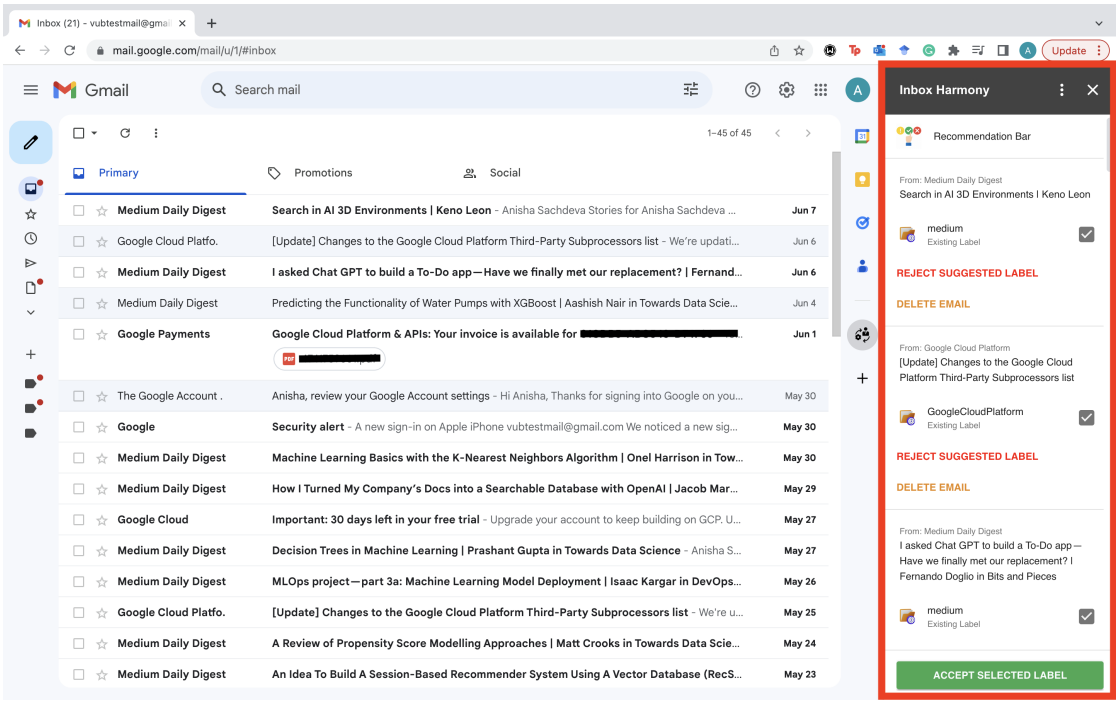


Figure 1.1: A screenshot of our *Inbox Harmony* email recommendation system in a Gmail account's user interface

Rather than using the traditional machine learning algorithms which are used to perform multi-class classification where each output instance is assigned to exactly one of the several mutually exclusive classes, we used a multi-label classification algorithm. A multi-label classifier can assign an output instance to multiple classes simultaneously. Using the multi-label classification, the system enables a user to not pigeonhole an email in a single folder. Further, our system is enabled to provide suggestions irrespective of the fact whether a user has already defined any folder(s) or not.

1.3 Thesis Outline

This thesis is comprised of 7 chapters. In the first chapter, we provide the general context of the thesis. Here, we present the motivation and research question behind the thesis along with mentioning our contribution towards the topic. The second chapter contains the study of related work for this thesis. In this chapter, we discuss some PIM principles along with how emails have become an integral part of our lives and the requirement to handle and organise the overwhelming amount of emails. We then discuss various automation that have been performed to better enable a user to handle the email overload. In the third chapter, we discuss the analysis and results of the survey that we conducted to understand the current email management practices. Based on

our findings in chapter three, we propose a new recommendation system for email management in chapter four. In this chapter, we discuss the theoretical Inbox Harmony solution along with listing a set of requirements that our recommendation system should fulfil. In chapter five, we discuss the technical details of our proposed prototype implementation. We discuss the various technologies and frameworks used along with the rationale behind some design choices. We also discuss some challenges faced while implementing the prototype and the possible solution implemented. Chapter six is divided into two parts. At first, we give a walkthrough of the recommendation system and then we discuss the evaluation of the presented Inbox Harmony solution. In the final chapter, we discuss the potential improvements that we discovered during the implementation phase or while evaluating our system. This thesis document is concluded with a thorough discussion of the contributions.

Chapter 2

Related Work

2.1 Personal Information Management

The term *Personal Information Management* first appeared in the 1980s, when the personal computer's ability to help human beings with processing and managing information was realised. Often, the history of PIM is traced to the 1945 essay, *As We May Think* by Vannevar Bush where he envisioned the Memex, a device to store the exceeding information such as books and communications which could be consulted later [9]. In 1988, Lansdale stated that personal information is any information held by an individual in any form ranging from files, notes and folders to personal records [15]. This information may not necessarily be confidential but it is for one's own use without which the individual may feel deprived of. Proceeding further with the definition of more digital space, Henderson described PIM as *"the process of acquiring, storing, managing, retrieving and using digital documents"* [13]. Based on the aforementioned two studies, various other research has been conducted establishing the formal definition of PIM as *"the study of the activities a person performs in order to acquire or create, store, organise, maintain, retrieve, use, and distribute information in each of its many forms (paper and digital, in e-mails, files, web pages, text messages, tweets, posts, etc.) as needed to meet life's many goals (every day and long term, work-related and not) and to fulfil life's many roles and responsibilities (as a parent, spouse, friend, employee or member of the community)."* [28]

The three fundamental activities concerning personal information management can be summarised as *keeping*, *organising* and *finding* activities. It was observed by Jones et al. [28] that when a user encounters some information, they take a decision if the information needs to be addressed and whether they should keep it or discard it. To keep the information, various organisational techniques like tagging, labelling, filing and piling are used. In order to find the right information at a later point in time, it is important to store it in the right place with sufficient completeness and in the right form. Searching and navigating (location-based search) are the two most commonly used methods to re-find the information. The details about each organising and finding technique are discussed in the following section.

2.1.1 Personal Information Collection

Personal Information Collections (PICs) are the conscious compilations made by a user to manage and organise the information in a particular collection. Some examples of PICs are:

1. The organisation of papers in an office including the piles of papers having the most recent paper at the top and least recent ones at the bottom. Also, the folders containing different other papers.
2. The organisation of digital documents on a desktop including folder and file system.
3. Digital songs managed on different digital platforms.
4. Articles managed on cloud repositories like Dropbox and Google Drive.

Nowadays, another interesting personal information collection (PIC) has emerged which was originally not designed to store and manage information but only for the sole purpose of communication. That PIC is the *'email'*. In the following sections, we will discuss how email is used as a PIC with the challenges faced to manage, organise and retrieve emails, along with some solutions provided in the current state of the art.

2.2 Email Overload

At first, in 1988, Summer [23] addressed the influence of the heavy use of emails in business organisations by interviewing and surveying employees. She established the base for the fact that email was replacing the previous methods of communication like phone calls and memos for various business activities like organising and scheduling events. Following her research, Mackay [16] interviewed various users and claimed two principles. First, how the email usage was highly diverse in the sense that different users had varied numbers of folders with different sizes of inboxes and second, email was more than just a communication system. She mentioned that users were already archiving emails for later subject retrieval, prioritising emails to sequence the order of their work and delegation of tasks was taking place via emails.

Although initially email was designed with the intent of communication, over the years, it is now also being used for functions like *task management* and *personal archiving*. This supplementary use of email, which it was not designed for is often regarded as *"email overload"*, a term coined by Whittaker and Sidner [26]. It was noted by Whittaker and Sidner [26] that many people operate their inbox as a task manager where they typically keep the emails in the inbox as a "to-do" reminder list and simultaneously move the already responded or unimportant emails to different folders as an archiving technique. Features like reminders and calendars to schedule meetings enable the email client to act like a task manager. In any organisation, email has now become the centre where all the work is received and regulated. This obviously results in an overwhelming volume of emails leading to email management challenges.

2.2.1 Email Handling and Organisation

One of the main challenges is to manage and organise the incoming email messages. As early as 1988, Mackay [16] was the first one to notice three different forms of email management. First, the *prioritisers*, such users were the ones with a very active social network and would receive a large number of daily email messages. Because of the enormous amount of messages, they felt like they are on the *edge of losing control* of their email messages. They identify various categories, prioritise the incoming messages and respond accordingly. Second, are known as the *archivers*. These users do not delete any emails and tend to hoard emails with the thought that they might be of some use in the future. Archivers are most interested in categorising and retrieving email messages.

The third type of users are the ones who receive the least number of emails and generally use this medium of communication to delegate work to the other users. They manage their tasks via email.

Mackay was the first one to closely examine the diversity present in email management. However, from the history of PIM, Malone examined the different organisational strategies observed in a paper-based office which are very well comparable even to today's digital world. Malone [17] distinguished between, filing and piling strategies. He draws the difference between the *"neat"* and the *"messy"* office, where the neat office contains some kind of categorisation while organising the papers and the messy office contains very relaxed characterisation in piles. The neat office has sorted documents with a structured layout, whereas the messy office is the opposite containing an unstructured layout. In brief, the two main units of desk organisation are *"files"* and *"piles"*. In files, there is categorisation present in the form of naming a folder or arranging different folders in some order (chronological or alphabetical). Whereas, in piles, the documents are not titled or grouped together in any order. Piles generally have a haphazard structure. In a similar fashion, today, some people generate different folders to maintain their emails whereas others do not clean their inbox at all and it becomes a pile of emails. In 1996, Whittaker and Sidner precisely described the three types of organisational strategies which are generally used to maintain the inbox, namely, (i) no filers (people who do not use any folder to manage their inbox), (ii) frequent filers (people who clean their inbox daily and maintain the folder hierarchy in emails), (iii) spring filers (people who clean their inbox intermittently) [26].

People falling into the category of no filers or spring filers are not entirely arbitrary. In one of the situations where specific users like managers, tend to receive a large number of emails on a daily basis, with the requirement of being away from the work desk for long hours, it becomes difficult to manage and clean the inbox [26]. Another major reason for the reluctance to filing strategy is that often people cannot decide how to categorise an email [15]. Hence, the compensating strategy becomes to pile emails in the inbox.

Frequent filers have the advantage of having their inbox as an absolute "to-do" list with a relatively smaller number of emails in their inbox. However, it is difficult to move emails from the inbox and put them in different folders. The reason is, a substantial amount of effort is required to generate and maintain the folders. To move an email to a folder, either there should be a suitable folder existing or a new folder has to be created. Moreover, existing folders can be of two types, *known* and *unknown* folders [4]. A known folder is one of which the user is already aware of its position and name, whereas, an unknown folder is one, of which the user is aware, that it exists but does not remember the location where it resides. The two main factors depending upon which the retrieval time of an unknown folder is affected can be described as, (i) the frequency of filing which enables a user to vaguely remember where a folder may exist in a hierarchy [26] (ii) if the total number of folders surpass the number of visible folders on the screen, it would then require more time to scroll the list of folders [4].

While arranging emails in different folders could be a time-consuming task, another interesting categorisation technique is *category tagging* (in Outlook), also known as *labelling* (in Gmail) and *flagging* (in Apple Mail). With tagging, a user is not required to pigeonhole an email in one folder, rather, they can create a tag based on the rich, semantic metadata in order to group similar emails.

Once a user finds an appropriate folder and moves the email from inbox to that folder or tags an email, the next challenge faced is to retrieve the message.

2.2.2 Re-finding Emails

Mostly, people tend to store all their emails, be it in the inbox or in organised folders so that they can retrieve the message later, whenever required. The two most common scenarios for retrieval are:

1. When people go back to older emails to get some important information such as contact details, attachments or reference materials.

2. Often people tend to postpone replying to an email as they might have insufficient time at that moment to reply or they might require to collect information (inputs) from their co-workers. Hence, re-finding information in emails becomes an important aspect of email management.

Organisation and retrieval of information are closely related to each other. The email management approach practised by the frequent filers to deliberately create folders or tags anticipating the context of retrieval is known as *preparatory* organisation, while, the re-finding behaviour such as scrolling or searching adopted by the no filers (or spring filers) is known as *opportunistic* organisation [27]. The main trade-off between these two approaches is that the preparation of folders requires time and effort to remember which email is stored where which might not succeed at the time of retrieval. Whereas, with the opportunistic methodology it would also require more retrieval time to retract something from the pile, along with the chance of overlooking some important email in a cluttered inbox [26].

However, as verified by Koren et al. [14], as many as 70% of Yahoo! Mail client users have never created even one folder and as established by Grbovic et al. [11] only 10% of users can be classified as frequent filers with moving more than 10 emails into folders over a time span of 6 months. This clearly states that users prefer an opportunistic approach over a preparatory effort. In a study performed by Whittaker et al. [27], it was found that people who create and organise information in folders in an email client are more inclined to rely on those folders only at the time of retrieval. Though, those filers are not more successful at retrieving. As the folder accesses would take longer time on average, filers were less efficient at re-finding. Another possible reason could be that folders might be poorly managed and organised and at times not well suited for retrieval. Categorising a piece of information into exactly one folder can be pretty tricky.

Searching and scrolling are the other two prevalent re-finding behaviours that are more efficient leading to more successful retrievals. Though search requires more cognitive power than navigation, it is still decisive in the case of re-finding emails. In the case of filers, given a large number of emails, it would be difficult for a user to navigate into folders which then will also require cognitive power to remember the correct folder to look into. Searching and scrolling are anyways needed. While querying in email search, people rarely mention the complete context in their search, rather they give a short query and rely on the faceting and sorting traits in the email client's interface. The most important queries in email search are people-centric, that is, queries consisting of a person's name or email address. Ai et al. [2], through their research, highlighted the fact that people are mostly adapted to their email collections which helps them to easily recognise relevant results given by the email search engines. Further, the survey helped to understand that people tend to remember more while searching in an email as compared to a web search. Surprisingly search is not only used to find information and emails to reply to but it is also used to find particular emails in order to organise them in folders. Based on the fact that search operators are a subtle way of embodying filters into email clients like Gmail, Dredze et al. [10], proposed a "Search Operator Suggestion System" which would display a suggestion for 10 search operators to the user for their current mailbox view. For clear visibility and

to gain more clicks from the user, the suggestions were placed just below the search pane and above the threaded messages. The operators were ranked using a statistical machine learning ranker. In the user survey, a lot of users (54%) were happy to manage their inbox including actions like labelling emails or archiving, using the search operators.

Now an interesting question could be, if search is such an efficient way of re-finding information in emails, then why should we investigate some automation for the creation and management of folders? Simple, because foldering was not a reaction to the growing demands of re-finding emails. Rather, it was a response to rationalise the inbox against email overload in order to better see the inbox as a "to-do" list than as a pile of unrelated tasks.

2.3 Email Automation

With the ever-increasing email overload, there has been enormous research and studies going on in the area of automating numerous aspects of email processing.

Back in 1990, *procmail*, an autonomous mail processor was first rolled out which enabled users to define regular expression scripts to filter emails. As it was a rule-based system, it was often difficult to define the accurate rules in order to obtain the desired results along with the necessity of writing correct syntax.

With the advancement in Machine Learning, Artificial Intelligence and Information Retrieval, Re:Agent, an intelligent email agent was developed in 1998 by Boone [6]. An email consists of tagged headers like `From` header, `Date` header or `Subject` header, but the body of the email is generally an unstructured text. Considering the predefined set of words as concept features and extracting those features from emails, Boone [6] tried to separate different emails into *work* or *other* folders. Using a neural network and nearest neighbour algorithm, they trained 2/3 of the 1210 emails. The remaining emails acted as the test data. It was noted that if only keywords were supplied to differentiate between emails and put them into folders they often gave a poor performance. However, if example emails were provided, the results had better accuracy for both the neural network and nearest neighbour algorithm. Keywords possibly degraded the performance because it is difficult to select keywords that are complete as well as statistically compelling. This study and performance of Re:Agent helped the PIM researchers to move forward on the idea of using machine learning to learn the concepts present within an email and further using those concepts as features to learn actions (such as filtering, prioritising, foldering) which could be performed on similar emails. Following the footsteps of Boone, Mock built an add-in for Microsoft Outlook 2000 to automatically classify incoming emails based on existing user folders [18]. While the early work required to train the classifier by actually moving emails from the inbox to different folders, Mock pointed out a valid statement that some users purposefully leave emails in inbox as a reminder for some task. This may be disturbed by automatic filling and maybe the data would also not be trained accurately. As a solution, they proposed an *automatic grouping of the inbox*. This solved both issues, first, emails remained in the inbox which could be used as reminders and second, the grouping of emails in the inbox itself, made the inbox organised and manageable. The user could later file or delete the email from the inbox as required. They built classifiers based on existing user-created folders. Features from emails from a particular folder were extracted and added to the classifier. They used the nearest neighbour (NN) classifier. It would extract the features from the target message which has to be classified, compare it to all the emails and based on some similarity index, it would classify the email to a group in the inbox.

When Mock was automating the grouping of emails in the inbox, at the same time, Takkinen and Shahmehri came up with the conceptual model named as CAFE to manage the information

in emails. CAFE is an acronym for *Categorisation Assistant For Email* [24] and this application was based on the e-mail client known as Exmh. This is an interesting model as it offered three different representations of a mailbox which further depends on which "mode" a user wants to use the mailbox in. They offered three modes, namely, Busy, Cool and Curious mode. The functionality and representation of each mode are described as follows:

1. Busy Mode: This mode is designed for situations where a user has limited time to read new messages. The representation of the mailbox is so that only important messages amongst the latest messages are displayed. The user is shown three main folders named `Important`, `2nd Class` and `Junk`. Using the Naive Bayes theorem, the unseen messages are prioritised in one of the three folders. Along with the three main folders mentioned above, the standard folders like inbox, outbox, draft and to-do folders are also available in the busy mode.

2. Cool Mode: This is the default mode. The representation of the mailbox consists of the user-defined folders. Based on some user-defined rules, the incoming messages are filtered into appropriate folders. Emails that are not filtered as per the rules remain in the inbox. However, they later can be manually moved to different folders by the user. A user can calmly go through their mailbox in this mode.

3. Curious Mode: This mode is particularly designed to enable a user to dedicate their time to find, organise (or even reorganise) previously stored emails. The representation of the mailbox is completely different, where, groups of messages are shown. The summary of each group is displayed at the header along with the top 10 common words. Scatter/Gather algorithm is used and users can interactively choose a group and investigate further as per their needs.

Today, if we talk about Microsoft Outlook, seeing the threads of messages and replies on the left-hand side of the mailbox with the functionality to zoom in to any of the threaded messages comes naturally to us, however, this was not a common functionality back in the early 2000s. Venolia et al. [25] were the first ones to suggest a similar prototype improving the UI of the email client, Microsoft Outlook 2000 in order to manage the conversational threads. They identified five major activities around how people were using email, namely,

1. Flow: While people are concentrating on other tasks, they still want to be updated with the flow of incoming emails.

2. Triage: While users were away from checking their emails for a long time, they require to catch up with all accumulated emails.

3. Task Management: Users often used their inbox as "to-do" lists.

4. Archive: Users store the emails with the intention of referring to them later

5. Retrieving: Once emails are archived, users require some method to retrieve them.

Though Venolia et al. provided the study of issues related to all five activities and suggested possible solutions, still they focused on the first two activities and demonstrated the prototype for a thread-oriented display in Microsoft Outlook 2000.

Rohall et al. [21] also described visualisation techniques highlighting email messages in the form of a calendar where under each date message by the selected user would be visible. On days when there was no email exchange, such dates were omitted. Rohan et al. argue that such an interface would enable a user to scan the emails similar to how they would experience scanning piles of physical emails. It was easy to see in such an interface in which email messages

contained images or with the help of colour coding, which all messages were part of the same thread. Such visualisations would help the user quickly recognise dependencies and relationships between different messages.

Similar to Rohall et al., Gwizdka [12] also proposed a 2D interface to display the pending tasks described (directly or indirectly) by email messages. The pending tasks are automatically placed as dots on a 2D plane where the x-axis represents the task attributes and the y-axis represents the time and subject. The interface provided looks like an email task view and not a traditional inbox.

Sudarsky and Hjelsvold [22] proposed another visualisation approach to support email browsing and Boolean querying. They tried to alleviate the challenges of a large mailbox by implementing automatic clustering of emails based on the sender's email addresses. Their prototype consists of three components, namely, hierarchical view, temporal view and query interface. In a hierarchical view, they use the sender's email address to differentiate and group messages based on various types of top-level domain names. For example, all the emails with commercial enterprises are automatically grouped in a folder named as `com`, because the sender's email address would have been comprised of `.com` or all the messages from universities, would be grouped in a folder called `edu` as the sender's email address would have `.edu` in it. A Windows Explorer-like interface is used to display the folders along with the number of emails in each folder. One could expand each folder to look at the individual emails. In addition to the top-level domain view, they also provided with company view where messages are clustered using the sub-domain which consists of the company name and a sender view where messages are grouped basis on the first letter of the sender's name. This automatic clustering forming the hierarchical view helps the user to have a high-level view of their inbox. Complementing the hierarchical view, a temporal view is also available. The temporal view consists of a 2D grid with the x-axis representing time and the y-axis representing the list of senders. When a user would select a node in the hierarchical view, a square icon with the number of messages appears on the timeline. To search more efficiently, a user could narrow down the search space in a hierarchical view. While searching for a particular subject, if that subject is not present in particular messages, then the square icon will be represented as grey-coloured indicating no match. Sudarsky and Hjelsvold also enabled the Boolean search functionality. Conversational threads were also visualised.

In 2003, another interesting layout to display Microsoft Outlook 2000, not just as a mailbox with the left-hand side pane displaying the folders and the right-hand side displaying the inbox messages, Bellotti et al. [5] redesigned a user's traditional email experience. They developed *"Taskmaster"* with the focus on not just displaying email messages but also exhibiting information about task and project management. By performing some initial studies and interviews with various participants, they identified specific problems related to managing tasks in emails. For example, users had to keep track of email threads manually or they had to manage deadlines and reminders which could be associated with particular message content. In their prototype, they called tasks as *"thrasks"*, i.e. threaded tasks. Basically, rather than finding similarities between emails, they grouped emails based on factors like the reply to an email. So threads of an email were grouped together. As displayed in Figure 2.1, Taskmaster's layout was divided into three parts. The top pane was responsible to act as a "to-do" list, which included the new messages at the bottom of the pane. The middle pane was a list view of selected thrask content like attachments and URLs. The bottom pane previewed the content of an individual selected item. In this approach, the folders of threaded messages were optimised as reminders along with archiving of ongoing tasks. When earlier drafts were saved in a separate folder, where the users might forget their reference to the original message, Taskmaster enabled the users to save the draft messages in the middle pane encouraging extended-response messages. Other visual features included warning bars, which changed to red when the deadline was approaching and

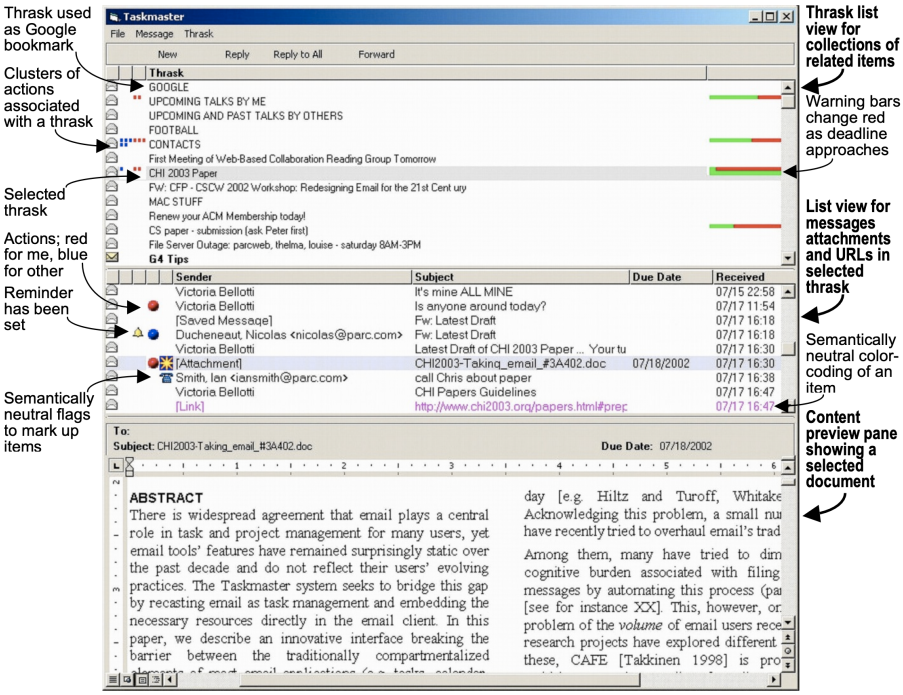


Figure 2.1: Taskmaster's user interface

action clusters represented with miniature balls which indicated the number of actions to be taken either by oneself or another. Their results showed that users were happy and satisfied with the redesign of the classical email client to a task management tool. The current popular email clients like Outlook, Gmail or Apple Mail use a similar user interface approach to group messages based on metadata like, reply to and forward messages. Rather than showing an individual message, it is shown as a part of a thread, which enables a user to better relate it with the ongoing context. Threads are now the rudimentary organising element for email management.

While the research to improve the visualisation of the mailbox was going on and various improvements were taking place, simultaneously, experimentation in the field of email classification into different folders based on content and structure was also progressing. A novel approach to extract the structure and content of emails from pre-defined folders and use that for email classification was proposed by Aery and Chakravarthy [1]. They pointed out that text classification differs in documents and in emails as documents are richer in content which makes the identification of topics easier. Hence, their approach was based on the assumption that every folder consists of representative emails, which could be utilised to extract common and recurring structures through graph mining techniques which would further automate the classification of incoming emails. The general strategy adopted was to first pre-process the emails present in already existing folders, like the removal of stop words, ranking of words based on their frequency etc. Next, a canonical representation in the form of a star graph was used to represent the emails. The central vertex was the broad concept like, header or body, and its edge labels consisted of metadata like from, to or subject and contains draft or final. Further, using graph mining techniques, some substructures of representative emails were extracted. As the number of substructures would be large, the basis on the identification of subsets required for discriminating incoming emails, some substructures would be retained and the rest would be pruned.

Based on ranking, the best 'n' substructures were selected as representative substructures and each incoming email was compared with the selected 'n' substructures for all folders. The email was then filed to the folder which contained the highest-ranked substructure. Though the performance of the system, eMailSift was compared with traditional models like Naive Bayes and the effect of folder or training set size was studied, however, there was no user acceptance testing done for the system. eMailSift was about proposing innovative research for the classification of emails using graph mining techniques.

Taking a step further from this novel and innovative solution for email classification, Koren et al. [14] tried categorising emails which were not based on a single user's foldering practice but they considered and shared information regarding foldering techniques between various users. They collected 200 million foldered messages from anonymised Yahoo! Mail users. Due to privacy concerns only the subject and sender fields of each message was analysed. From the large number of foldered emails, 6000 tags were extracted. From these 6000 tags, 819 non-singleton clusters containing synonymous tags were identified and each cluster was given the name of the most frequently used synonym. Additionally, some tags (like personal names, offensive tags or non-English tags) were removed which were not a good fit for the global learning task. The number of tags was lowered down from 6000 to 2000. Each email i.e. its subject and sender field, was then represented as a set of features. Some standard information retrieval processes like the creation of a vector space (tf-idf) representation followed by the application of various classifiers like Naive Bayes, FlatNormalizedHinge, HierNormalizedHinge were performed and the performance of each was compared. HierNormalizedHinge was the chosen classifier which would shortlist a number of suggested ranked tags for each email message. Further, in a new phase, the system would decide how good a tag is for each email. Out of the suggested tags, the system decides which tags to keep and which should be filtered out. They trained independent binary classifiers on all the email messages and suggested tags. At last, they trained a binary classifier for each tag by contrasting emails labelled by the tag with unlabelled email messages. Though this system proposed by Koren et al. [14] was rigorous, sophisticated and time-sensitive, however, they only considered the subject and from the field, dismissing the content which might have caused different results.

