Statistics Undergraduate Students Association University of California, Berkeley

An Analysis of Sentiment in the Enron Corpus

Arun Ramamurthy, Serena Jiang, Dennis Yang

Introduction

In 1985, Kenneth Lay led a merger between energy companies, Houston Natural Gas and InterNorth to form the Enron Corporation. Named as “America’s Most Innovative Company” by *Fortune* for six consecutive years, Enron was a renowned American energy, commodities, and services company which reported an annual revenue of $111 billion during 2000. However, a significant part of Enron’s public success can actually be apportioned to its illegal activities; through accounting loopholes, special purpose entities, and falsified financial reporting, Enron was able to hide billions of dollars in debt from failed partnerships and deals. Ultimately, their fraud and corruption was revealed, in a series of events that culminated with the Enron Scandal and the company’s bankruptcy on December 2, 2001.

Many cite November of 1997 as the starting point of Enron’s fraud, when it bought out a partner’s stake in the company called JEDI and sold the stake to a firm it created, thus beginning a complex series of transactions that enabled it to hide its debts. From then on, Enron appeared to be on the rise, culminating on its stock achieving a high of $90.75 per share on August 23, 2000. Then, on August 14, 2001, Jeffrey Skilling resigned as CEO, leading to Enron’s faltering price on the stock market. Two months later, Enron announced restatements in its financial statements, resulting in reports of $638 million in third quarter losses, and $1.2 billion reduction in equity. On October 22, 2001, the SEC announced several of their investigations into suspicious deals made with Enron, leading to its stock price falling from $20.65 to $5.40. On November 8, 2000, Enron revised documents with SEC revealing $586 million in losses. Only 11 days later, Enron restated third quarter earnings to disclose $690 million in debt due in 8 days. Ultimately, on November 28, 2001, Enron shares plunged below $1 after shareholders filed a $40 billion lawsuit after discovering the fraud. At the time, its $64.3 billion in lost assets made Enron the largest corporate bankruptcy in U.S. history.

Many executives at Enron were indicted for a variety of charges and several were sentenced to prison. This also led to the *de facto* dissolution of Arthur Andersen, which was found guilty of tampering with evidence and destroying documents related to the SEC investigation. Their auditing license was voided, effectively ending one of the five largest audit and accountancy partnerships in the world. Enron's shareholders lost $74 billion in the form of pensions and stock prices before the company's bankruptcy, but as the company owned $67 billion to creditors, employees and shareholders barely received any severance from Enron.

One positive adjustment from the Enron Scandal was the Sarbanes-Oxley Act, legislation passed by the Senate Committee on Banking, Housing, and Urban Affairs and the House Committee on Financial Services. In the hope of preventing future injustices, this federal law set new and expanded requirements for management to adhere to. Specifically, this article increased the penalties for destroying, modifying, and faking records, primarily for the purpose of embezzling investors and shareholders or impeding federal investigations. It also increased the accountability of auditing firms to remain unbiased and independent of their clients while increasing financial disclosure of companies' relationships with other unconsolidated entities.

Since then, the Enron Scandal has become a heavily researched and documented incident in corporate policy and culture. One of the most important artifacts preserved is the Enron Corpus; collected from the Houston headquarters, Enron in May 2002 by the FERC, the Enron Corpus is a large dataset of over 600,000 emails from over 150 employees, mostly senior management of the Enron Corporation. A year later, a professor at University of Massachusetts Amherst bought a copy of the dataset for $10,000 and released the corpus to fellow researchers. Finally, in a collaboration effort between MIT and SRI International, the data was stripped of over-personal and financial data, and made freely available to the general public. Given the Enron Corpus' accessibility, several studies have been done on it relating to business communication, and more generally, electronic communication during that time.

As an extension to these, this project is a preliminary investigation into the various trends of social behavior in a corporate setting. This data is very interesting because of the time frame it spans; there are emails from 1998 to 2002, a time period that includes both the rise and fall of the Enron Corporation. In this paper, we analyze the Enron Corpus to answer the following questions:

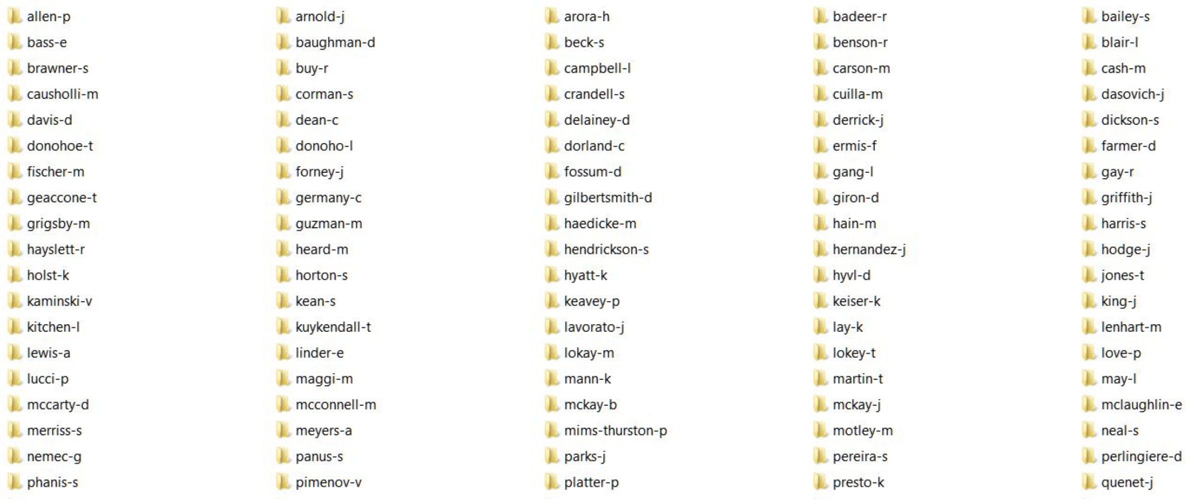
1. What trends do the emails exhibit, and how do they correlate? Specifically, we investigate the frequency of emails sent by user, time of day, day of the week, and by date.
2. How does the sentiment of the emails change over time? Does it correlate with season or day of the week?

Data Preparation

Data Collection and Cleaning

The Enron Corpus is freely available for download at:
https://www.cs.cmu.edu/~./enron/enron_mail_20150507.tgz

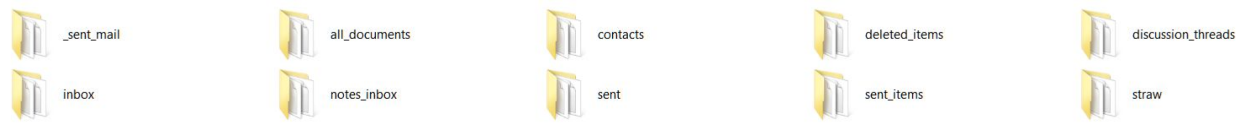
The link leads to a single zipped file called 'maildir', which contains a total of 150 folders, each associated with a single employee's email.



allen-p	arnold-j	arora-h	badeer-r	bailey-s
bass-e	baughman-d	beck-s	benson-r	blair-l
brawner-s	buy-r	campbell-l	carson-m	cash-m
causholli-m	corman-s	crandell-s	cuilla-m	dasovich-j
davis-d	dean-c	delainey-d	derrick-j	dickson-s
donohoe-t	donoho-l	dorland-c	ermis-f	farmer-d
fischer-m	forney-j	fossum-d	gang-l	gay-r
geaccone-t	germany-c	gilbertsmith-d	giron-d	griffith-j
grigsby-m	guzman-m	haedicke-m	hain-m	harris-s
hayslett-r	heard-m	hendrickson-s	hernandez-j	hodge-j
holst-k	horton-s	hyatt-k	hyvl-d	jones-t
kaminski-v	kean-s	keavey-p	keiser-k	king-j
kitchen-l	kuykendall-t	lavorato-j	lay-k	lenhart-m
lewis-a	linder-e	lokay-m	lokey-t	love-p
lucci-p	maggi-m	mann-k	martin-t	may-l
mccarty-d	mcconnell-m	mckay-b	mckay-j	mclaughlin-e
merriss-s	meyers-a	mims-thurston-p	motley-m	neal-s
nemec-g	panus-s	parks-j	pereira-s	perlingiere-d
phanis-s	pimenov-v	platter-p	presto-k	quenet-j

