**Intelligent E-mail Prediction System**

A PROJECT REPORT SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

BACHELOR OF TECHNOLOGY IN

INFORMATION TECHNOLOGY

SUBMITTED BY

ANKIT GUPTA 2K10/IT/016

CHIRAG GUPTA 2K10/IT/023

SANCHIT GUPTA 2K10/IT/056

VASEEM AHMED KHAN 2K10/IT/068

UNDER THE GUIDANCE

OF

Dr. N S Raghava

Associate Professor



DEPARTMENT OF INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY

(FORMERLY DELHI COLLEGE OF ENGINEERING) MAY 2014

**CERTIFICATE**

This is to certify that Project Report entitled **" Intelligent E-mail Prediction System”** submitted by **Ankit Gupta, Chirag Gupta, Sanchit Gupta** and **Vaseem Ahmed Khan** for partial fulfilment of the requirement for the award of degree Bachelor of Engineering in Department of Computer Engineering is a record of the candidate work carried out by them under my supervision.

Prof N S Raghava

Assistant Professor

Information Technology Department

Delhi Technological University

**ACKNOWLEDGEMENT**

It gives us a great sense of pleasure to present the report of the B.E Project titled **Intelligent E-mail Prediction System** undertaken during our Final Year. We owe special debt of gratitude to Prof NS Raghava, Department of Information Technology, Delhi Technological University, Delhi for his constant support and guidance throughout the course of our work. His sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavours have seen light of the day.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind assistance and cooperation during the development of our project.

**Signature Signature**

**Name** Ankit Gupta **Name** Chirag Gupta

**Roll No**. 2K10/IT/016 **Roll No**. 2K10/IT/023

**Signature Signature**

**Name** Sanchit Gupta **Name** Vaseem Ahmed Khan

**Roll No**. 2K10/IT/056 **Roll No**. 2K10/IT/068

**ABSTRACT**

Nowadays, email has become one of the most critical personal and business applications and email users would experience serious consequences if email messages could not be available or experience high volume of messages which lead to congestions, overloads and limited storage space coupled with unstructured messages in mail boxes.

A few years ago, the means of communication are via letters by post, telegraph, fax, couriers to mention a few but now the focus has changed to a faster means of obtaining quick responses and faster ways of communication-emails. We propose a new framework to help organised and prioritized email better; An intelligent email prediction system (IEPS).

The goal is to organise emails better in mail boxes, prioritise emails based on the focus of the email content. The intelligent email prediction system helps to improve email users’ performances, saves time, very effective and efficient tool and is cost effective for businesses and for personal use.

The system is evaluated against a corpus of human-judged predictions, reaching satisfactory level of performance.

|  |
| --- |
|  |
| **TABLE OF CONTENTS** |
|  |
|  |
|  |
| |  |  |  |  |  | | --- | --- | --- | --- | --- | | *Certificate* |  |  |  | *ii* | | *Acknowledgement* |  |  |  | *iii* | | *Abstract* |  |  |  | *iv* | | *Table of Content(s)* |  |  |  | *v* | | *List of Figure(s)* |  |  |  | *vi* | | **Chapter 1 Introduction** |  |  |  | 1 | | **Chapter 2 E-Mail** |  |  |  | 2 | | 2.1 Operation Overview |  |  |  | 3 | | 2.2 Message Format |  |  |  | 4 | | 2.3 Types |  |  |  | 8 | | 2.4 Uses |  |  |  | 9 | | **Chapter 3 Machine Learning** |  |  |  | 14 | | 3.1 Introduciton |  |  |  | 14 | | 3.2 Algorithm Types |  |  |  | 15 | | 3.3 Approaches |  |  |  | 20 | | 3.4 Applications |  |  |  | 22 | | 3.5 Extracting features from unstructured data |  |  |  | 23 | | **Chapter 4 Machine Learning Algorithms** |  |  |  | 25 | | 4.1 Logistic Regression |  |  |  | 26 | | 4.2 Support Vector Machine |  |  |  | 27 | | 4.3 Random Forest |  |  |  | 28 | | 4.4 Decision Tree |  |  |  | 30 | | **Chapter 5 Implementation** |  |  |  | 32 | | **Chapter 6 Evaluation And Results** |  |  |  | 37 | | **Conclusion** |  |  |  | 44 | | **References** |  |  |  | 45 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **LIST OF FIGURES**   |  |  | | --- | --- | | 1.How email works | 3 | | 2 Email header | 13 | | 3 Decision boundary | 15 | | 4 supervised learning | 17 | | 5 Unsupervised learning | 18 | | 6 The logistic function | 26 | | 7 Project approach | 33 | | 8 Login | 39 | | 9 E-Mail Data | 39 | | 10 JSon Training Data | 40 | | 11.Final Result Obtained | 40 | | 12.Features Extracted from email database | 41 | | 13. Features Extracted from email database | 42 | |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

**Chapter 1**

**Introduction**

The aim of IEPS(intelligent email prediction system) is to help email users save lots of time while checking, reading and searching for email messages reduce email overloads and congestions which are caused as a results of high volume of email messages in mail box, organise mail boxes better and all these makes life easier and improves user’s performances and productivity.

One of the lines of work developed within IEPS is the use of machine learning techniques for information management namely, text classification of email messages and determination of messages that needs reply and why. IEPS is an automated system that learns to determine whether email messages received in a mail box needs a reply or no action is to be taken.

Whittaker and Sidner analyzed the use of email to perform task management, personal archiving, and asynchronous communication and referred to the three as “email overload”. They concluded:

(1) Users perform a large variety of work-related tasks with email.

(2) As a result, users are overwhelmed with the amount of information in their mailbox.

A quotation from interviews conducted by Whittaker characterizes some frustrations:

“Waiting to hear back from another ... employee can mean delays in accomplishing a particular task, which can ... have significant impact on our overall operations. ... it can be critical or just frustrating.”

“One of my pet-peeves is when someone does not get back to me, but I am one of the worst offenders. I get so many emails ... that I cannot keep up.”

IEPS will enable email users to both manage their email inboxes and at the same time manage their time more efficiently. The existing solutions by Tyler explained that “regular user of email has, at one time or another, sent a message and wondered, “When will I get a response to this email?” Or, “How long should I wait for a response to this message before taking further action?” This work grew from the belief that an interesting, relatively unexplored aspect of email usage is its implicit timing information”.

**Chapter 2**

**E-Mail**

Electronic mail, most commonly referred to as email or e-mail since ca. 1993, is a method of exchanging digital messages from an author to one or more recipients. Modern email operates across the Internet or other computer networks. Some early email systems required that the author and the recipient both be online at the same time, in common with instant messaging. Today's email systems are based on a store-and-forward model. Email servers accept, forward, deliver, and store messages. Neither the users nor their computers are required to be online simultaneously; they need connect only briefly, typically to a mail server, for as long as it takes to send or receive messages.

Historically, the term electronic mail was used generically for any electronic document transmission. For example, several writers in the early 1970s used the term to describe fax document transmission. As a result, it is difficult to find the first citation for the use of the term with the more specific meaning it has today.

An Internet email message consists of three components, the message envelope, the message header, and the message body. The message header contains control information, including, minimally, an originator's email address and one or more recipient addresses. Usually descriptive information is also added, such as a subject header field and a message submission date/time stamp.

Originally a text-only (ASCII) communications medium, Internet email was extended to carry, e.g. text in other character sets, multi-media content attachments, a process standardized in RFC 2045 through 2049. Collectively, these RFCs have come to be called Multipurpose Internet Mail Extensions (MIME). Subsequent RFC's have proposed standards for internationalized email addresses using UTF-8.

Electronic mail predates the inception of the Internet and was in fact a crucial tool in creating it, but the history of modern, global Internet email services reaches back to the early ARPANET. Standards for encoding email messages were proposed as early as 1973. Conversion from ARPANET to the Internet in the early 1980s produced the core of the current services. An email sent in the early 1970s looks quite similar to a basic text message sent on the Internet today.

Email is an information and communications technology. It uses technology to communicate a digital message over the Internet. Users use email differently, based on how they think about it. There are many software platforms available to send and receive. Popular email platforms include Gmail, Hotmail, Yahoo! Mail, Outlook, and many others.

Network-based email was initially exchanged on the ARPANET in extensions to the File Transfer Protocol (FTP), but is now carried by the Simple Mail Transfer Protocol (SMTP), first published as Internet standard 10 in 1982. In the process of transporting email messages between systems, SMTP communicates delivery parameters using a message envelope separate from the message (header and body) itself.

**2.1 Operation Overview**

The diagram to the right shows a typical sequence of events that takes place when Alice composes a message using her mail user agent (MUA). She enters the email address of her correspondent, and hits the "send" button.