Another weakness of Koren et al.'s [14] research was that with the wide range of tags, their approach would vary the level of abstraction or specificity. For example, a tag like 'amazon' could overlap with related tags like 'shopping'. However, moving forward to 2014, based on Ailon et al.'s [3] study stating that 60% of the traffic on Yahoo! Mail was machine-generated, Grbovic et al. [11] proposed to differentiate between personal and machine-generated messages for generic classification into the mentioned categories. There could be a clear distinction in semantics and syntax between machine-generated emails and human-written emails. They focused on the fact that due to a huge number of machine-generated messages having similar structures, the data would be less sparse leading to more precision in the classification techniques. To test this hypothesis, they collected a large data set consisting of Yahoo! Mail's email traffic for 6 months which was about more than 500 billion email messages. The content and subject along with other metadata such as from field were used for the experiments. However, for guidelines related to the privacy of users, they included the content of emails only for users who voluntarily opted-in and were made sure that the content was not being examined by humans. Next, they used Latent Dirichlet Allocation (LDA) approach to find a set of topics from various folders generated by 40 million users. To find the correct number of K, i.e. the number of topics from the collected set of folders, they iterated over various values ranging from $K = 5$ to $K = 50$. They made sure that each topic K would cover at least 50% of the total amount of emails which were labelled. They noted that with a higher value of K, the topics that were being covered were niche down to very specific domains like Cooking, and Astrology with a small coverage of labelled

emails. They tested and obtained $K = 6$ as an ideal value. The 6 common topics from all the labelled emails came out as 'human', 'career', 'finance', 'shopping', 'social' and 'travel'. The topic 'human' consisted of emails having personal communications. Further to reduce more sparsity of different email contents and as an efficient feature extraction method, they aggregated the email messages on the domain level. For each email, three main features were extracted namely content features, address features and behavioural features. Content features were extracted from the subject and body, address features were extracted from the sender's email address which included the subdomain information like `.edu` or `.gov`, and behavioural features were extracted from the actions taken by the sender and recipient on a given email message. After the feature extraction, to create a well-labelled training dataset with the labels as the 6 common topics along with an additional label distinguishing between the human and machine-generated emails, they used 3 techniques: manual labelling, heuristic labelling and automatic labelling. With the privacy concerns on reading personal data, only 5 paid and responsible editors were hired who skimmed 18,000 messages coming from 14,000 users. Around 400 inconsistent labels were then created. For heuristic labelling, where rules are obtained from real-world knowledge they identified facts such as human senders would follow patterns like <first name>.<last name> in their email id whereas machine-generated emails would usually have keywords like mailer-daemon or no-reply present in the email id. Using heuristic labelling, they gathered a data set containing 80,000 corporate machine senders and 60,000 human senders. For automatic labelling, they applied folder-based Latent Dirichlet Allocation (LDA) voting and majority voting and provided a label for 210,000 senders. Next, for the classification they used both online and offline approaches. The online mechanism is an incremental machine learning mechanism where a model is evolved over time as new data becomes available to it. Their online system was further divided into three stages, first was lightweight classification which would use some hard-coded rules to quickly classify a considerable amount of email traffic. In the next stage as sender-based classification, the process included searching for a sender in a lookup table which comprised of a list of senders along with their known categories. The last stage was the final online heavy-weight classification. The email traffic which could not have been classified in the second stage, i.e. the sender could not be located in the lookup table, only those emails were sent to the third stage. This traffic which could not be covered by the second stage was around 8%. For this small amount of traffic, they could actually afford heavier computation and used all relevant features from the email subject, content and sender's name to classify an email. However, the topic 'human' which was discovered as a common topic in labelled email, was taken aback as it has highly subjective folders. Therefore, for the experiments and evaluation, the authors used 5 sender-based labels as 'career', 'finance', 'shopping', 'social' and 'travel' along with 1 label distinguishing between human and machine-sender. To provide an unbiased and fair test data set, they knit picked around 6000 human-written email messages and took 35% of the total machine-generated email messages. They implemented Logistic Regression Vowpal Wabbit[1]' implementation due to its high scalability. For their test cases, the model performed pretty well with Recall and Precision being 0.9 in most of the cases. They achieved more than 85% of the true positive rate. Not only their classifier performed extremely well, but they also uncovered some interesting facts like, 34.5% of machine-generated emails had subjects with a character count of more than 30, whereas only 5.8% of human-generated emails had such long subjects. Similarly, 77.5% of machine-generated emails have more than 300 characters, whereas only 1% of human-written emails were this long. Also, 41% of machine-generated emails had 3 URLs in the content, however, the same number of URLs was not found in the email content written by human senders.

[1] `https://github.com/VowpalWabbit/vowpal_wabbit`

An interesting fact to be mentioned here would be that during this time only, in June 2013 (as per Wikipedia[2]), Gmail had officially launched the usage of 5 tabs namely, `Primary`, `Promotions`, `Forums`, `Social` and `Updates` to automatically categorise emails in these 5 general and broad categories. This could be beneficial for the users to better maintain their inbox. For example, they could select all the emails under one category and if they seem to be of no importance, then delete those at once.

Boryczka et al. [8] presented a new approach to automatically categorise email messages into different folders by applying *Ant Colony Decision Tree* (ACDT). Boryczka et al. [7] in previous research had presented an optimised Decision Tree algorithm. To optimise the decision tree algorithm, they used Ant Colony Optimization (ACO). Decision trees are commonly used for data mining tasks in classification and Ant Colony Optimization (ACO) is a meta-heuristic algorithm motivated by the behaviour of ants which live in colonies and how they try to find an optimal path between their colony and food. In this research for the classification of emails, they used the same Ant Colony Decision Tree approach, along with some elements of *Social Network Analysis* (SNA). SNA is a field of study in Sociology which explores connections and social structures by examining the interactions and relationships between individuals or groups. It uses mathematical models and network theory to map and assess social networks. Boryczka et al. [8] transformed the publicly available e-mail data set, *Enron*[3] to a decision table along with analysing the list of recipients as part of analysing the communication network. They formed the decision table with data like the sender's name, a boolean value indicating if someone else was marked in the carbon copy (CC) field or not, length (number of characters) in the message body including white spaces and the first three words of the subject. They included the carbon copy (CC) information in order to inspect the communication network between the email ids of all the recipients and the sender. In their modified version of Ant Colony Decision Tree (ACDT), the list of recipients would have an impact on the decision class, the classifier will so select. They also analysed the frequency by which users would send emails to each other using SNA, which further impacted the classifier to choose the decision folder. They trained their algorithm and tested it on an unseen dataset (which was basically a subset of 10 mailboxes from Enron dataset). Their algorithm achieved better accuracy as compared to traditional algorithms like, Naive Bayes. From the experiments on the Enron dataset, they proved that their proposed approach of including SNA elements in ACDT significantly improved the task to classify emails into different folders.

Another interesting study and the main motivation behind our thesis was the research conducted by Park et al. [19] in which they had investigated in detail regarding the automation needs of today's users which are currently not possible in popular email clients. To explore the gap between the automation required by a user and the automation provided by the present email clients, they conducted a series of 3 different need-finding examinations.

The first examination method was an open-ended survey. Before forming the survey, they identified common categories of needs by organising a formative workshop with 13 Computer Science students as the participants. They identified some common needs like 'automated content processing', 'attention management and prioritisation' or 'filing and labelling'. Based on these needs, the questionnaire for the survey was prepared. 77 participants from different fields of study took part in the survey. The results from the open-ended survey were broadly divided into the following 6 automation needs:

1. *Richer Data Model:* As of now, the attributes provided by the email clients are limited in the form of sender, receiver and date. The participants in the survey expressed the need to include latent data associated with the email such as the priority of the email, if there

[2] `https://en.wikipedia.org/wiki/Gmail_interface`
[3] `https://en.wikipedia.org/wiki/Enron_Corpus`

is a need for any reply to the message or not and if there is any deadline associated with the email or not.

2. *Context Sensitive Rules:* Here, the context is in reference to the inbox. It could be *internal* or *external* context with respect to the inbox. As an example, a participant defined a rule as, "If the recipient hasn't replied for n days, send them a reminder". This is an example of an internal context rule. It depends on the state of the inbox. Whereas, a rule like, "Don't notify emails from these campus mailing lists during my summer vacation", depends on external factors like time or location related to a user.

3. *Attention Management:* These automation needs were related to efficiently managing a user's attention by levering the notification behaviour in email clients. Rules defined by users here were related to pushing notifications on the basis of time or on the accumulation of a certain number of emails.

4. *Sender Affordances:* Such rules were associated with sending emails to the right person at the right time. Examples demonstrated by some participants were, "Send email only to people in the lab right now because I need a staple" or "For event announcement emails, self-destruct the emails from recipients' inbox after the event".

5. *Altering Inbox Presentation:* Participants expressed to have different visualisations for email threads. They wanted to divide large email messages and group together small email messages.

6. *Content Processing and Aggregation:* These automation requirements were related to extracting content from the email and then taking appropriate action on the same. For example, extracting an event's date and place from an email's subject and adding it automatically to the calendar.

During the second probe, Park et al. examined the existing codes present on GitHub (an online code repository) to understand how programmers try to customise their inbox as per their needs. After filtering all the scripts present on GitHub, they came down to 500 scripts. They studied all those scripts and noted down 8 major categories which the scripts were trying to solve. Some most common categories were,

1. Triggering actions based on content: Most of the scripts included the extraction of content from the email body and then taking actions like sending an appropriate reply or forwarding the content of other users.

2. Altering Inbox Presentation: Next most common scripts were related to altering the inbox visualisation. These scripts were trying to mimic the view of social networking platforms in the inbox.

3. Inbox Zero: Many scripts were related to cleaning inbox by spam removal. These scripts were aimed at having near to 0 emails in an inbox at any given time.

Once they studied about the user's automation needs and scripts present online, they designed and deployed *YouPS*, the email automation tool. YouPS was basically a web interface where users could write their own automation rules using Python syntax. Those email rules were then executed on the content of IMAP API mailboxes. YouPS provided a Python environment to the user with some pre-defined methods such as `get_history(contact)` which would return back the interaction history with the provided contact name or `set_email_mode(my_email_mode)`

where a user could define their own email mode, such as a vacation mode to restrict incoming emails for a certain time period. These methods including `get_history(contact)` and `set_email_mode(my_email_mode)` were designed specifically from the automation needs discovered in the survey phase. For the evaluation, they asked 12 programmers to test the application YouPS for a week by defining their own rules. As a result, the programmers demonstrated that they were able to implement 40% of the rules of their choice. They could not implement the rest due to the complications in the current email filters and the lack of additional attributes provided by the email clients.

This research by Park et al. helped us to understand the state of the art in the area of email automation. We studied how users want to gain more control over how their inbox should look at different points in time and how email clients lack such automation in the current scenario. One particular thing that we noticed in this research was that the participants belonged to the Computer Science field. To define their own rules in the YouPS system, they indeed required the knowledge of Python. Also, mostly the automation rules discussed here were related to how and when should a user receive emails.

From all our research we understood that we may not be able to play around with how the inbox looks but one thing was clear, users do want to manage and organise their inbox. To date, a lot of studies have been performed to differentiate between spam and non-spam emails, however, they do not really help the user to declutter their inbox. Other than spam filters, Gmail has provided additional classifications as 'Social' or 'Primary'. Further, to the best of our knowledge, no recent survey or study has been performed acknowledging the current user's email practices.

From an initial informal conversation with friends and family, we noticed that the problem of email (inbox) overload still exists and the automation solution to help the users achieve *"Inbox Zero"* is somewhere lacking. We planned to create a recommendation system for an existing email client which could suggest users to move emails from their inbox to different folders. However, to get a better understanding of the current user's email practices to manage their inbox and to get some insights like which is the most popular email client and if users would like our solution in the form of a recommendation system to declutter their inbox or not, we first conducted a survey. In the following chapter, we discuss the questionnaire and the results of the survey in detail.

Chapter 3

Survey

In this chapter, we discuss the survey that we conducted in order to better understand current email practices.

3.1 Survey Method

We prepared a questionnaire containing a total of 31 questions[1]. Typically, not all 31 questions were presented to a participant. The sequence of questions was dynamically composed, depending on the participant's answers to the previous questions. Out of the 31 questions, 24 were multiple choice questions, i.e. participants had to select an option from the given choices as their answer, 1 was a slider scale question where participants had to rate on a scale using a slider control and 6 were descriptive questions, where participants could express their views in written form. We started with the question of whether a participant uses emails or not. If they do use emails, they were asked more about their day-to-day usage of emails and if not, they were simply asked the reason for not using emails along with some demographic questions.

For the participants who chose to be using emails, the survey was broadly divided into six sections. The first section contained general questions related to emails like which email client they use, on which device do they mostly use emails and so on. The second section consisted of questions like how the participants use and maintain their inbox. This section had questions like, how many emails they have in their inbox and whether they tag their emails or make folders to manage emails. The third section was dedicated to questions related to creating and using folders to manage their emails. Questions like how many folders they have and how often do they move emails to different folders were asked in this section. The fourth section included questions related to rule-based filtering present in today's email clients. The fifth section contained questions to understand if participants would like to have a recommendation bar in their email client to filter their inbox or not. The last section contained some demographic questions. We purposefully placed the demographic questions at the end of the survey as these were not the most interesting questions and we wanted to grab the attention of the participant for questions related to their email usage at first.

The survey was published on *Qualtrics*[2]. The confidentiality of answers was maintained by anonymising the responses in accordance with General Data Protection Regulation (GDPR) standards.

[1]For a detailed look, the questionnaire is attached in Appendix A at the end
[2]https://www.qualtrics.com

To invite participants for our survey, we used social media channels like LinkedIn, Facebook and Twitter, where we posted about the survey and requested the audience to fill it in. We received a total of 244 responses. The complete dataset of our survey has been published and made publicly available on Zenodo[3], providing a valuable resource for researchers and professionals interested in email management and information organisation. However, while analysing the dataset we observed that 11 responses out of the 244 responses were incomplete. We observed that these 11 participants had only answered the first question asking whether they use emails or not. They had answered *"yes"*, however, did not fill out the rest of the survey. Hence, we did not consider these responses for our analysis. For the remaining 233 responses, the average time taken to fill out the survey was between 7 to 10 minutes. The detailed study and analysis of the complete 233 responses are presented in the following sections.

3.2 Survey Analysis

In the following section, we provide a detailed analysis of the survey.

3.2.1 Section 1: General Questions Related to Email Usage

Section 1 consisted of 4 questions where we asked the participant, if they use emails or not, which device they mostly use emails on, which email client they usually use and for what purposes (personal or work-related) do they utilise the email functionality?

From the 233 responses, 230 participants chose to be using emails and only 3 were not using any emails. One of the 3 participants who chose not to be using emails mentioned that they feel email is an outdated and insecure technology. They preferred using decentralised chatting applications as a safe and encrypted alternative to email. However, they did mention using emails as a formal way of communication.

Next, as represented in Figure 3.1, out of the 230 participants, 127 participants (55%) were mainly using emails on big screens like laptops or desktop computers and 101 participants (44%) of the total were using mobile phones to manage their email accounts.

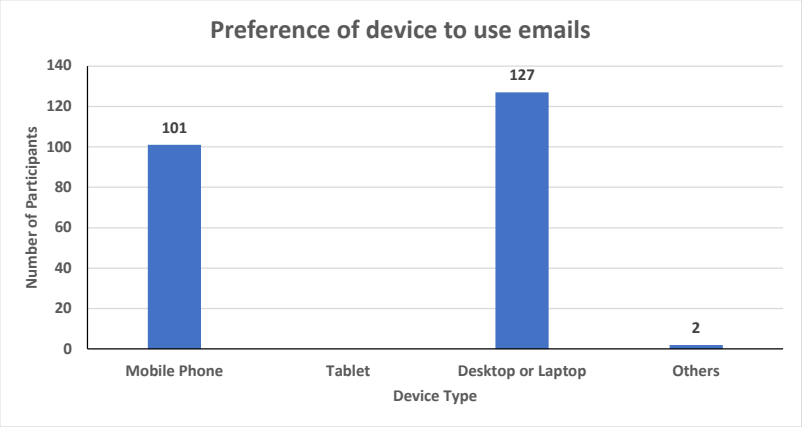


Figure 3.1: Big screens like desktops or laptops were preferred by the participants

[3] https://zenodo.org/record/8028567

Gmail came out to be the most popular email client service with 186 out of 230 participants (81%) using it. Following it was Outlook, which is used by 123 participants (53%) out of the total participants. Other email service providers like Apple Mail, Thunderbird, Yahoo! Mail, Telenet webmail and Spark constituted a small number of users using it as illustrated in Figure 3.2. Note that for this question, a user could select more than one email client as we believe a user could have email accounts with more than one email service provider.

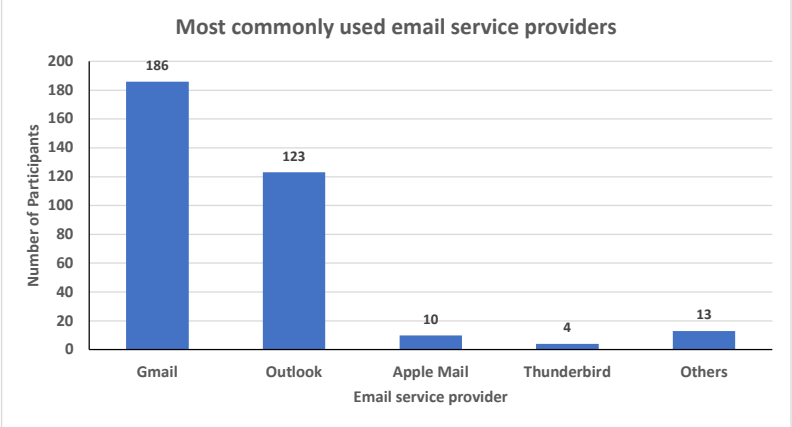


Figure 3.2: Gmail is the most popular email service provider

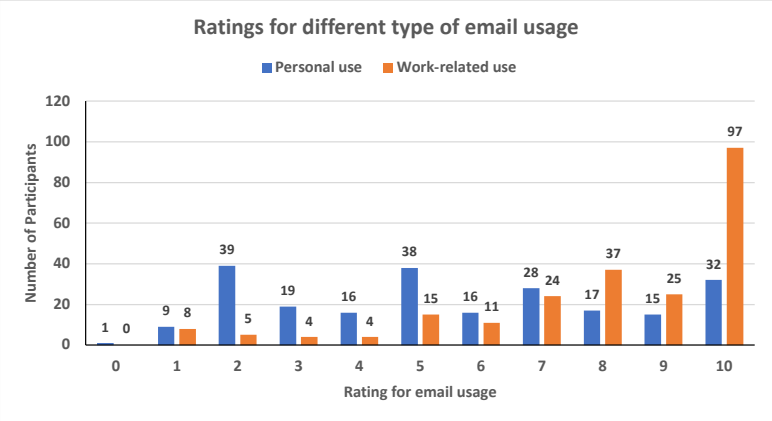


Figure 3.3: Mixed responses for emails used for personal purposes. However, most of the participants indicated to be using emails for work-related purposes.

Further, to evaluate the purpose of using emails in daily life, we had a question with 2 sliders as options, each slider ranging from 0 to 10 where 0 meant no use at all and 10 meant a lot of usage. Using one slider, users could select the amount of usage of emails in their personal life and using the other they could select the amount of work-related usage. We did not want to have this question as a black-and-white question where users could only select one of the two answers because, from our experience, we know that emails are part of personal as well as work life. Hence, we had a slider to mark the relevance of emails in both phases, personal and work-related. We received mixed responses for personal usage with a mean of 5.57. However,

most of the participants agreed to use emails a lot for work-related purposes. Hence, we received the mean for work-related usage as 8.03 on a scale of 0–10.

Figure 3.3 shows that 97 participants said to be using emails a lot (10 out of 10) for work-related purposes. However, there was a mixed response on using emails for personal purposes. 39 participants rated 2 out of 10, 38 participants rated 5 out of 10 and 32 rated 10 out of 10.

3.2.2 Section 2: Inbox Management

Section 2 consisted of 5 questions, where we asked the participants, how often they check their emails, how many emails they approximately receive in a day, how many emails approximately are there in their inbox, how they use their inbox and how they manage emails in the inbox.

We observed that 142 out of 230 participants (62%), check their emails several times a day and 37 participants (16%) check their emails several times an hour. 179 (142 + 37) participants check their emails at least several times a day. 40 participants (17%) check their emails once a day. Only a small fraction, i.e., 11 out of 230 participants (5%) check their emails once a week as shown in Figure 3.4.

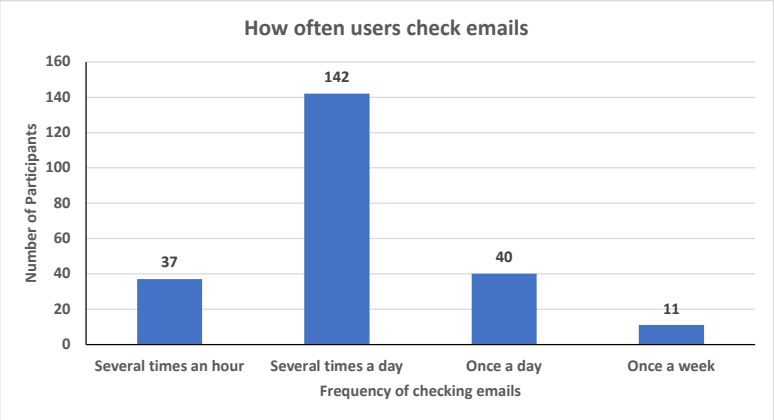


Figure 3.4: 179 (142 + 37) participants out of the total population (78%) indicated to be checking emails several times a day

Further, 108 out of 230 participants (47%), would receive 11–50 emails in a day and around the same number 103 participants (45%) would receive 0–10 emails in a day as highlighted in Figure 3.5. Though, not many but still 15 participants (7%) reported to be receiving 51–100 emails a day.

Interestingly, we additionally observed that out of the 15 participants who said to be receiving 51–100 emails a day, 10 claimed to be using emails a lot for work-related purposes by marking a 10 on the slider when we asked for what major purposes they use emails. Among the remaining 4 participants, 2 rated the use of emails for work at a 7/10, while the other 2 rated it at a higher 9/10. However, 1 participant said the usage of work-related email was 5/10. A small fraction, 4 out of 230 participants (2%) reported to be receiving more than 100 emails, out of which 2 agreed to be heavily receiving emails for work purposes (10/10) and 2 agreed to be depending on emails as 7/10 for work purposes. Briefly, we can say that receiving a huge number of emails can strongly be related to using emails a lot for work-related purposes.

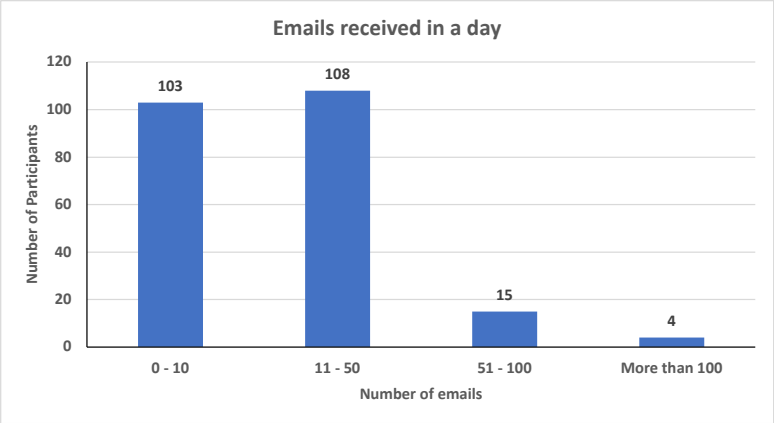


Figure 3.5: Most participants receive less than 50 emails a day

Trends between size of inbox and inbox maintenance practices

Next, to inspect the general size of the inbox and email management practices for maintaining the inbox, we had 3 co-related questions. We started by asking about the number of emails present in a user's inbox which could be 0–100, 101–500 or more than 500 emails. This was followed by a question on how they use their inbox with the options as, do they consider their inbox as a to-do list and regularly move emails from their inbox to manually created folders or do they occasionally move emails or do they *not* move emails at all? The third co-related question was how they manage emails in their inbox where a user could select more than one option from either they move emails to manually created folders or they use tags to manage emails in their inbox or they leave emails as is in the inbox. We also provided a descriptive field to let us know if they have some other strategy of managing emails in the inbox.

At first, we observed that 144 out of 230 participants (63%) had more than 500 emails in their inbox, 45 participants (20%) had 101–500 emails in their inbox and 41 participants (18%) had 0–100 emails in their inbox as illustrated in Figure 3.6.

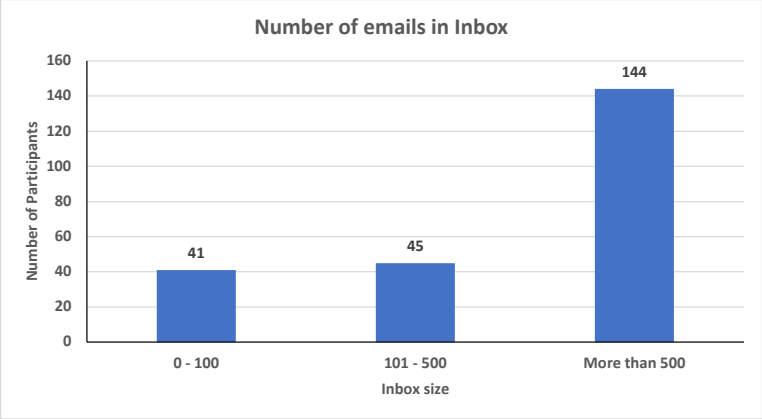


Figure 3.6: 144 of the total participants (63%) said to be having more than 500 emails in their inbox

Further, we divided the responses into three groups based on the number of emails present in the inbox. For ease of use, we write Inbox Size instead of writing 'the number of emails present in the inbox'.

1. *Inbox size: > 500*
2. *Inbox size: 101–500*
3. *Inbox Size: 0–100*

We studied in detail about each category of the inbox as per the number of emails contained in them. For each category, we first studied how a participant uses their inbox based on the following options:

1. use their inbox as a to-do list and regularly move emails from inbox to folders
2. occasionally move emails from their inbox to folders
3. leave their inbox as it is and do not move any email at all
4. follow some other email management practice

Once, we knew about a participant's usage of the inbox, we examined their email management practices. From this exercise, we were trying to find links like, whether a participant chose to be treating an inbox as a to-do list, then we would expect their email management practice as creating and maintaining folders.

Trends for Inbox Size: > 500

144 out of 230 participants, had more than 500 emails in their inbox. As shown in Figure 3.7, 17 out of these 144 participants claimed to be using their inbox as a to-do list and keeping only those emails in their inbox which require some action, or else they would move emails from their inbox to other folders. 28 out of these 144 participants claimed to be occasionally moving emails from their inbox to manually created folders. 98 out of these 144 participants agreed to not moving emails from their inbox to manually created folders. 1 out of 144 participants said to be following the strategy to only delete spam emails and let the other emails be in the inbox.

To-Do List: 17 participants (12%) said to be having more than 500 emails in their inbox and using their inbox as a to-do list by moving emails from their inbox to other folders. However, in the following question which was related to email management practice, only 12 out of these 17 participants agreed to be moving emails into folders and 5 contradicted the fact that they would indeed not move any email from their inbox. Hence, we can conclude that only 12 out of the total 230 participants have more than 500 emails in their inbox even though they regularly move emails from their inbox to other folders and consider their inbox as a to-do list.

