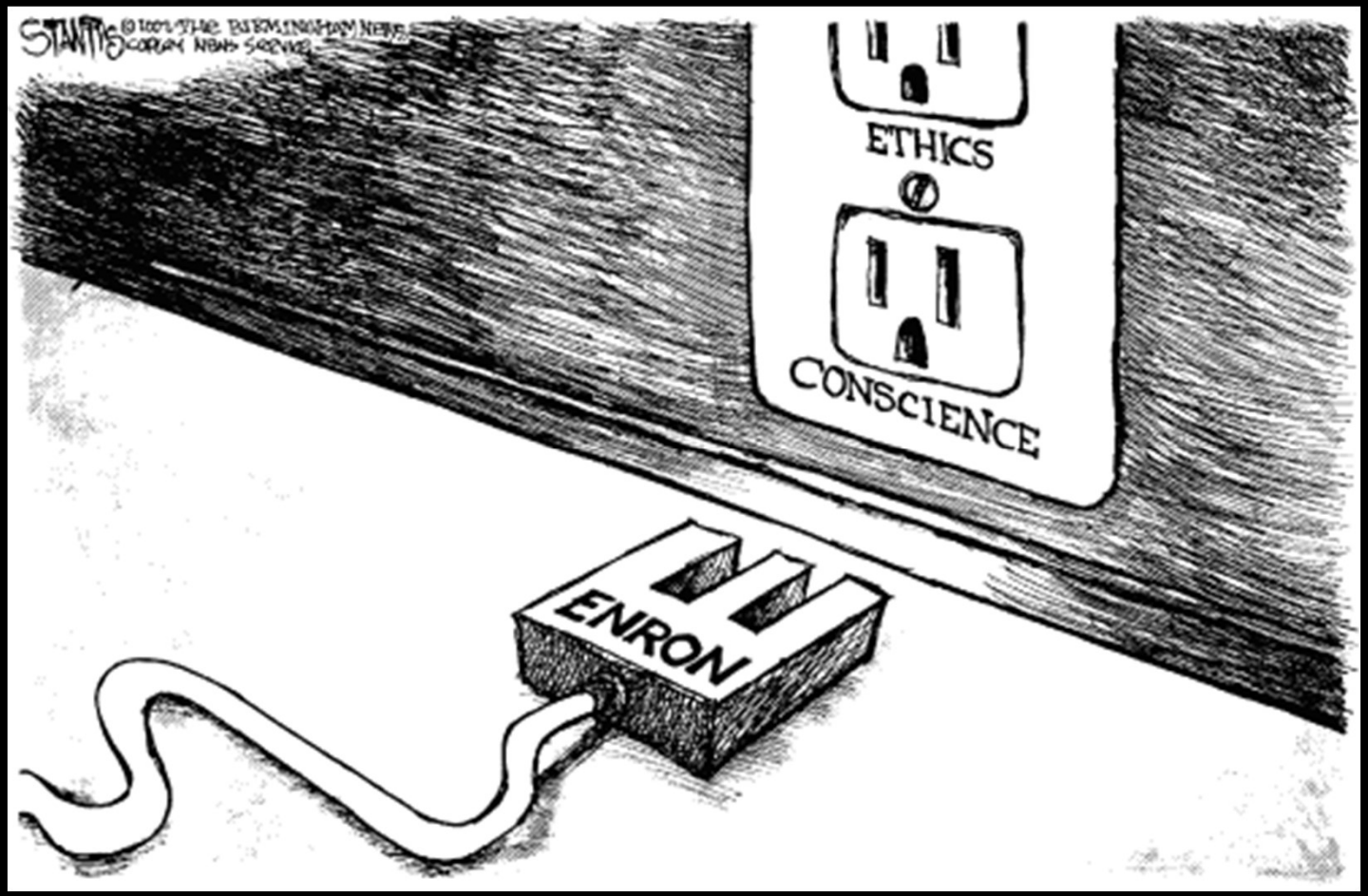
A Smoking Gun Classifier

Exploring the Enron Email Corpus


HOUSTON CHRONICLE
Guilty! Guilty!
Verdict will mean prison for ex-Enron chiefs


STANTIS
ETHICS
CONSCIENCE
ENRON

Why Emails?

- interesting class of written text
- open domain problem
- a useful solution
- presents unique challenges and solutions
  - network theory

- “I am distributing this dataset as a resource for researchers who are interested in improving current email tools, or understanding how email is currently used. This data is valuable; to my knowledge it is the only substantial collection of "real" email that is public. The reason other datasets are not public is because of privacy concerns. In using this dataset, please be sensitive to the privacy of the people involved (and remember that many of these people were certainly not involved in any of the actions which precipitated the investigation.)”