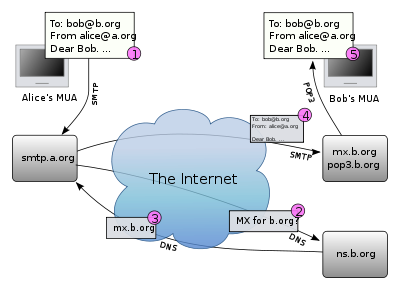


Figure 1

1. Her MUA formats the message in email format and uses the Submission Protocol (a profile of the Simple Mail Transfer Protocol (SMTP), to send the message to the local mail submission agent (MSA), in this case smtp.a.org, run by Alice's internet service provider (ISP).
2. The MSA looks at the destination address provided in the SMTP protocol (not from the message header), in this case bob@b.org. An Internet email address is a string of the form localpart@exampledomain. The part before the @ sign is the local part of the address, often the username of the recipient, and the part after the @ sign is a domain name or a fully qualified domain name. The MSA resolves a domain name to determine the fully qualified domain name of the mail server in the Domain Name System (DNS).
3. The DNS server for the b.org domain, ns.b.org, responds with any MX records listing the mail exchange servers for that domain, in this case mx.b.org, a message transfer agent (MTA) server run by Bob's ISP.
4. smtp.a.org sends the message to mx.b.org using SMTP.

This server may need to forward the message to other MTAs before the message reaches the final message delivery agent (MDA).

1. The MDA delivers it to the mailbox of the user bob.
2. Bob presses the "get mail" button in his MUA, which picks up the message using either the Post Office Protocol (POP3) or the Internet Message Access Protocol (IMAP).

**2.2 Message Format**

The Internet email message format is now defined by RFC 5322, with multi-media content attachments being defined in RFC 2045 through RFC 2049, collectively called Multipurpose Internet Mail Extensions or MIME. RFC 5322 replaced the earlier RFC 2822 in 2008, and in turn RFC 2822 in 2001 replaced RFC 822 – which had been the standard for Internet email for nearly 20 years. Published in 1982, RFC 822 was based on the earlier RFC 733 for the ARPANET.

Internet email messages consist of two major sections:

Header – Structured into fields such as From, To, CC, Subject, Date, and other information about the email.

Body – The basic content, as unstructured text; sometimes containing a signature block at the end. This is exactly the same as the body of a regular letter.

The header is separated from the body by a blank line.

**Message header**

Each message has exactly one header, which is structured into fields. Each field has a name and a value.

Informally, each line of text in the header that begins with a printable character begins a separate field. The field name starts in the first character of the line and ends before the separator character ":". The separator is then followed by the field value (the "body" of the field). The value is continued onto subsequent lines if those lines have a space or tab as their first character. Field names and values are restricted to 7-bit ASCII characters. Non-ASCII values may be represented using MIME encoded words.

**Header fields**

Email header fields can be multi-line, and each line should be at most 78 characters long and in no event more than 998 characters long. Header fields defined by RFC 5322 can only contain US-ASCII characters; for encoding characters in other sets, a syntax specified in RFC 2047 can be used. Recently the IETF EAI working group has defined some standards track extensions, replacing previous experimental extensions, to allow UTF-8 encoded Unicode characters to be used within the header. In particular, this allows email addresses to use non-ASCII characters. Such characters must only be used by servers that support these extensions.

The message header must include at least the following fields:

**From**: The email address, and optionally the name of the author(s). In many email clients not changeable except through changing account settings.

**Date**: The local time and date when the message was written. Like the From: field, many email clients fill this in automatically when sending. The recipient's client may then display the time in the format and time zone local to him/her.

The message header should include at least the following fields:

**Message-ID**: Also an automatically generated field; used to prevent multiple delivery and for reference in In-Reply-To: (see below).

**In-Reply-To**: Message-ID of the message that this is a reply to. Used to link related messages together. This field only applies for reply messages.

RFC 3864 describes registration procedures for message header fields at the IANA; it provides for permanent and provisional message header field names, including also fields defined for MIME, netnews, and http, and referencing relevant RFCs.

Common header fields for email include:

**To**: The email address(es), and optionally name(s) of the message's recipient(s). Indicates primary recipients (multiple allowed), for secondary recipients see Cc: and Bcc: below.

**Subject**: A brief summary of the topic of the message. Certain abbreviations are commonly used in the subject, including "RE:" and "FW:".

**Bcc**: Blind Carbon Copy; addresses added to the SMTP delivery list but not (usually) listed in the message data, remaining invisible to other recipients.

**Cc**: Carbon Copy; Many email clients will mark email in one's inbox differently depending on whether they are in the To: or Cc: list.

**Content-Type**: Information about how the message is to be displayed, usually a MIME type.

**Precedence**: commonly with values "bulk", "junk", or "list"; used to indicate that automated "vacation" or "out of office" responses should not be returned for this mail, e.g. to prevent vacation notices from being sent to all other subscribers of a mailing list. Sendmail uses this header to affect prioritization of queued email, with "Precedence: special-delivery" messages delivered sooner. With modern high-bandwidth networks delivery priority is less of an issue than it once was. Microsoft Exchange respects a fine-grained automatic response suppression mechanism, the X-Auto-Response-Suppress header.

**References**: Message-ID of the message that this is a reply to, and the message-id of the message the previous reply was a reply to, etc.

**Reply-To**: Address that should be used to reply to the message.

**Sender**: Address of the actual sender acting on behalf of the author listed in the From: field (secretary, list manager, etc.).

**Archived-At**: A direct link to the archived form of an individual email message.

Note that the To: field is not necessarily related to the addresses to which the message is delivered. The actual delivery list is supplied separately to the transport protocol, SMTP, which may or may not originally have been extracted from the header content. The "To:" field is similar to the addressing at the top of a conventional letter which is delivered according to the address on the outer envelope. In the same way, the "From:" field does not have to be the real sender of the email message. Some mail servers apply email authentication systems to messages being relayed. Data pertaining to server's activity is also part of the header, as defined below.

SMTP defines the trace information of a message, which is also saved in the header using the following two fields:

**Received**: when an SMTP server accepts a message it inserts this trace record at the top of the header (last to first).

**Return**-**Path**: when the delivery SMTP server makes the final delivery of a message, it inserts this field at the top of the header.

Other header fields that are added on top of the header by the receiving server may be called trace fields, in a broader sense.

**Authentication-Results**: when a server carries out authentication checks, it can save the results in this field for consumption by downstream agents

**Received-SPF**: stores results of SPF checks in more detail than Authentication-Results

Auto-Submitted: is used to mark automatically generated messages

**VBR-Info**: claims VBR whitelisting

**Message body**

**Content encoding**

Email was originally designed for 7-bit ASCII.[66] Most email software is 8-bit clean but must assume it will communicate with 7-bit servers and mail readers. The MIME standard introduced character set specifiers and two content transfer encodings to enable transmission of non-ASCII data: quoted printable for mostly 7 bit content with a few characters outside that range and base64 for arbitrary binary data. The 8BITMIME and BINARY extensions were introduced to allow transmission of mail without the need for these encodings, but many mail transport agents still do not support them fully. In some countries, several encoding schemes coexist; as the result, by default, the message in a non-Latin alphabet language appears in non-readable form (the only exception is coincidence, when the sender and receiver use the same encoding scheme). Therefore, for international character sets, Unicode is growing in popularity.

**Plain text and HTML**

Most modern graphic email clients allow the use of either plain text or HTML for the message body at the option of the user. HTML email messages often include an automatically generated plain text copy as well, for compatibility reasons.

Advantages of HTML include the ability to include in-line links and images, set apart previous messages in block quotes, wrap naturally on any display, use emphasis such as underlines and italics, and change font styles. Disadvantages include the increased size of the email, privacy concerns about web bugs, abuse of HTML email as a vector for phishing attacks and the spread of malicious software.

Some web based Mailing lists recommend that all posts be made in plain-text, with 72 or 80 characters per line for all the above reasons, but also because they have a significant number of readers using text-based email clients such as Mutt.

Some Microsoft email clients allow rich formatting using RTF, but unless the recipient is guaranteed to have a compatible email client this should be avoided.

In order to ensure that HTML sent in an email is rendered properly by the recipient's client software, an additional header must be specified when sending: "Content-type: text/html". Most email programs send this header automatically.

**2.3 Types**

Web-based email (webmail)

Many email providers have a web-based email client (e.g. AOL Mail, Gmail, Outlook.com and Yahoo! Mail). This allows users to log into the email account by using any compatible web browser to send and receive their email. Mail is typically not downloaded to the client, so can't be read without a current Internet connection.