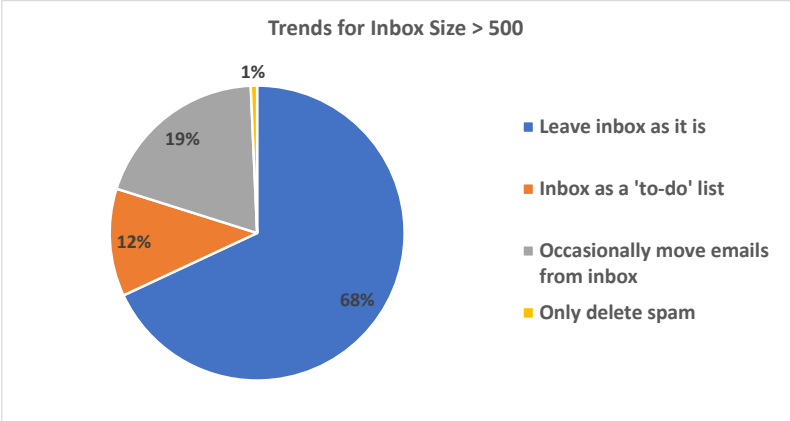


Figure 3.7: Most of the participants who had more than 500 emails in their inbox said to be leaving their inbox as it is without moving any emails from their inbox to other folders

Occasionally move emails: 28 participants (19%) said to be having more than 500 emails in their inbox when they occasionally move emails from their inbox to other folders. 20 out of these 28 participants agreed to be moving emails into folders in the following email management practice question. However, 4 contradicted on the part that they would indeed not move any email from their inbox. Hence, we can conclude that only 20 participants out of the total 230, have more than 500 emails in their inbox when they occasionally move emails from their inbox to other folders.

Leave inbox as it is: 98 participants (68%) said to be having more than 500 emails in their inbox when they do not move emails from their inbox to other folders. In the following question which was related to email management practice, 71 of these 98 participants exclusively said to be leaving the inbox as it is, additionally, 13 out of 98 said to be tagging emails in the inbox itself. However, 6 contradicted by first selecting to leave their inbox as it is and then selecting the option for moving emails from their inbox to folders in the email management question. There were another 4 respondents who selected both options to be moving emails as well as leaving their inbox as it is in the same question of email management practice. For these 4 cases, we can consider that these users do practice foldering technique, though it might be occasional.

Trends for Inbox Size: 101–500

45 out of 230 participants had 101–500 emails in their inbox. As shown in Figure 3.8, 6 out of these 45 users claimed to be using their inbox as a to-do list and keeping only those emails in inbox which require some action, or else they would move emails from inbox to manually created folders. 11 out of these 45 participants claimed to be occasionally moving emails from their inbox to manually created folders. 27 out of these 45 participants agreed to not moving emails from inbox to manually created folders. 1 out of 45 participants said to be using Personal Knowledge Management (PKM) practice and organising their emails in Google folders to reduce clutter in their inbox.

To-Do List: 6 participants (13%) said to be having 101–500 emails in their inbox and using their inbox as a to-do list by moving emails from their inbox to other folders. However, in the following question which was related to email management practice, only 2 out of these 6 participants agreed to be moving emails into folders and 4 contradicted the fact that they would indeed not move any email from their inbox. Hence, we can conclude that only 2 participants

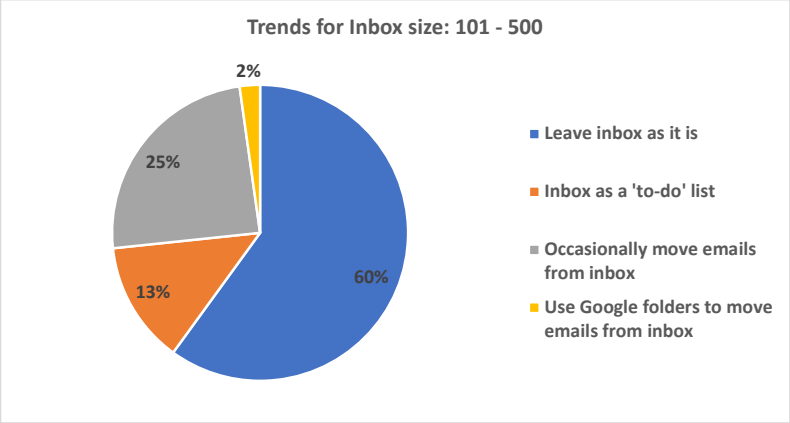


Figure 3.8: Most of the participants who had 101–500 emails in their inbox said to be leaving their inbox as it is without moving any emails from their inbox to other folders

out of the total 230, have 101–500 emails in their inbox even though they regularly move emails from their inbox to folders and consider inbox as a to-do list.

Occasionally move emails: 11 participants (25%) said to be having 101–500 emails in their inbox when they occasionally move emails from their inbox to other folders. 10 out of these 11 participants agreed to be moving emails into folders in the following email management practice question. However, one mentioned to be tagging emails. For this one participant, we assume that they might not have selected the option to move emails into folders accidentally, as the further responses by the person indicated that they create and maintain folders. Hence, we can conclude that only 11 out of the total 230 participants have 101–500 emails in their inbox when they occasionally move emails from their inbox to other folders.

Leave inbox as it is: 27 participants (60%) said to be having 101–500 emails in their inbox when they do not move emails from inbox to other folders. In the following question which was related to email management practice, 21 of these 27 participants exclusively said to be leaving the inbox as it is, additionally, 3 out of 27 said to be tagging emails in the inbox itself. There were another 3 respondents that selected both options to be moving emails as well as leaving inbox as it is in the same question of email management practice. For these 3 cases, we can consider that these users do practice foldering technique, though it might be occasional.

Trends for Inbox Size: 0–100

41 out of 230 participants had 0–100 emails in their inbox. As shown in Figure 3.9, 20 out of these 41 users claimed to be using their inbox as a to-do list and keeping only those emails in their inbox which require some action, or else they would move emails from their inbox to manually created folders. 9 out of these 41 participants claimed to be occasionally moving emails from their inbox to manually created folders. 12 out of these 41 participants agreed to not moving emails from their inbox to manually created folders.

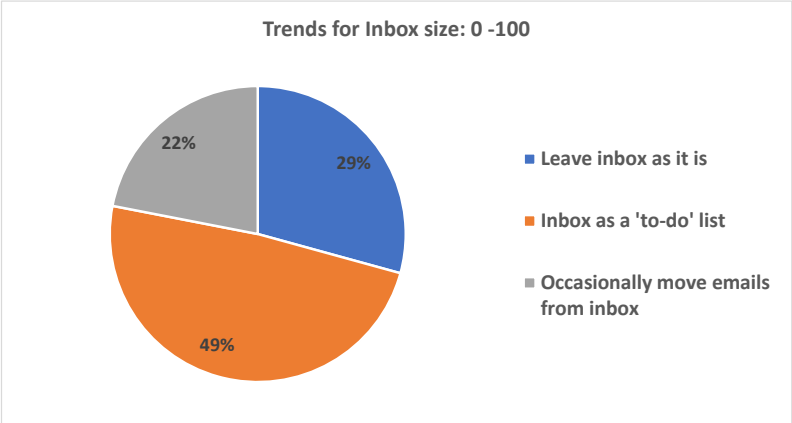


Figure 3.9: Most of the participants who had 0–100 emails in their inbox said to be using their inbox as a to-do list and only keeping those emails that require some action in the inbox, else they would move emails from their inbox to other folders

To-Do List: 20 participants (49%) said to be having 0–100 emails in their inbox and using their inbox as a *to-do* list by moving emails from their inbox to other folders. Interestingly, 17 out of the 20 participants agreed to be moving emails from their inbox to folders, 2 said to be using tagging technique and 1 particularly mentioned customising rules in their email client to move specific emails, otherwise, they would manually move emails from their inbox to different folders. Hence, we can conclude that only 20 out of the total 230 participants have 0–100 emails in their inbox when they regularly move emails from their inbox to folders and consider their inbox as a to-do list.

Occasionally move emails: 9 participants (22%) said to be having 0–100 emails in their inbox when they occasionally move emails from their inbox to other folders. Though 8 out of these 9 participants agreed to be moving emails into folders in the following email management practice question, however, one said to be tagging emails. For this one participant with this small number of emails in their inbox, we assume that they might be deleting the unnecessary emails as a decluttering strategy. Hence, we can conclude that only 8 out of the total 230 participants have 0–100 emails in their inbox when they occasionally move emails from their inbox to other folders.

Leave inbox as it is: 12 participants (29%) said to be having 0–100 emails in their inbox when they do not move emails from inbox to other folders. Looking at the small number of emails in their inbox along with 2 out of these 12 users explicitly mentioning that they delete emails, we assume that all these 12 users might be deleting the unnecessary emails as a de-clutter strategy to manage their inbox.

3.2.3 Section 3: Email Management using Folders

Section 3 consisted of 5 questions, where we asked the participants, how many folders they have in their mailbox (including default folders like *spam* or *archive*), whether they manually create folders in their mailbox, how often they create folders, how often they move emails from their inbox to other folders and how much time (on average) they spend every day to manually sort their inbox.

A general question of how many folders are there in the user's current inbox was shown to *all* the participants who said to be using emails. Here we asked the participants to also include the default folders like junk mail in the count of folders. We wanted to have an idea of the average number of folders in a user's mailbox.

As illustrated in Figure 3.10, 159 out of 230 participants had 0–10 folders, 50 participants had 11–50 folders, 9 participants had 51–200 folders and 12 participants had more than 200 folders.

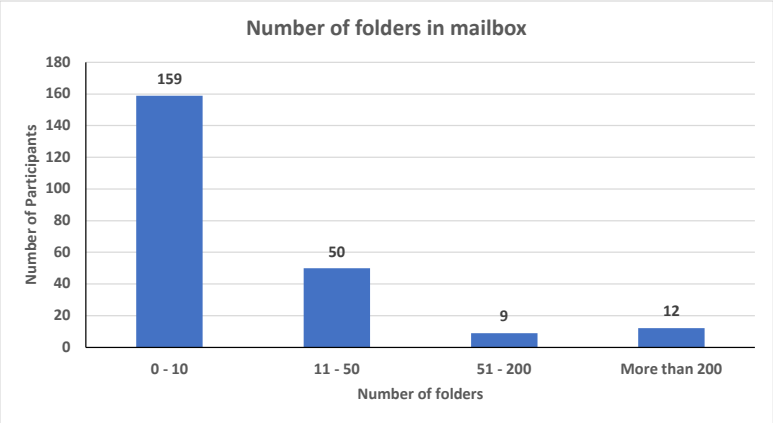


Figure 3.10: 159 of the total participants (69%) said to be having 0–10 folders in their mailbox

Trends between the number of folders and email management practices

Now, we wanted to study a bit more about how the number of folders can be associated with the different email management practices. For example, we would expect that those who have a large number of folders say more than 51, would be regularly creating folders and moving emails from inbox to those folders. To explore this in detail, we divided the participants into 3 groups. The first group contained 159 participants who had 0-10 folders, the second group contained 50 participants who had 11–50 folders and the third group contained 21 participants who had more than 51 folders in their mailbox.

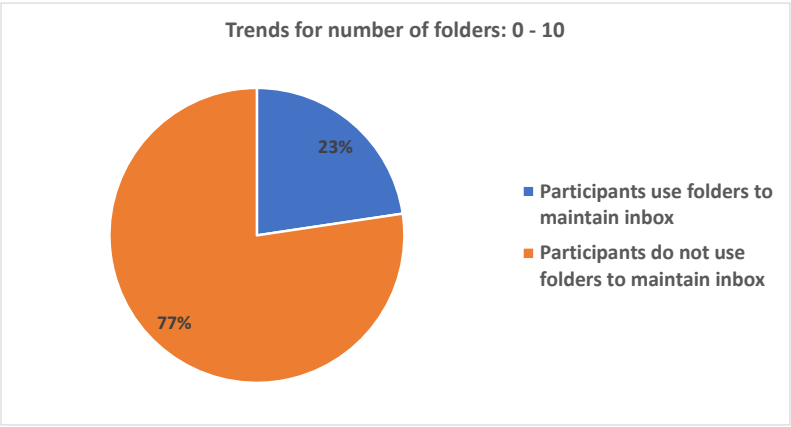


Figure 3.11: Most of the participants who had 0–10 folders in their mailbox indicated not using folders to manage their mailbox

Trends for number of folders: 0–10

For 159 participants who had 0–10 folders, we observed that 36 out of these 159 participants (23%) were maintaining their inbox by moving emails to different folders. Most participants (33 out of 36) said to be creating folders once a month. Out of the remaining 123 participants (77%), most said to be leaving their inbox as it is and some said to be tagging or deleting emails to maintain their inbox. As shown in Figure 3.11, about 77% of the 159 participants do not use folders to manage their inbox, it made sense to have fewer folders if they do not move emails from their inbox to different folders.

Trends for number of folders: 11–50

For 50 participants who had 11–50 folders as shown in Figure 3.12, 39 out of these 50 participants (78%) were maintaining their inbox by moving emails to different folders. they mostly said to be creating folders once a month. Intuitively, we can say that if someone creates folders once a month, we would expect a lower number of folders in their mailbox. Out of the remaining 11 participants, most said to be leaving their inbox as it is and some said to be tagging emails to maintain their inbox.

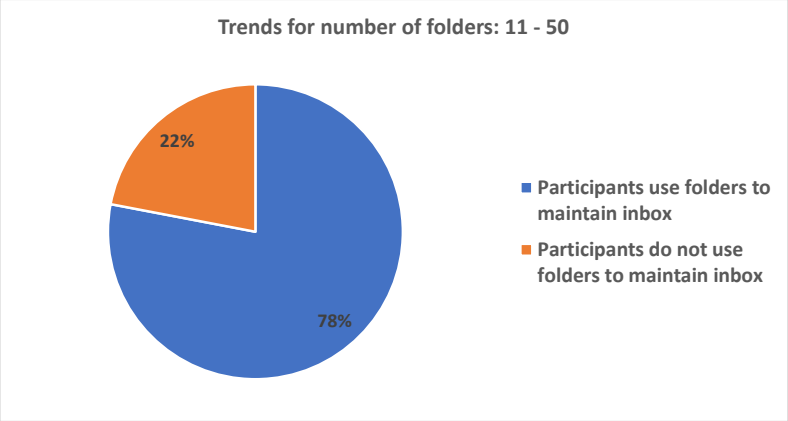


Figure 3.12: Most of the participants who had 11–50 folders in their mailbox agreed to use folders to manage their mailbox

Trends for number of folders: > 50

For 21 participants who had more than 50 folders, we observed that 10 out of these 21 participants were maintaining their inbox by moving emails to different folders. There were 11 contradictory responses, which said to be leaving their inbox as is and some said to be tagging emails to maintain their inbox. Though these 11 participants were not using the folders functionality to manage the inbox and they had explicitly mentioned not creating any folder, still they chose to be having over more than 200 folders (out of these 11 also, 9 said to be having more than 200 folders and 2 said to be having 51-200 folders). We concluded that there was a major inconsistency in these 11 responses.

Inbox sorting by moving emails from inbox to folders

We had an explicit question asking the participants whether they manually create folders to manage their inbox or not. However, this question was not displayed to everyone but only to those participants who *did not* select moving emails from their inbox to folders as their inbox management practice. Here, we wanted to give the benefit of the doubt that maybe participants would have created folders in their email client, however, they might not be moving emails to those folders (due to any unknown reason). Also, we assumed that the participants who selected the moving emails option would obviously be creating folders and then only they would be moving emails, so we did not want to repeat the question for those.

148 participants had not chosen to be moving emails from their inbox to folders, so we asked them explicitly if they create folders. 22 out of these 148 participants agreed to be creating folders and 126 said that they do not create folders. As shown in Figure 3.13, we observed that only 97 out of 230 participants (42%) were creating and maintaining folders to manage their inbox.

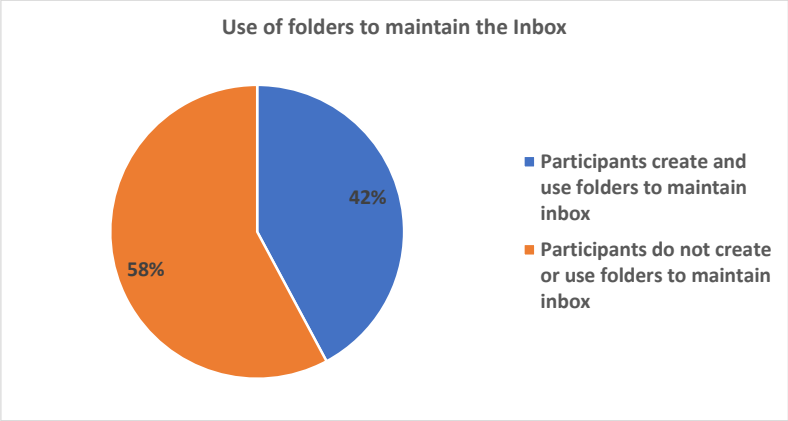


Figure 3.13: 97 of the total participants (42%) said to be using folders to maintain their inbox, whereas 133 (58%) do not create and maintain folders to manage their inbox

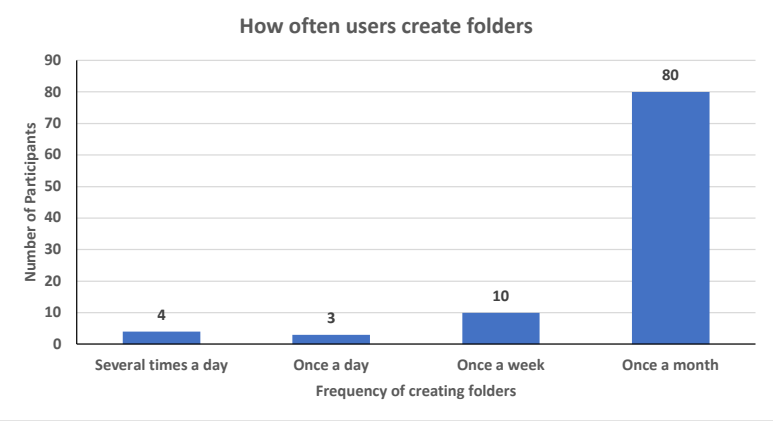


Figure 3.14: 80 out of the 97 participants who create and maintain folders said to be creating folders once a month

Now, the 97 participants who were creating and maintaining folders to manage the inbox were asked, how often they manually create folders. As illustrated in Figure 3.14, 80 out of these 97 participants said to be creating folders once a month, 10 were creating folders once a week whereas a small number of users like 3 and 4 were creating folders once a day and several times a day respectively.

We could infer that only 42% of all the 230 participants were creating folders to maintain their inbox and from that 42%, mostly (80%) were creating folders just once a month. Further, we asked these 97 participants how often they move emails from inbox to other folders and how much time they spend every day (on average) to manually sort their inbox.

As shown in Figure 3.15, 55 out of 97 participants said to be moving emails from their inbox to other folders once a week, 19 said to be moving emails once a day, 18 said to be moving emails several times a day and 5 said to be moving several times an hour.

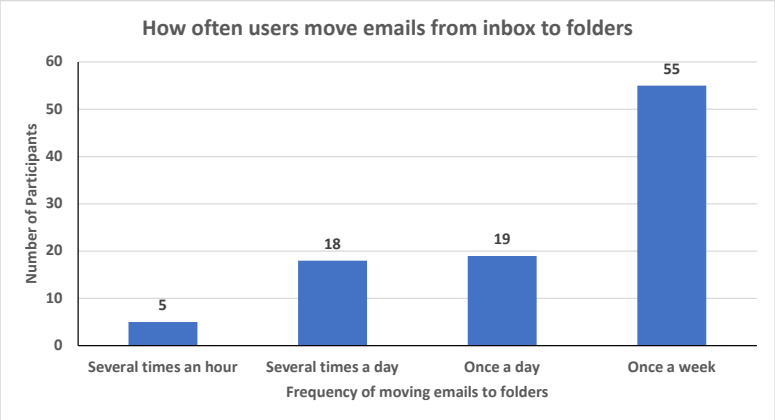


Figure 3.15: 55 out of the 97 participants who create and maintain folders said to be moving emails from their inbox to other folders once a week

As shown in Figure 3.16, 66 out of 97 participants said to be spending less than 10 minutes (on average) to manually sort their inbox, 19 said to be spending 10 to 20 minutes and 12 said to be spending more than 20 minutes to sort their inbox.

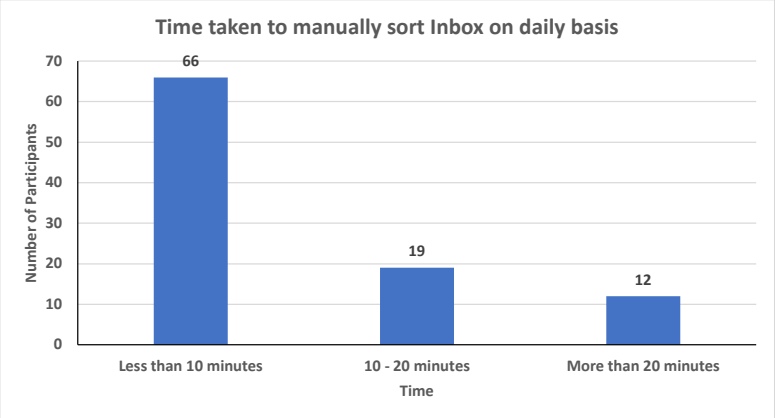


Figure 3.16: 66 out of the 97 participants who create and maintain folders said to be spending less than 10 minutes (on average) to manually sort their inbox

3.2.4 Section 4: User Awareness of Rule-based Filtering Present in Current Email Clients

We wanted to know how many people are aware of defining automatic rules in their email clients using which emails could be directly moved to predefined folders rather than storing them all in the inbox. As highlighted in Figure 3.17, 142 out of 230 participants were aware of this functionality whereas 88 were still unaware of this feature.

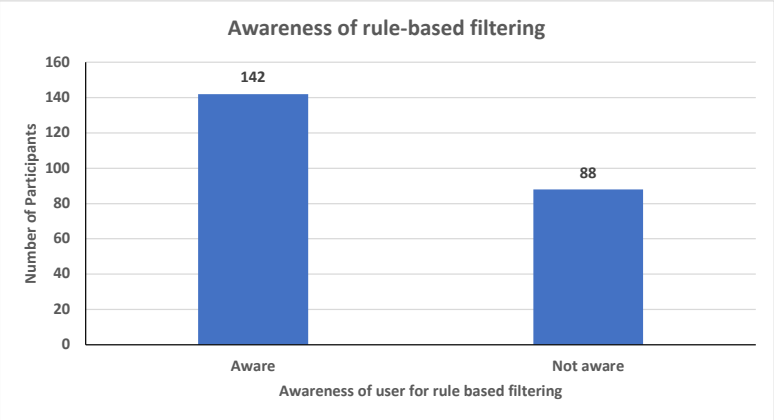


Figure 3.17: 142 of the total 230 participants (62%) were aware of the rule-based filtering present in current email clients

Next, as shown in Figure 3.18, among the 142 participants who knew about defining rules, we observed that 62 of these (44%) had actually never used this functionality. 59 participants (41%) claimed to have used it sometimes, whereas, only a small number 21 (15%) said to be often creating rules to automatically move emails to different folders.

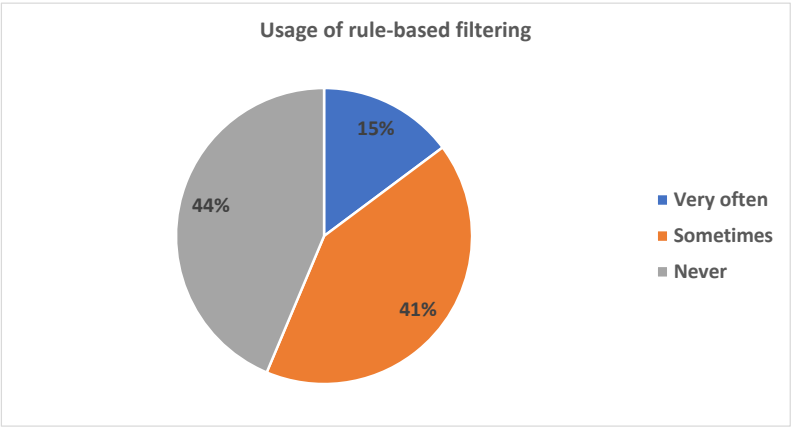


Figure 3.18: 62 out of the 142 participants (44%) who knew about rule-based filtering never used it to automatically filter the inbox

We can deduce that only a few people (80 out of the total 230 participants) define rules to automatically refine their inbox. There could be a few reasons why people do not define

rules. One possibility could be that they are not at all aware of the functionality, another reason could be that they find it difficult to create rules as their choice of rules might be complex for the system. To examine a little more, we asked the 62 participants who knew that they could define rules in their email client however they never chose to use the functionality about the reason for not creating rules. 24 participants indicated that it takes a lot of time to create rules, 14 indicated that they find it difficult to create rules and the remaining 24 chose the 'others' option and filled in a descriptive field with their reason for not defining rules. Some interesting reasons to be noted here are,

- Some do not like to directly move emails into different folders, and they prefer to first take an appropriate action on an email (which may include deletion of an email) and then move emails to other folders
- Some find it difficult to define rules for the proper filtering of emails. They feel that too many rules would rather create a mess and put emails in the wrong folders. For them, filtering out email manually seemed to be an easier option.
- Some have categorised email accounts on the basis of their usage. They have separate accounts for personal use, professional use and one for advertisements or subscribed emails. For these users, the most chaotic email account is the one which receives spam or advertisement emails. They usually delete all of those emails as their email management practice.
- Some said for them filtering emails manually seems to be an easier option.
- Some said for their already messy inbox, they will now have to spend a lot of time to organise it and then define rules.
- Some said they never took the initiative and time to look at how the rules are created as they either do not like creating rules or with less number of incoming emails they do not need to create rules.

Further, we asked *all* the users whether they find it difficult to categorise emails in different folders. As shown in Figure 3.19, 145 out of 230 participants said that they might want to classify an email in two different folders at the same time when doubting between two (or more) folders for the best classification of their email. However, the remaining 85 participants were pretty confident with their binary email classification.

3.2.5 Section 5: User's Preference for Recommendation System in their Mailbox

We showed some email management options to the participants to examine how they would prefer to maintain their mailbox. The options were as follows:

1. The system automatically moves emails to a folder when they arrive, based on the rules you defined earlier. Note that this implies that you would never see those emails in your inbox, but only in the specified folders.
2. The system automatically classifies (without predefined rules) and moves emails to specific folders. Note that this implies that you would never see those emails in your inbox, but only in the specified folders.

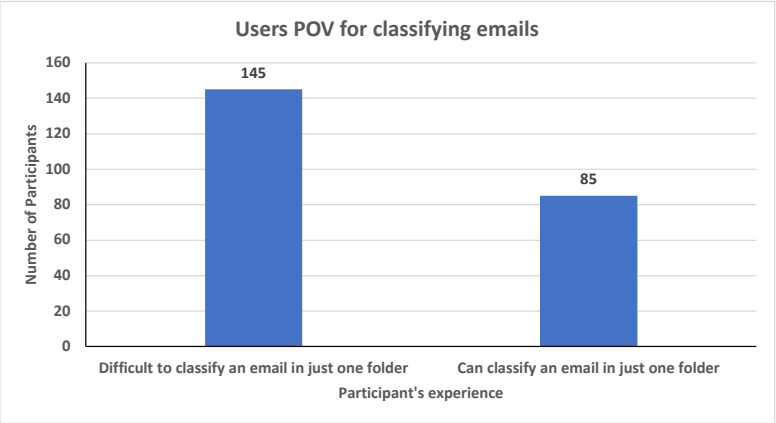


Figure 3.19: 145 out of 230 participants (63%) find it difficult to classify an email in just one folder

3. The system automatically recommends (without predefined rules) and moves emails to specific folders, after asking confirmation from the user.

4. The user manually drags and drops emails to different folders.

As shown in Figure 3.20, 104 out of 230 participants (45%) agreed that they would prefer a recommendation system that would move the suggested emails after asking for confirmation from the user. 71 participants (31%) agreed with the current automation provided by the email clients where they can define rules and the emails would directly be moved to specified folders without ever coming into the inbox. 40 participants (17%) said that they would prefer to manually drag and drop emails to different folders. Only 15 participants (7%) said that they would prefer an automatic system without any predefined rules to move emails directly into folders, without ever coming into the inbox.

