Assignment 1 Project Report

Team members: Zixi Jiang, Peizhen Li, Xiaoyu Wang, Yuchen Zhang, Xiuwen Zhang, Nat Zheng
GitHub Repo: https://github.com/Anthonyive/DSCI-550-Assignment-1

About the dataset

After we converted the original dataset into separated json files using Tika, we firstly found the dataset was a little messy. Within 4029 json files that we created, each file has different numbers and types of attributes. Several files are metadata of attachments of another email, and some files are a list of email addresses without any attributes.
Secondly, since each of our team members did different tasks in task 5, the structures of the outputs were slightly different and we did not have a uniform standard for the unknown and missing values. Some of us put 'unknown', some put 'N/A' and others didn't contain the features in their output if the value of the feature is unknown. To resolve the problems, we did another data-cleaning process so that we had the same format for every missing value and the same structure for all the outputs to be integrated together.
Our final TSV file contains 28 features for each email.

Expanding the dataset

In this section, we will provide some clarification on our features generation process. In addition, we will illustrate what kind of query or questions could be answered by these additional features.

We mainly used keywords scanning to determine the attack type. A few features are rather straightforward, being able to be extracted from the metadata or text content. Below are a few attributes that we would love to provide detailed explanation on how we obtained them.

- Attacker title: Since the attackers' titles are always at the beginning of their names, we found the attribute about the sender's name and retrieved the title by first splitting the name and taking the first word. For those senders' names without blank space, we made a list for the titles and searched to classify them.

- Urgency of the email: In order to detect the urgency of attacking emails, we used the Python NLTK.tokenize.WordPunctTokenizer() method to segment and count the frequency of the words "urgent" and "now" in "separated by email". In addition, we made statistics on the total frequency of "urgent" and "now" in "fradulent_emails".Through counting, it can be seen that in most json files of "separated by email", the frequency of urgent and now is 0 or 1. This shows that the frequency of urgency words are much lower in most fraudulent emails. The highest frequency of emergence of urgent and now is 5.

- Attacker location: The easiest way to track the attacker location is by extracting the IP address of the email. From this dataset, there are only approximately 500 ip addresses that can be retrieved. We used an IP API to obtain the latitude and longitude for each IP. For the rest of the emails that does not provide an ip address, we imported a library called geoparser to extract locations that are mentioned in the email content, geoparser.country_city provides the country and the city noticed. Using the google geocode API will also be able to provide us the latitude and longitude of the location.

- Attacker relationship: We used the model from "Deep Learning has (almost) all the answers: Yes/No Question Answering with Transformers" in Medium. The model uses google research datasets about boolean questions called BoolQ where BoolQ is a question answering dataset for yes/no questions containing 15942 examples. The model has an evaluation accuracy of more than 80%. The goal of using this model is to extract answers from a given boolean question and a given email content. We will take the answers that are answered yes and convert it into human-readable form. For instance, if we asked "Did we meet online?" and the answer was yes, we will store the output as "online". In some instances, the attacker may have multiple relationships with the victim, this case we will store them all.

- Attacker email sentiment: We used the method of analyzeSentiment from Google Cloud Natural Language API. For each email, we used the API to get a sentiment score. A negative score indicates Negative, a positive score means Positive and a score equals 0 means neutral.

- Attacker language style: We used Pyspellchecker package to check the email body and generated the number of wrong-spelled words. We detected wrong capitalizations in the text and calculated the ratio of the wrong_cap words to the total number of words.

- Attacker estimated age: We used USC Data Science AgePredictor, but instead of using the model it provided, we used our pre-trained model using Blog Authorship Corpus from kaggle. The Blog Authorship Corpus consists of the collected posts of 19,320 bloggers gathered from blogger.com in August 2004. We mainly used the age and text columns. We separated the corpus into training, testing, evaluating with correspondingly 80%, 10%, 10%. The training accuracy can get up to 53.205% while the evaluation accuracy can get up to 56.23%. These accuracy results are slightly higher than the model provided by the age predictor. We hope this dataset can give better results. As we were testing, we used "I am 18 years old", and our model successfully predicted the age is 18 while other models struggled.

Features generated by joining external dataset by dataset

Dataset I: Prevalence of Selected Measures Among Adults Aged 20 and Over

How likely do these attackers have some diseases?