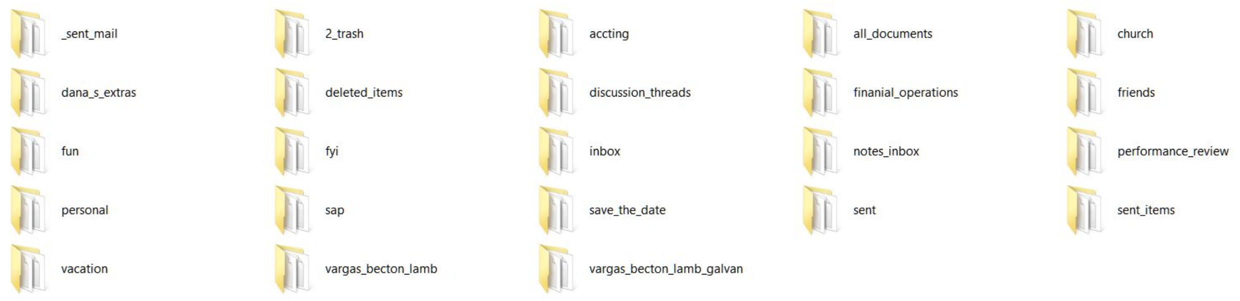
Each employee’s private directory is then further split up into categories, like “inbox” or “sent”. However, since each employee has their own organization habits, the subfolders are inconsistent. Take, for example, the following directories:

“maildir/allen-p”:



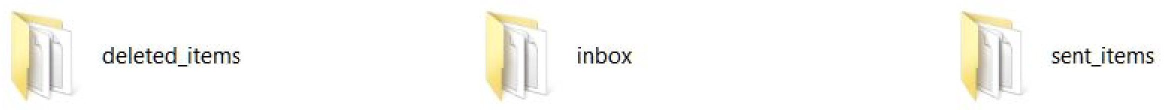
_sent_mail	all_documents	contacts	deleted_items	discussion_threads
inbox	notes_inbox	sent	sent_items	straw

“maildir/davis-d”:



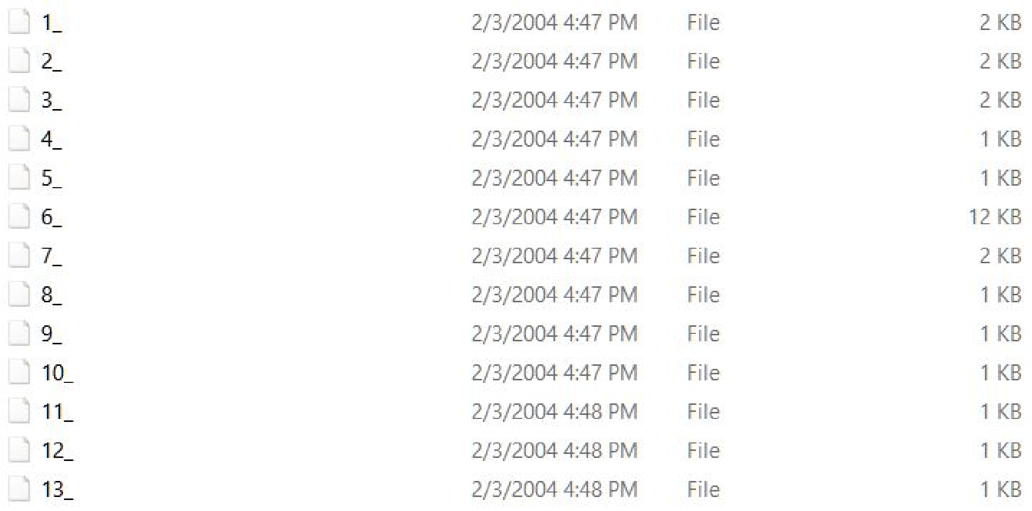
_sent_mail	2_trash	accting	all_documents	church
dana_s_extras	deleted_items	discussion_threads	finanial_operations	friends
fun	fyi	inbox	notes_inbox	performance_review
personal	sap	save_the_date	sent	sent_items
vacation	vargas_becton_lamb	vargas_becton_lamb_galvan		

”maildir/holst-k”:



deleted_items	inbox	sent_items

After delving into each of these categories, we find a list of emails, each named with the syntax, “x_”, where x is a natural number. As an example, below are the contents of “maildir/allen-p/sent”:



1_	2/3/2004 4:47 PM	File	2 KB
2_	2/3/2004 4:47 PM	File	2 KB
3_	2/3/2004 4:47 PM	File	2 KB
4_	2/3/2004 4:47 PM	File	1 KB
5_	2/3/2004 4:47 PM	File	1 KB
6_	2/3/2004 4:47 PM	File	12 KB
7_	2/3/2004 4:47 PM	File	2 KB
8_	2/3/2004 4:47 PM	File	1 KB
9_	2/3/2004 4:47 PM	File	1 KB
10_	2/3/2004 4:47 PM	File	1 KB
11_	2/3/2004 4:48 PM	File	1 KB
12_	2/3/2004 4:48 PM	File	1 KB
13_	2/3/2004 4:48 PM	File	1 KB

The first step in entering this data into a dataframe was to form a delimited text file containing the file path, date, sender, recipient, subject, and body of every single email. We wrote a Python script, which can be found in the Appendix, to recursively run through each directory and store all relevant information into a single list object. Then, after all directories are read, the script writes the list into a single line in a single text file, delimited by a symbol. We had to choose a symbol that wasn't present in any of the email files, and arbitrarily chose "º" as our delimiter. Given this text file, we used R to read the emails into a dataframe. After a small amount of cleaning to account for the date formatting, we are left with a full dataframe containing 495,129 observations of email information.

Polarity Calculation

Sentiment analysis refers to a set of powerful NLP techniques that shed light on the *context* of text body. Ordinarily, given a set of sentences, we can only make naive statements about the meaning of the text, by, for instance, counting the occurrence of particular words or calculating the mean length of each sentence. But with sentiment analysis, we are able to approximate the actual meaning of the text, and, in particular, the attitude of the writer at the time. The main quantitative measure of sentiment we analyzed is polarity, which indicates to what degree the emotional content of a body of text is positive or negative. Polarity is usually represented as a real number, with positive numbers indicating positive emotions, negative numbers indicating negative emotions, and numbers close to zero indicating neutrality. For example, the polarity of, "I am extremely glad this morning," would be a high positive value, whereas the sentence, "Nobody in this room feels happy," would have a negative polarity.