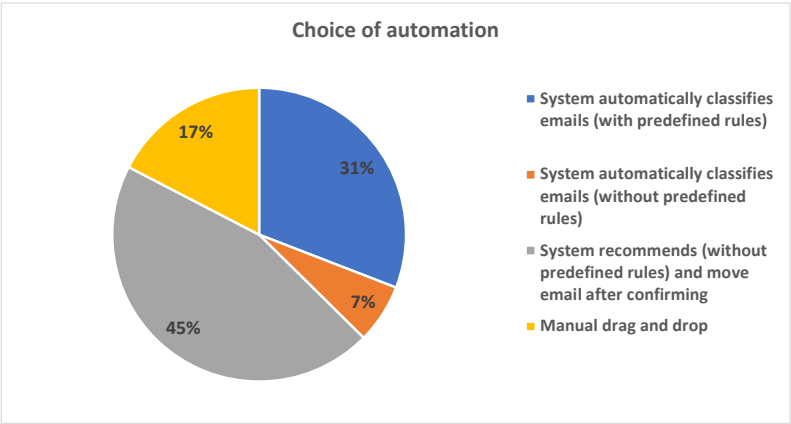


Figure 3.20: 104 out of 230 participants (45%) preferred to have a system that could automatically recommend (without predefined rules) and moves emails to specific folders, after asking for confirmation from the user

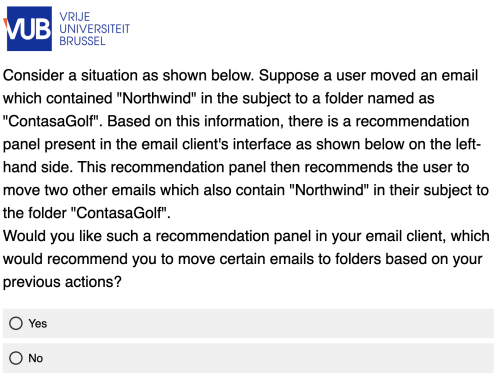


Figure 3.21: Situational question presented to the participants

Once the participants chose their preferred system, we presented them with the situational question as shown in Figure 3.21. Participants were presented a recommendation pane in the mailbox that could recommend similar actions on emails present in their inbox on the basis of previous actions taken by them on their emails. As highlighted in Figure 3.22, this was basically a mock-up of our intended recommendation system. We purposefully displayed the system after asking about a user's preferred email management technique just to avoid any bias in their response. We wanted to know if the audience would be interested in such a system or not.

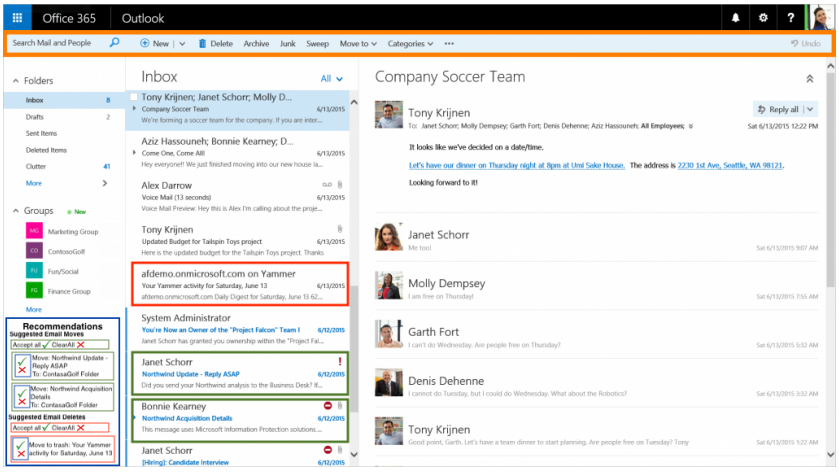


Figure 3.22: The mock-up image presented along with the situational question to the participants

As shown in Figure 3.23, 185 out of 230 participants (80%) were interested in such a recommendation panel, however, the remaining 45 (20%) participants were not interested.

As we wanted to make a recommendation pane for our audience, it was very important here to study these 45 participants who were not interested in the solution. We wanted to know why do they not like the solution in order to maybe include something in our system to make it better or to know about the reason for the incompetence of our system.

We observed that 15 out of these 45 participants preferred the present rule-based (where rules are defined by users) filtering technique over other automation or manual filtering. However, there

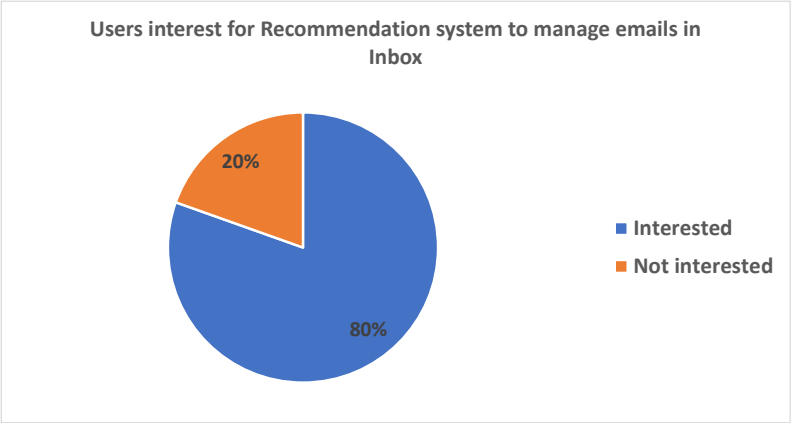


Figure 3.23: 185 out of 230 participants (80%) were interested in the mock-up of our intended recommendation system

was one participant of these 15 participants who also mentioned in a previous question that they do not create rules as it is a time-consuming process. Also, many of these 15 participants agreed to be leaving their inbox as it is and not really take advantage of the present rule-based filtering. However, one could also argue that rather than foldering, they might prefer to have one inbox and delete the unnecessary emails, which we would also agree to. Moving forward, 2 out of the 45 participants preferred having automatic rule-based filtering where they do not have to define the rules, however, based on AI-suggested rules, emails should directly move to folders without coming to the inbox even once. 13 out of 45 participants preferred to manually drag and drop emails to different folders from the inbox. Manual filtering would provide a chance for the user to first take action on an email and then filter it out from the inbox as required rather than directly moving emails to different folders.

Now, interestingly, as many as 15 out of these 45 participants who were not interested in the solution presented in the mock-up had actually chosen the exact same solution presented in words in the previous question. They had preferred an automatic system that could recommend moving emails to specific folders, but only after receiving a confirmation from the user. Clearly, there was a mismatch in both the consecutive answers of these 15 users. We suspect some miscommunication or lack of understanding (of the situational question) over here.

3.2.6 Section 6: Demographic Questions

At last, we had a number of demographic questions to know more about our participants. These questions were displayed to all 233 participants irrespective of if they were using or not using emails.

Our participants were mostly young adults aged between 18 to 30 years. Next, 140 out of 233 participants identified themselves as males, 82 as females, 5 as gender-queer (or non-binary) and the remaining 5 preferred not to say. As the author of the survey belongs to India and is now studying in Belgium, there is a bias in the nationalities of the participants who took part in the survey. We received 125 out of 233 responses (54%) from India and 50 (22%) responses from Belgium. Some other nationalities which took part in the survey were Pakistani, Malaysian, German, Swiss, Nigerian, Iranian and Albanian. Our participants were mostly either students or employed. Only 8 participants out of the total were either retired or non-employed. As

highlighted in Figure 3.24, those who were employed were mostly working in the information technology sector followed by the education and training sector, followed by the business and financial operation sector.

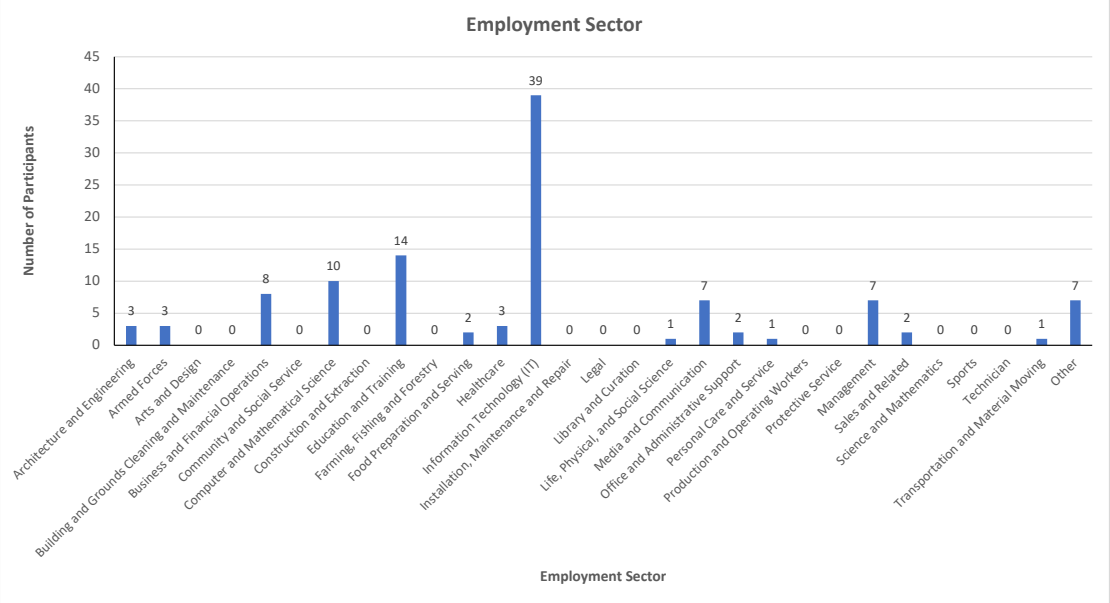


Figure 3.24: Most of the employed participants were working in the IT sector

3.3 Survey Result

From the survey, we inferred that more than 50% of the total 230 participants using emails preferred to access them on big screens like desktops or laptops and the most commonly used email client is Gmail. Further, people mostly use emails for work-related purposes. A lot of information is shared via email, hence 62% agreed to be checking their email several times a day, while 47% said to be receiving 11–50 emails a day. This signifies that emails are still very much a part of our day-to-day lives.

While there is hype for the *Inbox Zero* approach for email management and many tutorials and methods regarding the same are present online, in our survey, we however observed that only 41 of the total 230 participants (18%) had 0–100 emails in their inbox. Most of these participants were following the Inbox Zero approach by either moving emails from their inbox to other folders or deleting emails from their inbox. 20 out of these 41 participants were using their inbox as their to-do list.

The rest of the 189 participants (82%) had more than 100 emails in their inbox. As an interesting observation, most of these participants (66%) leave their inbox as it is and do not move any email from their inbox, which explains the reason for having a large number of emails in their inbox.

Further, we attempted to find an association between the number of folders and different email management practices. We divided the participants into 3 groups. The first group consisted of 159 participants who had 0–10 folders, the second group consisted of 50 participants who had 11 - 50 folders and the third group consisted of 21 participants who had more than 50 folders. We

noted that most of the participants (69%) had 0–10 folders in their mailbox and they were *not* creating or maintaining folders as an email management practice. The lower number of folders would generally include folders like spam or archive which a user would not maintain, but they were present in the mailbox by default. The remaining 31% of the total participants had more than 10 folders and most of these participants were using folders to manage their mailbox. We observed that participants who maintained folders to manage their mailbox usually would have more than 10 folders.

In 2011, Koren et al. [14] had observed that 70% of the Yahoo! Mail users had not created even a single folder. Not much has changed in 12 years, as we also observed from our detailed analysis that only 97 out of 230 participants (42%) were creating and maintaining folders as their email management practice. Still, 58% of the users *do not* create and maintain folders. There could be various reasons like the time and effort required to create and maintain the folders because of which the users might not be utilising them to declutter their inbox. However, the current email clients provide rule-based filtering as an automation technique. Using this filtering technique, users can define some folders along with some rules. Now, using those rules, emails can be stored directly in the predefined folders rather than getting accumulated in the inbox. Though this automation has been in the market for a long time now, still only 142 out of 230 participants (62%) were aware of this functionality. We went into some more depth and noted that only 80 users out of the 230 participants were using the rule-based filtering in their inbox. The remaining 150 participants (65%) were either unaware or had never used rule-based filtering. As discussed in detail in Section 3.2.4, many users did not prefer having rule-based filtering because they believed in, *"out of sight is out of mind"* and wanted to have emails in their inbox at least once. Some also said that if they will start with rule-based filtering now, it will take a lot of time to clean the already messy inbox. Overall, creating and maintaining folders manually seemed like a time-consuming task and users did not prefer rule-based filtering because of reasons like the time and effort required to create the rules and the need to get the emails in their inbox at least once before moving them to different folders.

We also observed that users do not prefer an automatic system which would classify emails into different folders directly without any rules defined by them. It would become difficult for a user to check and process emails in different folders when they do not know where to expect any email. Today, Gmail has defined 5 categories like `Promotions`, `Social` or `Updates` and many emails directly go into these categories without ever coming to the `primary` inbox. These categories are enabled by default which increases the chances of missing an email due to the automatic classification by Gmail. The emails going to these categories would never appear in the primary inbox. From our survey analysis, we suspect users would not prefer such automation.

Further, we noticed that many users would prefer a recommendation system that could help them move emails from their inbox to different folders without any predefined rules by the users.

With all these observations, we were motivated to develop a recommendation system that could suggest similar movement on emails present in the inbox based on actions taken by the users on previous emails. With this approach, not only the user will be benefited with no requirement to create rules, but they would also get the emails first in the inbox and then by their confirmation they could simply move the emails with a click of a button to the folder suggested by the intelligent system. We also observed that 80 out of the 97 participants who create and maintain folders said to be creating folders once a month, so we thought to include functionality to automatically define folders for emails present in the inbox which could be grouped together. With this additional functionality, the users would get a suggestion to create a new folder and classify similar emails from the inbox to that folder. Another big advantage of this system would be that it would not just help to categorise future emails but could also help to declutter the already messy inbox.

As noted from the survey, 80% of the participants were interested in the mock-up presented to them. This formed a strong motivation for us to move in the forward direction to build such a recommendation system. Another observation was that more than 50% of the participants use `Gmail` and mostly use emails on big screens like laptops or desktops, we went ahead and made an add-on for Gmail which could be used on the laptop or desktop interface. In the following chapters, the ideal solution and implementation are discussed.

Chapter 4

Solution

In this chapter, we discuss the ideal user interface design of a recommendation system to improve the organisation of emails in a mailbox. As described by Whittaker and Sidner [26] in their early research and verified by our survey, users do not desire automatic filing. They rather prefer to be made aware of the incoming email message at first and then prefer to take suitable action on the same. Therefore, we propose a recommendation system for existing email clients. As highlighted in Figure 4.1, our vision is to have a small recommendation pane in the existing user interface of the inbox, where users can be shown a number of suggestions to move or delete emails present in their inbox.

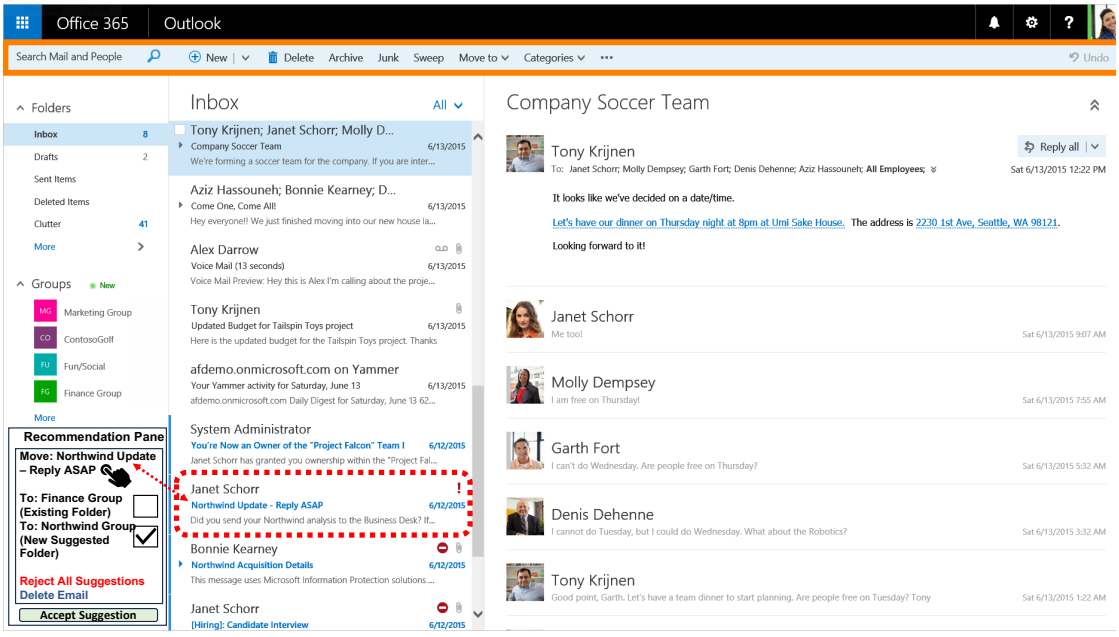


Figure 4.1: Mock-up displaying a recommendation system to manage emails in the Outlook user interface

The architectural diagram shown in Figure 4.2 illustrates the four key components of the recommendation system for emails. The system comprises the Simple Mail Transfer Protocol

(SMTP), the mail server, the Internet Message Access Protocol (IMAP) and the recommendation system. When a user sends an email, the email client utilises the SMTP to transmit the email from the email client to a mail server. While all emails are stored in the mail server, they can be retrieved using the IMAP. The recommendation system establishes a connection with the mail server via IMAP, allowing it to access and retrieve the email data. Following this, the recommendation system processes the data, extracts relevant information and generates machine learning-based recommendations. Further, using the IMAP emails can be moved from the inbox to other folders. IMAP is also responsible to synchronise the email data across different devices.

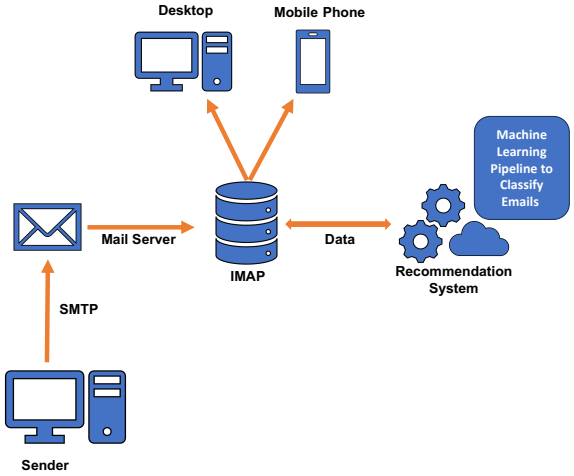


Figure 4.2: An architectural diagram of the recommendation system for emails

The development of the recommendation system is threefold. At first, the system extracts the required data from a user's email account. Secondly, it sends the data to a back-end application where machine learning algorithms are implemented to classify the raw text. After the completion of the text classification task, the suggestions are sent to the front-end application. The front-end application then displays the recommendations in an email client's user interface. Figure 4.3 illustrates a visual representation of the proposed recommendation system's different modules.

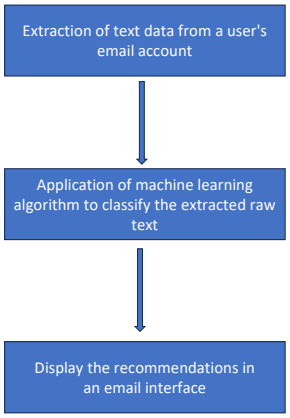


Figure 4.3: Recommendation system modules

4.1 Data Extraction

In order to generate and display recommendations for emails present in the inbox, the system first requires to extract the data from the mailbox. An email consists of some data and metadata. The data of an email refers to the content of the message. The metadata refers to the data about the email itself. Metadata consists of information regarding an email like the subject, the sender and recipient of the email, the date and time when the email was received (or sent), whether there are any carbon copy or blind carbon copy recipients and a message ID. Figure 4.4 highlights the data and metadata related to an email. Both data and metadata of an email can be helpful while building a classifier to organise a mailbox. Data such as subject and content can provide text data which could be processed using Natural Language Processing (NLP) techniques to categorise emails. However, metadata such as the sender's information can further aid in classifying emails based on the domain knowledge received from the sender's email address.

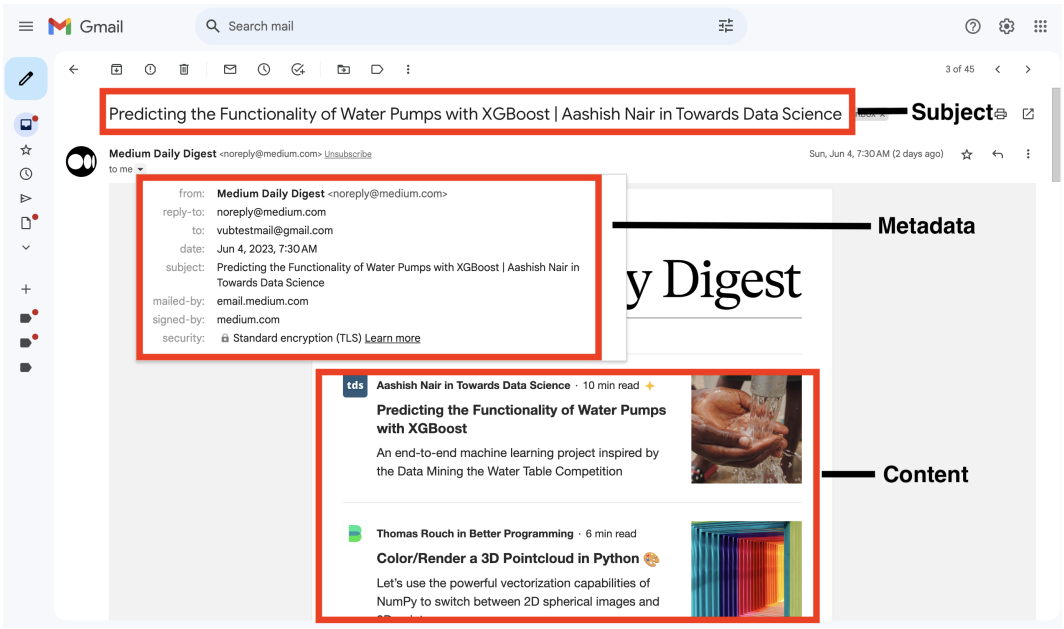


Figure 4.4: Data and metadata of an email

4.2 Data Processing and Generation of Recommendations

In order to generate recommendations, the raw data extracted needs to be processed using some machine learning algorithms. However, before applying any machine learning algorithm we need to understand a user's mailbox. We observed from our survey that 42% of the total population was creating and maintaining folders, while the remaining 58% do not manage emails in folders. This means that there are three possibilities which should be taken care of while developing the recommendation system. These three cases are:

1. *Recommendations based on emails present in the inbox and existing folders* - This is the case where the user creates and maintains folders on a regular basis to manage their inbox. Now, in this case, a user defines folders as per their needs and updates the folders as and

when required. Here, the recommendation system should gain knowledge of emails existing in already-defined folders and should suggest the user to move similar emails from the inbox to already-existing folders.

2. *Recommendations based on emails present only in the inbox* - This is the case where the user does not create any folders to manage their inbox. In this case, the system should identify similar emails in the inbox, group them together and suggest moving those emails from the inbox to a newly suggested folder. The system should also be able to provide a reasonable name for the newly suggested folder.

3. *Hybrid recommendations* - In this case, the user creates folders but may not be maintaining those on a regular basis. Now, it depends there might or might not be a lot of emails in the already-defined folders. If there are a lot of emails in the folders which could help our system to classify similar emails from the inbox, then the system should suggest them on the basis of already classified emails. Otherwise, our system should be able to group together emails present in the inbox based on the similarity between them and suggest moves to the newly suggested folder.

Next, we observed that users find it difficult to classify emails in different folders. They often do not feel confident to pigeonhole an email in a single folder. Hence, the system should not suggest just one folder for the movement of an email. Rather, it should suggest all the relevant folders that an email could belong to. The user can then select which folder(s) they want to move the email into. Further, for effective machine learning, it is crucial to adopt an iterative training process that enables the algorithms to continuously learn and improve the recommendations based on user feedback.

Hence the machine learning algorithm(s) should be selected such that they are able to label the inbox emails either based on the emails present in the already-existing folders or by finding similarities between the emails present in the inbox and grouping them together. Also, the system should be able to recommend more than one label for an email, if relevant. Through these recommendations, we aim at classifying not just incoming emails but also the emails which are already present in the inbox.

4.3 Design and Display of the Recommendation System

We aim at developing a recommendation system which could be used for any email client. However, as different email clients may have different frameworks and APIs involved, we aim to implement different plug-ins for specific clients. Using the plug-in and desired APIs, required information can be extracted from say an Outlook or a Gmail account and sent to the recommendation system in order to receive the list of suggestions. These suggestions can then be displayed to a user as an add-on in the email client's interface.

The add-on for the recommendation system can be displayed as an icon on either side of the inbox's interface. Here, we say either side because, in some email clients like Gmail, the add-ons are displayed on the left-hand side of the inbox. Whereas email clients like Outlook display the add-on on the right-hand side of the inbox. So, depending on the email client, the add-on icon for the recommendation system can be displayed. Whenever a user may want to see various suggestions for emails present in their inbox, they can easily click on the icon which would then expand into a scrollable pane to display the suggestions. Some guidelines which should be followed while displaying suggestions (or recommendations) in a scrollable pane are listed below:

1. The suggestion should comprise the *subject* of an email along with the *recommended folder(s)*. Metadata such as the *sender* of the email should also appear in the suggestion to provide the user with more relatable information.

2. As shown in Figure 4.1, while displaying the suggestion for folders, it should be mentioned whether it is an *existing* folder or a *newly suggested* folder by the system. Users tend not to remember all the existing folder names in their mailbox [4].

3. The suggestions should be displayed in a *chronological* order, meaning that they should appear in the same order as in the inbox. It should not be like a suggestion to move a pretty old email appears first, since this would require more cognitive load on the user to remember the relevance of the email. Users can relate more if they see the same order in the recommendation pane as in their inbox.

4. As highlighted in Figure 4.1, when a user hovers over a suggestion, the corresponding *email* should be *highlighted* in the inbox. This would enable users to easily relate between the subject displayed in the suggestion with where that email is in the inbox. When highlighted, they can easily have a look at the email if required. They can quickly remember the content of the email, which would help to take a better decision for the movement of the email from inbox to the suggested folder(s).

5. All the suggested labels for an email should appear together with a checkbox in front of each label. This would facilitate a user to simply select the checkbox to mark the label of their choice.

6. The recommendation pane should be scrollable, where a user can accept or reject multiple suggestions at once. To enable this functionality there should be a global *accept* button for all suggestions and a local *reject* button for each individual email's suggestion(s). A user can then select multiple suggestions of their choice by selecting the relevant checkboxes. After selecting the suggestion, the user can click on the accept button. However, if they would not like any of the presented suggestions for an email, they could click the reject button.

7. A user could either select or reject suggestions displayed by the system.

   - In case of accepting suggestion(s), there could be two possibilities. First that the suggested folder already exists. In this scenario, the system should move the email(s) from the inbox to the respective existing folder(s). Second, the suggested folder may be a new folder that needs to be added to the mailbox. In this scenario, the system should first create a new folder in the mailbox and then move the selected email(s) from the inbox to the newly created folder(s).

     If the user selected to move an email from the inbox to more than one folder, then that email should appear in all the folders.
   - In case of rejecting suggestion(s) for an email, that suggestion should disappear from the recommendation pane. If the user would have selected some other suggestions for other emails (to be accepted later on), those selected choices should remain intact while disappearing the rejected suggestion from the pane. Also, the system should be able to learn and recommend some new (and better) labels for those emails for which previous suggestions were rejected.

8. When a user would click on the global `accept` button, the page displaying the inbox should be refreshed with the expected movements of emails from inbox to various folders.

9. The display of suggestions should be different when a user opens the recommendation pane in the inbox and when they open the recommendation pane with an email. The recommendation pane should display the existing suggestions for all the emails when it is opened in the inbox. However, when a user opens an individual email, the recommendation pane should only display the suggestion(s) for that particular email (only if a suggestion exists).

10. Along with the movement of email, the recommendation pane should also enable the user to delete the email. A user might not want to move all emails to different folders, but they might delete some emails to manage their inbox.