Goals

- accuracy
- find useful features
- understand the corpus better

The Data

Metadata

- History
- Size
  - 500,000 files
  - 3,500 folders
  - 1.7GB
- Version

Data Hierarchy

- “maildir/”
- user level
- folder level
- email level*

User Level

allen-p	1/23/2020 12:08 PM	File folder
arnold-j	1/23/2020 12:27 PM	File folder
arora-h	1/23/2020 12:15 PM	File folder
badeer-r	1/23/2020 11:20 AM	File folder
bailey-s	1/23/2020 11:28 AM	File folder
bass-e	1/23/2020 11:45 AM	File folder
baughman-d	4/14/2020 12:28 AM	File folder
beck-s	1/23/2020 12:15 PM	File folder
benson-r	1/23/2020 11:20 AM	File folder
blair-l	1/23/2020 11:20 AM	File folder
brawner-s	1/23/2020 11:48 AM	File folder
buy-r	1/23/2020 12:03 PM	File folder
campbell-l	1/23/2020 12:35 PM	File folder
carson-m	1/23/2020 12:20 PM	File folder
cash-m	4/14/2020 12:23 PM	File folder

Folder Level

_sent_mail	1/23/2020 12:27 PM	File folder
2000_conference	1/23/2020 12:27 PM	File folder
active_international	1/23/2020 12:27 PM	File folder
all_documents	1/23/2020 12:26 PM	File folder
avaya	1/23/2020 12:26 PM	File folder
bmc	1/23/2020 12:27 PM	File folder
bridge	1/23/2020 12:27 PM	File folder
bristol_babcock	1/23/2020 12:26 PM	File folder
colleen_koenig	1/23/2020 12:27 PM	File folder
compaq	1/23/2020 12:27 PM	File folder
computer_associates	1/23/2020 12:26 PM	File folder
continental_airlines	1/23/2020 12:26 PM	File folder
cooper_cameron	1/23/2020 12:27 PM	File folder
corestaff	1/23/2020 12:27 PM	File folder
deleted_items	1/23/2020 12:27 PM	File folder

File Level

1	2/3/2004 8:19 PM	File
2	2/3/2004 8:19 PM	File
3	2/3/2004 8:19 PM	File
4	2/3/2004 8:19 PM	File
5	2/3/2004 8:19 PM	File
6	2/3/2004 8:19 PM	File
7	2/3/2004 8:19 PM	File
8	2/3/2004 8:19 PM	File
9	2/3/2004 8:19 PM	File
10	2/3/2004 8:19 PM	File
11	2/3/2004 8:19 PM	File
12	2/3/2004 8:19 PM	File
13	2/3/2004 8:19 PM	File
14	2/3/2004 8:19 PM	File
15	2/3/2004 8:19 PM	File

An Email

```
4]: ['Message-ID: <29790972.1075855665306.JavaMail.evans@thyme>',
     'Date: Wed, 13 Dec 2000 18:41:00 -0800 (PST)',
     'From: 1.11913372.-2@multexinvestornetwork.com',
     'To: pallen@enron.com',
     "Subject: December 14, 2000 - Bear Stearns' predictions for telecom in Latin",
     ' America',
     'Mime-Version: 1.0',
     'Content-Type: text/plain; charset=us-ascii',
     'Content-Transfer-Encoding: 7bit',
     'X-From: Multex Investor <1.11913372.-2@multexinvestornetwork.com>',
     'X-To: <pallen@enron.com>',
     'X-cc: ',
     'X-bcc: ',
     'X-Folder: \\Phillip_Allen_Dec2000\\Notes Folders\\All documents',
     'X-Origin: Allen-P',
     'X-FileName: pallen.nsf',
     '',
     "In today's Daily Update you'll find free reports on",
     'America Online (AOL), Divine Interventures (DVIN),',
     'and 3M (MMM); reports on the broadband space, Latin']
```

Note: We know that the first 15 or so lines at the beginning of every email are the same. It's biographical information that we can use as features in future machine learning algorithms. What's the problem? Well, when there is a long line, this disrupts the number of line pattern we could be using (look under *Subject*). Lets see if we can solve this in the next section.