To conduct this polarity calculation on our data, we used an R package called qdap (Quantitative Discourse Analysis Package). This package calculates the polarity of a sample of text by tagging polarized words. Each polarized word is assigned a context cluster (x_i^T) of words before and after (default number is four words before and two words after) to be considered as valence shifters. The word already has its own original weight based on a dataframe or hashset of positive and negative words and weights, but will ultimately change based on its valence shifters. Each word in this cluster is assigned as neutral (x_i^0), a negator (x_i^N), an amplifier (x_i^D), or a de-amplifier $(x_i^{D'})$. Neutral words are effectively ineffectual; they have no value in the equation and do not effect the word count (n). Each negator switches the positivity of the end polarity; two negators will result in the polarity staying the same, etc. The polarized word is then weighted by the number and position of each of its valence shifters. The researcher may provide a weight (c), in addition to the amplifiers and de-amplifiers (default is 0.8, but is ultimately constrained to a lower bound of -1). Lastly, this context cluster is summed and divided by the square root of the word count, resulting in the final polarity score (P). By dividing by the square root of the word count, this ultimately results in the density of the polarity. This algorithm is summarized mathematically below:

$$P = x_i^T / \sqrt{2}$$

$$x_i^T = \sum (1 + c * (x_i^A - x_i^D)) * w * (-1)^{\sum x_i^N}$$

$$x_i^A = \sum w_{neg} * x_i^a$$

$$x_i^D = max\,(x_i^{D'},\ -1)$$

$$x_i^{D'} = \sum - w_{neg} * x_i^a + x_i^d$$

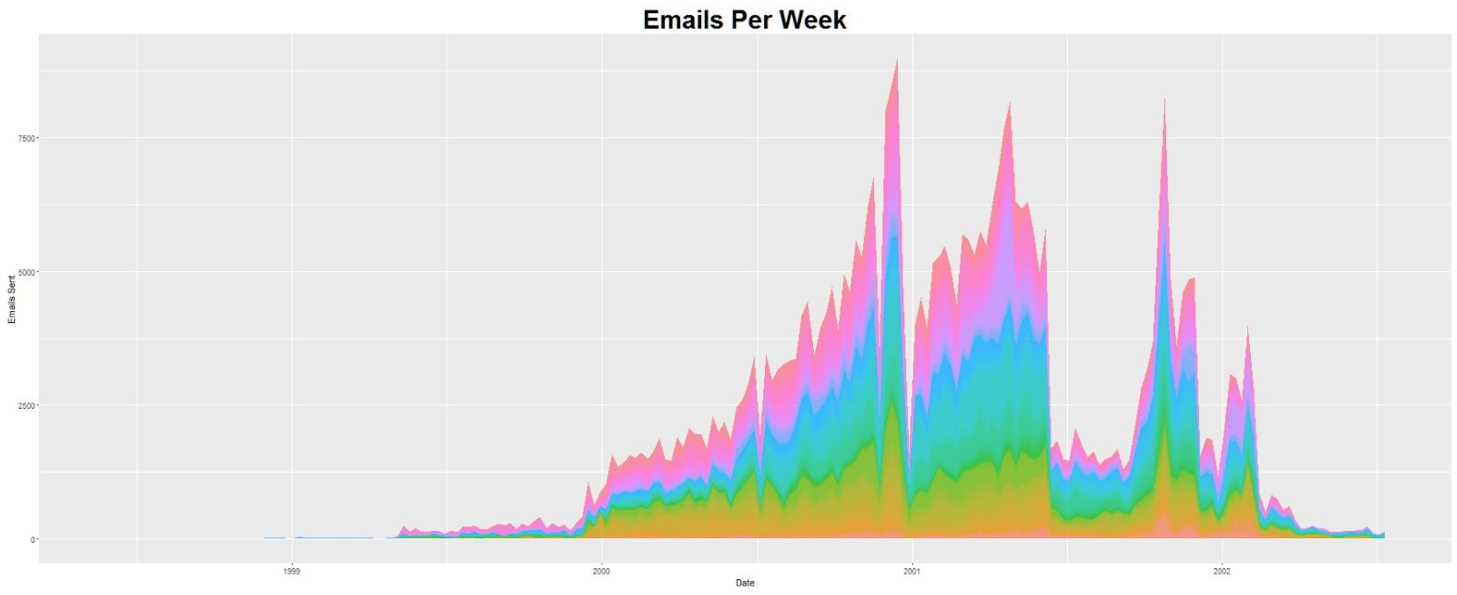
$$w_{neg} = \left(\sum x_i^N\right) mod\ 2$$

Unfortunately, this algorithm, while it does run in linear time with respect to the number of words in the body of text, demands a lot of computational power. To go through each email's body and calculate sentiment for all 495,129 emails requires over 50 hours on a typical local machine, and more than 150 hours on UC Berkeley's gandalf machine. Because of these time constraints, all sentiment analysis was conducted on a randomly selected subset of 10,000 emails. There is a chance this decision limited the power of our tests significantly, altering results. That being said, all research and methods used are entirely reproducible for the entire data set later, provided the necessary computational resources.

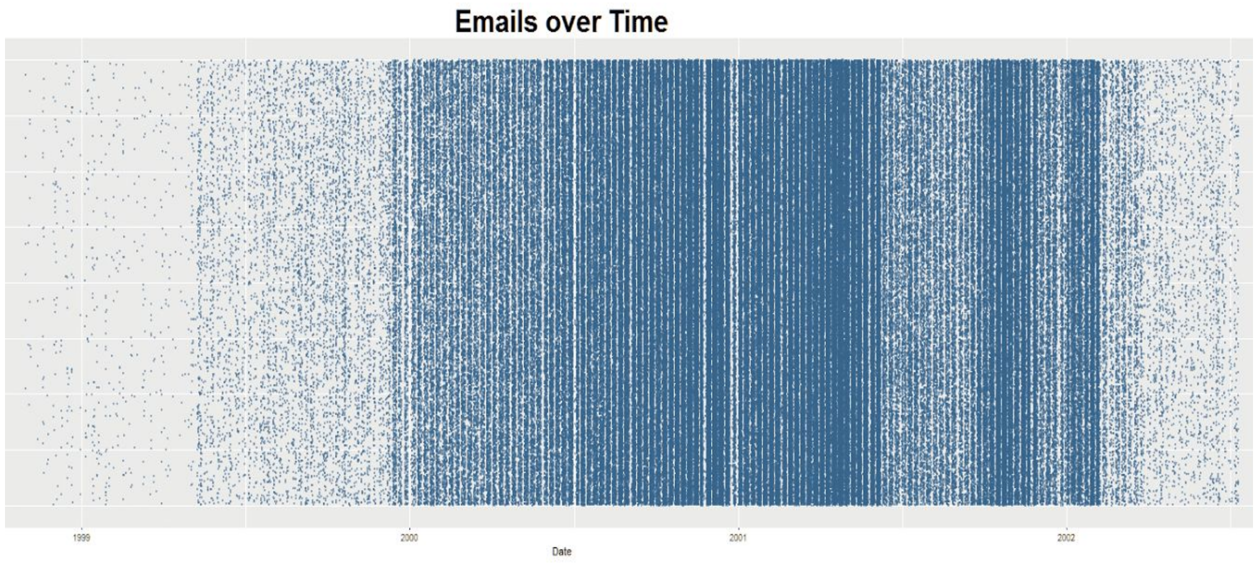
Analysis and Data Visualization

Preliminary Analysis

Before analyzing the sentiment and polarity of the emails, we first wanted to conduct basic analysis on the frequency and timing of the emails sent.