Once we formed the basic architecture for our system, we then explored different technologies to implement the system. Various design and technical choices to implement the system are discussed in the following chapter.

Chapter 5

Implementation

In this chapter, we discuss the technical details and various design choices that we made during the implementation of the recommendation system prototype. As we observed from the survey, Gmail is the most popular email service provider. Therefore, we built a recommendation system which can be integrated with Gmail's UI. Before going into the implementation details, it is important to mention that in Gmail, *labels* are used to categorise emails. Labels are similar to folders. However, a user can apply multiple labels to an email and then access the email from any of the assigned labels.

As depicted in Figure 5.1, the architectural diagram illustrates the integration of Gmail with the recommendation system. As a brief introduction, the Inbox Harmony add-on utilises the `Gmail Service` provided in *Google Apps Scripts* to extract and manipulate an individual's Gmail data programmatically. Google Apps Script is a scripting platform developed by Google to facilitate users to extend the functionalities of various Google products, including Gmail. After the data extraction, the extracted data is sent to the *Flask* micro web framework by utilising the `URL Fetch Service`. Within the Flask application, the machine learning pipeline is responsible for processing the raw data and generating recommendations. Subsequently, the recommendations are sent from the Flask application to the Google Apps Script by again utilising the `URL Fetch Service`. Finally, the recommendations are displayed to the user as an add-on in Gmail's UI by using the `Card Service` in Google Apps Script. Now, the user may accept or reject the recommendations so presented to them. The machine learning algorithm implemented in the Flask application learns from the user's feedback continuously to provide improved recommendations over time. This is an iterative learning process to provide users with improved, accurate and personalised recommendations.

In the previous chapter, we discussed the three broad steps involved while developing the recommendation system. They were data extraction, data processing and the generation of recommendations and finally, the design and display of the recommendation system. To implement the recommendation system for Gmail, we group the first and third step as part of the *front-end* development. Therefore, the front-end development includes the extraction of data from a user's Gmail account and the design of the user interface of the recommendation system to display the suggestions. The front end is developed using the Google Apps Script. The back-end development consists of the machine learning part of the recommendation system. The back end is developed using the Flask web framework. The chapter is divided into two broad sections, front-end and back-end development, to discuss the different parts of the implementation.

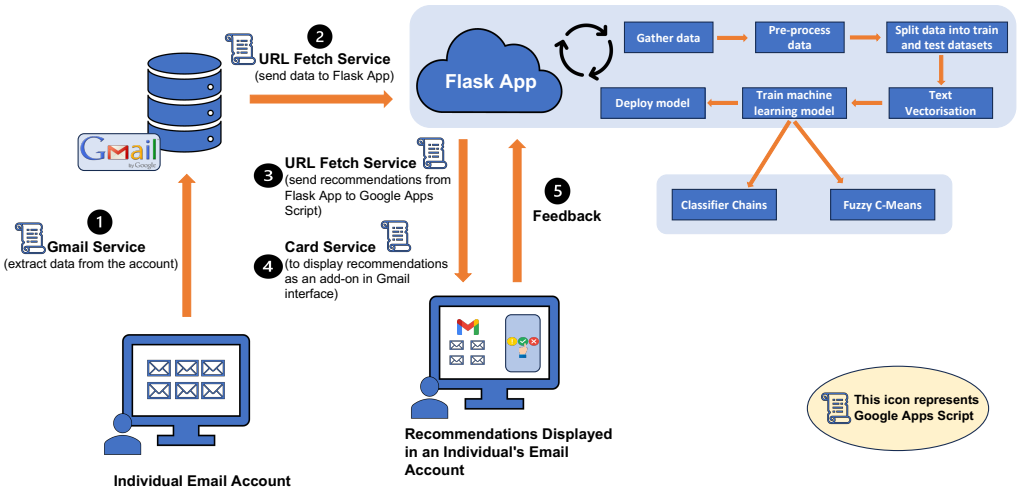


Figure 5.1: Recommendation system architecture

5.1 Front-End Development

To develop the recommendation system as a Gmail add-on, we selected *Google Apps Script (GAS)*. GAS is a cloud-based scripting platform to create web-based applications which can be easily integrated with Google products such as Gmail. Apps scripts are written in the same format as JavaScript. Google APIs and UI tools are built into Apps script as Services. We utilised the `Gmail Service` to extract the text data from a user's Gmail account, the `URL Fetch Service` to send the extracted text data and receive the recommendations from the Flask application and the `Card Service` to develop a UI to display the recommendations.

5.1.1 Gmail Service

Gmail Service provides various sets of classes and methods which enabled us to access and modify email threads and labels. We used 4 out of the 6 classes provided in the Gmail Service as mentioned below:

1. `GmailApp`[1] - This class provides access to Gmail threads, messages and labels. It is worth mentioning that Gmail threads and messages are two different components. A message refers to an individual email message and a thread refers to a grouped conversation that includes forwarded or replied messages.

2. `GmailLabel`[2] - This class provides the methods to interact with the user-created labels in Gmail.

3. `GmailMessage`[3] - This class provides the methods to extract and modify different components of a Gmail message.

[1] `https://developers.google.com/apps-script/reference/gmail/gmail-app`
[2] `https://developers.google.com/apps-script/reference/gmail/gmail-label`
[3] `https://developers.google.com/apps-script/reference/gmail/gmail-message`

4. `GmailThread`[4] - This class provides the methods to extract and modify different components of a Gmail thread.

GmailApp Class

We used various methods available in the GmailApp class to extract text data from a user's Gmail account. Some of the used methods are:

1. `getInboxThreads()` - This method is used to fetch all the threads present in the Inbox irrespective of the labels. In our application, we assume that the threads present in the inbox are not labelled. Hence by using the `getInboxThreads()` method, we are extracting the email threads which need to be labelled.

2. `getSpamThreads()` - This method is used to fetch all the threads present in Spam irrespective of the labels. Again we assume that threads present in Spam are not labelled. Hence by using the `getSpamThreads()` method, we are extracting the email threads which are marked as Spam.

3. `getUserLabels()` - This method is used to fetch a list of all the user-created labels.

4. `getMessageById()` - This method is used to fetch a message by its id.

Once our system extracts the list of Inbox threads, Spam threads and user-created labels, we then use `GmailLabel` and `GmailThread` classes to further retrieve the text information.

GmailLabel Class

A user may or may not have created labels. For those who would have created labels, using the `getThreads()` method present in the `GmailLabel` class, our system extracts the threads for each of the user-created labels.

GmailThread Class

Using the `getMessages()` method in `GmailThread` class, our system extracts all the messages belonging to a thread. If a thread contains a single message, it is further processed using the `GmailMessage` class. However, if a thread contains several messages including forwarded or replied messages then our system extracts the last message of the thread. The system extracts only the last message of a thread as the last message includes all the text from the previous messages. Extracting the content from each message of a thread would give us redundant text which might not be useful while modelling our machine learning algorithm. The last message of a threaded conversation (containing 3 messages) is shown in Figure 5.2. We can see that the last message contained the text from the first two messages. Once the message is extracted, the system uses the `GmailMessage` class to extract the text and the metadata for the extracted message.

GmailMessage Class

Various methods are defined in the `GmailMessage` class using which Inbox Harmony could extract the content, attachments and metadata related to a message. The metadata could be the subject, the message Id, the date and time of the message, the sender of the message, the recipient marked in the `CC` and `BCC` column of the message and the status as read or unread of the

[4] `https://developers.google.com/apps-script/reference/gmail/gmail-thread`

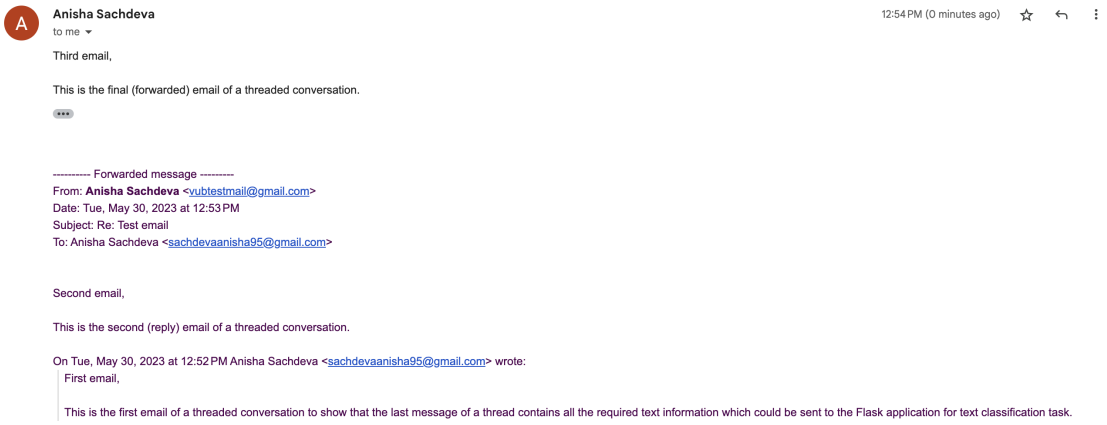


Figure 5.2: Threaded conversation

message. However, we only extract the content, subject, sender information and the message ID using `getPlainBody()`, `getSubject()`, `getFrom()` and `getId()` methods respectively from the `GmailMessage` class. We extracted the message ID as it is a unique identifier for each individual message and could be used as a primary key to identify and distinguish emails.

We did not enable the system to extract the attachments for a message as it could have been a time-consuming task without any additional advantage to improve the accuracy while classifying different text. Other information like date, time, cc'd and bcc'd recipients for a message would also not have provided any additional value for our text classifier, hence we chose not to extract this metadata.

Further, the data is extracted from the inbox, spam and user-created labels in the form of a two-dimensional array. The general structure of the data is:

```
[[location, subject, email_content, sender_name_and_id, message_id],
[inbox, subject_1, email_content_1, sender_name_and_id_1, message_id_1],
[inbox, subject_2, email_content_2, sender_name_and_id_2, message_id_2],
[spam, subject_3, email_content_3, sender_name_and_id_3, message_id_3],
[user_created_label_1, subject_4, email_content_4, sender_name_and_id_4, message_id_4],
[user_created_label_2, subject_5, email_content_5, sender_name_and_id_5, message_id_5]]
```

5.1.2 URL Fetch Service

Once the system extracted the required email data, it is then sent to the Flask application hosted on PythonAnywhere using the URL Fetch Service. The `fetch()` method of the `UrlFetchApp` class is used to get the URL where the Flask application resides and sends the extracted data to it. When the Flask application receives the data, it processes it using various machine learning algorithms (as described later in Section 5.2) and sends back a list of message IDs with respective suggested labels. This list of suggestions is then received using the `getContentText()` method defined in the `HTTPResponse` class. The HTTPResponse class and UrlFetchApp class are both part of the URL Fetch Service[5].

[5] `https://developers.google.com/apps-script/reference/url-fetch`

The system receives the list of suggestions from the Flask application as a JSON data array. The general structure of the received JSON data array is:

```
[[suggested_label_1, message_id_1],
[suggested_label_2, message_id_1],
[suggested_label_1, message_id_2],
[suggested_label_3, message_id_3]].
```

As Inbox Harmony is enabled to provide multiple suggestions, there might be some subjects having multiple label suggestions. For example, in the above-mentioned array, there are 4 suggestions generated by the system for 3 message IDs. The message_id_1 has suggested_label_1 and suggested_label_2 as suggestions, message_id_2 has suggested_label_1 as suggestion and message_id_3 has suggested_label_3 as a suggestion. From the JSON data array, a dictionary object is created where values are grouped together as per the corresponding keys, eliminating duplicates within each group. The dictionary for the above-mentioned JSON array would be:

```
{‘‘message_id_1’’:[‘‘suggested_label_1’’, ‘‘suggested_label_2’’],
        ‘‘message_id_3’’:[‘‘suggested_label_3’’],
        ‘‘message_id_2’’:[‘‘suggested_label_1’’]}
```

Now, the dictionary so created would not guarantee to be iterated over a specific order as the objects in JavaScript are not ordered. Therefore, the system first creates an ordered map containing the same key-value pairs as in the dictionary. However, now these key-value pairs are ordered as per the order of emails in the inbox. The ordered map so created ensures that the suggestions shown in the recommendation system will be chronologically ordered or we can say, the suggestions will follow the same order as emails in the inbox. The ordered map[6] for the above-mentioned dictionary would be:

```
[[‘‘message_id_1’’,[‘‘suggested_label_1’’, ‘‘suggested_label_2’’]],
        [‘‘message_id_2’’,[‘‘suggested_label_1’’]],
        [‘‘message_id_3’’,[‘‘suggested_label_3’’]]]
```

5.1.3 Card Service

The add-on UI is based on the ordered map as discussed in Section 5.1.2. While displaying recommendations, required information such as the subject and the sender's name is extracted using the unique message ID.
Google Apps Script provides two types of add-ons, *Google Workspace Add-ons* and *Editor Add ons*. They both differ in terms of which all applications they can extend and how they are created. While editor add-ons can just be extended by Google editor applications like Docs, Forms, Slides and Sheet, Google Workspace Add-ons can be extended by several Google Workspace apps like Drive, Calendar and Gmail. Hence, we used the `Google Workspace Add-ons` to build the add-on for Gmail.

To create the add-on's user interface, we could either use HTML and CSS or the `Card Service` provided by Apps script. However, during our research, we came across the restriction[7]

[6] We assume that the order of our hypothetical inbox is message_id_1, message_id_2 and message_id_3
[7] `https://developers.google.com/apps-script/add-ons/guides/workspace-restrictions`

documented by Google that while developing Google Workspace Add-ons, one must use card-based interfaces. Only Editor Add-ons can leverage HTML/CSS. Therefore, we used the `Card Service` to define the UI for our recommendation system.

The `Card Service` provides various sets of classes and methods to build an add-on using different widget components. We used the `CardService`[8] class which provided the ability to create a card to display the suggestions in the Gmail interface. Figure 5.3a highlights the Card anatomy which would help to better understand some methods which we used from the `CardService` class. The methods used to create the card are:

1. `newCardSection()` - This method is used to create a card section to hold widgets together that belong to an individual email and provide a visual partition between suggestions for different emails.

2. `newDecoratedText()` - This method is used to create a widget to display text in a card section. A `decoratedText` can include an icon, a top label and a bottom label. We used `decoratedText` to display the subject and the suggested label(s) for an email.

   - While displaying the subject of an email, we used the `setTopLabel()` method to display the sender's name above the subject of the email. The subject and the sender's name are displayed to the user to indicate the email for which the suggested labels are presented.
   - While displaying the suggested label(s) for an email, we used the `setBottomLabel()` method to display if the suggested label is an *existing label* or a *new label*. Further, we used the `setSwitchControl()` method to display a switch button to the right of the label. This switch button serves as a selection checkbox. Using this switch button, a user could select the label(s) that they would want to move their respective emails into. Once they would have selected their choice of labels for all the suggestions displayed in the scroll bar, they could simply click on the global *accept selected label* button to move emails into respective selected folders.

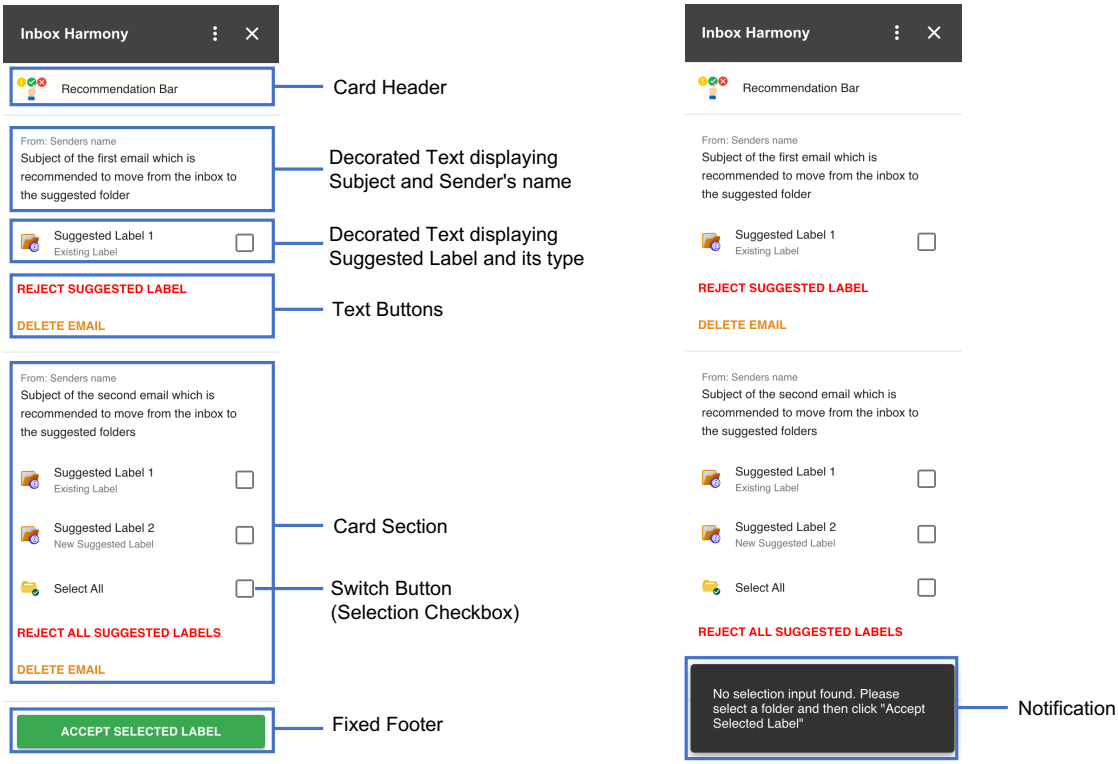
3. `newTextButton()` - This method is used to create a button widget to display the options to 'Reject Suggested Label' or to 'Delete Email'. These two are local buttons provided for each section which display label(s) suggested for an individual email. If a user does not like any of the suggested label(s), they can quickly reject all the labels by clicking the 'Reject Suggested Label' button. If a user may want to delete a particular email rather than moving it to another folder, they can click on the 'Delete Email' button.

4. `newFixedFooter()` - This method is used to create a fixed footer that is shown at the bottom of a card. We used this method to create the global button widget 'Accept Selected Label'. Using this global button, a user can easily move emails from their inbox to the selected folders.

5. `newNotification()` - This method is used to show various notifications to a user as feedback (or response) while interacting with a user interface element. As highlighted in Figure 5.3b, if a user clicks on the 'Accept Selected Label' button without selecting any label to move an email, the system shows a notification stating *"No selection input found. Please select a folder and then click Accept Selected Label"*.

6. `newCardHeader()` - This method is used to set the header of the card. In our system, we defined our card header as 'Recommendation Bar'.

[8] `https://developers.google.com/apps-script/reference/card-service/card-service`



(a) Card Anatomy

(b) Notification displayed on a card

Figure 5.3: A sample card-based UI

7. `newCardBuilder()` - This method is used to build the card objects.

8. `newActionResponseBuilder()` - This method is used as a builder for `ActionResponse` objects. `ActionResponse` objects are the response objects that are returned from a callback function when processing an action triggered by a user.

5.1.4 Authorisation Scopes

To enable the recommendation system (add-on), the user needs to authorise the *Google Apps Script (GAS)* project to provide access to their Gmail account. We configured the level of access required by the add-on by declaring authorisation scopes in the manifest[9] file of our GAS project. The authorisation scopes are defined as OAuth[10] 2.0 URI strings containing information about the kind of data and level of access, the add-on is asking permission for. Some of the Gmail API scopes used in our GAS project are:

1. '`https://www.googleapis.com/auth/gmail.readonly`' - This specific scope enables the GAS project to read all the resources and their metadata present in a user's Gmail account.

[9]Manifest file is a special file named `appsscript.json`, where we can define the configurations and settings for a GAS project.

[10]OAuth stands for 'Open Authorisation' which is an industry-standard authorisation protocol designed to enable an application to access data hosted by third-party web application without requiring credentials of identification of a user.

2. '`https://mail.google.com/`' - This scope enables the GAS project to fully access a user's Gmail account including actions like permanent deletions of emails. In Inbox Harmony, we provide the functionality to delete an email or move an email from a user's inbox to other folders, hence we are required to include the scope to have complete access to a user's mailbox.

3. '`https://www.googleapis.com/auth/script.external_request`' - This specific scope enables permission to make external requests from GAS project. In our system, we are required to send and receive data from the Flask application hosted on PythonAnywhere platform, hence we are required to include this external request scope in our manifest file.

As highlighted in Figure 5.4, before the user runs the recommendation system add-on for the first time, the add-on's UI displays a prompt requesting the user to authorise access to their account. Further, as illustrated in Figure 5.5, during the authorisation flow, the UI shows the human-understandable descriptions of the permissions that the GAS project is asking for. Once the user is comfortable sharing the required information, they can then see and utilise the functionalities present in the recommendation system.

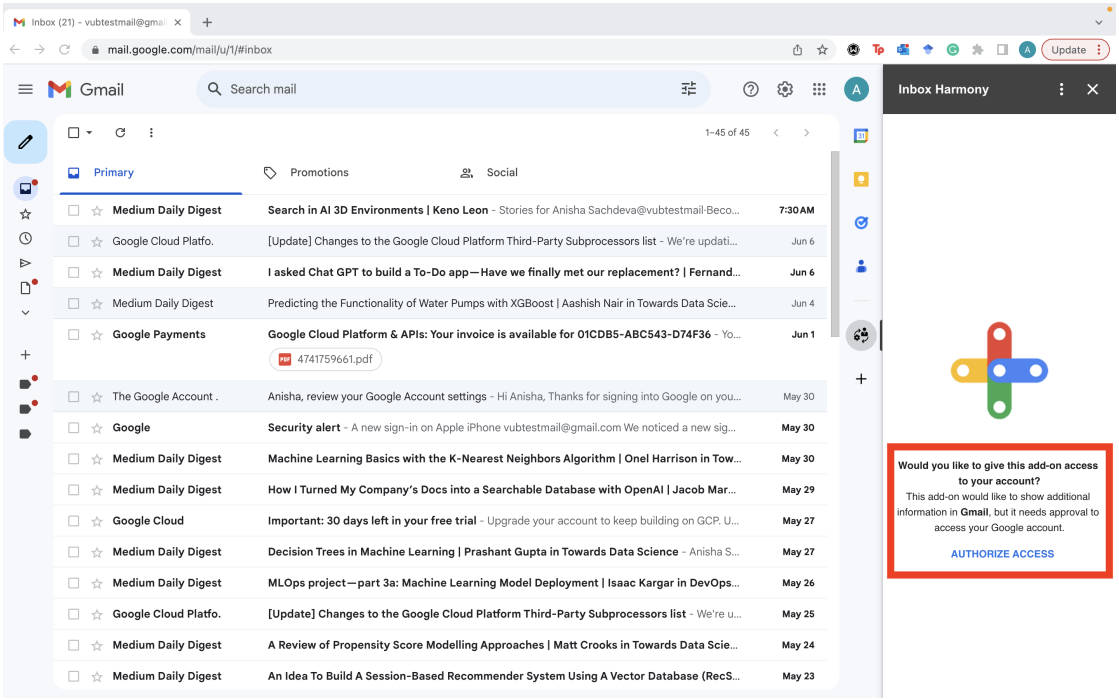


Figure 5.4: Prompt requesting users to authorise access to their Gmail account

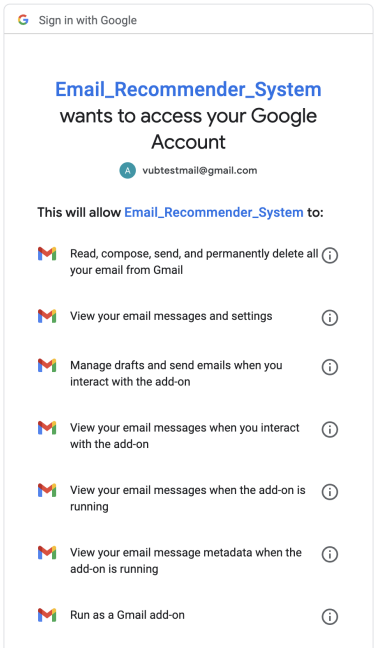


Figure 5.5: UI displaying the description of the permission the GAS project is asking for

5.1.5 Challenges

While developing the front end of our recommendation system (as an add-on) we faced a number of challenges. These challenges along with how we solved them are described below:

1. *Need to rebuild the card while interacting with the user and displaying a change in the UI* - In the add-on, we created two local buttons for each email suggestion as 'Reject Suggested Label' and 'Delete Email'. Our aim was to provide users with a seamless experience while selecting or rejecting suggestions. The idea was that a user could select all the suggestions (by clicking on the checkbox provided to the right of the suggested label) for various emails at once and then click the 'Accept Selected Label'. Meanwhile, if they want to reject some suggestions (or delete some emails) they could click the 'Reject Suggested Label' (or 'Delete Email') button and the particular email for which the suggestion has been rejected (or email needs to be deleted) should disappear keeping the rest of the selected suggestions as it is.

   While implementing this functionality, we came across `CardService` being a stateless model. To enable the users to see the change in the UI (in our case disappearance of the rejected suggestion), the card displaying the suggestions needed to be rendered again. This rebuilding of the card takes around 7–10 seconds causing a hindrance in the user's experience of using the add-on. A user would have to wait for 7–10 seconds every time they would reject or delete an email. At first, we thought of changing the colour of the reject and delete buttons as an interaction with the user to ensure that their choice has been saved. However, even changing the colour of the button required rebuilding the card. Hence, as highlighted in Figure 5.6, rather than making any changes in the UI of the card itself, our system displays a notification to the user indicating that their choice has been registered.

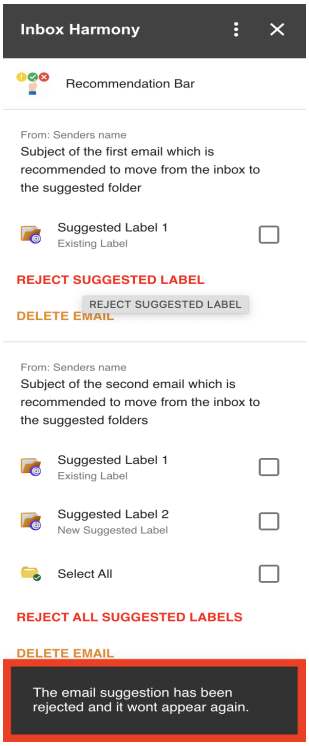


Figure 5.6: Notification pops up when a user rejects a suggestion

2. *Not possible to manipulate Gmail interface* - As Google Apps Script (GAS) runs on the server side and Gmail runs in a user's browser, one cannot use GAS to manipulate or modify Gmail's interface. This restricted us from automatically refreshing the page when a user would click the 'Accept Selected Label' button and highlighting the email in Gmail's interface when a user would hover over a subject shown in the recommendation pane.

   Using GAS we could not automatically refresh Gmail's interface on the click of 'Accept Selected Label' button and display the movement of emails from the inbox to the selected folders in the same Gmail tab (though the email movements had taken place). However, as an alternative, we used the `setOpenLink()` method provided in `CardService` class to reload the Gmail page in a new tab when the 'Accept Selected Label' button is clicked. Gmail which now got opened in the new tab displayed the expected movement of emails along with new suggestions (as per the accepted or rejected labels by the user). However, the downside of this alternative is that every time a user would click on the accept button to move some emails, Gmail would be opened in a new tab which might not be desirable.

   To enable a user to quickly refer to an email for which labels are suggested in the add-on, the ideal solution would be to highlight the email in Gmail's interface while the user would hover over the subject present in the recommendation pane. However, it was not possible to make changes to Gmail's interface. Therefore, to still provide a user with the content of an email whose subject is displayed in the recommendation pane, we provided the subject as a button. As shown in Figure 5.7, the user could click on the subject to see its content along with the sender's name. This would help the user to better understand the context of the email and take the decision to move or delete an email from their inbox.