The MIME type of this dataset is application/xml. This data represents the ratio of high total cholesterol, hypertension, and obesity in different populations grouped by age from 1999 to 2018. We joined the datasets by the feature keys of date and predicted age. For each email, based on the predicted age of the attacker and the year that email was sent, we get the probability of the email sender having hypertension, obesity, and high total cholesterol respectively as three new features.

- Feature 1: Hypertensions rate (the estimated probability of email sender having hypertension)
- Feature 2: Obesity rate (the estimated probability of email sender having obesity)
- Feature 3: High total cholesterol rate (the estimated probability of email sender having high total cholesterol)

It comes from data.gov as a XML file, published by National Center for Health Statistics, could be retrieved at https://catalog.data.gov/dataset/prevalence-of-selected-measures-among-adults-aged-20-and-over-united-states-1999-2000-2017

Dataset II: S&P 500 Historical Stock Exchanges

What is the stock price and volume on the specific day when the attacker sent the email?

The MIME type of this dataset is text/csv. This dataset displays S&P 500's historical stock data from 1998 to 2017. S&P 500 is a market-capitalization-weighted index of the 500 largest U.S. publicly traded companies which could be an indicator of the overall performance of the stock market on that day. The common attribute between S&P 500 historical dataset and fraud email dataset is the date. There are three attributes, daily open price, close price and volume extracted from this dataset. We were planning to utilize these three attributes to analyze the relationship between the performance of the stock market and the number of spam emails sent or number of victims have been duped of the day.

- Feature 1: open (stock' open price on the trading day)
- Feature 2: close (stock's close price on the trading day)
- Feature 3: volume (number of stocks has been trade on the trading day)

Reference: https://finance.yahoo.com/quote/%5EGSPC/history

Dataset III: Gender by Name Data Set by UCI Machine Learning Repository

What is the gender of the attacker?

The MIME type of this dataset is text/csv. This dataset combines raw counts for first/given names of male and female babies in those time periods, and then calculates a probability for a name given the aggregate count.

- Feature 1: male count
- Feature 2: female count
- Feature 3: unknown count (the count of extracted names that could not be identified by the dataset)

Gender by Name Data Set has a column of names and their corresponding genders. We use the spacy library to do entity extraction to extract names from email contents. Then, we use the dataset as a look up table and see whether these corresponding name's genders are male, female or unknown.
This dataset allowed us to answer what gender the attacker wanted to pretend when they were attacking the victim and potentially why they wanted to pretend to be like that. If the gender majority is female, maybe that is the majority of attackers wanted to pretend and they wanted to be approaching or obliging.
Dataset: https://archive.ics.uci.edu/ml/datasets/Gender+by+Name

Tika Similarity Analysis

x-coordinate	y-coordinate	Similarity_score
sep_by_email/3721.json	sep_by_email/2833.json	0.272727273
sep_by_email/3721.json	sep_by_email/729.json	0.206896552
sep_by_email/3721.json	sep_by_email/3371.json	0.166666667

Jaccard Similarity

x-coordinate	y-coordinate	Similarity_score
data/3721.json	data/2833.json	0.910866934
data/3721.json	data/729.json	0.881448098
data/3721.json	data/3371.json	0.889105341

Cosine Similarity

x-coordinate	y-coordinate	Similarity_score
../sep_by_email/3721.json	../sep_by_email/2833.json	0.597851171
../sep_by_email/3721.json	../sep_by_email/729.json	0.588767631
../sep_by_email/3721.json	../sep_by_email/3371.json	0.547351754

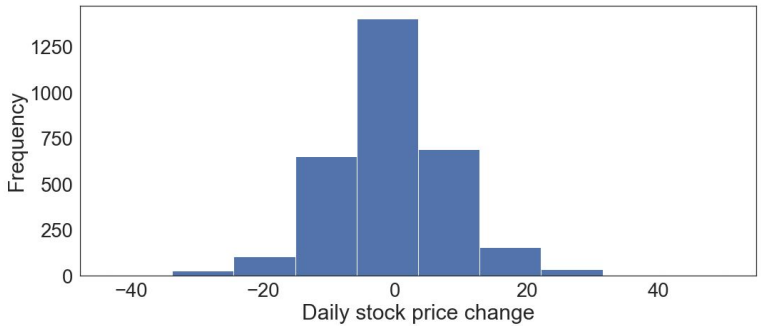
Edit Distance Similarity

These three images are snapshots of Jaccard Similarity, Cosine Similarity and Edit Distance Similarity accordingly. Jaccard Similarity overall provides the lowest score between two independent json files while Cosine Similarity presents the highest similarity score of two individual json files. Cosine similarity metric is