Techniques: Raw Parse

```python
MessageID_tup = email[0].split(":")
Date_tup = email[1].split(":")
From_tup = email[2].split(":")
To_tup = email[3].split(":")
Subject_tup = email[4].split(":")
MimeVers_tup = email[5].split(":")
ContentType_tup = email[6].split(":")
Encoding_tup = email[7].split(":")
XFrom_tup = email[8].split(":")
XTo_tup = email[9].split(":")
CC_tup = email[10].split(":")
BCC_tup = email[11].split(":")
XFolder_tup = email[12].split(":")
XOrigin_tup = email[13].split(":")
XFilename_tup = email[14].split(":")
header = [MessageID_tup, Date_tup, From_tup, To_tup, Subject_tup, MimeVers_tup, ContentType_tup, Encoding_tup,
         XFrom_tup, XTo_tup, CC_tup, BCC_tup, XFolder_tup, XOrigin_tup, XFilename_tup]
```

```python
text = email[16:]
for line in text:
    if line == '':
        text.remove(line)
text
```

```python
count = 0
for line in email:
    if line == '':
        pass
    else:
        if line[0].isspace():
            email[count-1] += line
            del email[count]
            print(count)
    count += 1
email[:20]
```

Techniques: Email.Parser Module

```
def readEmailHead(username, emailNum, corpus_root='maildir'):
    fname = f"/{corpus_root}/{username}/all_documents/{emailNum}"
    with open(fname) as fd:
        pp = email.parser.Parser()
        header = pp.parse(fd, headersonly=True) #where the magic happens.
    return header
```

Techniques: Email.Parser Module

```python
sample = readEmailHead('allen-p', 1, corpus_root=cr)
sample.items()
email_df = pd.DataFrame(columns=sample.keys())
email_df
```

```
[('Message-ID', '<29790972.1075855665306.JavaMail.evans@thyme>'),
 ('Date', 'Wed, 13 Dec 2000 18:41:00 -0800 (PST)'),
 ('From', '1.11913372.-2@multexinvestornetwork.com'),
 ('To', 'pallen@enron.com'),
 ('Subject',
  "December 14, 2000 - Bear Stearns' predictions for telecom in Latin\n America"),
 ('Mime-Version', '1.0'),
 ('Content-Type', 'text/plain; charset=us-ascii'),
 ('Content-Transfer-Encoding', '7bit'),
 ('X-From', 'Multex Investor <1.11913372.-2@multexinvestornetwork.com>'),
 ('X-To', '<pallen@enron.com>'),
 ('X-cc', ''),
 ('X-bcc', ''),
 ('X-Folder', '\\Phillip_Allen_Dec2000\\Notes Folders\\All documents'),
 ('X-Origin', 'Allen-P'),
 ('X-FileName', 'pallen.nsf')]
```

Message-ID	Date	From	To	Subject	Mime-Version	Content-Type	Content-Transfer-Encoding	X-From	X-To	X-cc	X-bcc	X-Folder	X-Origin	X-FileName

Techniques: Email.Parser Module

```
header_all = [] #a list of header data from all emails
sample_size = 5
curr_email = 1
while sample_size > 0:
    header = readEmailHead('allen-p', sample_size, corpus_root=cr)
    header_one = [] #a list of header data from one email
    for line in header.values():
        header_one.append(line)
    header_all.append(header_one)
    sample_size = sample_size - 1
email_df = pd.DataFrame(header_all, columns=sample.keys())
email_df
```

Techniques: OS Module

```
for name in os.listdir(path):
    relpath = path + name + "/"
    print(name + "'s folders: ")
    #for folder in os.listdir(relpath):
        #filepath = relpath + folder + "/"
        #print("folder<" + folder + ">: ")
        #for file in os.listdir(filepath):
```

```
allen-p's folders:
arnold-j's folders:
arora-h's folders:
badeer-r's folders:
bailey-s's folders:
bass-e's folders:
baughman-d's folders:
beck-s's folders:
```

Techniques: OS Module

```
emails = []
folders = []
users = []
fileErrors = 0
folderErrors = 0
totalEmails = 31000
for name in os.listdir(path):
    relpath = path + name + "/"
    print(name + " loaded")
    for folder in os.listdir(relpath):
        filepath = relpath + folder + "/"
        try:
            for file in os.listdir(filepath):
                try:
                    emails.append(readEmailHead(filepath+file))
                    users.append(name)
                    folders.append(folder)
                except:
                    #print("file error")
                    fileErrors += 1
                    continue
        except:
            #print("folder error")
            folderErrors += 1
            continue
totalErrors = fileErrors+folderErrors
accuracy = 1-totalErrors/totalEmails
print("users loaded with " + (str)(totalErrors) + " total errors, at a " + (str)(accuracy) + "% accuracy")
```

Viability of OS Walking

- coverage
- assumptions
- other corpora?

Data Exploration and Machine Learning

- Manual Annotation:
- problems
- benefits

- Unsupervised to Supervised:
- problems
- benefits

Future Directions and Core Takeaways

- summer project
- apply for fall conferences
- BPhil

- data cleaning takes forever
- exceptions are a must
- look hard for previous solutions
- sometimes just move on