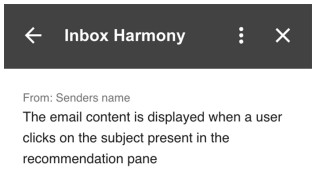


Figure 5.7: The content and sender's information are displayed on the click of subject

3. *Script limitations* - The current script runtime allowance is 6 minutes per execution. We noticed that in the case of a large size of the mailbox, it would certainly take more than 6 minutes to extract the desired data from all the emails, send and receive a response from the Flask Application and render the card to display the suggestions to the user. Hence, we divided the project into two scripts. One script was used just to extract the data from the mailbox, send it to the Flask application and receive a response from it. This response was then stored in cache which could be accessed by the second script of the same project. The second script would then take the data, reorganise it as explained in Section 5.1.2 and render the card to be displayed to the user.

5.2 Back-End Development

As discussed in Sections 5.1.1 and 5.1.2, once the data is extracted from a user's Gmail account it is sent to the *Flask* application hosted on *PythonAnywhere*[11]. The data received by the Flask application is a nested list containing the name of the label, the subject, the content, the name of the sender and the message ID for all the emails present in a user's inbox, spam and other user-created labels. As described earlier in the chapter, the data received is of the form:

```
    [[location, subject, email_content, sender_name_and_id, message_id],
   [inbox, subject_1, email_content_1, sender_name_and_id_1, message_id_1],
   [inbox, subject_2, email_content_2, sender_name_and_id_2, message_id_2],
    [spam, subject_3, email_content_3, sender_name_and_id_3, message_id_3],
[user_created_label_1, subject_4, email_content_4, sender_name_and_id_4, message_id_4],
[user_created_label_2, subject_5, email_content_5, sender_name_and_id_5, message_id_5]]
```

As we have the textual data from the emails, we focus on *text classification* which is a *Natural Language Processing (NLP)* task. NLP is a branch of Artificial Intelligence which give machines the ability to read, understand and derive meaning from human languages. Text classification is the process of organising text data into groups. As illustrated in Figure 5.8, the general pipeline followed for the text classification task is gathering data, pre-processing data, splitting data into training and testing data sets, text vectorisation, selecting and training the machine learning model and finally deploying the model to be tested. Each step of the pipeline is discussed in detail in the following sections.

5.2.1 Gather and Pre-process the Data

The data is extracted using Google Apps Script and sent to the Flask Application. To transform the raw text into a more suitable form for modelling we performed the following steps on the subject and content of the received emails:

[11] PythonAnywhere is a Python-specific web hosting platform.

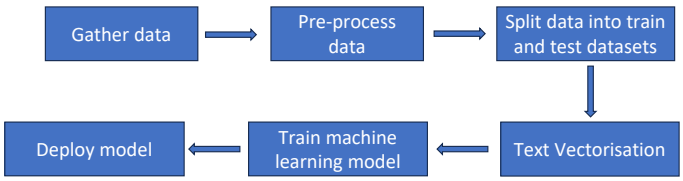


Figure 5.8: Text classification pipeline

1. *Tokenisation* - This step involves the splitting of text into tokens (or words).

2. *Text Cleaning* - This step involves converting the text to lowercase and removing non-alphanumeric characters, single characters (letters) surrounded by spaces, punctuation and special characters.

3. *Normalisation* - This step involves transforming words into their canonical form. For this step, we could either go for stemming or lemmatisation. Stemming is the process of reducing words to their root form (or stem). Lemmatisation is the process of determining the base form of words considering the part of speech and its context. Though stemming is simpler and faster, however, lemmatisation provides more accurate results. Hence, we decided to choose the lemmatisation technique to reduce the words to their base form.

4. *Stopword Removal* - This step involves the removal of common words in a text which usually do not provide any information. Words like articles, prepositions or conjunctions are eliminated from the text as they are not considered informative.

5.2.2 Split the Data

Once, the text data of the subject and content are cleaned, we split the data into training and testing data sets. The nested list received by the Flask application contains various sublists. Each sublist contains information about an individual email. The first element of the sublist contains the name of the label which could be inbox, spam or user-created label. This element helps to distinguish between the data which needs to be categorised and the data which would be used to train the machine learning model. If the first element of a sublist is 'inbox' it would mean that the email content of the sublist needs to be categorised and it becomes the test data. If the first element is anything other than 'inbox' it would mean that the email content of the sublist is already categorised as given in the first element and it is then included in the training data set.

5.2.3 Text Vectorisation

We can not directly use the text data to train the machine learning model. We need to convert the text data into a numerical or structured representation to be used as inputs for the machine learning model.

TF-IDF

We used *TF-IDF (Term Frequency-Inverse Document Frequency)* to transform the text data into vectors. This data consisted of text from the subject and content of an email. The text vectoriser, TF-IDF includes two concepts, term-frequency and inverse document frequency. As

the name suggests, term frequency counts the number of times a word appears in a document and inverse document frequency tells about how common (or uncommon) a word is in a corpus (collection of documents). We preferred to use TF-IDF as the text vectoriser as it is simple and computationally cheap.

One-Hot Encoding

For the text data present in the sender's detail (the domain name from where the email is sent), we observed that this text is categorical in nature. Hence, we used *One-Hot Encoding* to convert the categorical data (domain names) into numerical features.

5.2.4 Selection and Training of Machine Learning Model

As we cannot anticipate whether a user would have defined labels in their Gmail mailbox or not, the system should be enabled to handle both situations and provide suggestions accordingly. In the first situation where a user would have defined labels along with emails under those labels, the system should train the model with the already labelled data and then provide suggestions of labels for the unlabelled emails present in the inbox. In such scenarios where labelled data is available, the system should use a *supervised* machine learning model.

However, in the other scenario where a user has not defined any label, the system should find similarities within the unlabelled emails present in the inbox, group them together, provide a name to the group and then present the new suggested labels for the emails. In such scenarios where labelled data is *not* available, the system should use an *unsupervised* machine learning model. The flowchart illustrated in Figure 5.9 describes the criteria and selection of the machine learning model.

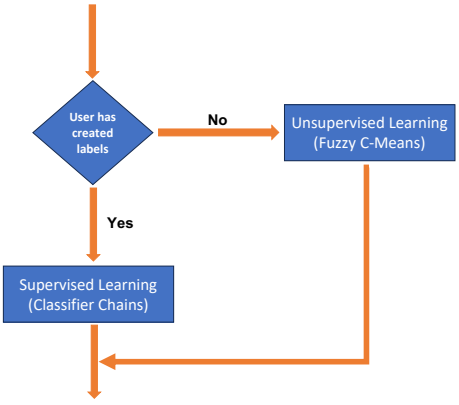


Figure 5.9: Machine learning model selection flowchart

Classifier Chains: Supervised Machine Learning Model

Rather than opting for traditional supervised machine learning models like Support Vector Machine or Naive Bayes Classifier, we selected the *Classifier Chains* model. The reason was simple, traditional algorithms perform multi-class classification i.e. each output instance is assigned to exactly one of the several mutually exclusive classes. However, as observed from the survey (Figure 3.19) users find it difficult to assign an email to exactly one folder, we chose a multi-label classifier. A multi-label classifier can assign an output instance in multiple classes simultaneously.

Some commonly used multi-label classification algorithms are Binary Relevance, Classifier Chains and Label Powerset. After some research[12] and looking at the simplicity of Binary Relevance and over-complexity of the Label Powerset, we considered Classifier Chains to be a good choice when label dependencies were important to us [20]. Now, the labelled emails were used to train the Classifier Chains model.

Fuzzy C-Means: Unsupervised Machine Learning Model

For the same reason as mentioned above, we did not want to go for hard clustering algorithms to group together the emails present in the inbox. We wanted to form clusters where each unlabelled email could belong to more than one cluster. Hence, we chose *Fuzzy Clustering* (also known as *Fuzzy c-means* or *Soft k-means*) to group together emails present in the inbox.

Latent Dirichlet Allocation: Topic Modelling

Once the groups are created using the Fuzzy c-means algorithm, we need to name those groups, and only then the system can suggest those names as labels to the users. We chose *Latent Dirichlet Allocation (LDA)* for automatically identifying topics from the subjects of the emails which were grouped together using the Fuzzy c-means algorithm. The length of topics for different groups could vary and usually, a big label name is not desirable. Hence, the system was enabled to select the top two words using LDA and display those as the suggested labels.

Thresholds

We defined thresholds for both supervised and unsupervised learning algorithms. This was done to ensure that the user is presented with the most accurate (probable) suggested label(s) for an email. Additionally, in the case of supervised learning where labelled emails are available, however, they might be small in number causing poor performance by the Classifier Chains model. As illustrated in Figure 5.10, our system is enabled to then perform unsupervised learning (using Fuzzy c-means) for the emails present in the inbox. Doing this we ensured to display the most probable suggestions to a user.

5.2.5 Deployment of Model(s)

The supervised and unsupervised models were deployed using the Flask application. Once the models were trained and labels were generated for the emails present in the inbox, the so-formed result was sent back to the Google Apps Script. The result was sent in the form of a nested list where each sublist contained a suggested label and the respective `message_id`. Section 5.1.2 provides further details of displaying the suggestions to the user. However, Inbox Harmony continually improves recommendations based on user feedback. Every time a user accepts or rejects suggested labels, the machine learning models incorporate this feedback to train themselves and to provide better recommendations over time.

5.2.6 Challenges

While developing the back end of our recommendation system we faced a few challenges. Those challenges are described below:

[12] `https://github.com/Jcharis/Python-Machine-Learning/blob/master/Multi_Label_Text_Classification_with_Skmultilearn/Multi-Label%20Classification%20with%20Python%20and%20Scikit-Multilearn-.ipynb`

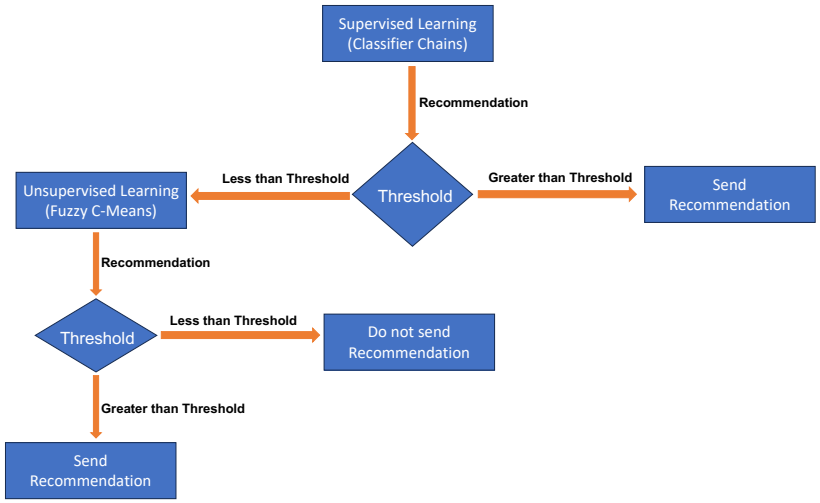


Figure 5.10: Displaying recommendations to users only if they meet the defined threshold, ensuring the display of the most accurate suggestions.

1. *Finding the appropriate web platform to host the Python application* - While we came across many options like Amazon Web Services (AWS), Google Cloud Platform (GCP), Heroku and Microsoft Azure to host our Python application, however, most of these options were paid (which was pretty expensive to make a prototype). Though GCP provided a free trial of three months with some free credits, however, the fast computational services provided by it were causing the rapid depletion of the free credits. After a lot of research, we came across PythonAnywhere which provided us with suitable computational services, free of cost. PythonAnywhere provided us with a web-based Python development environment where we hosted our Flask Application.

2. *Optimisation of machine learning algorithms* - As our application is based on real-time data for individual users, it was difficult to validate the hyper-parameters in our machine learning algorithms. We could have tuned the hyper-parameters if more email data was publicly extracted from various users to train the models. However, this was not possible.

Once we implemented the recommendation system, we enabled the system for a few users to evaluate it. In the following chapter, we discuss the evaluation of our recommendation system.

Chapter 6

Evaluation

In any research project, evaluation plays an important role to demonstrate the validity of the research, ensure the research quality and to identify areas of improvement. It is an essential component which adds reliability and depth to the research presented. While proposing a solution based on the user's experience and preferences for a desired functionality, what better could be to get the solution validated by the end user themselves? Ideally, we wanted to provide the Gmail add-on called *Inbox Harmony* to a group of users to use and play around with the add-on functionality for a couple of weeks. This way, the users will get some hands-on experience and we must better assess the real-world usage. Unfortunately, this was not possible due to certain limitations imposed by Google Apps Script like limited execution time, memory and API calls available for a script in a day. Therefore, we conducted a small qualitative study to get a first impression of the add-on application from a number of different users. The user study included ten participants. The goal of this study was to introduce the end users to the Inbox Harmony add-on, explain all the functionality and features of the add-on, let the users use the functionality for around ten minutes and receive constructive feedback regarding the application. This study involved both online and in-person participation. Four out of the 10 participants were present in person and we explained the system face-to-face. For the remaining 6 participants, media such as video calling and screen sharing were used to describe the system. 5 out of the 10 participants were females and the remaining 5 participants were males. All the participants were aged between 20 to 30. 3 out of the 10 participants were students and the remaining 7 were employed in different sectors like Information Technology and Finance.

The following chapter is divided into two parts. The first part includes the introduction and features illustration of the Inbox Harmony add-on for the users. The second part consists of the user evaluation which further includes two phases. After familiarising themselves with the add-on, users were given ten minutes to test the functionalities of the add-on in their own Gmail accounts. Following the testing, users were interviewed for ten minutes to understand their thoughts on the add-on, get some feedback on the UI and know about the scope of improvement. Subsequently, we asked users to fill out the User Experience Questionnaire (UEQ) to assess the subjective user experience of the add-on.

6.1 Inbox Harmony Illustration

Inbox Harmony displays suggestions in the add-on panel. These are individually personalised for every Gmail account. As shown in Figure 6.1, a user must first give permission to access their Gmail-related data, before being able to use the add-on. The add-on requires the data

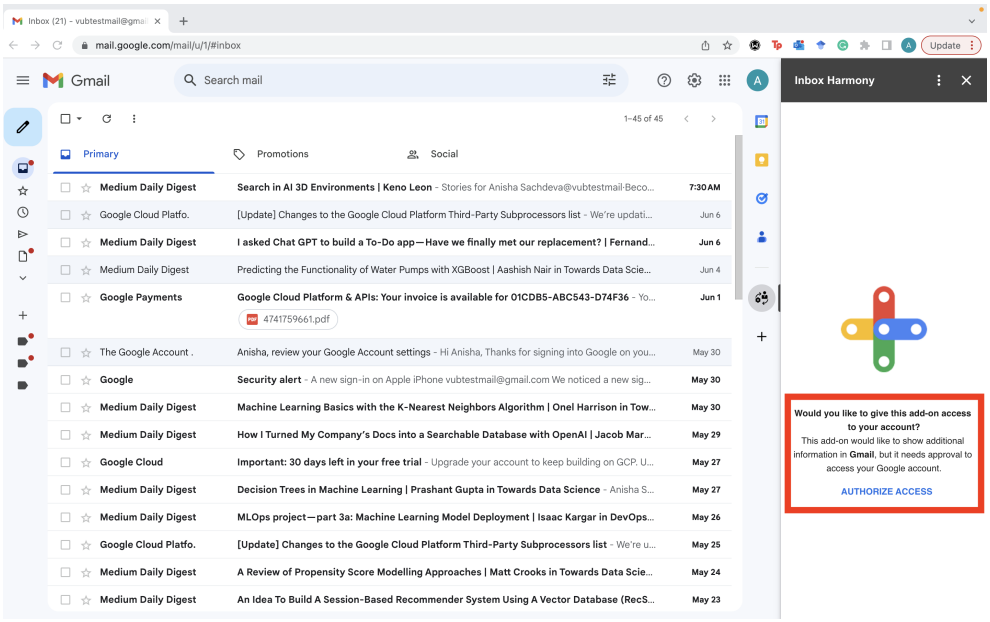


Figure 6.1: Inbox Harmony requesting the user to give the add-on access to their Gmail account

so that it can process the relevant information and provide customised suggestions. Further, as illustrated in Figure 6.2, during the authorisation flow the user is shown the easy-to-understand descriptions of permissions, the add-on is asking for.

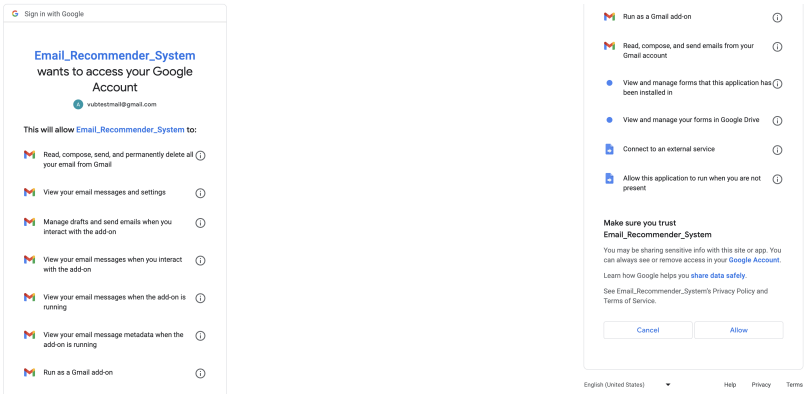


Figure 6.2: Easy to understand permissions displayed to the user

As soon as users grant the required permissions, recommendations are shown in their Gmail interface. The recommendations are chronologically sorted showing the recommendations corresponding to the most recent emails at the top. Sections are generated to display recommendation(s) for individual emails. Each section contains a subject and the sender's information to indicate the email for which suggestions are displayed. Below the indicative email information are the label(s) recommended for that email. These labels can either be already existing in the user's mailbox or can be newly suggested labels by the system. If multiple labels are suggested for an email, then, in addition to the list of labels, an option to select all of them at once is also provided. As highlighted in Figure 6.3, the email with the subject "Security Alert" from the sender "Google" is suggested labels: "GoogleCloudPlatform," *an existing label*, and "alert Google," a *newly proposed label*. An option to select all the labels simultaneously is also available. A user can then select the labels of their choice from the list of displayed recommendations and click on the "Accept Selected Label" button to get the desired movements of emails.

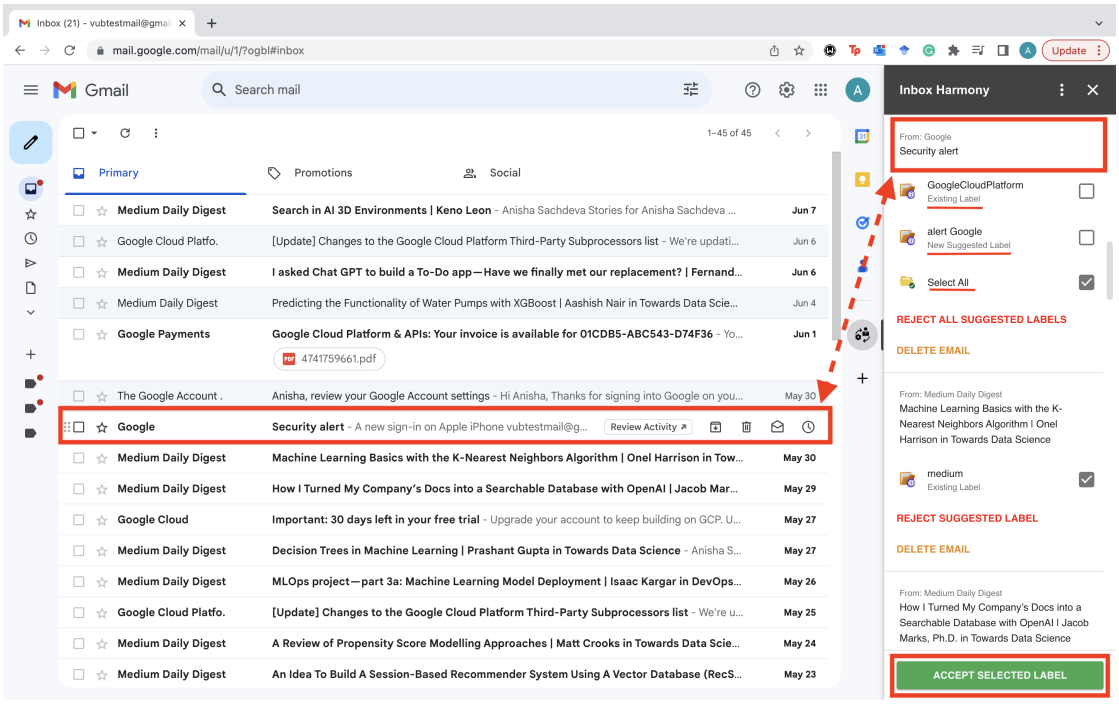


Figure 6.3: Illustration of selecting labels in Inbox Harmony

When a user does not like any of the labels suggested for an email, they can reject all the label(s) by clicking on *Reject All Suggested Labels*. As highlighted in Figure 6.4, when a user clicks on *Reject Suggested Label* for the email with the subject "Machine Learning Basics with the K-Nearest Neighbors Algorithm — Onel Harrison in Towards Data Science" from the sender "Medium Daily Digest", a notification would pop up indicating to the user that the rejected label has been successfully registered in the system and it would not be displayed to the user again. The system will then try to provide improved label suggestions for the email in the future.

When a user wants to delete a particular email they can select the *Delete Email* button provided for each individual email and the email would be moved from the inbox to the trash. As highlighted in Figure 6.5, when a user would click on *Delete Email* for the email with the subject "Security Alert" from the sender "Google", a notification would pop up indicating to

the user about the deletion. Unfortunately, the system can not reload the page itself to show the movement of the email from the inbox to the trash. The user will have to reload the Gmail page themselves to see that the email is no longer visible in their inbox.

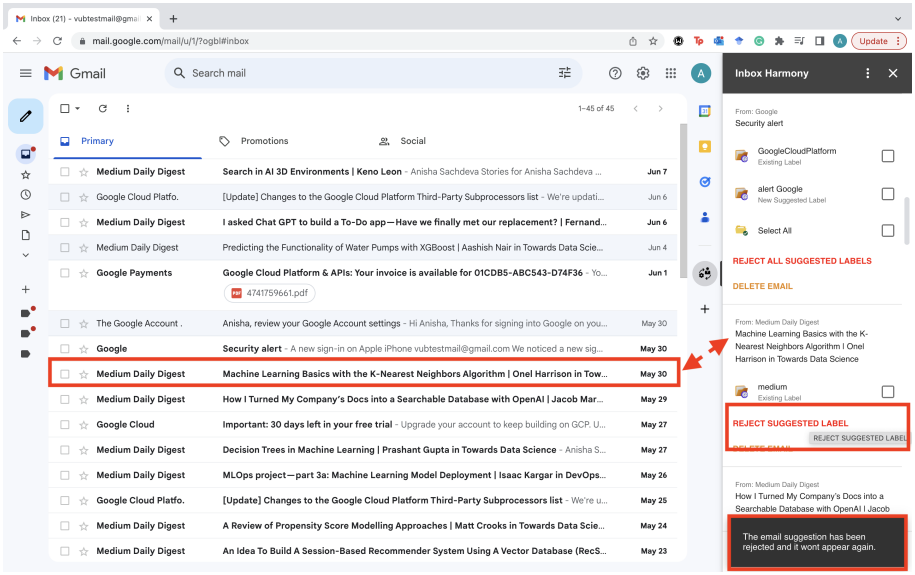


Figure 6.4: Illustration of rejecting a label in Inbox Harmony

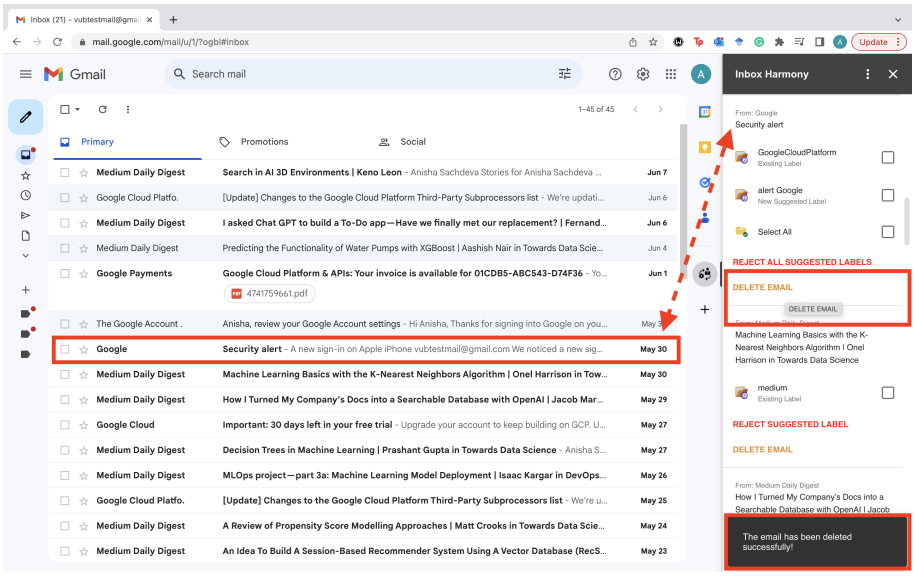


Figure 6.5: Illustration of deleting an email in Inbox Harmony

Figure 6.6 illustrates how a user can see the content of the email when they click on the subject present in Inbox Harmony.

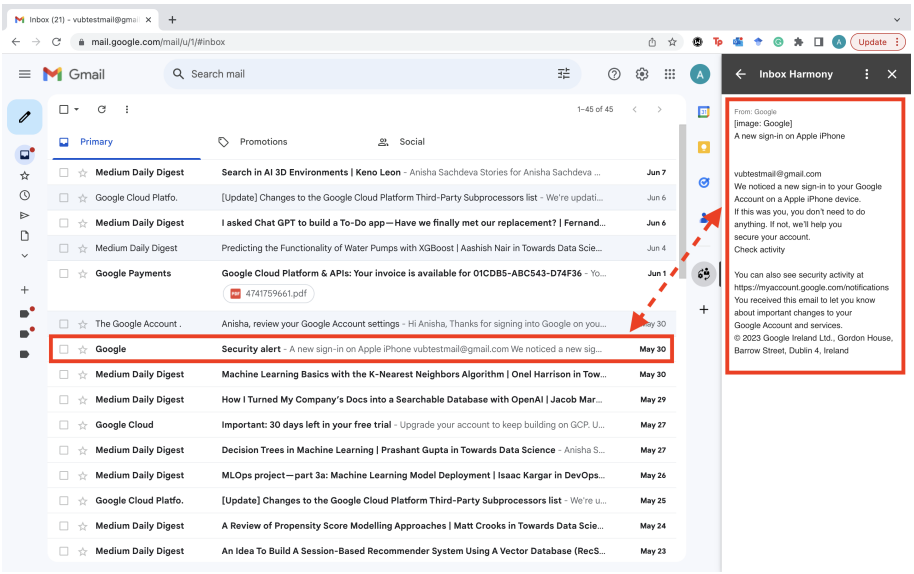


Figure 6.6: Users can view the content by clicking on the subject present in Inbox Harmony

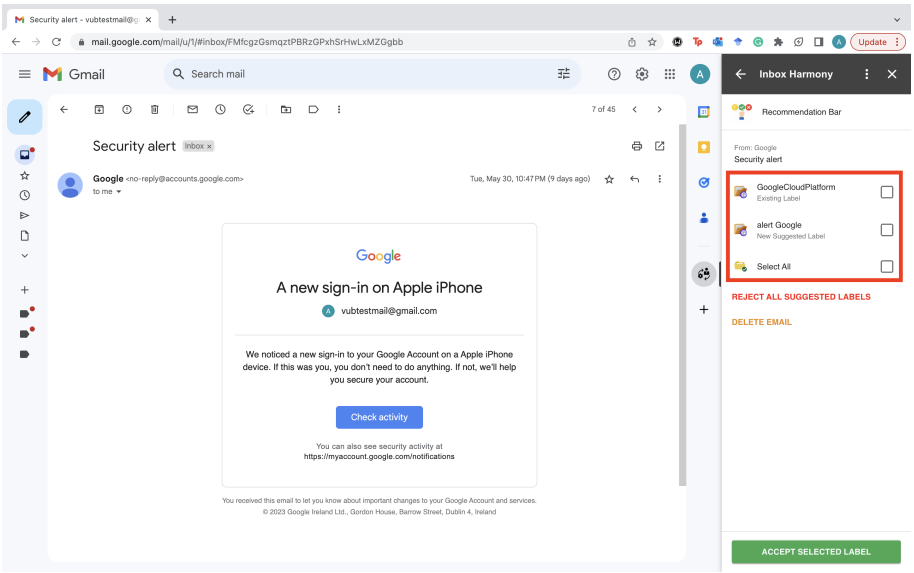


Figure 6.7: Inbox Harmony displays the suggestion for a particular email when it is opened

When a user opens an email from their inbox, the add-on's interface changes dynamically. As shown in Figure 6.7, it only displays the suggested labels for that particular email. However, there might be a case where there are indeed no suggested labels for an email in the inbox. In such cases, the system displays a message indicating to the user that there is no suggestion for

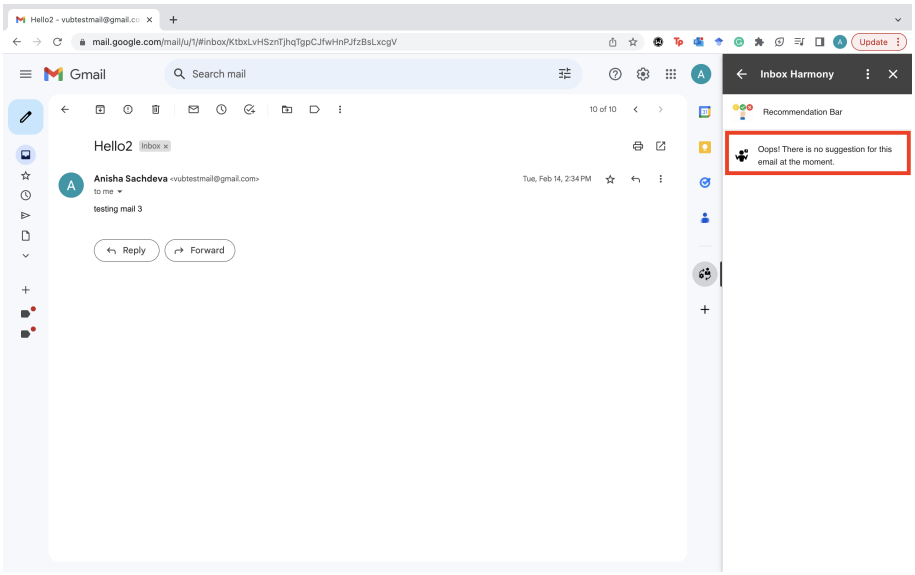


Figure 6.8: Inbox Harmony displays a message notifying the user that there is currently no suggestion for the email

that email at the moment. The message is highlighted in Figure 6.8. Further, as displayed in Figure 6.9, when a user opens an already labelled email, they see the message "Seems like this email is already labelled. Currently, our system suggests label(s) for the emails present only in Inbox."