POP3 email services

POP3 is the acronym for Post Office Protocol 3. In a POP3 email account, email messages are downloaded to the client device (i.e. a computer) and then they are deleted from the mail server. It is difficult to save and view messages on multiple devices. Also, the messages sent from the computer are not copied to the Sent Items folder on the devices. The messages are deleted from the server to make room for more incoming messages. POP supports simple download-and-delete requirements for access to remote mailboxes (termed maildrop in the POP RFC's). Although most POP clients have an option to leave messages on the server after downloading a copy of them, most e-mail clients using POP3 simply connect, retrieve all messages, store them on the client device as new messages, delete them from the server, and then disconnect.

IMAP email servers

IMAP refers to Internet Message Access Protocol. With an IMAP account, a user's account has access to mail folders on the mail server and can use any compatible device to read messages, as long as such a device can access the server. It shows the headers of messages, the sender and the subject and the device needs to request to download specific messages. Usually mail is left in folders in the mail server.

MAPI email servers

Messaging Application Programming Interface (MAPI) is a messaging architecture and a Component Object Model based API for Microsoft Windows.

**2.4 Uses**

**Flaming**

Flaming occurs when a person sends a message with angry or antagonistic content. The term is derived from the use of the word Incendiary to describe particularly heated email discussions. Flaming is assumed to be more common today because of the ease and impersonality of email communications: confrontations in person or via telephone require direct interaction, where social norms encourage civility, whereas typing a message to another person is an indirect interaction, so civility may be forgotten.

**Email bankruptcy**

Also known as "email fatigue", email bankruptcy is when a user ignores a large number of email messages after falling behind in reading and answering them. The reason for falling behind is often due to information overload and a general sense there is so much information that it is not possible to read it all. As a solution, people occasionally send a boilerplate message explaining that the email inbox is being cleared out. Harvard University law professor Lawrence Lessig is credited with coining this term, but he may only have popularized it.

**In business**

Email was widely accepted by the business community as the first broad electronic communication medium and was the first 'e-revolution' in business communication. Email is very simple to understand and like postal mail, email solves two basic problems of communication: logistics and synchronization (see below).

LAN based email is also an emerging form of usage for business. It not only allows the business user to download mail when offline, it also allows the small business user to have multiple users' email IDs with just one email connection.

**Pros**

The problem of logistics: Much of the business world relies upon communications between people who are not physically in the same building, area or even country; setting up and attending an in-person meeting, telephone call, or conference call can be inconvenient, time-consuming, and costly. Email provides a way to exchange information between two or more people with no set-up costs and that is generally far less expensive than physical meetings or phone calls.

The problem of synchronisation: With real time communication by meetings or phone calls, participants have to work on the same schedule, and each participant must spend the same amount of time in the meeting or call. Email allows asynchrony: each participant may control their schedule independently.

**Cons**

This section possibly contains original research. Please improve it by verifying the claims made and adding inline citations. Statements consisting only of original research should be removed. (June 2009)

Most business workers today spend from one to two hours of their working day on email: reading, ordering, sorting, 're-contextualizing' fragmented information, and writing email.[77] The use of email is increasing worldwide:

**Information overload**: Email is a push technology – the sender controls who receives the information. Convenient availability of mailing lists and use of "copy all" can lead to people receiving unwanted or irrelevant information of no use to them.

**Inconsistency**: Email can duplicate information. This can be a problem when a large team is working on documents and information while not in constant contact with the other members of their team.

Despite these disadvantages, email has become the most widely used medium of communication within the business world. In fact, a 2010 study on workplace communication, found that 83% of U.S. knowledge workers felt that email was critical to their success and productivity at work.[78]

**Research on email marketing**

Marketing research suggests that opt-in email marketing can be viewed as useful by consumers if it contains information such as special sales offerings and new product information. Offering interesting hyperlinks or generic information on consumer trends is less useful.[79] This research by Martin et al. (2003) also shows that if consumers find email marketing useful, they are likely to visit a store, thereby overcoming limitations of Internet marketing such as not being able to touch or try on a product.

**Outside of Business**

Email users attach a higher level of formality, in regards to other ICTs. Users tended to associate the medium for communicating with professors, bosses, and those who they maintained professional relationships with; the degree of formality attached to email varies across users, so email is neither a more formal or less formal ICT than others.[80]

All users of email use the medium differently. With structural functionalism, people will attach various meanings to influence how they use the medium. For example, while one person might use email to communicate with their friends or fellow students on a weekly basis, and another may use it to keep in touch with family members on a daily basis. The situational setting (for example, a student will find him or herself in a different situation than a stay at home mom) in which the user finds himself or herself in shapes how and what email will be used for.

**Mobile**

Email has become widely used on smart phones. Mobile apps for email increase accessibility to the medium. While before users could only access email on computers, it is now possible for users to check their email out of the home and out of the library while on the go. Alerts can also be sent to the phone to notify them immediately of new messages. This has given email the ability to be used for more frequent communication between users and allowed them to check their email and write messages throughout the day.

It was found that US adults check their email more than they browse the web or check their Facebook, making email the most popular activity for users to do on their smart phones. 78% of the respondents in the study revealed that they check their email on their phone.[81] It was also found that 30% of consumers use only their smartphone to check their email, and 91% were likely to check their email at least once a day on their smartphone. However, the percentage of consumers using email on smartphone ranges and differs dramatically across different countries. For example, in comparison to 75% of those consumers in the US who used it, only 17% in India did.

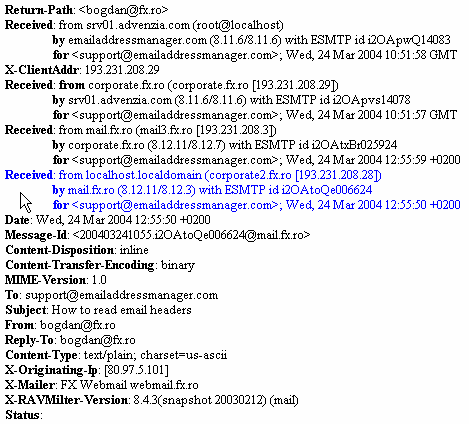


Figure 2

**Chapter 3**

**Machine Learning**

**3.1 Introduction**

Machine learning, a branch of artificial intelligence, concerns the construction and study of systems that can learn from data. For example, a machine learning system could be trained on email messages to learn to distinguish between spam and non-spam messages. After learning, it can then be used to classify new email messages into spam and non-spam folders.  
  
The core of machine learning deals with representation and generalization. Representation of data instances and functions evaluated on these instances are part of all machine learning systems. Generalization is the property that the system will perform well on unseen data instances; the conditions under which this can be guaranteed are a key object of study in the subfield of computational learning theory.

Machine Learning is about building programs with tunable parameters (typically an array of floating point values) that are adjusted automatically so as to improve their behavior by adapting to previously seen data.

Machine Learning can be considered a subfield of Artificial Intelligence since those algorithms can be seen as building blocks to make computers learn to behave more intelligently by somehow generalizing rather that just storing and retrieving data items like a database system would do.

A very simple example of a machine learning task can be seen in the following figure: it shows a collection of two-dimensional data, colored according to two different class labels. A classification algorithm is used to draw a dividing boundary between the two clusters of points:

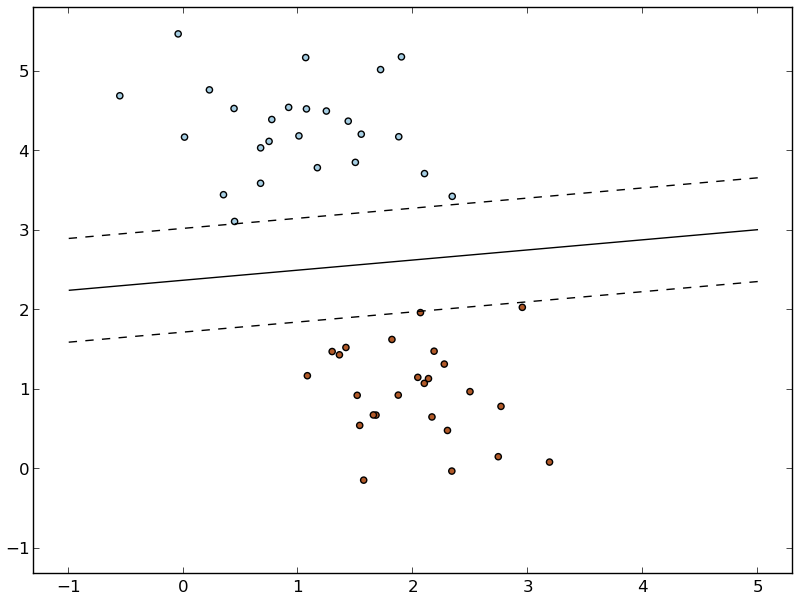
****

Figure 3

**3.2 Algorithm Types**

**Supervised Learning**

Machine learning can be broken into two broad regimes: supervised learning and unsupervised learning. We’ll introduce these concepts here, and discuss them in more detail below.