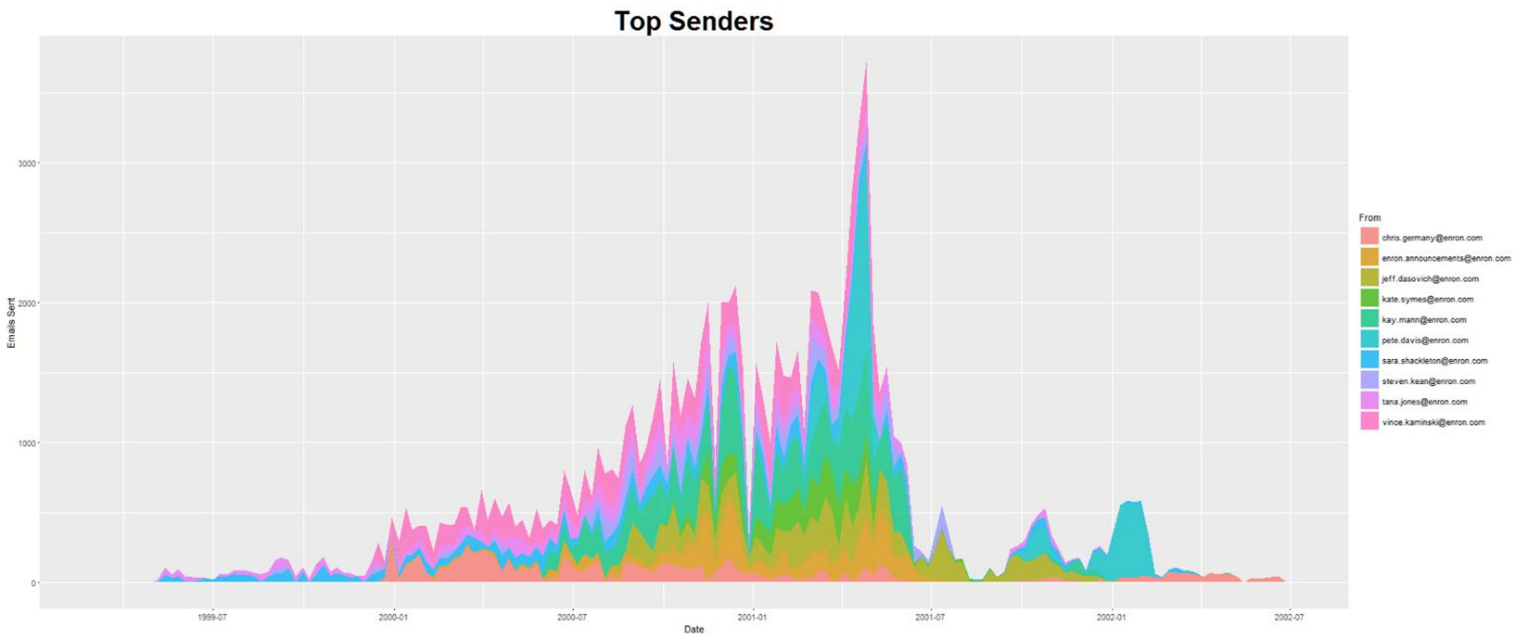


The graph above shows the frequency of emails sent per week over the course of several years. Although at first a clear upward trend in email frequency appears, there were several troughs and peaks affecting the data that we wanted to investigate.

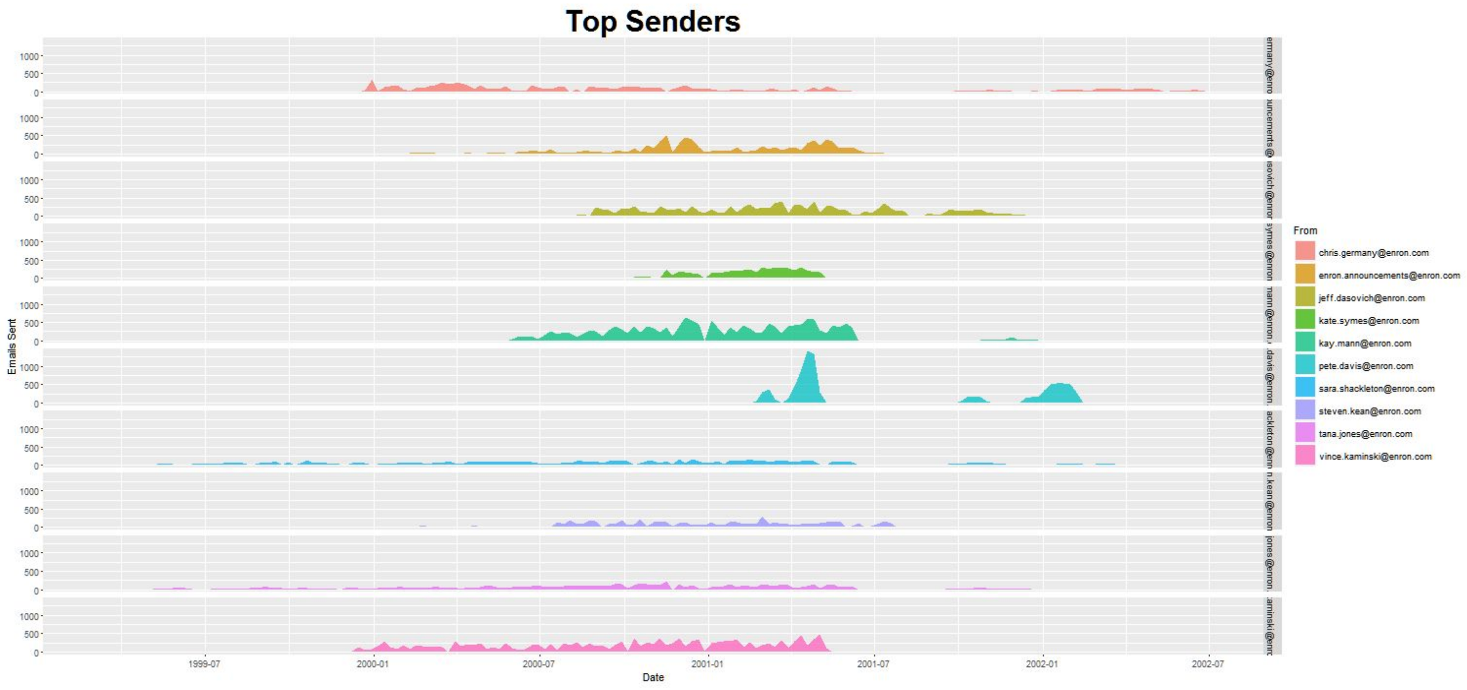


To investigate the sudden dips in the data, we created a scatter plot to help visualize the density of email frequency. Using the jitter function to arbitrarily spread the emails across the y-axis, the above graph is a representation of the “bands” of emails in the corpus; the density of the bands are indicative of email frequency. The importance of this graph is that it helps visualize the gaps in dataset. It’s clear by the absence of certain strips of glyphs that the Enron corpus is missing values from particular time periods, potentially because of the removal of all financial data by the FERC when they released this data set. We hypothesise that these redactions are partly what caused the dips in email frequency. Additionally, as the corpus was collected by taking database images from several different sources, the sudden jump in email frequency seen in late 1999 seems to be caused by the addition of another branch of the Enron email database to the corpus around that time.

To investigate the peaks, we turned to the most active email senders. A graph of a similar frequency plot for the subset of the top ten senders is below.



While most of these users follow the same trend as the rest of the employees, there are a couple of deviations, namely emails sent by pete.davis@enron.com. Upon faceting, these deviations become more visually apparent.

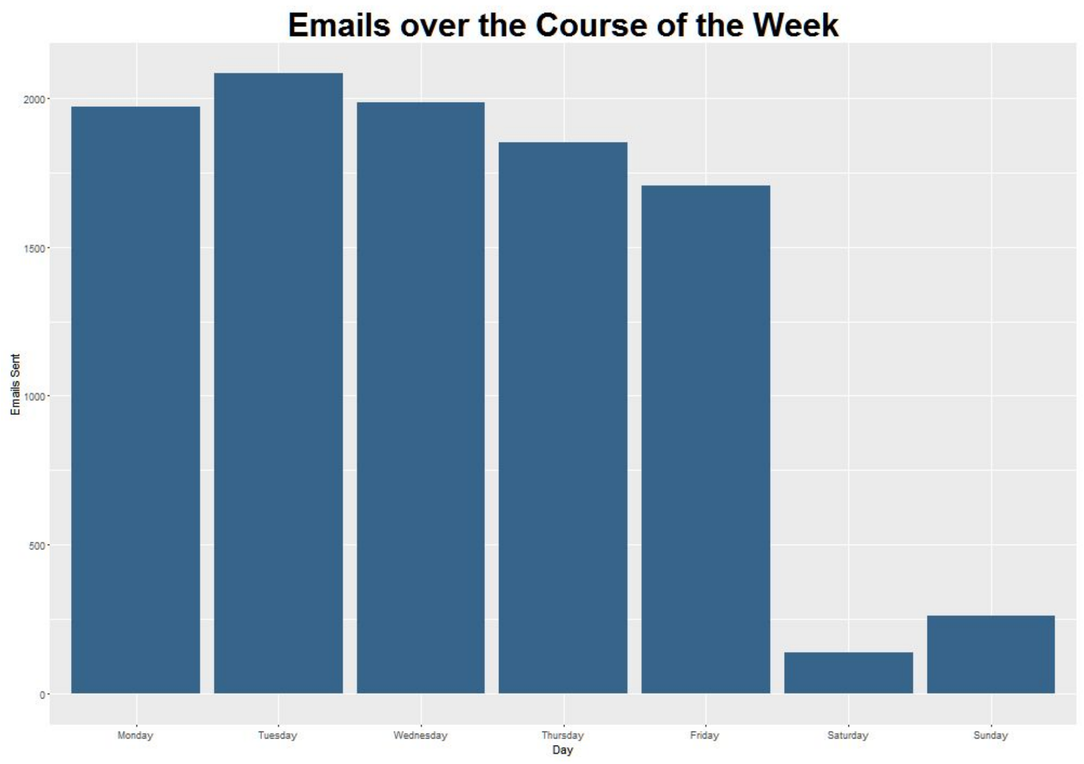


After looking through the email files for pete.davis@enron.com, we find that Pete Davis was some sort of system administrator, who would over the period of four separate occasions, send mass emails of program success logs to himself, cc'ing many relevant coworkers. An example of the body of such an email is seen below.