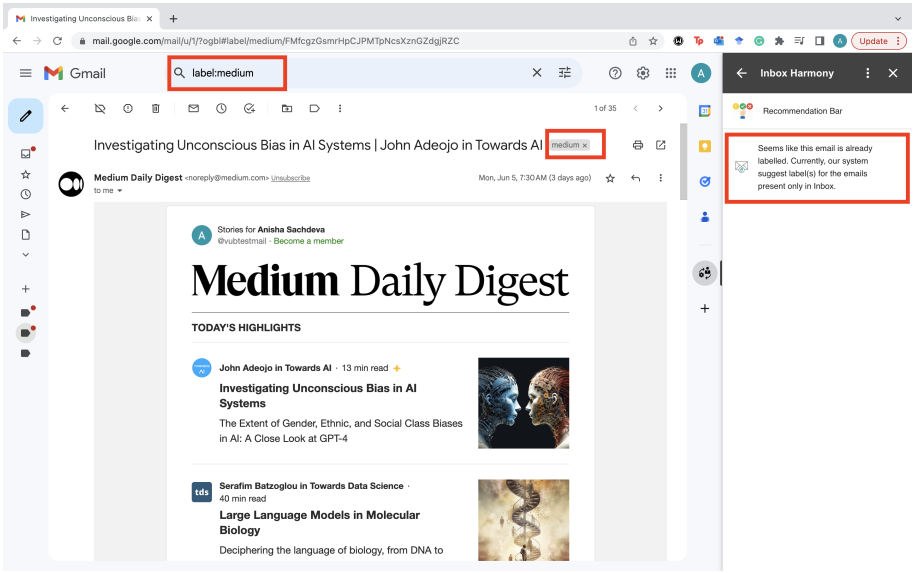


Figure 6.9: Inbox Harmony displays a message notifying the user that the email is already labelled

6.2 Evaluation Activity

After illustrating all the functionalities, the participants used the system in their own Gmail accounts to get a better look and feel of the system. Ideally, we wanted the participants to use the system for a few weeks. This would have enabled us to compare the number of suggestions shown to the users, how many they accept or reject and how well the machine learning algorithm learns over time. Unfortunately, as mentioned earlier this was quite not possible with the limitations posed by Google Apps Script. While we encouraged the participants to explore and use the features themselves for about ten minutes, we acknowledge that actions taken during this limited time frame and in a controlled environment would not fully reflect the real-world usage patterns. However, this was a good time frame for the user to explore the system effectively. Since they were using the add-on on their own Gmail account they could better assess the recommendations displayed to them in terms of relevance and accuracy. They could provide feedback on the functionality, usability and general experience of using the add-on.

Following the exploration and usage of the add-on, we conducted an interview to gather constructive feedback. We started the interview by asking general questions about the Gmail account on which the participant tested the add-on. We observed that all the participants had shared their secondary account details to test the add-on. There were two major reasons why they did not share their primary email id. The first reason was that participants were usually using their Gmail account as their secondary email account. Their work email account or student email account was primary for them. In their Gmail account, they would usually receive promotional emails. The second reason was the participant felt more comfortable sharing the secondary email id as the add-on required to read and extract the content from their account. We observed that the minimum number of emails in a participant's Gmail account was 59, while the maximum number of emails noted was 26k. Except for one participant with 59 emails, all other participants had more than 1000 emails in their inbox. During the interview, we noted that the participant had 59 emails because they would delete all the emails from their inbox every month and they had recently deleted most of the emails from the inbox.

Furthermore, we observed that 4 out of the 10 participants had created labels to manage their inbox. The remaining six participants were not organising their inbox using labels. Of the four participants who had created labels, except one the other three had at least 30 emails in each of the defined labels in their respective mailboxes. For these three participants, they received recommendations to move some emails from inbox to their existing folders. These participants assessed the recommendations to be accurate for classifying emails from inbox to existing labels. These participants then moved some email from their inbox to the existing folders using the add-on. They all were pretty happy with the functionality. There is an interesting case to be mentioned here. Of the six participants who did not have defined any label, two of these were inquisitive enough to create two or three labels and classify some emails in those labels while testing the add-on. They wanted to evaluate the correctness of the system while categorising emails present in the inbox when given some labelled data. They were pleased to see the results. The system very well categorised the unlabelled emails. One of them mentioned this system could even help them to first categorise emails together in a label and then they could mass delete emails at once to clean their inbox. They would not need to search for specific emails and then delete them.

All the participants received suggestions to move some emails from the inbox to some newly suggested labels. Participants were ecstatic to see how the system could group and label similar emails. Out of curiosity, everyone tested to move some emails from the inbox to the newly suggested folder. Though they liked the functionality of how labels were automatically created for them along with moving emails, however, a general comment was made on the naming

of new labels. As an area of improvement, participants suggested generating better names for the labels. Although they were satisfied with the grouping of emails, they could not relate all the emails (within the group) with the suggested label name. One participant suggested using domain names while generating a label's name rather than the most frequent words in the group of subjects. We agree with the feedback on better names for new labels. However, naming a label would always be subjective as per individual preferences.

In Gmail, emails are typically displayed on pages with each page containing a certain number of emails. By default, this number is set as 50 and can range between 10–100. Taking this into consideration, we displayed a maximum of 32 suggestions at any given point in time to ensure that participants could easily access the most recent suggestions for the emails present on the first page. Further, to identify Inbox Harmony's accuracy, we logged the number of accepted and rejected recommendations. We observed that the range of accepted recommendations was between 3–8. One out of the 10 participants had accepted three recommendations and 3 participants had accepted eight recommendations. The one participant who accepted the three recommendations was interested to test the functionality of creating and moving emails from their inbox to some newly suggested folders. The ones who accepted more than five recommendations were utilising Inbox Harmony to organise their mailbox. They found the recommendations to be accurate enough to make changes to their mailbox. Further, we observed that the number of rejected recommendations ranged between 0–5. While 3 participants did not reject any recommendation, one participant rejected 5 recommendations. The reason for rejecting the 5 recommendations was the poor quality of the name of the newly suggested label. They received suggestions for new labels with names that followed a pattern of a numeric value followed by a word. Though they were satisfied with the grouping of emails; however, they suggested improving the names for new labels. Four participants rejected one or two labels to test the rejection functionality. Due to the time constraint on participants, we can say that these actions are not the actual demonstration of using Inbox Harmony. However, the comparatively high rate of acceptance of recommendations indicates the accuracy of recommendations displayed to the users.

Further, we received some more interesting feedback. One of which was to change the folder icon used in the add-on. Participants suggested being consistent with the Gmail icons and using the same label icon[1] as used in the Gmail interface. Though this is a minute change, however, small changes can help a lot to improve the user's experience. Another feedback was to provide an option to filter the emails as per labels. Right now, we display the labels for individual emails. The participant suggested it would be good if we could display emails for individual labels as well. It would help the participant to check which all emails have been automatically grouped by the system and what is the label suggested for that group of emails. We agree with this assessment. With the restrictions formed by Google Apps Script such as no use of HTML/CSS and only the Card Service could be used to design the UI, we were quite limited with the space. Hence, though we thought to add this functionality at the start of the project, we later dropped the idea.

Having discussed all areas of improvement as suggested by the participants, we would like to mention that participants expressed a high level of satisfaction and confidence in the add-on. Participants felt the add-on to be clear, concise and easy to use. One of the participants mentioned using the add-on space efficiently in Gmail's overall interface. Further, we asked the participants if they would like to use the Inbox Harmony add-on in their primary email id. Without any hesitation, they all happily agreed to be taking advantage of the tool if it could be present in their primary email id. 7 out of 10 participants indicated that they would regularly use the add-on, specifically at least once a day. The remaining three users indicated that they

[1] `https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcSHvmyLHnQ_rdS54uFz1ce4gnKPIElF8y9wDw&usqp=CAU`

would use the add-on on a weekly basis, specifically during their email sorting routine.

Following the interview, the participants were requested to fill in a User Experience Questionnaire (UEQ). UEQ is a standardised questionnaire used to assess and evaluate an overall software product. UEQ consists of 26 adjective pairs and a participant is asked to mark their agreement or disagreement with the adjectives on a scale of 1 to 7. These 26 adjective pairs represent six scales which are important for a good user experience. These six scales are:

1. *Attractiveness* - This scale measures the visual appeal and aesthetics of the application.
2. *Perspicuity* - This scale measures the comprehensibility and clarity of the application's functionality.
3. *Efficiency* - This scale measures the effectiveness and efficiency while performing the tasks.
4. *Dependability* - This scale measures a user's trust, confidence and reliance on the application.
5. *Stimulation* - This scale measures the level of motivation, excitement and curiosity that the application can simulate in a user.
6. *Novelty* - The scale measures the originality of the designed solution.

On receiving all the responses from the ten participants, we filled in the raw responses from the PDF(s) to the UEQ tool. Figure 6.10 highlights the raw responses filled in the UEQ tool[2]. Every column in the UEQ corresponds to a question, and each row represents the participant's score for that question. To interpret the results, the UEQ tool automatically computes the mean and variance for the six scales which are mentioned above. The questionnaire is not formed to provide an overall score for the user experience. It provides quantitative data to measure the effectiveness of the six scales and qualitative data to measure the pragmatic and hedonic quality. Pragmatic quality emphasises the functional aspects of the application and hedonic quality emphasises the non-functional or aesthetic aspects of the application.

	Items																									
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
	7	7	3	1	1	7	7	7	1	1	7	1	7	6	6	7	1	1	1	7	1	7	1	1	1	7
	7	6	1	1	1	7	7	6	2	1	7	1	6	7	6	7	3	1	1	7	1	7	1	1	1	7
	6	7	3	1	2	4	5	4	4	3	6	2	7	6	5	6	4	3	3	6	2	6	2	3	4	5
	6	7	2	1	1	6	7	7	4	2	7	1	6	6	6	6	1	2	1	7	2	6	2	1	3	6
	7	7	1	1	1	7	7	7	1	1	7	1	7	7	7	7	1	1	1	7	1	7	1	1	1	7
	7	7	1	1	1	7	7	7	1	1	7	1	7	7	7	7	4	1	1	7	1	7	1	1	1	7
	7	7	4	1	3	4	5	4	6	2	7	1	7	7	5	7	2	3	3	6	2	7	1	4	1	7
	6	6	2	1	1	7	7	6	4	1	6	2	6	7	6	6	1	1	1	7	1	7	2	3	2	7
	7	7	1	2	1	6	7	2	2	1	7	1	7	7	7	7	1	1	1	7	2	7	1	1	1	7
	7	7	2	1	2	7	6	5	1	1	7	2	7	7	7	6	2	2	2	6	1	7	1	2	1	7

Figure 6.10: Raw data filled in the UEQ tool

The UEQ scale for mean varies between the range -3 and +3 with -3 representing highly negative evaluation and +3 representing extremely positive evaluation. Figure 6.11 highlights the mean values for all six parameters. We have received positive responses for all six scales. The mean values for all six scales are ≥ 2.2, which is commendable. Figure 6.12 represents the

[2]For a detailed overview, all the filled-in UEQ forms are attached in Appendix B

UEQ Scales (Mean and Variance)	
Attractiveness	2,550
Perspicuity	2,750
Efficiency	2,400
Dependability	2,200
Stimulation	2,425
Novelty	2,375

Figure 6.11: UEQ Mean Scale

UEQ scale mean values in the form of a graph. As illustrated in Figure 6.13, the attractiveness, pragmatic quality and hedonic quality of Inbox Harmony is quite high. That is Inbox Harmony meets the user's practical needs efficiently with a pleasing user experience.

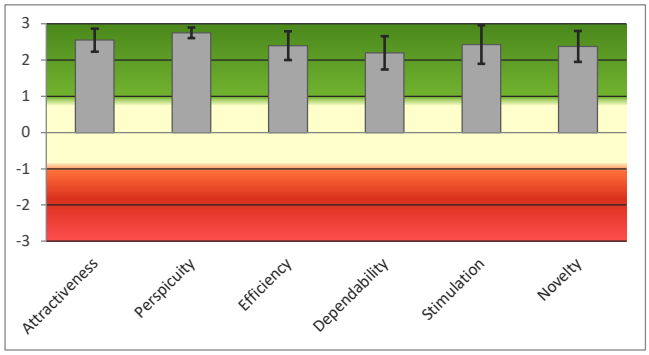


Figure 6.12: UEQ Mean Graph

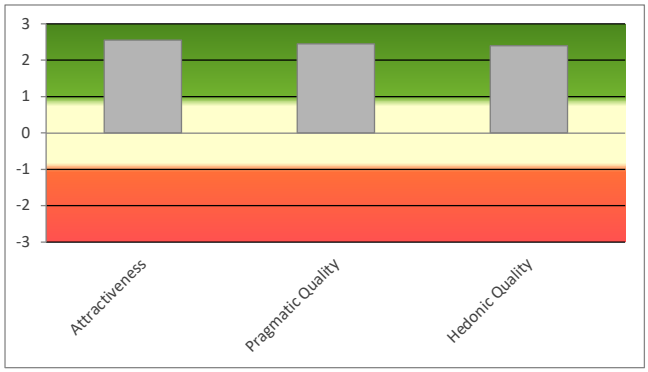


Figure 6.13: UEQ Mean Graph for Pragmatic and Hedonic Quality

Chapter 7

Future Work and Conclusion

In this chapter, we discuss some additional ideas and potential improvements discovered during the implementation and evaluation phase. Finally, we conclude the work performed in this thesis with a thorough discussion of our contributions.

7.1 Future Work

The positive feedback obtained from the evaluation phase affirms the success of Inbox Harmony's design, implementation and usability. However, this research around the recommendation system for emails is a prototype which can be optimised not just for the Gmail email client, but also to support any other major email client. If the add-on is made available for all the major email clients, the evaluation of the recommendation system can be greatly benefited from observing the average number of acceptances and rejections by users. In our current evaluation, we had limited time for users to explore the add-on. However, if users have the opportunity to use it in their daily routine over an extended period (such as weeks), a more comprehensive evaluation can be conducted. Further, in order to inspire future researchers to build on our prototype, we outline a number of key areas of work that could benefit from additional investigation.

7.1.1 Platform Compatibility

One of the implementation choices that we made was to implement the add-on exclusively for Gmail's user interface. With time restrictions, we leveraged Gmail-specific APIs to tightly integrate the add-on with Gmail's UI and features. During the evaluation phase, we observed that users were satisfied with the recommendations they received and they were positive about using this recommendation pane in other email clients such as Microsoft Outlook. Hence, a possible modification could be to design an interface compatible with various email clients. Instead of a Gmail-specific API, it would require standardised protocols (like Internet Messaging Access Protocol) or APIs that are supported by various email clients. With the research being quite generalisable, the machine learning pipeline developed can be leveraged with different APIs to enable users to access the recommendations regardless of the specific email client they use.

7.1.2 Improved UI

We were limited to using the `Card Service` in Google Apps Script to build the add-on. Freedom of choice while designing the UI was missing for us. Thereby, we were restricted to use the partic-

ular size and style of different fonts and buttons offered by the `Card Service`. In the future, we might enhance a user's experience by designing the add-on using technologies like HTML/CSS or JavaScript. The flexibility offered by these technologies can enable the incorporation of different UI designs. For example, in Inbox Harmony due to `Card Service` being stateless and the need to rebuild the card to make any visible interactive changes to the UI, we could not change the colour of buttons on some interaction with the user, and we could also not let disappear the rejected suggestions after a click of the reject button. Using other web development tools, one can overcome these limitations and improve the overall user experience.

7.1.3 Improved Machine Learning Algorithms

Class Imbalance Problem

The class imbalance problem in machine learning refers to a situation where the number of instances belonging to different classes is significantly unequal. in this case, one class (known as the minority class) has remarkably fewer instances compared to another class or classes (known as the majority class or classes). The machine learning algorithms tend to be biased towards the majority class. This type of situation can be encountered in mailboxes. It is possible that a user has created many folders; however, most of the emails are labelled only in a few (say one or two) folders. This can be problematic for the recommendation system as it might not accurately suggest emails to move in the minority label(s). In such cases, synthetic data can be generated by using techniques like SMOTE (Synthetic Minority Over-sampling Technique). SMOTE generates artificial samples by interpolating between existing instances. Solving the class imbalance problem can improve the accuracy of the recommendation system.

Availability of Real-World Email Dataset

The availability of real-world email datasets containing all the information as in a real email account can help to increase the accuracy of the machine learning models used. With the increased training dataset, one can tune the hyperparameters used in the machine learning models. For example, while clustering the emails present in the inbox, the optimal number of clusters can be decided if we would have a bigger dataset to train and test the Fuzzy C-Means model on.

Online Machine Learning Model

To enable feedback learning in our Inbox Harmony prototype, our machine learning models were re-trained every time a user would click on accept or reject button. We had enabled bulk acceptance, i.e. users could select multiple labels at once and then accept all the movement of emails. During this movement of emails, our models were re-trained to provide better results as per the action taken by the user. However, a better approach that can be tried in future would be Online Machine Learning. Here, the models can be trained on an initial dataset to establish its initial state. Later, as the feedback (accept or reject label) arrives in the model, the model can incorporate the feedback in the training phase to update its parameters and make predictions. Such models adapt and learn continuously and incrementally and refine the predictions based on the most recent observations.

Improve Label Names

During the evaluation phase, it was observed that the label names generated by LDA (Latent Dirichlet Allocation, a topic modelling technique in machine learning) could not demonstrate

sufficient generalisation while generating the label names. It would be interesting to investigate further how can we assign meaningful, descriptive yet small label names for a cluster formed by grouping similar emails in the inbox. Though the naming process can be subjective as per individual's choice. However, we assume having the sender's domain name in the newly suggested label could be helpful.

7.2 Conclusion

In conclusion, the development and implementation of Inbox Harmony, a recommendation system to manage and organise emails has emerged as a promising solution to the ever-increasing email overload problem. Throughout this thesis, we have explored the challenges faced by users while managing emails in their mailbox and have proposed a solution in the form of a recommendation system which can provide personalised recommendations to move various emails from the inbox to other folders. More specifically, we first performed a survey to understand existing email management practices. We attempted to understand users' utilisation patterns and behaviours within their email inboxes. Based on the information gained from the empirical study, Inbox Harmony, a recommendation system for Gmail's UI was proposed and developed.

Amongst the contributions, we count the insights gained from the initial survey on the use and management of emails that we conducted. As per our research, since 2011 no empirical study has been conducted to understand whether users are utilising the folder-based email classification system. Despite the existing research emphasising the demand for more automation in email clients, we identified a notable gap in the literature concerning the actual utilisation of the available automation by email clients. We discovered the challenges faced by the users while using the current automation (rule-based filtering) offered by email clients. We then identified users' preferences for a simple system that allows email to first come into the inbox, enabling them to take any required action, and subsequently moving those emails automatically to other relevant folders.

The second contribution is the Inbox Harmony recommendation solution that we developed to be integrated with Gmail's user interface. While developing the recommendation system, we were very well aware of the following two scenarios. In the first scenario, users have created and maintained folders and the second one, users have not created any folders. We designed the machine learning pipeline to efficiently handle both scenarios and provide users with the most relevant recommendations. Additionally, based on our survey we observed that users often find it difficult to categorise an email in a single folder. Taking inspiration from this observation, rather than going for traditional machine learning algorithms, we went one step ahead and chose multi-label classification algorithms. This enabled the system to display multiple label suggestions for an email, expanding the capability to offer a broad range of relevant labels for users to choose from. A participant mentioned in the survey that they do not use rule-based filtering in their mailbox as their inbox is already messy and they cannot clean it. However, our solution is powerful but also simple enough to enable the users to not only filter the incoming emails but also clean an already chaotic inbox. To provide improved recommendations, our system learns from the feedback it receives from users.

Furthermore, we evaluated the system with ten participants where they tested Inbox Harmony's functionalities in their individual Gmail account. They evaluated the system on the basis of the personalised recommendations they received from the system. Their positive response indicated their liking for the easy-to-use yet powerful system to manage their inbox. Participants' enthusiastic feedback and expressed desire to incorporate the recommendation system regularly into their daily routine while using emails served as strong encouragement for us.

To conclude, we have presented the artefacts and strong indication that the presence of a recommendation system as an add-on to an email client's user interface has the potential to encourage users to effectively manage and organise emails in their mailbox. The current Inbox Harmony prototype can act as a solid baseline for further improvements of the user interface and the development of more sophisticated machine learning pipelines.

Bibliography

[1] Manu Aery and Sharma Chakravarthy. "eMailSift: Email Classification Based on Structure and Content". In: *Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 2005)*. Houston, USA, Nov. 2005, p. 8. DOI: 10.1109/ICDM.2005.58.

[2] Qingyao Ai et al. "Characterizing Email Search Using Large-scale Behavioral Logs and Surveys". In: *Proceedings of the 26th International Conference on World Wide Web (WWW 2017)*. Perth, Australia, Apr. 2017, pp. 1511–1520. DOI: 10.1145/3038912.3052615.

[3] Nir Ailon et al. "Threading Machine Generated Email". In: *Proceedings of the 6th ACM International Conference on Web Search and Data Mining (WSDM 2013)*. Rome, Italy, Feb. 2013, pp. 405–414. DOI: 10.1145/2433396.2433447.

[4] Olle Bälter. "Keystroke Level Analysis of Email Message Organization". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2000)*. The Hague, The Netherlands, Apr. 2000, pp. 105–112. DOI: 10.1145/332040.332413.

[5] Victoria Bellotti et al. "Taking Email to Task: The Design and Evaluation of a Task Management Centered Email Tool". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2003)*. Ft. Lauderdale, USA, Apr. 2003, pp. 345–352. DOI: 10.1145/642611.642672.

[6] Gary Boone. "Concept Feature in Re:Agent, an Intelligent Email Agent". In: *Proceedings of the 2nd International Conference on Autonomous Agents (AGENTS 1998)*. St. Paul, USA, May 1998, pp. 141–148. DOI: 10.1145/280765.280791.

[7] Urszula Boryczka and Jan Kozak. "Ant Colony Decision Trees– A New Method for Constructing Decision Trees Based on Ant Colony Optimization". In: *Proceedings of the 2nd International Conference on Computational Collective Intelligence. Technologies and Applications: Second International Conference (ICCCI 2010)*. Kaohsiung, Taiwan, Nov. 2010, pp. 373–382. DOI: 10.1007/978-3-642-16693-8_39.

[8] Urszula Boryczka, Barbara Probierz, and Jan Kozak. "Automatic Categorization of Email into Folders by Ant Colony Decision Tree and Social Networks". In: *Proceedings of the 8th KES International Conference on Intelligent Decision Technologies (KES-IDT 2016)*. Puerto de la Cruz, Spain, June 2016, pp. 71–81. DOI: 10.1007/978-3-319-39627-9_7.

[9] Vannevar Bush. "As We May Think". In: *The Atlantic Monthly* 176.1 (1945), pp. 101–108.

[10] Mark Dredze, Bill N Schilit, and Peter Norvig. "Suggesting Email View Filters for Triage and Search". In: *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI 2009)*. Pasadena, USA, July 2009, pp. 1414–1419.

[11] Mihajlo Grbovic et al. "How Many Folders Do You Really Need? Classifying Email into a Handful of Categories". In: *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, (CIKM 2014)*. Shanghai, China, Nov. 2014, pp. 869–878. DOI: 10.1145/2661829.2662018.

[12] Jacek Gwizdka. "Reinventing the Inbox: Supporting the Management of Pending Tasks in Email". In: *Extended Abstracts of the 2002 Conference on Human Factors in Computing Systems (CHI 2002)*. Minnesota, USA, Apr. 2002, pp. 550–551. DOI: 10.1145/506443.506476.

[13] Sarah Henderson. "Genre, Task, Topic and Time: Facets of Personal Digital Document Management". In: *Proceedings of the 6th ACM SIGCHI New Zealand Chapter's International Conference on Computer-Human Interaction (CHINZ 2005)*. Auckland, New Zealand, July 2005, pp. 75–82. DOI: 10.1145/1073943.1073957.

[14] Yehuda Koren et al. "Automatically Tagging Email by Leveraging Other Users' Folders". In: *Proceedings of the 17th ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD 2011)*. San Diego, USA, Aug. 2011, pp. 913–921. DOI: 10.1145/2020408.2020560.

[15] Mark W Lansdale. "The Psychology of Personal Information Management". In: *Applied Ergonomics* 19.1 (1988), pp. 55–66. DOI: 10.1016/0003-6870(88)90199-8.

[16] Wendy E. Mackay. "Diversity in the Use of Electronic Mail: A Preliminary Inquiry". In: *ACM Transactions on Information Systems (TOIS 1988)* 6.4 (1988), pp. 380–397. DOI: 10.1145/58566.58567.

[17] Thomas W Malone. "How do People Organize their Desks? Implications for the Design of Office Information Systems". In: *ACM Transactions on Information Systems (TOIS 1983)* 1.1 (1983), pp. 99–112. DOI: 10.1145/357423.357430.

[18] Kenrick J. Mock. "An Experimental Framework for Email Categorization and Management". In: *Proceedings of the 24th Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR 2001)*. Louisiana, USA, Sept. 2001, pp. 392–393. DOI: 10.1145/383952.384033.

[19] Soya Park et al. "Opportunities for Automating Email Processing: A Need-Finding Study". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2019)*. Scotland, UK, May 2019, p. 374. DOI: 10.1145/3290605.3300604.