In Supervised Learning, we have a dataset consisting of both *features* and *labels*. The task is to construct an estimator which is able to predict the label of an object given the set of features. A relatively simple example is predicting the species of iris given a set of measurements of its flower. This is a relatively simple task. Some more complicated examples are:

* given a multicolor image of an object through a telescope, determine whether that object is a star, a quasar, or a galaxy.
* given a photograph of a person, identify the person in the photo.
* given a list of movies a person has watched and their personal rating of the movie, recommend a list of movies they would like (A famous example is the Netflix Prize).

What these tasks have in common is that there is one or more unknown quantities associated with the object which needs to be determined from other observed quantities. Supervised learning is further broken down into two categories,*classification* and *regression*. In classification, the label is discrete, while in regression, the label is continuous. For example, in astronomy, the task of determining whether an object is a star, a galaxy, or a quasar is a classification problem: the label is from three distinct categories. On the other hand, we might wish to determine the age of an object based on such observations: this would be a regression problem: the label (age) is a continuous quantity.

**Unsupervised Learning** addresses a different sort of problem. Here the data has no labels, and we are interested in finding similarities between the objects in question. In a sense, you can think of unsupervised learning as a means of discovering labels from the data itself. Unsupervised learning comprises tasks such as dimensionality reduction, clustering, and density estimation. For example, in the iris data discussed above, we can used unsupervised methods to determine combinations of the measurements which best display the structure of the data. As we’ll see below, such a projection of the data can be used to visualize the four-dimensional dataset in two dimensions. Some more involved unsupervised learning problems are:

* given detailed observations of distant galaxies, determine which features or combinations of features are most important in distinguishing between galaxies.
* given a mixture of two sound sources (for example, a person talking over some music), separate the two (this is called the blind source separation problem).
* given a video, isolate a moving object and categorize in relation to other moving objects which have been seen.

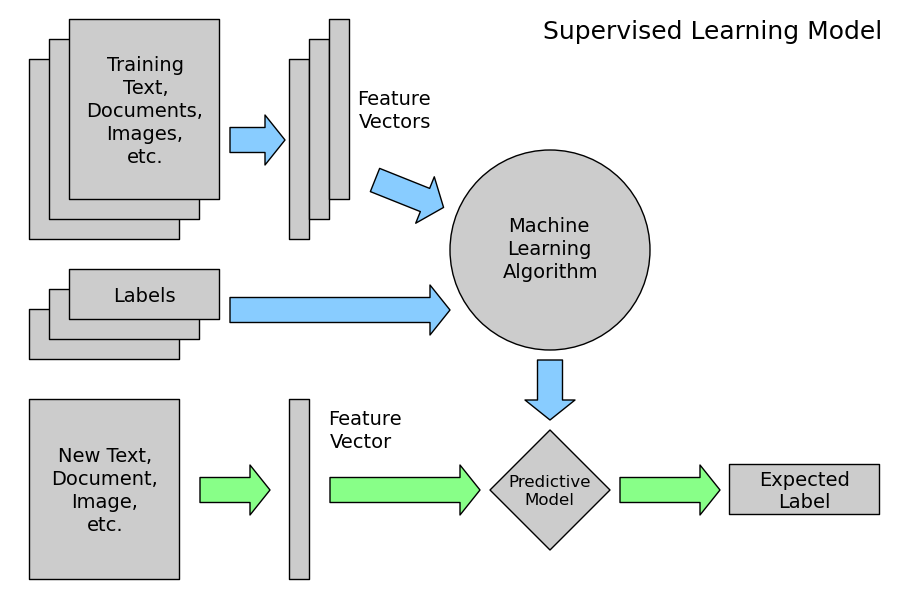
[](http://www.astroml.org/sklearn_tutorial/auto_examples/plot_ML_flow_chart.html)

Figure 4

**Regression**

Regression is the task of predicting the value of a continuously varying variable (e.g. a price, a temperature, a conversion rate...) given some input variables (a.k.a. the features, “predictors” or “regressors”). We’ll explore a detailed example of regression with astronomical data in *Regression: Photometric Redshifts of Galaxies*.

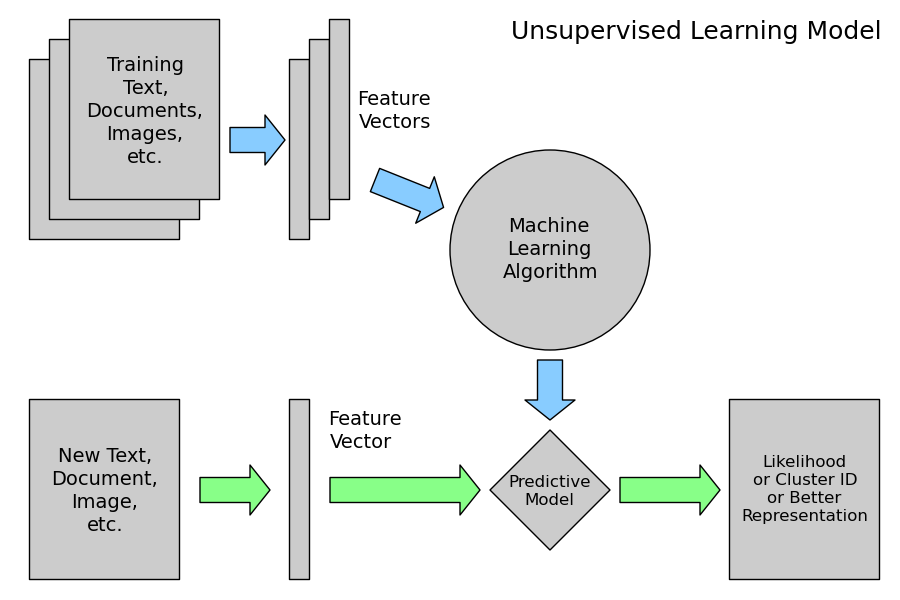
[](http://www.astroml.org/sklearn_tutorial/auto_examples/plot_ML_flow_chart.html)

Figure 5

Unsupervised Learning overview

An unsupervised learning algorithm only uses a single set of observations X with shape (n\_samples, n\_features) and does not use any kind of labels.

An unsupervised learning model will try to fit its parameters so as to best summarize regularities found in the data.

The following introduces the main variants of unsupervised learning algorithms, namely dimensionality reduction and clustering.

**Clustering**

Clustering is the task of gathering samples into groups of similar samples according to some predefined similarity or dissimilarity measure (such as the Euclidean distance).

Machine learning and data mining

These two terms are commonly confused, as they often employ the same methods and overlap significantly. They can be roughly defined as follows:

* Machine learning focuses on prediction, based on *known* properties learned from the training data.
* Data mining focuses on the discovery of (previously) *unknown* properties in the data. This is the analysis step of Knowledge Discovery in Databases.

The two areas overlap in many ways: data mining uses many machine learning methods, but often with a slightly different goal in mind. On the other hand, machine learning also employs data mining methods as "unsupervised learning" or as a preprocessing step to improve learner accuracy. Much of the confusion between these two research communities (which do often have separate conferences and separate journals, ECML PKDD being a major exception) comes from the basic assumptions they work with: in machine learning, performance is usually evaluated with respect to the ability to *reproduce known* knowledge, while in Knowledge Discovery and Data Mining (KDD) the key task is the discovery of previously *unknown* knowledge. Evaluated with respect to known knowledge, an uninformed (unsupervised) method will easily be outperformed by supervised methods, while in a typical KDD task, supervised methods cannot be used due to the unavailability of training data.

**Theory**

The computational analysis of machine learning algorithms and their performance is a branch of theoretical computer science known as computational learning theory. Because training sets are finite and the future is uncertain, learning theory usually does not yield guarantees of the performance of algorithms. Instead, probabilistic bounds on the performance are quite common.

In addition to performance bounds**,** computational learning theorists study the time complexity and feasibility of learning. In computational learning theory, a computation is considered feasible if it can be done in polynomial time. There are two kinds of time complexity results. Positive results show that a certain class of functions can be learned inpolynomial time. Negative results show that certain classes cannot be learned in polynomial time.

**3.3 Approaches**

**Decision tree learning**

**Decision tree learning** uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value. It is one of the predictive modelling approaches used in statistics, data mining and machine learning. More descriptive names for such tree models are **classification trees** or **regression trees**. In these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels.

In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data but not decisions; rather the resulting classification tree can be an input for decision making. This page deals with decision trees in data mining.

