ML for SS: Traits and GPT

Dr. Richard M. Crowley
rcrowley@smu.edu.sg
https://rmc.link/

Overview

Papers

> Gentzkow, Shapiro and Taddy (2019) Econometrica.

- Shows the methodological benefits that can come from careful merging of econometrics and machine learning

> Eichstaedt et al. (2015) PS.

- Psychology + Social Media = Healthcare predictions.

> Kim, Muhn, and Nikolaev (2023) Working.

- Uses GPT as a proxy for how investors might [miss]use it

Technical Discussion: Linguistics

1. A bit on the methods from the papers
2. A bit on some methods that are open source (from the optional readings)
3. A bit on GPT at a high level
    - We'll construct a simple GPT in class!

Python

- Twitter Emotion Recognition
    - A neural network approach to labeling emotion latent to tweets
- A GPT made in Pytorch

Java + WEKA

- Personality Recognizer
    - A noisy but validated way to detect personality based on writing

> These tools tend to be released for just 1 language, so it's good to be flexible

Twitter Emotion Recognition

Emotion: Ekman’s 6 emotions

1. Anger
2. Surprise
3. Disgust
4. Enjoyment
5. Fear
6. Sadness

> Why is this useful?

- Can be a useful IV for a regression
  - E.g., understanding emotional response to government policies

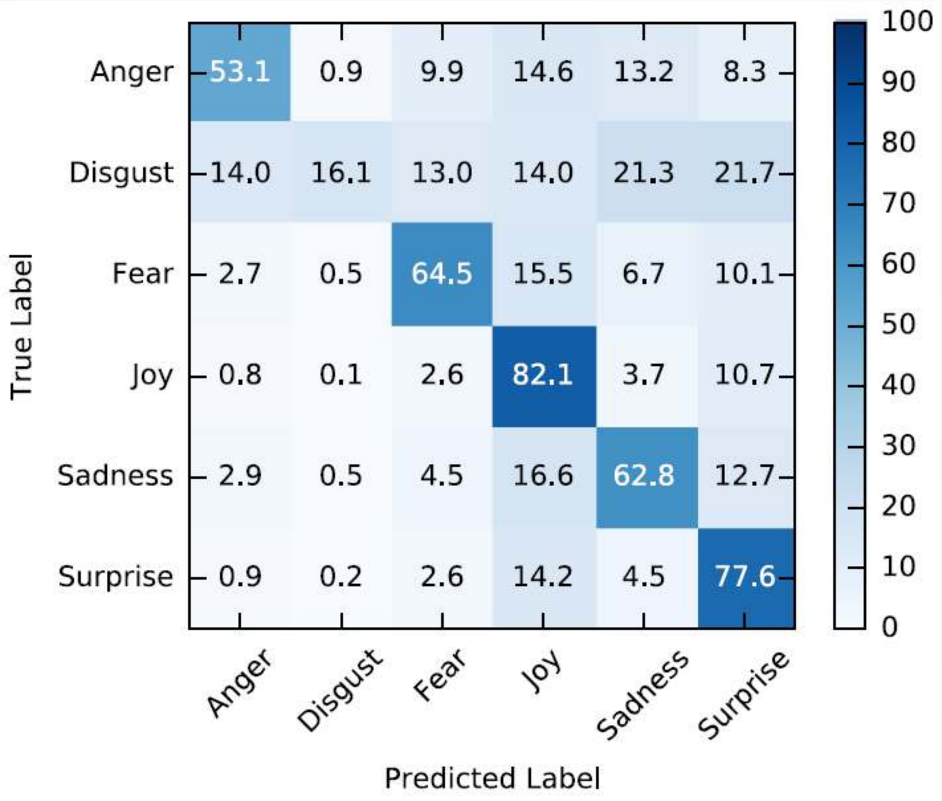
How do they do it?

1. Grabbed a collection of 73 *billion* tweets
2. Looked for tweets with hashtags directly matching the emotions, e.g., `#anger`
   - To enforce this as a label, the hashtag must be in the last 10% of the tweet (by token)
   - Removed duplicates and retweets as well
3. Use the pre-labeled tweets as a “weak supervision” to train an RNN
   - Also tried a CNN, but it doesn’t work as well
   - The open source version is applied character-by-character; superior to the tokenized version

> Other variations

1. Single class prediction vs multiclass
2. Other emotion classification schemes: Plutchik’s 8 emotions and Profile of Mood States (POMS)

How well does it work?



> Works well for Joy and surprise, and works alright for fear and sadness. Poor performance for disgust.

> Doesn’t control for sarcasm. No neutral class.