[20] Jesse Read et al. "Classifier Chains for Multi-label Classification". In: *Machine Learning* 85 (2011), pp. 333–359. DOI: 10.1007/s10994-011-5256-5.

[21] Steven L Rohall et al. "Email Visualizations to Aid Communications". In: *Proceedings of The IEEE Symposium on Information Visualization (InfoVis 2001)*. Vol. 12. 2001, p. 15.

[22] Sandra Sudarsky and Rune Hjelsvold. "Visualizing Electronic Mail". In: *International Conference on Information Visualisation (IV 2002)*. London, UK, July 2002, pp. 3–9. DOI: 10.1109/IV.2002.1028749.

[23] Mary Sumner. "The Impact of Electronic Mail on Managerial and Organizational Communications". In: *Proceedings of the ACM SIGOIS and IEEECS TC-OA 1988 Conference on Office Information Systems, (COCS 1988)*. California, USA, Mar. 1988, pp. 96–109. DOI: 10.1145/45410.45421.

[24] Juha Takkinen and Nahid Shahmehri. "CAFE: A Conceptual Model for Managing Information in Electronic Mail". In: *Thirty-First Annual Hawaii International Conference on System Sciences (HICSS 1998)*. Hawaii, USA, Jan. 1998, pp. 44–53. DOI: 10.1109/HICSS.1998.648295.

[25] Gina Venolia et al. "Supporting Email Workflow". In: *MSR-TR-2001-88:Microsoft Research* (2001).

[26] Steve Whittaker and Candace L. Sidner. "Email Overload: Exploring Personal Information Management of Email". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 1996)*. Vancouver, Canada, Apr. 1996, pp. 276–283. DOI: 10.1145/238386.238530.

[27] Steve Whittaker et al. "Am I Wasting My Time Organizing Email?: A Study of Email Refinding". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2011)*. Vancouver, Canada, May 2011, pp. 3449–3458. DOI: 10.1145/1978942.1979457.

[28] T. D. Wilson. "Personal information management (PIM)". In: *SIGIR Forum* 42.2 (2008). DOI: 10.1145/1480506.1480524.

Appendix A

Survey Questionnaire

Email Usage

Start of Block: Block 1

Terms & Conditions Dear volunteer,

Thank you for participating in our research on the use and management of emails. We greatly appreciate your interest and cooperation! This questionnaire will take about *10 to 15 minutes* to complete.

In order to protect your opinions, your answers are treated **confidentially** in accordance with the Belgian and European privacy legislation (cf. AVG or GDPR). All your answers will be processed **anonymously**, so your identity is never revealed.

Research team:
Anisha Sachdeva, Vrije Universiteit Brussel, Web and Information Systems (WISE)
Prof. Dr. Beat Signer, Vrije Universiteit Brussel, Web and Information Systems (WISE)

Contact address:

Anisha Sachdeva
Vrije Universiteit Brussel
DINF WISE
Pleinlaan 9
B-1050 Brussel

Prof. Dr. Beat Signer
Vrije Universiteit Brussel
DINF WISE
Pleinlaan 9
B-1050 Brussel

Email: anisha.sachdeva@vub.be Tel: +32 2 629 12 39

Email: bsigner@vub.be

A note on privacy
This survey is anonymous.
The record kept of your survey responses does not contain any

identifying information about you, unless a specific question in the survey has asked for this. If you have responded to a survey that used an identifying token to allow you to access the survey, you can rest assured that the identifying token is not kept with your responses. It is managed in a separate database, and will only be updated to indicate that you have (or have not) completed this survey. There is no way of matching identification tokens with survey responses in this survey.

End of Block: Block 1

Start of Block: Questionnaire

Q1 Do you use emails?

- Yes (1)
- No (2)

Display This Question:
If Do you use emails? = No

Q2 Why do you not use emails?

_____

Skip To: Q23 If Condition: Why do you not use emails? Is Not Empty. Skip To: Which age group describes you? .

Q3 On which device do you most often use emails? By use, we mean activities like *opening*, *reading*, *replying* or *organising* emails in different folders.

- ○ Mobile phone (1)
- ○ Tablet (2)
- ○ Desktop or laptop computer (3)
- ○ Other (please specify) (4) _____

Q4 Which email client do you mostly use? Note that for this question you can select more than one option.

- ☐ Gmail (1)
- ☐ Outlook (2)
- ☐ Apple Mail (3)
- ☐ Thunderbird (4)
- ☐ Other (please specify) (5) _____

Q5 For what purpose do you usually use emails? Please select a number between 0 (not at all) and 10 (very much) to specify the use of emails for *personal use* as well as *work-related use*.

	0 1 2 3 4 5 6 7 8 9 10
Personal use ()	
Work-related use ()	

Q6 How often do you check your emails?

- ○ Several times an hour (1)
- ○ Several times a day (2)
- ○ Once a day (3)
- ○ Once a week (4)

Q7 Approximately how many emails do you receive per day?

- ○ 0–10 (1)
- ○ 11–50 (2)
- ○ 51–100 (3)
- ○ More than 100 (4)

Q8 Approximately how many emails are currently in your inbox? Note that we only consider your inbox and *not* any subfolders that might be present in your mailbox.

- ○ 0–100 (1)
- ○ 101–500 (2)
- ○ More than 500 (3)

Q9 How do you use your inbox?

- ○ As a to-do list. I only keep those emails in the inbox which require some action; otherwise I move emails to other manually created folders. (1)
- ○ I occasionally move emails from my inbox to other folders. (2)
- ○ I keep emails in my inbox and do not manually move them to other folders. (3)
- ○ Other (please specify) (4) _____

Q10 How do you manage emails in your inbox? Note that for this question you can select more than one option.

- ☐ By moving emails to manually created folders (1)
- ☐ By tagging emails (2)
- ☐ I leave them in the inbox (3)
- ☐ Other (please specify) (4) _____

Display This Question:

If How do you manage emails in your inbox? Note that for this question you can select more than one... != By moving emails to manually created folders

Q11 Do you manually create folders to maintain the inbox?

- ○ Yes (1)
- ○ No (2)

Display This Question:

If How do you manage emails in your inbox? Note that for this question you can select more than one... = By moving emails to manually created folders

Or Do you manually create folders to maintain the inbox? = Yes

Q12 How often do you manually create a folder in your mailbox?

- ○ Several times a day (1)
- ○ Once a day (2)
- ○ Once a week (3)
- ○ Once a month (4)

Q13 How many folders do you have in your mailbox? Note that this includes default folders such as *Junk Email* or *Archive*.

- ○ 0–10 (1)
- ○ 11–50 (2)
- ○ 51–200 (3)
- ○ More than 200 (4)

Display This Question:

If How do you manage emails in your inbox? Note that for this question you can select more than one... = By moving emails to manually created folders

Or Do you manually create folders to maintain the inbox? = Yes

Q14 How often do you move emails from your inbox to other folders?

- ○ Several times an hour (1)
- ○ Several times a day (2)
- ○ Once a day (3)
- ○ Once a week (4)

Display This Question:

If How do you manage emails in your inbox? Note that for this question you can select more than one... = By moving emails to manually created folders

Or Do you manually create folders to maintain the inbox? = Yes

Q15 How much time do you spend every day (on average) to manually sort your inbox?

- ○ Less than 10 minutes (1)
- ○ 10 – 20 minutes (2)
- ○ More than 20 minutes (3)

Q16 Did you know that email clients like Outlook, Gmail or Apple Mail offer the functionality to create a rule according to which emails can be directly moved to different predefined folders rather than storing them all in your inbox? For example, you can create a rule that if an email contains certain words in the subject, then it should be automatically moved to a specific folder.

- ○ Yes (1)
- ○ No (2)

Display This Question:

If Did you know that email clients like Outlook, Gmail or Apple Mail offer the functionality to crea... = Yes

Q17 How often do you create rules to automatically move emails to different folders?

- ○ Very often (1)
- ○ Sometimes (2)
- ○ Never (3)

Display This Question:
If How often do you create rules to automatically move emails to different folders? = Never

Q18 Why do you not create rules to manage the inbox?

- ○ It takes a lot of time to create rules (1)
- ○ I find it difficult to create rules (2)
- ○ Other (please specify) (3) _____

Q19 Do you have suggestions for other types of rules that you would like to have in your email client?

_____

Q20 Do you feel that it is difficult to categorise emails in different folders? For example, you might want to classify an email in two different folders at the same time, when doubting between two (or more) folders for the best classification of your email.

- ○ Yes (1)
- ○ No (2)

Q21 What would you prefer from the following options?

- ○ The system automatically moves emails to a folder when they arrive, based on the rules you defined earlier. Note that this implies that you would never see those emails in your inbox, but only in the specified folders. (1)
- ○ The system automatically classifies (without predefined rules) and moves emails to specific folders. Note that this implies that you would never see those emails in your inbox, but only in the specified folders. (2)
- ○ The system automatically recommends (without predefined rules) and moves emails to specific folders, after asking confirmation from the user. (3)
- ○ I manually drag and drop emails to different folders. (4)

Q22 Consider a situation as shown below. Suppose a user moved an email which contained "Northwind" in the subject to a folder named as "ContasaGolf". Based on this information, there is a recommendation panel present in the email client's interface as shown below on the left-hand side. This recommendation panel then recommends the user to move two other emails which also contain "Northwind" in their subject to the folder "ContasaGolf".

Would you like such a recommendation panel in your email client, which would recommend you to move certain emails to folders based on your previous actions?

- ○ Yes (1)
- ○ No (2)

Page Break

Q23 Which age group do you belong to?

- ○ Younger than 18 (1)
- ○ 18 - 30 (2)
- ○ 31 - 50 (3)
- ○ Older than 50 (4)

Q24 Which gender do you identify most with?

- ○ Male (1)
- ○ Female (2)
- ○ Genderqueer/Non-Binary (3)
- ○ Not Listed (4) _____
- ○ Prefer not to say (5)

Q25 What is your nationality?

▼ Afghan (1) ... Zimbabwean (225)

Q26 Which of the following categories best describes your employment status?

- ○ Student (1)
- ○ Employed (2)
- ○ Non-Employed (3)
- ○ Retired (4)

Display This Question:
If Which of the following categories best describes your employment status? = Employed

Q27 In which of the following sectors do you work?

▼ Architecture and Engineering (1) ... Other (30)

Display This Question:
If In which of the following sectors do you work? = Other

Q28 As you selected *'other'* in the previous question, request you to please specify your occupation in the text box below.

_____

Q29 Do you have any suggestions or comments that you would like to add to this survey?

_____

Display This Question:
If Do you use emails? = Yes

Q30 If you would willing to potentially have a short follow-up discussion in order that we can learn even more about your use of emails, then please leave your email here (your email will only be used for the purpose of this survey and not be further distributed).

_____

Q31 If you would later like to see the results of the survey, then please leave your email here (your email will only be used for the purpose of this survey and not be further distributed).

_____

End of Block: Questionnaire

Appendix B

UEQ Responses

Please make your evaluation now.

For the assessment of the product, please fill out the following questionnaire. The questionnaire consists of pairs of contrasting attributes that may apply to the product. The circles between the attributes represent gradations between the opposites. You can express your agreement with the attributes by ticking the circle that most closely reflects your impression.

Example:

attractive	○	⊗	○	○	○	○	○	unattractive

This response would mean that you rate the application as more attractive than unattractive.

Please decide spontaneously. Don't think too long about your decision to make sure that you convey your original impression.

Sometimes you may not be completely sure about your agreement with a particular attribute or you may find that the attribute does not apply completely to the particular product. Nevertheless, please tick a circle in every line.

It is your personal opinion that counts. Please remember: there is no wrong or right answer!

Please assess the product now by ticking one circle per line.

	1	2	3	4	5	6	7		
annoying	○	○	○	○	○	○	●	enjoyable	1
not understandable	○	○	○	○	○	○	●	understandable	2
creative	○	○	●	○	○	○	○	dull	3
easy to learn	●	○	○	○	○	○	○	difficult to learn	4
valuable	●	○	○	○	○	○	○	inferior	5
boring	○	○	○	○	○	○	●	exciting	6
not interesting	○	○	○	○	○	○	●	interesting	7
unpredictable	○	○	○	○	○	○	●	predictable	8
fast	●	○	○	○	○	○	○	slow	9
inventive	●	○	○	○	○	○	○	conventional	10
obstructive	○	○	○	○	○	○	●	supportive	11
good	●	○	○	○	○	○	○	bad	12
complicated	○	○	○	○	○	○	●	easy	13
unlikable	○	○	○	○	○	●	○	pleasing	14
usual	○	○	○	○	○	●	○	leading edge	15
unpleasant	○	○	○	○	○	○	●	pleasant	16
secure	●	○	○	○	○	○	○	not secure	17
motivating	●	○	○	○	○	○	○	demotivating	18
meets expectations	●	○	○	○	○	○	○	does not meet expectations	19
inefficient	○	○	○	○	○	○	●	efficient	20
clear	●	○	○	○	○	○	○	confusing	21
impractical	○	○	○	○	○	○	●	practical	22
organized	●	○	○	○	○	○	○	cluttered	23
attractive	●	○	○	○	○	○	○	unattractive	24
friendly	●	○	○	○	○	○	○	unfriendly	25
conservative	○	○	○	○	○	○	●	innovative	26

Please assess the product now by ticking one circle per line.

	1	2	3	4	5	6	7		
annoying	○	○	○	○	○	○	●	enjoyable	1
not understandable	○	○	○	○	○	●	○	understandable	2
creative	●	○	○	○	○	○	○	dull	3
easy to learn	●	○	○	○	○	○	○	difficult to learn	4
valuable	●	○	○	○	○	○	○	inferior	5
boring	○	○	○	○	○	○	●	exciting	6
not interesting	○	○	○	○	○	○	●	interesting	7
unpredictable	○	○	○	○	○	●	○	predictable	8
fast	○	●	○	○	○	○	○	slow	9
inventive	●	○	○	○	○	○	○	conventional	10
obstructive	○	○	○	○	○	○	●	supportive	11
good	●	○	○	○	○	○	○	bad	12
complicated	○	○	○	○	○	●	○	easy	13
unlikable	○	○	○	○	○	○	●	pleasing	14
usual	○	○	○	○	○	●	○	leading edge	15
unpleasant	○	○	○	○	○	○	●	pleasant	16
secure	○	○	●	○	○	○	○	not secure	17
motivating	●	○	○	○	○	○	○	demotivating	18
meets expectations	●	○	○	○	○	○	○	does not meet expectations	19
inefficient	○	○	○	○	○	○	●	efficient	20
clear	●	○	○	○	○	○	○	confusing	21
impractical	○	○	○	○	○	○	●	practical	22
organized	●	○	○	○	○	○	○	cluttered	23
attractive	●	○	○	○	○	○	○	unattractive	24
friendly	●	○	○	○	○	○	○	unfriendly	25
conservative	○	○	○	○	○	○	●	innovative	26

Please assess the product now by ticking one circle per line.

	1	2	3	4	5	6	7		
annoying	○	○	○	○	○	◉	○	enjoyable	1
not understandable	○	○	○	○	○	○	◉	understandable	2
creative	○	○	◉	○	○	○	○	dull	3
easy to learn	◉	○	○	○	○	○	○	difficult to learn	4
valuable	○	◉	○	○	○	○	○	inferior	5
boring	○	○	○	◉	○	○	○	exciting	6
not interesting	○	○	○	○	◉	○	○	interesting	7
unpredictable	○	○	○	◉	○	○	○	predictable	8
fast	○	○	○	◉	○	○	○	slow	9
inventive	○	○	◉	○	○	○	○	conventional	10
obstructive	○	○	○	○	○	◉	○	supportive	11
good	○	◉	○	○	○	○	○	bad	12
complicated	○	○	○	○	○	○	◉	easy	13
unlikable	○	○	○	○	○	◉	○	pleasing	14
usual	○	○	○	○	◉	○	○	leading edge	15
unpleasant	○	○	○	○	○	◉	○	pleasant	16
secure	○	○	○	◉	○	○	○	not secure	17
motivating	○	○	◉	○	○	○	○	demotivating	18
meets expectations	○	○	◉	○	○	○	○	does not meet expectations	19
inefficient	○	○	○	○	○	◉	○	efficient	20
clear	○	◉	○	○	○	○	○	confusing	21
impractical	○	○	○	○	○	◉	○	practical	22
organized	○	◉	○	○	○	○	○	cluttered	23
attractive	○	○	◉	○	○	○	○	unattractive	24
friendly	○	○	○	◉	○	○	○	unfriendly	25
conservative	○	○	○	○	◉	○	○	innovative	26

Please assess the product now by ticking one circle per line.

	1	2	3	4	5	6	7		
annoying	○	○	○	○	○	●	○	enjoyable	1
not understandable	○	○	○	○	○	○	●	understandable	2
creative	○	●	○	○	○	○	○	dull	3
easy to learn	●	○	○	○	○	○	○	difficult to learn	4
valuable	●	○	○	○	○	○	○	inferior	5
boring	○	○	○	○	○	●	○	exciting	6
not interesting	○	○	○	○	○	○	●	interesting	7
unpredictable	○	○	○	○	○	○	●	predictable	8
fast	○	○	○	●	○	○	○	slow	9
inventive	○	●	○	○	○	○	○	conventional	10
obstructive	○	○	○	○	○	○	●	supportive	11
good	●	○	○	○	○	○	○	bad	12
complicated	○	○	○	○	○	●	○	easy	13
unlikable	○	○	○	○	○	●	○	pleasing	14
usual	○	○	○	○	○	●	○	leading edge	15
unpleasant	○	○	○	○	○	●	○	pleasant	16
secure	●	○	○	○	○	○	○	not secure	17
motivating	○	●	○	○	○	○	○	demotivating	18
meets expectations	●	○	○	○	○	○	○	does not meet expectations	19
inefficient	○	○	○	○	○	○	●	efficient	20
clear	○	●	○	○	○	○	○	confusing	21
impractical	○	○	○	○	○	●	○	practical	22
organized	○	●	○	○	○	○	○	cluttered	23
attractive	●	○	○	○	○	○	○	unattractive	24
friendly	○	○	●	○	○	○	○	unfriendly	25
conservative	○	○	○	○	○	●	○	innovative	26

Please assess the product now by ticking one circle per line.

	1	2	3	4	5	6	7		
annoying	○	○	○	○	○	○	○	enjoyable	1
not understandable	○	○	○	○	○	○	○	understandable	2
creative	○	○	○	○	○	○	○	dull	3
easy to learn	○	○	○	○	○	○	○	difficult to learn	4
valuable	○	○	○	○	○	○	○	inferior	5
boring	○	○	○	○	○	○	○	exciting	6
not interesting	○	○	○	○	○	○	○	interesting	7
unpredictable	○	○	○	○	○	○	○	predictable	8
fast	○	○	○	○	○	○	○	slow	9
inventive	○	○	○	○	○	○	○	conventional	10
obstructive	○	○	○	○	○	○	○	supportive	11
good	○	○	○	○	○	○	○	bad	12
complicated	○	○	○	○	○	○	○	easy	13
unlikable	○	○	○	○	○	○	○	pleasing	14
usual	○	○	○	○	○	○	○	leading edge	15
unpleasant	○	○	○	○	○	○	○	pleasant	16
secure	○	○	○	○	○	○	○	not secure	17
motivating	○	○	○	○	○	○	○	demotivating	18
meets expectations	○	○	○	○	○	○	○	does not meet expectations	19
inefficient	○	○	○	○	○	○	○	efficient	20
clear	○	○	○	○	○	○	○	confusing	21
impractical	○	○	○	○	○	○	○	practical	22
organized	○	○	○	○	○	○	○	cluttered	23
attractive	○	○	○	○	○	○	○	unattractive	24
friendly	○	○	○	○	○	○	○	unfriendly	25
conservative	○	○	○	○	○	○	○	innovative	26

Please assess the product now by ticking one circle per line.

	1	2	3	4	5	6	7		
annoying	○	○	○	○	○	○	●	enjoyable	1
not understandable	○	○	○	○	○	○	●	understandable	2
creative	●	○	○	○	○	○	○	dull	3
easy to learn	●	○	○	○	○	○	○	difficult to learn	4
valuable	●	○	○	○	○	○	○	inferior	5
boring	○	○	○	○	○	○	●	exciting	6
not interesting	○	○	○	○	○	○	●	interesting	7
unpredictable	○	○	○	○	○	○	●	predictable	8
fast	●	○	○	○	○	○	○	slow	9
inventive	●	○	○	○	○	○	○	conventional	10
obstructive	○	○	○	○	○	○	●	supportive	11
good	●	○	○	○	○	○	○	bad	12
complicated	○	○	○	○	○	○	●	easy	13
unlikable	○	○	○	○	○	○	●	pleasing	14
usual	○	○	○	○	○	○	●	leading edge	15
unpleasant	○	○	○	○	○	○	●	pleasant	16
secure	○	○	○	●	○	○	○	not secure	17
motivating	●	○	○	○	○	○	○	demotivating	18
meets expectations	●	○	○	○	○	○	○	does not meet expectations	19
inefficient	○	○	○	○	○	○	●	efficient	20
clear	●	○	○	○	○	○	○	confusing	21
impractical	○	○	○	○	○	○	●	practical	22
organized	●	○	○	○	○	○	○	cluttered	23
attractive	●	○	○	○	○	○	○	unattractive	24
friendly	●	○	○	○	○	○	○	unfriendly	25
conservative	○	○	○	○	○	○	●	innovative	26

Please assess the product now by ticking one circle per line.

	1	2	3	4	5	6	7		
annoying	○	○	○	○	○	○	○	enjoyable	1
not understandable	○	○	○	○	○	○	○	understandable	2
creative	○	○	○	○	○	○	○	dull	3
easy to learn	○	○	○	○	○	○	○	difficult to learn	4
valuable	○	○	○	○	○	○	○	inferior	5
boring	○	○	○	○	○	○	○	exciting	6
not interesting	○	○	○	○	○	○	○	interesting	7
unpredictable	○	○	○	○	○	○	○	predictable	8
fast	○	○	○	○	○	○	○	slow	9
inventive	○	○	○	○	○	○	○	conventional	10
obstructive	○	○	○	○	○	○	○	supportive	11
good	○	○	○	○	○	○	○	bad	12
complicated	○	○	○	○	○	○	○	easy	13
unlikable	○	○	○	○	○	○	○	pleasing	14
usual	○	○	○	○	○	○	○	leading edge	15
unpleasant	○	○	○	○	○	○	○	pleasant	16
secure	○	○	○	○	○	○	○	not secure	17
motivating	○	○	○	○	○	○	○	demotivating	18
meets expectations	○	○	○	○	○	○	○	does not meet expectations	19
inefficient	○	○	○	○	○	○	○	efficient	20
clear	○	○	○	○	○	○	○	confusing	21
impractical	○	○	○	○	○	○	○	practical	22
organized	○	○	○	○	○	○	○	cluttered	23
attractive	○	○	○	○	○	○	○	unattractive	24
friendly	○	○	○	○	○	○	○	unfriendly	25
conservative	○	○	○	○	○	○	○	innovative	26

Please assess the product now by ticking one circle per line.

	1	2	3	4	5	6	7		
annoying	○	○	○	○	○	⊖	○	enjoyable	1
not understandable	○	○	○	○	○	⊖	○	understandable	2
creative	○	⊖	○	○	○	○	○	dull	3
easy to learn	⊖	○	○	○	○	○	○	difficult to learn	4
valuable	⊖	○	○	○	○	○	○	inferior	5
boring	○	○	○	○	○	○	⊖	exciting	6
not interesting	○	○	○	○	○	○	⊖	interesting	7
unpredictable	○	○	○	○	○	⊖	○	predictable	8
fast	○	○	○	⊖	○	○	○	slow	9
inventive	⊖	○	○	○	○	○	○	conventional	10
obstructive	○	○	○	○	○	⊖	○	supportive	11
good	○	⊖	○	○	○	○	○	bad	12
complicated	○	○	○	○	○	⊖	○	easy	13
unlikable	○	○	○	○	○	○	⊖	pleasing	14
usual	○	○	○	○	○	⊖	○	leading edge	15
unpleasant	○	○	○	○	○	⊖	○	pleasant	16
secure	⊖	○	○	○	○	○	○	not secure	17
motivating	⊖	○	○	○	○	○	○	demotivating	18
meets expectations	⊖	○	○	○	○	○	○	does not meet expectations	19
inefficient	○	○	○	○	○	○	⊖	efficient	20
clear	⊖	○	○	○	○	○	○	confusing	21
impractical	○	○	○	○	○	○	⊖	practical	22
organized	○	⊖	○	○	○	○	○	cluttered	23
attractive	○	○	⊖	○	○	○	○	unattractive	24
friendly	○	⊖	○	○	○	○	○	unfriendly	25
conservative	○	○	○	○	○	○	⊖	innovative	26

Please assess the product now by ticking one circle per line.

	1	2	3	4	5	6	7		
annoying	○	○	○	○	○	○	●	enjoyable	1
not understandable	○	○	○	○	○	○	●	understandable	2
creative	●	○	○	○	○	○	○	dull	3
easy to learn	○	●	○	○	○	○	○	difficult to learn	4
valuable	●	○	○	○	○	○	○	inferior	5
boring	○	○	○	○	○	●	○	exciting	6
not interesting	○	○	○	○	○	○	●	interesting	7
unpredictable	○	●	○	○	○	○	○	predictable	8
fast	○	●	○	○	○	○	○	slow	9
inventive	●	○	○	○	○	○	○	conventional	10
obstructive	○	○	○	○	○	○	●	supportive	11
good	●	○	○	○	○	○	○	bad	12
complicated	○	○	○	○	○	○	●	easy	13
unlikable	○	○	○	○	○	○	●	pleasing	14
usual	○	○	○	○	○	○	●	leading edge	15
unpleasant	○	○	○	○	○	○	●	pleasant	16
secure	●	○	○	○	○	○	○	not secure	17
motivating	●	○	○	○	○	○	○	demotivating	18
meets expectations	●	○	○	○	○	○	○	does not meet expectations	19
inefficient	○	○	○	○	○	○	●	efficient	20
clear	○	●	○	○	○	○	○	confusing	21
impractical	○	○	○	○	○	○	●	practical	22
organized	●	○	○	○	○	○	○	cluttered	23
attractive	●	○	○	○	○	○	○	unattractive	24
friendly	●	○	○	○	○	○	○	unfriendly	25
conservative	○	○	○	○	○	○	●	innovative	26

Please assess the product now by ticking one circle per line.

	1	2	3	4	5	6	7		
annoying	○	○	○	○	○	○	■	enjoyable	1
not understandable	○	○	○	○	○	○	■	understandable	2
creative	○	■	○	○	○	○	○	dull	3
easy to learn	■	○	○	○	○	○	○	difficult to learn	4
valuable	○	■	○	○	○	○	○	inferior	5
boring	○	○	○	○	○	○	■	exciting	6
not interesting	○	○	○	○	○	■	○	interesting	7
unpredictable	○	○	○	○	■	○	○	predictable	8
fast	■	○	○	○	○	○	○	slow	9
inventive	■	○	○	○	○	○	○	conventional	10
obstructive	○	○	○	○	○	○	■	supportive	11
good	○	■	○	○	○	○	○	bad	12
complicated	○	○	○	○	○	○	■	easy	13
unlikable	○	○	○	○	○	○	■	pleasing	14
usual	○	○	○	○	○	○	■	leading edge	15
unpleasant	○	○	○	○	○	■	○	pleasant	16
secure	○	■	○	○	○	○	○	not secure	17
motivating	○	■	○	○	○	○	○	demotivating	18
meets expectations	○	■	○	○	○	○	○	does not meet expectations	19
inefficient	○	○	○	○	○	■	○	efficient	20
clear	■	○	○	○	○	○	○	confusing	21
impractical	○	○	○	○	○	○	■	practical	22
organized	■	○	○	○	○	○	○	cluttered	23
attractive	○	■	○	○	○	○	○	unattractive	24
friendly	■	○	○	○	○	○	○	unfriendly	25
conservative	○	○	○	○	○	○	■	innovative	26