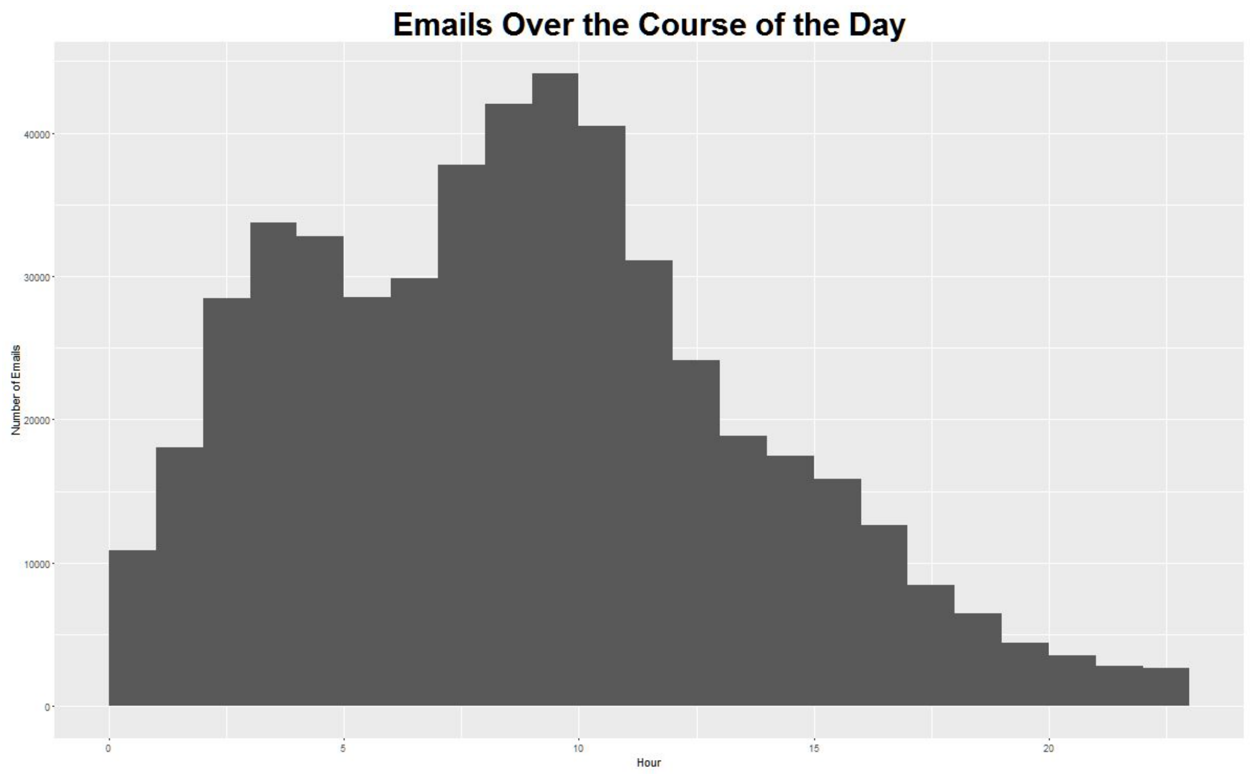
```
Start Date: 10/18/01; HourAhead hour: 7;  No ancillary schedules awarded.  No variances detected.

    LOG MESSAGES:

PARSING FILE -->> O:\Portland\WestDesk\California Scheduling\ISO Final Schedules\2001101807.txt

Error retrieving HourAhead price data - process continuing...
```

Apart from Pete Davis, the rest of the top ten email senders don't appear to any surprising spikes or dips in emails sent. While they all do have fluctuations, none are significant like Pete Davis' to warrant additional research into - probably a combination of varying levels of work, position, and personal factors.

We then analyzed the corpus for email frequency as a function of weekday and hour.



The above plot shows the typical expected trend of email frequency for a company in early 2000s, with far fewer email activity over the weekends. Investigation of the timing of email activity, however, reveals slightly more nuanced results.

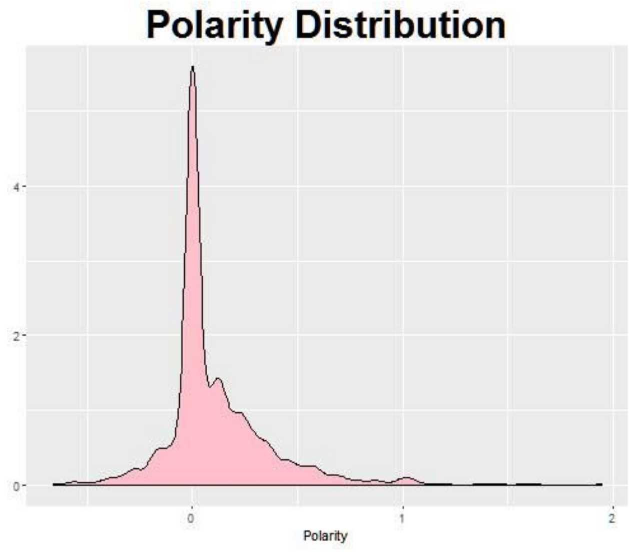


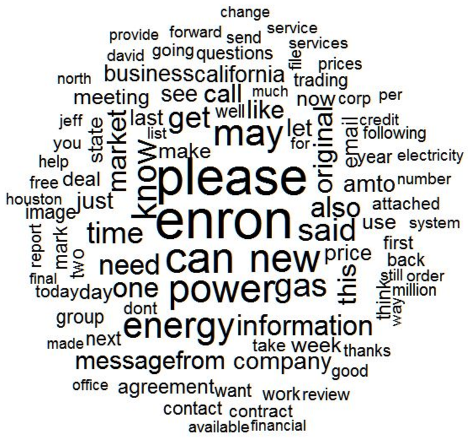
The first two immediate surprises to this graph are that there are two peaks, the one centered near 10 AM is understandable, but the smaller peak centered near 5 AM was very confusing as the majority of people wake up after 6 AM. Upon further analysis of the emails, we realized that the majority of the emails were split between two time zones - the Pacific Time Zone and the Central Time Zone. Despite Enron being originally based in Houston, all the emails were recorded in PT. The first peak represents all the emails sent from California, where 5 AM PT correlates with 7 AM CT, a very reasonable time for a lot of emails sent. After the first peak, the number of emails decreases, until it begins increasing again at 7 AM PT, when all the workers in California start to wake up and go to work. The largest peak at 10 AM PT occurs at a time when people at both time zones are working and communicating, resulting in the maximum number being sent.

The sum of two approximately normal distributions of differing centers and spread can result in a bimodal bell-curve distribution. In the above graph, this concept manifests because of the two-hour time difference between Enron's two main branches, its headquarters in Houston, and another center based in California. Overall, though, emails peak around 10 AM, and continuously decrease in frequency after, which is understable as most people check their emails at the beginning of the work day.

Sentiment Analysis

We began our sentiment analysis by plotting the distribution of polarity for the random sample of 10,000 emails from the Enron corpus.