Decision tree learning is a method commonly used in data mining.[1] The goal is to create a model that predicts the value of a target variable based on several input variables. An example is shown on the right. Each interior node corresponds to one of the input variables; there are edges to children for each of the possible values of that input variable. Each leaf represents a value of the target variable given the values of the input variables represented by the path from the root to the leaf

**Association rule learning**

Association rule learning is a popular and well researched method for discovering interesting relations between variables in large databases. It is intended to identify strong rules discovered in databases using different measures of interestingness. Based on the concept of strong rules, Rakesh Agrawal et al. introduced association rules for discovering regularities between products in large-scale transaction data recorded by point-of-sale (POS) systems in supermarkets. For example, the rule  found in the sales data of a supermarket would indicate that if a customer buys onions and potatoes together, he or she is likely to also buy hamburger meat. Such information can be used as the basis for decisions about marketing activities such as, e.g., promotional pricing or product placements. In addition to the above example from market basket analysis association rules are employed today in many application areas including Web usage mining, intrusion detection, Continuous production, and bioinformatics. As opposed to sequence mining, association rule learning typically does not consider the order of items either within a transaction or across transactions

**Artificial neural networks**

In computer science and related fields, **artificial neural networks** (**ANNs**) are computational models inspired by animals' central nervous systems (in particular the brain) that are capable of machine learning and pattern recognition. They are usually presented as systems of interconnected "neurons" that can compute values from inputs by feeding information through the network.

For example, in a neural network for handwriting recognition, a set of input neurons may be activated by the pixels of an input image representing a letter or digit. The activations of these neurons are then passed on, weighted and transformed by some function determined by the network's designer, to other neurons, etc., until finally an output neuron is activated that determines which character was read.

Like other machine learning methods, neural networks have been used to solve a wide variety of tasks that are hard to solve using ordinary rule-based programming, including computer vision and speech recognition.

**Inductive logic programming**

**Inductive logic programming** (**ILP**) is a subfield of machine learning which uses logic programming as a uniform representation for examples, background knowledge and hypotheses. Given an encoding of the known background knowledge and a set of examples represented as a logical database of facts, an ILP system will derive a hypothessed logic program which entails all the positive and none of the negative examples.

Schema: *positive examples* + *negative examples* + *background knowledge* => *hypothesis*.

Inductive logic programming is particularly useful in bioinformatics and natural language processing. The term *Inductive Logic Programming* was first introduced in a paper by Stephen Muggleton in 1991. The term "*inductive*" here refers to philosophical (i.e. suggesting a theory to explain observed facts) rather than mathematical (i.e. proving a property for all members of a well-ordered set) induction.

**3.4 Applications**

Applications for machine learning include:

* Machine perception
* Computer vision, including object recognition
* Natural language processing
* Syntactic pattern recognition
* Search engines
* Medical diagnosis
* Bioinformatics
* Brain-machine interfaces
* Cheminformatics
* Detecting credit card fraud
* Stock market analysis
* Classifying DNA sequences
* Sequence mining
* Speech and handwriting recognition
* Game playing
* Software engineering
* Adaptive websites
* Robot locomotion
* Computational advertising
* Computational finance
* Structural health monitoring
* Sentiment analysis (or opinion mining)
* Affective computing

**3.5 Extracting features from unstructured data**

The previous example deals with features that are readily available in a structured dataset with rows and columns of numerical or categorical values.

However, most of the produced data is not readily available in a structured representation such as SQL, CSV, XML, JSON or RDF.

Here is an overview of strategies to turn unstructed data items into arrays of numerical features.

|  |  |
| --- | --- |
| **Text documents:** | Count the frequency of each word or pair of consecutive words in each document. This approach is called Bag of Words  Note: we include other file formats such as HTML and PDF in this category: an ad-hoc preprocessing step is required to extract the plain text in UTF-8 encoding for instance. |
| **Images:** | * Rescale the picture to a fixed size and **take all the raw pixels values** (with or without luminosity normalization) * Take some transformation of the signal (gradients in each pixel, wavelets transforms...) * Compute the Euclidean, Manhattan or cosine **similarities of the sample to a set reference prototype images** aranged in a code book. The code book may have been previously extracted from the same dataset using an unsupervised learning algorithm on the raw pixel signal.   Each feature value is the distance to one element of the code book.   * Perform **local feature extraction**: split the picture into small regions and perform feature extraction locally in each area.   Then combine all the features of the individual areas into a single array. |
| **Sounds:** | Same strategy as for images within a 1D space instead of 2D |

Practical implementations of such feature extraction strategies will be presented in the last sections of this tutorial.

**CHAPTER 4**

**Machine Learning Algorithms**

**4.1 Logistic Regression**

In statistics, logistic regression or logit regression is a type of probabilistic statistical classification model. It is also used to predict a binary response from a binary predictor, used for predicting the outcome of a categorical dependent variable (i.e., a class label) based on one or more predictor variables (features). That is, it is used in estimating empirical values of the parameters in a qualitative response model. The probabilities describing the possible outcomes of a single trial are modeled, as a function of the explanatory (predictor) variables, using a logistic function. Frequently (and subsequently in this article) "logistic regression" is used to refer specifically to the problem in which the dependent variable is binary—that is, the number of available categories is two—and problems with more than two categories are referred to as multinomial logistic regression or, if the multiple categories are ordered, as ordered logistic regression.

Logistic regression can be binomial or multinomial. Binomial or binary logistic regression deals with situations in which the observed outcome for a dependent variable can have only two possible types (for example, "dead" vs. "alive"). Multinomial logistic regression deals with situations where the outcome can have three or more possible types (e.g., "disease A" vs. "disease B" vs. "disease C"). In binary logistic regression, the outcome is usually coded as "0" or "1", as this leads to the most straightforward interpretation. If a particular observed outcome for the dependent variable is the noteworthy possible outcome (referred to as a "success" or a "case") it is usually coded as "1" and the contrary outcome (referred to as a "failure" or a "non case") as "0". Logistic regression is used to predict the odds of being a case based on the values of the independent variables (predictors). The odds are defined as the probability that a particular outcome is a case divided by the probability that it is a non-case.

An explanation of logistic regression begins with an explanation of the logistic function, which always takes on values between zero and one:

F(t) = \frac{e^t}{e^t+1} = \frac{1}{1+e^{-t}},

and viewing t as a linear function of an explanatory variable x (or of a linear combination of explanatory variables), the logistic function can be written as:

F(x) = \frac {1}{1+e^{-(\beta_0 + \beta_1 x)}}.

This will be interpreted as the probability of the dependent variable equaling a "success" or "case" rather than a failure or non-case. We also define the inverse of the logistic function, the logit:

g(x) = \ln \frac{F(x)}{1 - F(x)} = \beta_0 + \beta_1 x ,

and equivalently:

g(x) = \ln \frac{F(x)}{1 - F(x)} = \beta_0 + \beta_1 x ,

A graph of the logistic function F(x) is shown in Figure:

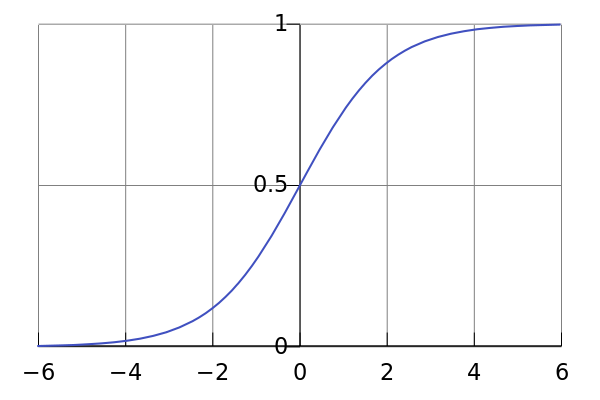


Figure 6

**4.2 Support Vector Machine**

In machine learning, support vector machines (SVMs, also support vector networks) are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on.

More formally, a support vector machine constructs a hyper plane or set of hyper planes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyper plane that has the largest distance to the nearest training data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.

Whereas the original problem may be stated in a finite dimensional space, it often happens that the sets to discriminate are not linearly separable in that space. For this reason, it was proposed that the original finite-dimensional space be mapped into a much higher-dimensional space, presumably making the separation easier in that space. To keep the computational load reasonable, the mappings used by SVM schemes are designed to ensure that dot products may be computed easily in terms of the variables in the original space, by defining them in terms of a kernel function K(x,y) selected to suit the problem. The hyper planes in the higher-dimensional space are defined as the set of points whose dot product with a vector in that space is constant. The vectors defining the hyper planes can be chosen to be linear combinations with parameters i of images of feature vectors that occur in the data base. With this choice of a hyper plane, the points x in the feature space that are mapped into the hyper plane are defined by the relation: \textstyle\sum_i \alpha_i K(x_i,x) = \mathrm{constant}.Note that if K(x,y) becomes small as y grows further away from x, each term in the sum measures the degree of closeness of the test point x to the corresponding data base point x-i. In this way, the sum of kernels above can be used to measure the relative nearness of each test point to the data points originating in one or the other of the sets to be discriminated. Note the fact that the set of points x mapped into any hyperplane can be quite convoluted as a result, allowing much more complex discrimination between sets which are not convex at all in the original space.