more accurate since the cosine index is used to distinguish between plagiarized files in general. On the other hand, Jaccard index will provide a more accurate result when identifying mirror objects, such as mirror sites; while Edit Distance metric is used to calculate the shortest distance converting one word to the other, which is not quite appropriate in this context. The relative high similarity score produced by Cosine metric indicates that the attributes and features that we extracted from these phishing emails are constructed with similar contents, structures and semantics. For further research, after adding some other extra features, we hope we will be able to leverage this dataset as a benchmark to cluster phishing attacks.

Unintended consequences

We have a dataset about the stock price for the day when the attackers sent these emails. When we joined the stock data, we thought that the change of stock price might affect those attackers since the stockholders might be influenced by the great change of stock price and hence made some unsmart decisions. However, according to the histogram about the daily stock price change, we could find the distribution is normalized. Hence a greater change of stock price would not significantly influence the attackers.



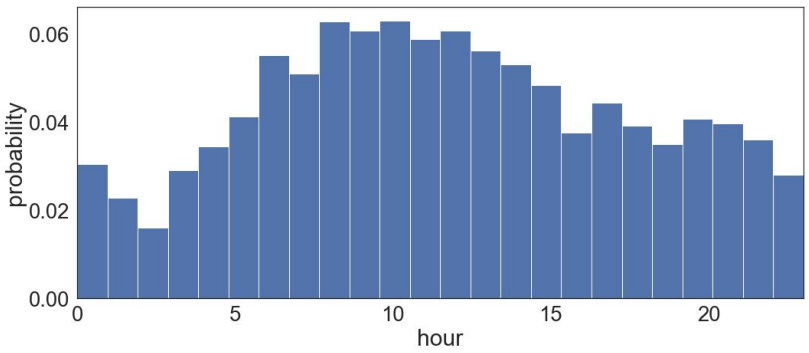
Deep thinking

1. *Are there clusters of attackers with similar features that tend to attack victims the same way?*

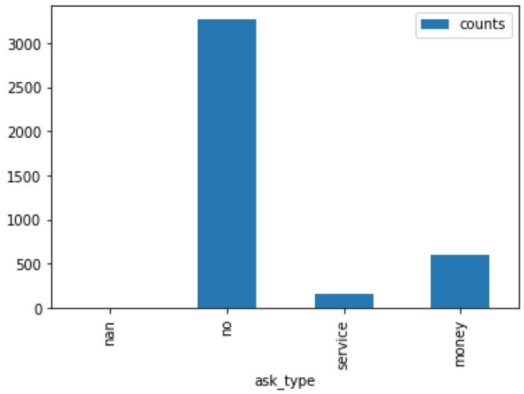
Emails that have cosine similarity equaling to 0.91 share similar attack stylometrics in terms of the preference of saying now or urgent. They also have similar attacker relationships (mostly online, friend of friend and in person). Male mentionings are much higher than females.

2. *Does the time of day of attack matter?*

In order to answer this question, we have drawn a histogram illustrating the time when these attack emails were sent. To make the plot clear, we picked up the nearby hour for each time point. According to the plot, the attackers are more likely to send emails during 8 a.m to 12 p.m in a random day. It is 1.5 times more likely for them to send emails in the morning compared to the afternoon. As a result, the time of day would be a minor matter.



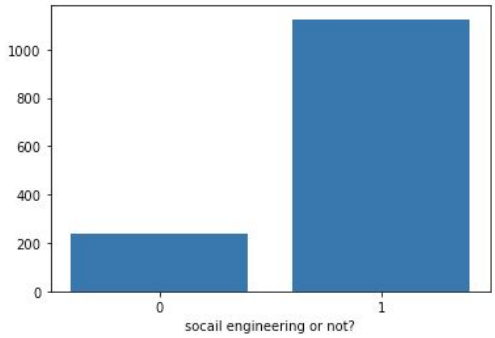
3. *Are specific types of asks more prevalent?*



As shown from this graph, the ask_type “no” constitutes the majority of the database. Asking for money or providing money is more prevalent than providing service within these scamming emails. After analysing this question, we realized that we should include more keywords that are sensitive to any type of services provided or any kinds of money transaction to improve the accuracy of “ask_type” attributes.