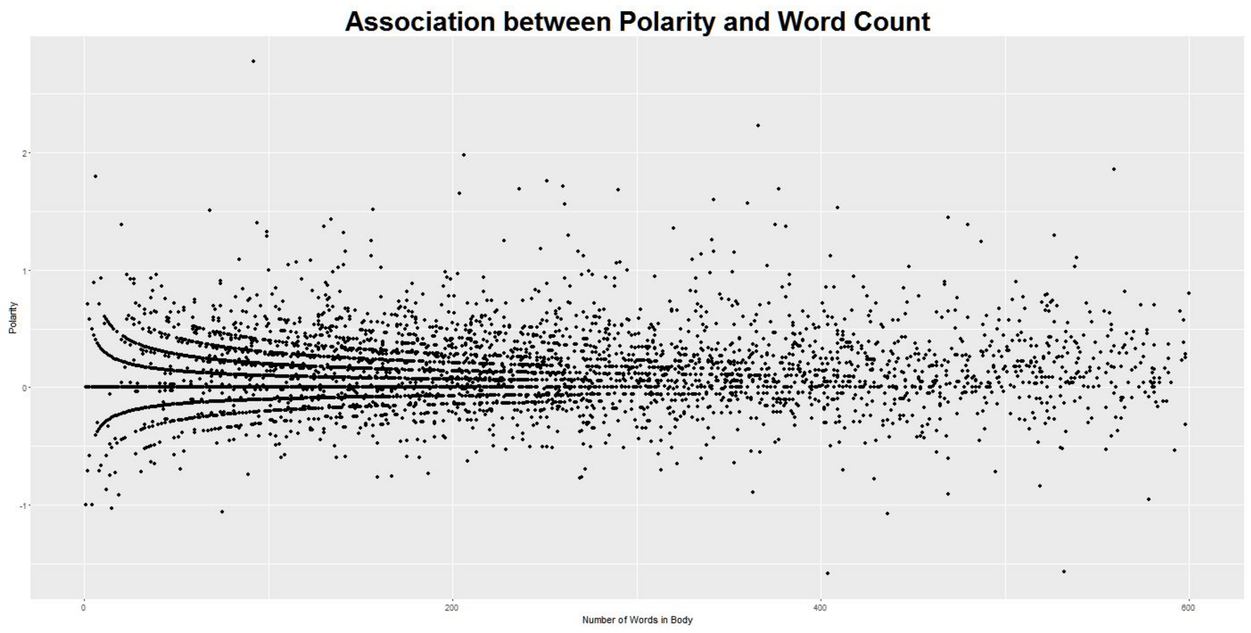




The graph above to the left shows the polarity distribution of all the emails. It's important to note that most emails have a polarity of zero as most words don't have a negative or positive connotation. Secondly, the distribution is skewed right. This is supported statistically, as the polarity has a mean of $\mu =$ 0.111261 and a median of $\widehat{x} = 0$. This shows that the polarity of more emails is positive, which is very understandable in a workplace environment - people tend to make an effort to be more amicable and polite.

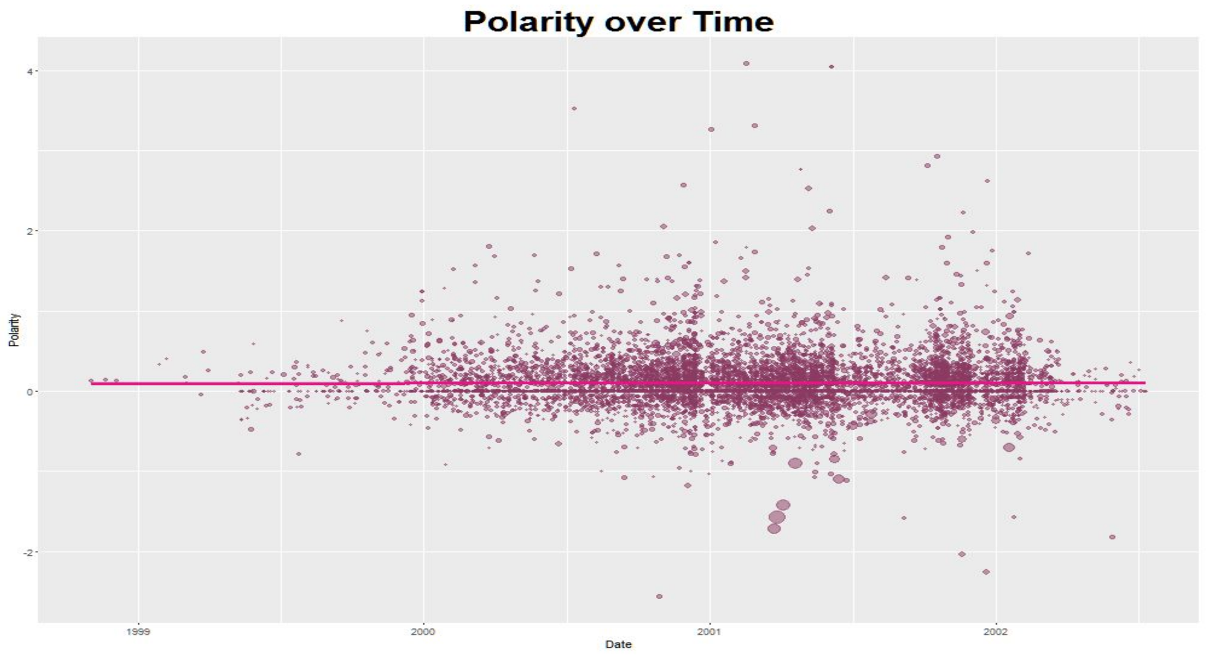
This explanation is further supported by the word cloud to the right, where the most frequent words in the sample text are the most prominent. The most noticeable of which is "please," which along with "enron", are the two most frequent words in all of the email samples. Several other very common positive words to show up on the diagram are "may", "well", and "good"; in fact, among the 100 words to be shown, the only remotely negative word is "don't". Many of the other most common words make sense in the context of Enron being an energy and commodities corporation, such as “market”, “power”, “trading”, “gas”, “financial”, “trading”, and “electricity”.

We then turn to the bulk of sentiment analysis: investigating associations between polarity and other factors. First, we graphed the relationship between the polarity of an email and the number of words in its body.