Classifying data is a common task in machine learning. Suppose some given data points each belong to one of two classes, and the goal is to decide which class a new data point will be in. In the case of support vector machines, a data point is viewed as a p-dimensional vector (a list of p numbers), and we want to know whether we can separate such points with a (p − 1)-dimensional hyper plane. This is called a linear classifier. There are many hyper planes that might classify the data. One reasonable choice as the best hyper plane is the one that represents the largest separation, or margin, between the two classes. So we choose the hyper plane so that the distance from it to the nearest data point on each side is maximized. If such a hyper plane exists, it is known as the maximum-margin hyper plane and the linear classifier it defines is known as a maximum margin classifier; or equivalently, the perceptron of optimal stability.

**4.3 Random Forests Algorithm**

Random forests are an ensemble learning method for classification (and regression) that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes output by individual trees. The algorithm for inducing a random forest was developed by Leo Breiman and Adele Cutler and "Random Forests" is their trademark. The term came from random decision forests that was first proposed by Tin Kam Ho of Bell Labs in 1995. The method combines Breiman's "bagging" idea and the random selection of features, introduced independently by Ho and Amit and Geman in order to construct a collection of decision trees with controlled variance.

Random forests can be used to rank the importance of variables in a regression or classification problem in a natural way. The following technique was described in Breiman's original paper and is implemented in the R package Random Forest.

The first step in measuring the variable importance in a data set is to fit a random forest to the data. During the fitting process the out-of-bag error for each data point is recorded and averaged over the forest (errors on an independent test set can be substituted if bagging is not used during training).

To measure the importance of the j-th feature after training, the values of the j-th feature are permuted among the training data and the out-of-bag error is again computed on this perturbed data set. The importance score for the j-th feature is computed by averaging the difference in out-of-bag error before and after the permutation over all trees. The score is normalized by the standard deviation of these differences.

Features which produce large values for this score are ranked as more important than features which produce small values.

This method of determining variable importance has some drawbacks. For data including categorical variables with different number of levels, random forests are biased in favor of those attributes with more levels. Methods such as partial permutations can be used to solve the problem. If the data contain groups of correlated features of similar relevance for the output, then smaller groups are favored over larger groups.

**Algorithm**

The training algorithm for random forests applies the general technique of bootstrap aggregating, or bagging, to tree learners. Given a training set X = x1, …, xn with responses Y = y1 through yn, bagging repeatedly selects a bootstrap sample of the training set and fits trees to these samples:

For b = 1 through B:

Sample, with replacement, n training examples from X, Y; call these Xb, Yb.

Train a decision or regression tree fb on Xb, Yb.

After training, predictions for unseen samples x' can be made by averaging the predictions from all the individual regression trees on x':

\hat{f} = \frac{1}{B} \sum_{b=1}^B \hat{f}_b (x')

or by taking the majority vote in the case of decision trees.

In the above algorithm, B is a free parameter. Typically, a few hundred to several thousand trees are used, depending on the size and nature of the training set. Increasing the number of trees tends to decrease the variance of the model, without increasing the bias. As a result, the training and test error tend to level off after some number of trees have been fit. An optimal number of trees B can be found using cross-validation, or by observing the out-of-bag error: the mean prediction error on each training sample xᵢ, using only the trees that did not have xᵢ in their bootstrap sample.

**4.4 Decision Trees Algorithm**

Decision tree learning uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value. It is one of the predictive modeling approaches used in statistics, data mining and machine learning. More descriptive names for such tree models are classification trees or regression trees. In these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels.

In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data but not decisions, rather the resulting classification tree can be an input for decision making. This page deals with decision trees in data mining.

Decision trees used in data mining are of two main types:

Classification tree analysis is when the predicted outcome is the class to which the data belongs.

Regression tree analysis is when the predicted outcome can be considered a real number (e.g. the price of a house, or a patient’s length of stay in a hospital).

The term Classification And Regression Tree (CART) analysis is an umbrella term used to refer to both of the above procedures, first introduced by Breiman et al. Trees used for regression and trees used for classification have some similarities - but also some differences, such as the procedure used to determine where to split.

Some techniques, often called ensemble methods, construct more than one decision tree:

Bagging decision trees, an early ensemble method, builds multiple decision trees by repeatedly resampling training data with replacement, and voting the trees for a consensus prediction.

A Random Forest classifier uses a number of decision trees, in order to improve the classification rate.

Boosted Trees can be used for regression-type and classification-type problems.

Rotation forest - in which every decision tree is trained by first applying principal component analysis (PCA) on a random subset of the input features.

Decision tree learning is the construction of a decision tree from class-labeled training tuples. A decision tree is a flow-chart-like structure, where each internal (non-leaf) node denotes a test on an attribute, each branch represents the outcome of a test, and each leaf (or terminal) node holds a class label. The topmost node in a tree is the root node.

There are many specific decision-tree algorithms. Notable ones include:

* ID3 (Iterative Dichotomiser 3)
* C4.5 (successor of ID3)
* CART (Classification And Regression Tree)
* CHAID (CHi-squared Automatic Interaction Detector). Performs multi-level splits when computing classification trees.
* MARS: extends decision trees to better handle numerical data.
* Conditional Inference Trees. Statistics-based approach that uses non-parametric tests as splitting criteria, corrected for multiple testing to avoid over-fitting. This approach results in unbiased predictor selection and does not require pruning.

ID3 and CART were invented independently at around same time (b/w 1970-80), yet follow a similar approach for learning decision tree from training tuples.

**CHAPTER 5**

**Implementation**

**Intelligence Email Reply Management System**

Intelligent Email reply management system is designed to handle incoming email messages for the IEPS and transferring the mails to the email predictor which then analyse each email fields and contents to determine if they need reply and classify them into various state (need reply-0, do not need reply-1 and others-2) based on the sensitivity of the content of the email content.

This is a decision making system that could determine if emails received require a reply and the model is shown in Figure. For any given email datasets, there are multiple email conversations and to capture these different conversations, the system assumes that if one email was a reply to the sender’s original message, then such a mail may require attention as this may have element of request.

We used machine learning techniques for finding interrogative words, questions marks, most frequent words, most used phrases and embed WorldNet in order build a model that provide a focus to each mail and determine whether email message require a reply. We implemented a machine learning approach to solve the problem of IEPS system.

Machine learning is learning the theory automatically from the data, model fitting, or learning from examples. It is also an automated extraction of useful information from a body of data by building a good probabilistic model.

Importance of Machine Learning

Our work involves machine learning because it is the underlying method that enables us to generate high statistical output. These are the importance of machine learning as applied in our work:

* New knowledge about tasks is constantly being discovered by humans. Like vocabulary changes, and there is constant stream of new events in the world. Continuing redesign of a system to conform to new knowledge is impractical, but machine learning methods might be able to tract much of it.
* Environments change over time, and new knowledge is constantly being discovered. A continuous redesign of the systems “by hand” may be difficult. So, machine that can adapt to changing environment would reduce the need for constant redesign.



Figure 7

The Figure 1 shows the schematic model of our email management. A schematic model of the architecture for email words and phrases extractions from incoming email messages. Email messages are passed as input data to the feature extractor where all the rare words and frequent words and phrases are extracted and passed to the analyser. The analyser, then check the meaning of each words with the embedded WorldNet [6] for the meaning and our predictor then select most meaningful words that occur often in the mail and build a storage of words and phrases model and store words and phrases learned and their meaning. Our predictor will intelligently choose the most frequent words and meanings to determine the categories that the email messages belong to: need reply-0, do not need reply-1, others-2. The predictor’s decision is based on human training data that has been learned and begin to be more self intelligent as new mails arrives.