4. *Is there a set of frequently co-occurring features that induce the email to be read?* #to be added

One feature is about social engineering. We found all the emails that were read by the victims and checked if the emails were with social engineering or not. According to the bar plot, among about 1400 emails, over 100 of them had social engineering. So it is clear that the construction of social engineering is very likely to induce the email to be read.



5. *What insights do the “indirect” features you extracted tell us about the data?*

Due to the large amount of information in fraudulent emails, we cannot directly extract data about the frequency of word urgency. After counting, the total number of urgent frequencies is 15381, and the frequency of now is 14929. However, by extracting the frequency of words such as "urgent" and "now" in the json file of each email, in most json files "separated by email", The highest frequency of urgent and now is 5 but most of the frequency of urgent and now is between 0 and 1. It is clear that the data shows the frequency of urgent words is much lower in the most fraudulent emails.

6. *What clusters of Attacks and Attacker stylometrics made the most sense? Why?*

From the perspective of Attacks, social engineering made the most sense. Using “friendship”,“intimate” relationship or any types of urgent words can easily attract victims’ attention. This technique is more plausible if scammers would ever send out deceiving emails. Reconnaissance is not capable of explicitly indicating whether victims fell into the trap or not; while malware and credential phishing will quickly alarm victims’ cautiousness.

In terms of Attacker stylometrics, date/time of the email attack, attacker IP known as Phisher. Attackers are likely to send out emails during the day time since there are more people to check their emails from morning to noon. The website to check the attacker IP is a credible source and therefore checking the sender's IP is a sustainable and plausible way to detect a phisher.

7. *Also include your thoughts about Apache Tika – what was easy about using it? What wasn't?*

For people like us who are not familiar with Java, Apache Tika has some drawbacks in terms of ease of use. We actually have some experience with Bash/Zsh, so figuring out paths is relatively easy. However, although tika has a GUI, the GUI cannot process large files, so Tika is not like the type of programs/applications that just need to be downloaded and "plug-and-play".
Also, Tika generated output sometimes gives us unexpected results. For instance, we fed the entire fraudulent email dataset into Tika, and Tika generated around 20 attributes for each email, which were normal. However, when we concatenated all the outputs generated by Tika, we surprisingly had thousands of attributes. Some of them seem to be wrong keys.
That being said, Tika actually did what it needed to do and was fast to implement.

Contribution

Yuchen Zhang ● Task 4: Tika data generation ● Task 5b vi, ix: Attacker relationship and attacker estimated age ● Task 6, Dataset III (Name gender): Data generation ● Task 7, Tika similarity: generate edit-distance similarities and circle-packing visualization ● Miscellaneous: README.md, package management.	Xiuwen Zhang ● Task 5a: reconnaissance, social engineering, malware, credential phishing ● Task 5b iv, x: what does attacker offer, ip risk score and level ● Task 6 Dataset II: data-cleaning, extract attributes and merge new features into email dataset ● Task 7: Tika similarity: generate cosine similarity ● Attribute analysis
Nat Zheng ● Task 5b v: locate attacker's location by ip and email content ● Task 6 Dataset II : scrape S&P's historical stock data from 1998 to 2017 on Yahoo Finance ● Attribute analysis & generate visualization	Zixi Jiang ● Task 5b iii, viii: extract data/time attributes from the metadata; calculate wrong capitalization ratio ● Task 6 Dataset I: find obesity dataset and identify relevant attributes ● Attribute analysis & visualizations
Peizhen Li ● Task 5b ii,viii:Urgency of the attack email (word strength, "urgent", "now") ● Task 6 Dataset III(Name gender): Dataset finding(We found shooting incident data at first ,but we changed to name and gender data); Extract data ● Attribute analysis & generate visualization	Xiaoyu Wang ● Task 5b i,vii,viii(1): attacker title, email sentiment, misspellings ● Task 6 Dataset I: convert json file to usable attributes ● Task 7: Tika similarity: generating jaccard_similarity ● Attribute analysis & generate visualization