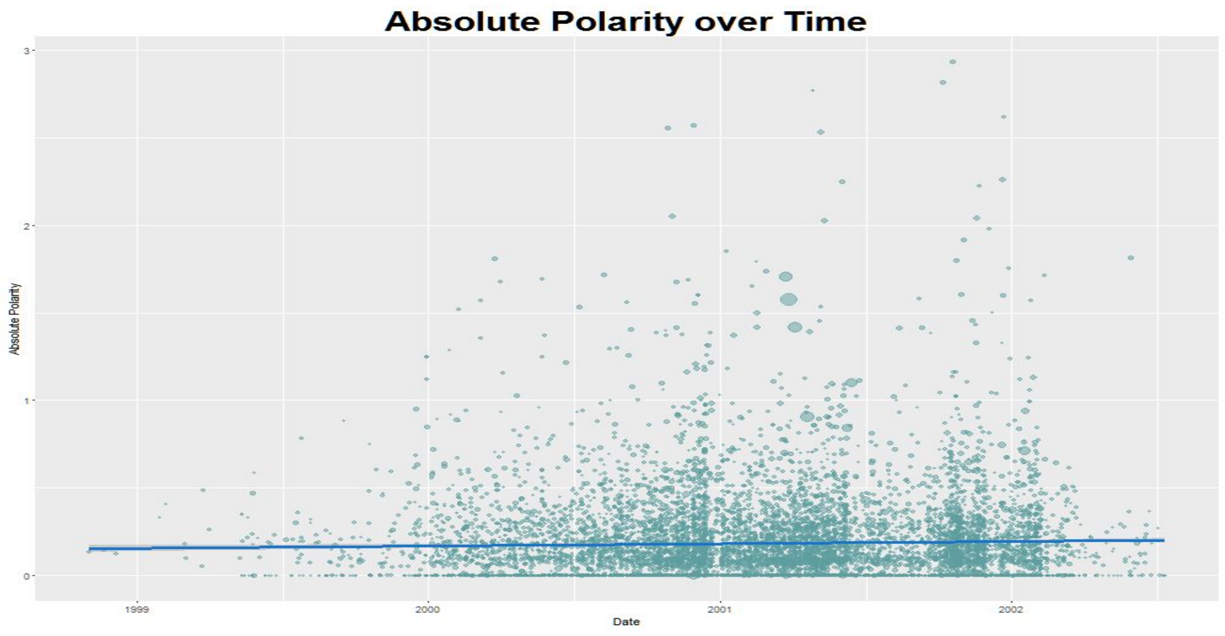


This plot indicates two key facts about the qdap algorithm for polarity calculations. Firstly, a text body with fewer words gives more weight to each individual word. So, if an email is very short, even just one or two non-neutral words can very easily sway the overall polarity. As bodies of text grow longer, the high frequency of neutral words balance out the fewer non-neutral words, resulting in less extreme polarity values. Secondly, the default word associations qdap uses seem to stem from about about a dozen categories. For example, words like “good” and “well” seem to be associated with the same group, whereas words like “miserable” and “failure” belong to another group with a more extreme initial polarity value. These categories manifest themselves as curved lines within the above graph, which deviate with amplification and word count.

Next, we began investigating relationships between polarity and time.

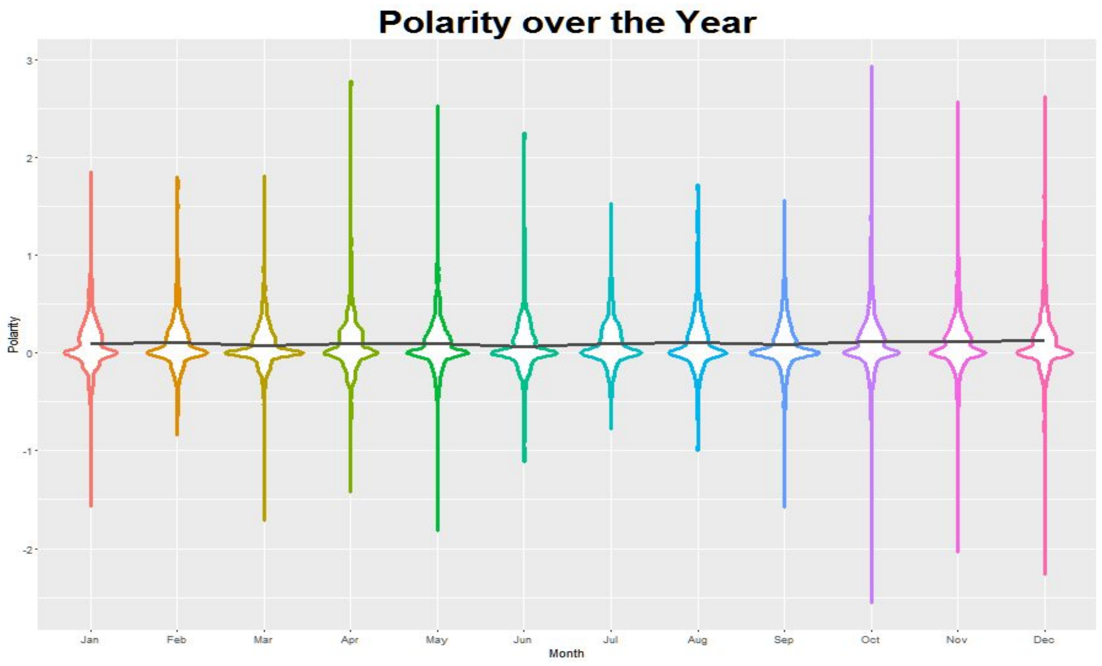


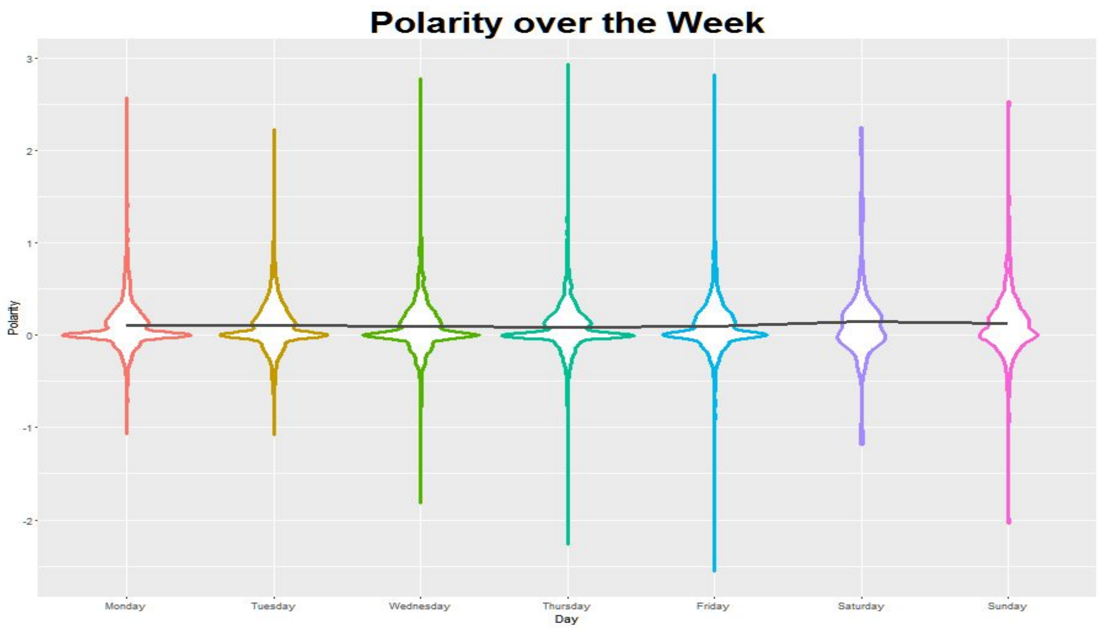
We hypothesized that Enron's disastrous crash might inspire more negative sentiment in the emails of its employees, but our results showed that the polarity of messages remained relatively consistent over time. This graph also shows that most emails are somewhat positive, rather than negative in nature, with most of the emails above the x-axis. However, cursory examination seemed to suggest that the polarity of messages may have become more extreme over time. So, we plotted the absolute value of the polarity to check.



After graphing the absolute polarity of Enron emails over time, we found that the sentiment of emails did get more extreme over time, but not by a notable amount. In this case, sentiment of employee emails was a poor predictor of company success.