**Intelligent Email Prediction System**

This is an automated machine learning system that determines if emails received require a reply. We also implemented WorldNet. WorldNet is a large lexical database of English, developed under the direction of George A. Miller. Nouns, verbs, adjectives and adverbs are grouped into sets of cognitive synonyms (synsets), each expressing a distinct concept. Synsets are interlinked by means of conceptual-semantic and lexical relations. We embed WorldNet into our email feature analyser as shown in Figure 1 in order to analyse meaning of words and short phrases in email messages and as our analyser finishes with the meaning of words, then the processed email messages are then passed to our propose predictor which then determine the category of the email messages based on the intelligent feedback retrieved from the feature analyser to determine if such a mail belongs to:

􀂄 Need reply – 1

􀂄 Do not need reply – 0

􀂄 Others- 2

The IEPS learns by example as human participants analysed over 10000 email messages to determine these three aforementioned categories. Further analysis on the methodology used is explicated in section 4. Human participants involved in this research training and testing of this IEPS are from various backgrounds and professions. We have 200 participants from Europe, Asia, Africa, America and Australia and their profession ranges from postgraduate students (Researchers, IT discipline, Banking, medical, Environments, Academia etc), undergraduate students (Art and Science majors), bankers, doctors, business owners, tourism, Air line agents, and many more.

**IEPS Participants’ Methodology**

Two hundred email users at 40 different departments of an IT company and 200 University students were interviewed and observed during their morning reading of emails. Each of them checked the followings:

􀂆 Sender’s email address

􀂆 Subject of each email messages

􀂆 Cc/Bcc field

􀂆 Content of the emails

􀂆 Previous conversation

The email users analysed all their emails ranging from private emails, business emails, to public emails. Each of their email messages were analysed to determine email messages that require reply. The human participant employ heuristic approach to solving the difficulties in prioritising email messages with the following assumptions:

A. Sender’s email address: If an email message is from certain people: CEO, Manager, Head of department, debt collector, hospital etc then assume it may require a reply and assign a score of 1 to this on the scoring board provide with the predictor system

B. Subject of each email messages: If the subject of the email messages are similar or almost related to the phrases stored in the database of words and phrases on our system- please reply soon, let me hear from you, Is there any news today?, Are we on the same project or not. A score of 3 is allocated to the scoring board.

C. Cc/Bcc Filed: If a mail is Cc or Bcc to others, such a mail may require attention because mail copied to others may be as a result of a group project or task that an individual or groups need to be aware of or act upon as soon as possible. Such a mail may require reply. A score of 2 is allocated to the scoring board.

D. Email content: If a mail contain words and phrases such as: interrogative words- could, when, where, how? Rarely used phrases: meeting at noon, is it alright, because of the yearly budget, based on what we saw, etc. such mail may denote a request and may denote a reply. A score of 3 is allocated to this area because content of email messages focus on what the email is all about and that is why human participant spent more time in analysing this area of research.

E. Previous conversation: An existing email conversation within the same subject indicated that majority of such messages require attention. A score of 1 is assigned to the scoring board making a total of 10 points.

Based on the results of these interviews, and testing, our IEPS predictor is trained to learn the features mentioned above and be to know the type of categories each email received belongs to and be more intelligent as times goes on when received new messages.

**Machine Learning Techniques**

Email messages in mail boxes could become large amount of data that could have hidden correlations and may be hard to find specific emails messages in mail box after a long time. We experiment with machine learning techniques to learn and be self knowledgeable about email features namely:

􀁸 sender’s email address (domain from where this email is coming from)

􀁸 previous email conversation-which may suggest any request made previously 􀁸 subject field for any phrases that suggest interrogation or statements of commitment

􀁸 attachment found in email messages.etc

Our technique is capable of learning email features that could be used to determine whether an email require a reply and is capable of becoming more intelligent when it receives a new email with different format ranging from public email, e-commerce, private and business emails. The technique keeps learning and that makes it more efficient and effective learning approach without any supervision.

**CHAPTER 6**

**Evaluation and Results**

The following observations were made from the initial interviews.

􀁸 All users scanned new messages several times in order to read and determine the most important messages and at the same time categorize them.

􀁸 Messages related to events, meetings, venues are considered as important. 􀁸 Replies are important as they often contain a solution to a problem posted by the recipient and there are usually elements of request.

􀁸 Subject field of email messages do give a clue about what the mail is all about. 􀁸 For most users, carbon copies and Blind copies were judged by their receivers as important as other messages.

􀁸 Messages containing interrogative words are many and most recipients consider such messages as need reply.

Email messages are categorised by our intelligent email reply prediction system to require a reply or not in three mail categories as shown in Figure 9.

The three possible predictions for the email messages are:

􀁸 Need reply: Email messages that are categorised to need a reply indicated that such a mail passed the threshold set by human analyser. The threshold value assigned for email messages that require a reply is 7 and any messages that score 7 or above out of total score of 10 will be assigned the tag “need reply -1”.

􀁸 Do not need reply: Messages that scoreless that the threshold value 7 will be assigned a tag “Do not need reply- 0”.

􀁸 Others: Messages that could not belong to either of the categories above will be categorised here. These email messages that are in this categories are: email messages from friends that does not require any urgency, auto respond reply messages, advertisement email messages, junk emails, email messages with email address: nonreply@myname.com, informative emails etc. Intelligent email reply prediction system was evaluated using precision and recall over 10,000 email Enron datasets from over 120 email boxes owned by 200 people from Enron Corpus as shown below: Recall = group found and correct (needs reply) total group correct (rightly predicted) Precision = group found and correct (needs reply) total group found (Total email found)

The Table 1 shows the results of interviews perform on a small scale basis for the first and second interviews. As seen on the table below, IEPS has reduced email users’ time spent on un-necessary browsing through thousands of email messages as this solution has made the mail box well organised and well structured.

Email messages are well prioritized and therefore overload is reduced and high volume of email messages are well controlled by the proposed IEPS. The system performs better than the existing email prediction systems.

**Outputs & Screenshots**

Login

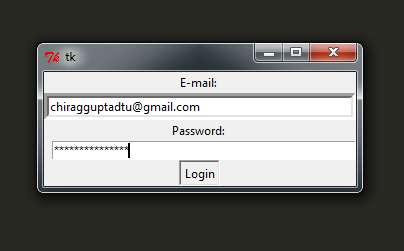


Figure 8

e-mail DATA

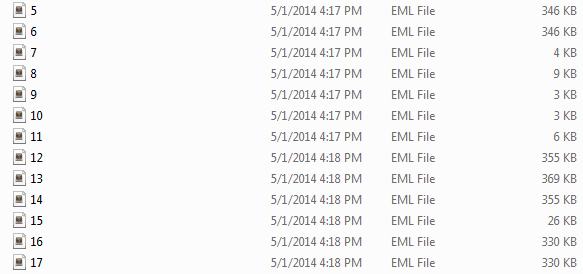


Figure 9

JSON Training Data



Figure 10

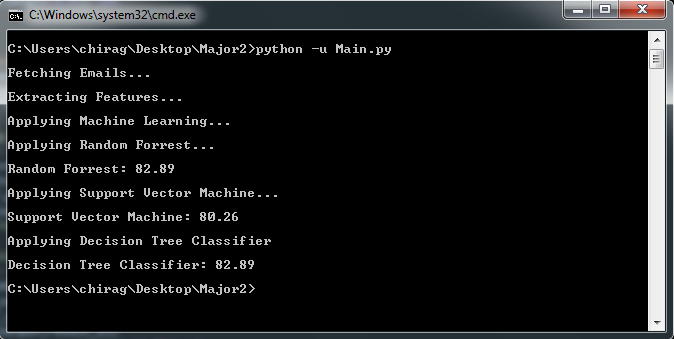


Figure 11

The figure 11 shows the process/steps our project goes through and the final result obtained from three different machine learning algorithms in term of accuracy of the email reply prediction.