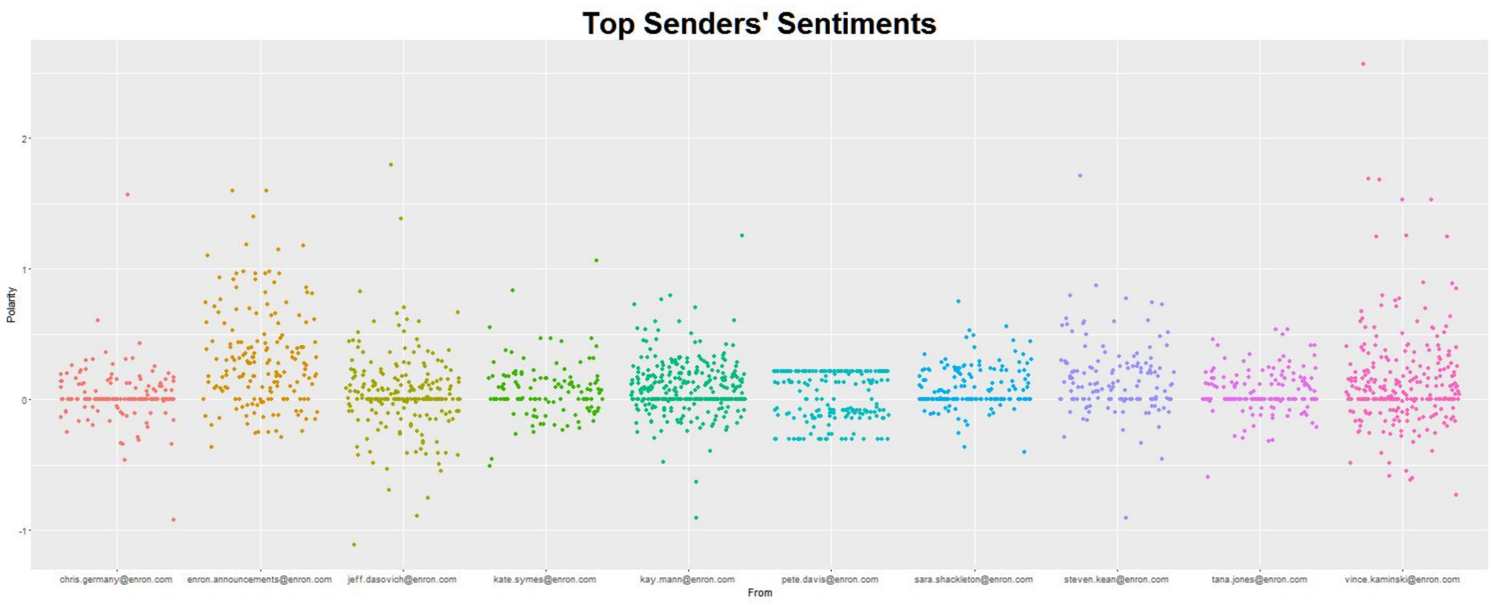
Next, we investigated whether there were any cyclical patterns in the sentiment of emails.





Again, we found no trends. Overall, the sentiment of an Enron employee email appears to be independent of the time or events surrounding its sending. We suspect that the neutrality of Enron emails could be attributed to their corporate setting, or that our sample is not representative of the corpus as a whole.

We ended our sentiment analysis by looking at the polarity of the emails sent by the ten users with the highest email activity.



While most users seem to follow the same trend displayed in the plot of polarity distribution, with most emails being neutral with an inclination towards positivity, pete.davis@enron.com and enron.announcements@enron.com stand out.

As stated above, Pete Davis sends success-failure reports to the rest of the company. Because of this, there appear to be two clusters in his polarity results, one for success reports, and another for failure reports. The Enron announcement email also displays unique behaviour; since their emails are usually memos to large groups of employees, their polarity seems to be skewed more so towards positive language, typical of an official company manifestation.

Conclusion

This preliminary investigation of sentiment in the Enron corpus leaves us at a good starting point for more in-depth analysis. The natural progression would be to recreate the polarity graphs with the entire corpus, and then to compare sentiment to more factors, such as stock prices, legal events, and email path. Additionally, sentiment analysis can be taken even further, with the use of statistical testing to measure the effect of one employee's tone on the people he sends the email too, or to validate if particular employees have more of an effect than others. As perhaps the largest publicly available representation of corporate culture and language, the Enron corpus stands today as an interesting, useful, and complex dataset, worthy of still more complex queries and investigations.

Appendix

enron.py

```
import os
import re
import time

header_elements = ["Date: ", "From: ", "To: ", "Subject: "]

def main(i_path, o_path = "", b_path = "broken.txt", search = False, phrase = "", limit =
517401):
    def look_for(path, phrase, o_path = ""):
        nonlocal counter
        nonlocal limit
        if counter > limit:
            return
        else:
            if os.path.isdir(path):
                for folder in os.listdir(path):
                    look_for(os.path.join(path, folder), phrase, o_path)
            else:
                email = open(path, 'r')
                for line in email:
                    if o_path != "":
                            print(counter, "/", limit, "|\t", path)
                    if phrase in line:
                        counter += 1
                        if o_path == "":
```

```
                        print(counter, "/", limit, ":\t", path, "\n", line, "\n\n")
                    else:
                        output.append(path + "\n" + line)

    def make_vec(path):
        nonlocal counter
        nonlocal limit
        if counter > limit:
            return
        else:
            if os.path.isdir(path):
                for folder in os.listdir(path):
                    make_vec(os.path.join(path, folder))
            else:
                email = open(path, 'r')
                body = []
                header = []
                element_len = [6, 6, 4, 9]
                body_start = False
                mult = -1
                for line in email:
                    if body_start:
                            body.append(line)
                            continue
                    if line.find("X-FileName: ") != -1:
                        if (len(header) != 4):
                            broken.append(path)
                            return
                        else:
                            body_start = True
                            continue
                    else: # e = [0,3] --> looking for Date, From, To, Subject
                        for e in range(len(header), 4):
                            n = line.find(header_elements[e])
                            if n != -1:
                                if e == 2 and mult == -1: # special case for "To: "
                                    mult = line.find(",")
                                    if mult != -1:
                                        header.append(line[n+element_len[e]:mult])
                                    else:
                                        header.append(line[n+element_len[e]:])
                                else:
                                    header.append(line[n+element_len[e]:])
                # email parsed successfully
                print(counter, "/", limit, ":\t", path)
                header = "º".join(header)
                body = "".join(body)
                output.append(path)
                output.append(header)
                output.append(body)
                counter+=1

    start = time.time()
    counter = 1
    output = []
    broken = []
    if search:
        look_for(i_path, phrase, o_path)
        if o_path != "":
            outFile = open(o_path, 'w')
            outFile.write("\n\n".join(output))
            outFile.close()
    else:
        make_vec(i_path)
        a = [s.replace('\n', '').replace('\r', '') for s in output]
        outFile = open(o_path, 'w')
        outFile.write("º".join(a) + "º")
        outFile.close()
        outFile = open(b_path, 'w')
        outFile.write("\n".join(broken))
        outFile.close()
    print("[", limit, " files took ", time.time() - start, " seconds ]")

# prints the first valid 20,000 files located to a new text file, "enron_short.txt"
main("maildir", o_path = "enron_short.txt", limit = 20000)
# prints the first 1000 hits to the phrase, "corporate losses" to console
main("maildir", search = True, phrase = "corporate losses", limit = 10)
```

count.py

```
def count(i_path):
    files = 0
    def look(path):
        nonlocal files
        if os.path.isdir(path):
            for folder in os.listdir(path):
                    look(os.path.join(path, folder))
        else:
            files+=1
            print("File ", files, ":\t", path)
    look(i_path)
    print("TOTAL:\t", files)
count("maildir")
```

enron_batch.R

```
library(qdap)
library(dplyr)
```

```
s = Sys.time()
t <- c()

# Reading
inPath <- "enron_full.txt"

enron <- scan(file=inPath, sep="º", quote="", what=character()) %>% head(-1) %>% matrix(ncol
= 6, byrow = TRUE) %>% as.data.frame(stringsAsFactors=FALSE)
names(enron) <- c("Path", "Date", "From", "To", "Subject", "Body")

dateify <- function(d) {
  as.POSIXct(substr(d, 6, nchar(d) - 11), format = "%d %b %Y %H:%M:%S")
}

enron$Date <- dateify(enron$Date)

ancient <- as.POSIXct("1998-01-01 09:22:03"
enron <- enron %>% filter(Date >= ancient)

t[1] = Sys.time() - s

# Sentiment Analysis
sp <- polarity(enron$Subject)[[1]]
t[2] = Sys.time() - s
bp <- polarity(enron$Body)[[1]]
t[3] = Sys.time() - s
enron$body.pol = bp$polarity
enron$body.pWords = bp$pos.words
enron$body.nWords = bp$neg.words
enron$body.wc = bp$wc
enron$subject.pol = sp$polarity
enron$subject.pWords = sp$pos.words
enron$subject.nWords = sp$neg.words
enron$subject.wc = sp$wc

outPath <- "enron.rds"
saveRDS(enron, file = outpath)
t
```