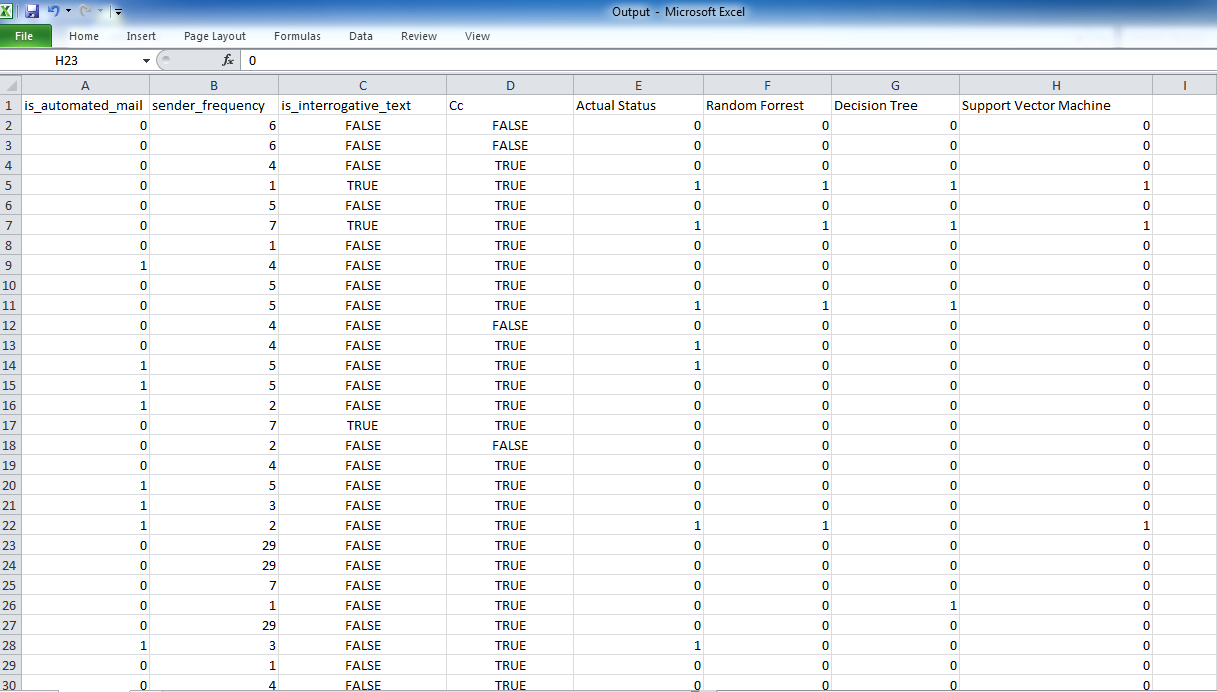


Figure 12

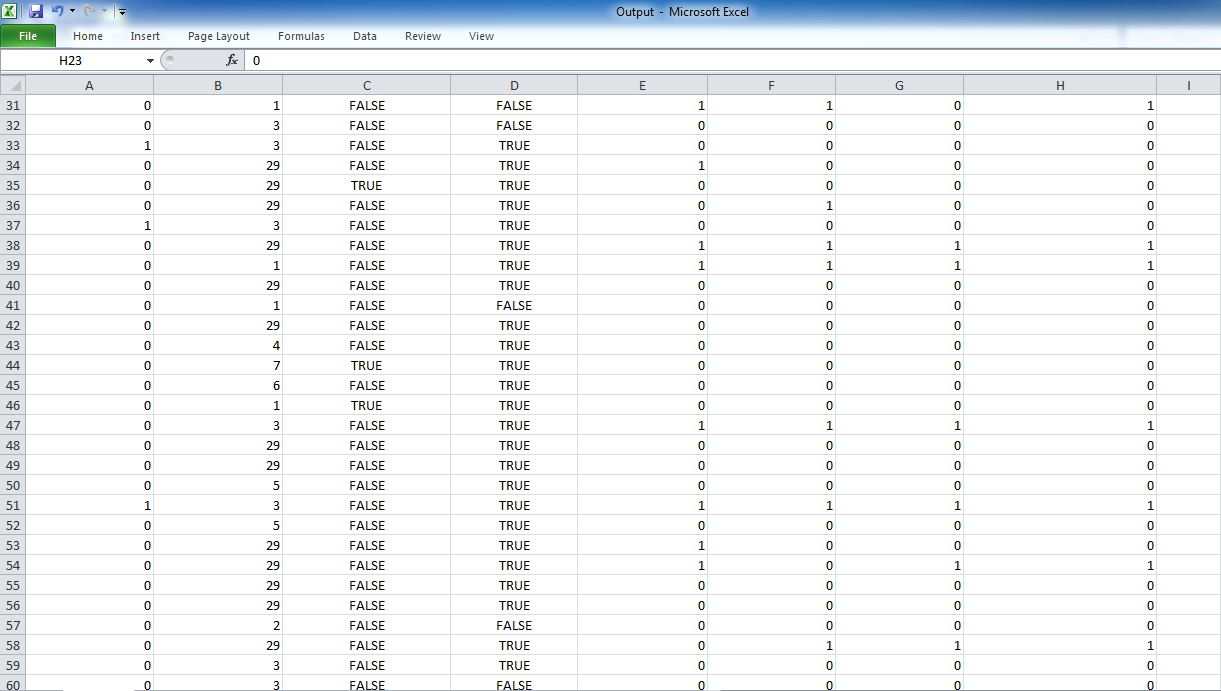


Figure 13

Figure 12 and Figure 13 shows all the features extracted from the email database. The features are :

1. Is the email automated ?
2. Does the email contain interrogative text ?
3. Number of previous email messages exchanged with sender.
4. Is the email is Cc ?

The last four columns shows the comparison between the actual status of the emails and the predictions of the different machine learning algorithms.

**Conclusion**

**Conclusion**

The main focus of this project was to study how to generate accurate measures to determine the mail that require a reply and the one that does not require reply.

We analyse the features of emails and study email conversation structure, IEPS system learns from human participant categorised data tests, which we maintain that this area of research has not been sufficiently investigated in previous research on intelligent email prediction system.

We build a novel structure: Intelligent email management prediction system (IEMPS), interrogative words, mails from specific domains and many more. Our future plan includes improving the IEMPS with more sophisticated linguistic analysis.

In order to verify the generality of our findings, we are working on evaluating our methods with different real life datasets: creating the gold standard for a large real datasets.

**REFERENCES**

1. S. Whittaker, C. Sider, 1996, ‘Email overload: exploring personal information management of email’, CHI ‘96’, pp.276-283. ACM Press.

2. Tyler, J., & Tang, J. When can I expect an email response? A study of rhythms in email usage. In Proc. of ECSCW 2003, Oulu Univ. Press (2003), 238-258.

3. M. Dredze, J. Blitzer, F. Pereira, J 2005, ‘Reply Expectation Prediction for Email Management’, CEAS’2005, 2nd Conference on Email and Anti-Spam, Stanford University, CA.

4. G. Salton, A. Wong, S.S. Yang, 2005, “A vector space model for automatic indexing”, Communications of the ACM, vol.18, p.613-620.

5. Ducheneaut, N., & Belloti, V. Email as habitat: An exploration of embedded personal information management. Interactions 8, 5 (2001), 30-38.

6. Kraut, R.E. & Attewell, P. Media use in a global corporation: Electronic mail and organizational knowledge, in Culture of the Internet. Lawrence Erlbaum Associates, Mahwah, NJ,n USA, (1997), 323-342.

7. Mackay, W. Diversity in the use of electronic mail: A preliminary inquiry. ACM Transactions on Office Information Systems 6,4 (1988), 380-397.

8. Sproull, L., & Kiesler, S. Connections: New ways of working in the networked organization. MIT Press: Cambridge, MA, USA, 1991.

9. Bryan Klimt and Yiming Yang. The Enron corpus: A new dataset for email classification research. In European Conference on Machine Learning, 2004.

10. Vitor R. Carvalho and William Cohen. Recommending recipients in the enron email corpus. Technical Report CMU-LTI-07-005, Carnegie Mellon University, Language Technologies Institute, 2007.

11. William W. Cohen, Vitor R. Carvalho, and Tom M. Mitchell. Learning to classify email into ”speech acts”. In Empirical Methods in Natural Language Processing (EMNLP), 2004.

12. Carman Neustaedter, A.J. Bernheim Brush, Marc A. Smith, and Danyel Fisher. The social network and relationship finder: Social sorting for email triage. In Proc. of the Conf. on Email and Anti-Spam (CEAS), Mountain View, CA, 2005.

13. Richard B. Segal and Jeffrey O. Kephart. MailCat: An intelligent assistant for organizing e-mail. In AAAI, 1999. 15. Jianqiang Shen, Lida Li, Thomas G. Dietterich, and Jonathan L. Herlocker. A hybrid learning system for recognizing user tasks from desk activities and email messages. In IUI, 2006.

14. S. Whittaker, V. Bellotti, and P Moody. Introduction to this special issue on revisiting and reinventing e-mail. Human-Computer Interaction, 20(1-2):1–9, 2005.

15. Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

16. Belloti, V. & Ducheneaut, N. Email as habitat: An exploration of embedded personal information management. Interactions 8, 5 (2001), 30-38.

17. , S.S. Yang , G. Salton, A. Wong, 2005, “A vector space model for automatic indexing”, Communications of the ACM, vol.18, p.613-620.

18. F. Pereira, J, M. Dredze, J. Blitzer, 2005, ‘Reply Expectation Prediction for Email Management’, CEAS’2005, 2nd Conference on Email and Anti-Spam, Stanford University .

19. Yiming Yang and Bryan Klimt. The Enron corpus: A new dataset for email classification research. In European Conference on Machine Learning, 2004.

20. Kiesler, S. Connections & Sproull, L.: New ways of working in the networked organization. MIT Press: Cambridge, MA .

21 William Cohen and Vitor R. Carvalho. Recommending recipients in the enron email corpus. Technical Report CMU-LTI-07-005, Carnegie Mellon University, Language Technologies Institute.