Example of running the algorithm Setup

```python
import os;
os.environ['KERAS_BACKEND'] = 'theano'

import pandas as pd

from emotion_predictor import EmotionPredictor

# Import the model
model = EmotionPredictor(classification='ekman', setting='mc', use_unison_model=True)

# Somewhat randomly pulled from Twitter
tweets = [
    "What saddens me most is that we seem to be fast becoming an “us vs them” society.",
    "I don't understand why the government hasn't factored in waiting for vaccinations being available to u
    "Switched to Windows 11! Looking and feeling rlly great so far Star-struck",
    "I got a pint of Windows 11! IT’S SO GOOD",
    "I’m not even a top 100 earning Twitch streamer, what the fuck is my community even doing out there??"
]
```

Example of running the algorithm: Output

- Most likely label

```python
predictions = model.predict_classes(tweets)
predictions['Emotion']
```

	Tweet	Emotion
0	What saddens me most is that we seem to be fas...	Sadness
1	I don't understand why the government hasn't f...	Fear
2	Switched to Windows 11! Looking and feeling rl...	Joy
3	I got a pint of Windows 11! IT’S SO GOOD	Surprise
4	I’m not even a top 100 earning Twitch streamer...	Anger

- Percentages

```python
probabilities = model.predict_probabilities(tweets)
probabilities
```

	Tweet	Anger	Disgust	Fear	Joy	Sadness	Surprise
0	What saddens me most is that we seem to be fas...	0.010681	0.005633	0.076239	0.006292	0.898160	0.002987
1	I don't understand why the government hasn't f...	0.000313	0.002712	0.995694	0.000672	0.000421	0.000187
2	Switched to Windows 11! Looking and feeling rl...	0.010332	0.002005	0.061332	0.360945	0.320342	0.245043
3	I got a pint of Windows 11! IT’S SO GOOD	0.023655	0.004097	0.082532	0.173595	0.094921	0.621201
4	I’m not even a top 100 earning Twitch streamer...	0.430638	0.060563	0.085490	0.022956	0.318132	0.082221

Example of running the algorithm: Output

- If using in a neural network, the embedding level is also made available

```python
embeddings = model.embed(tweets)
embeddings.shape
```

```
(6, 801)
```

Neural network embeddings

This is similar to the more complex embeddings we discussed last week. This model *will* give you a high-dimensional representation of your text. However, it only retains the information useful for it's classification exercise; e.g., there is information loss, and it is intentional.

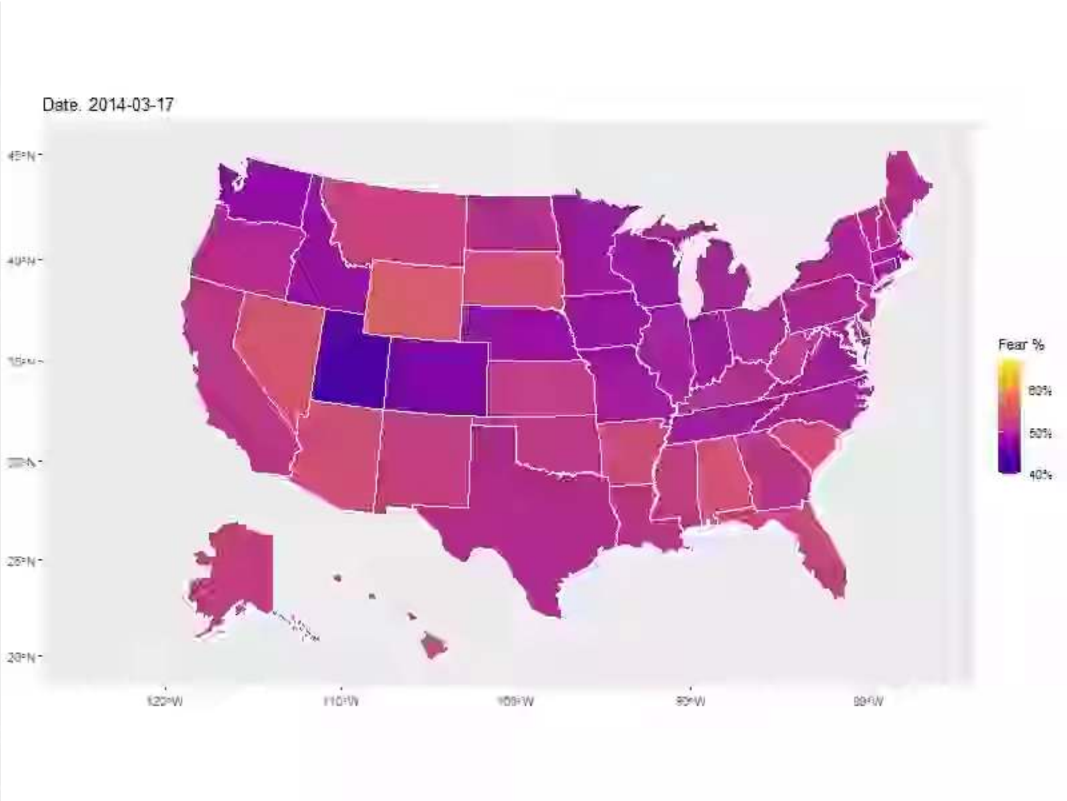
Try it out!

- The authors of the paper put out a public Binder for the algorithm
  - Binder is a cloud hosted Jupyter notebook

> Click here to access it [If it doesn’t work the first time, just use Ctrl+Shift+R to force a full refresh]

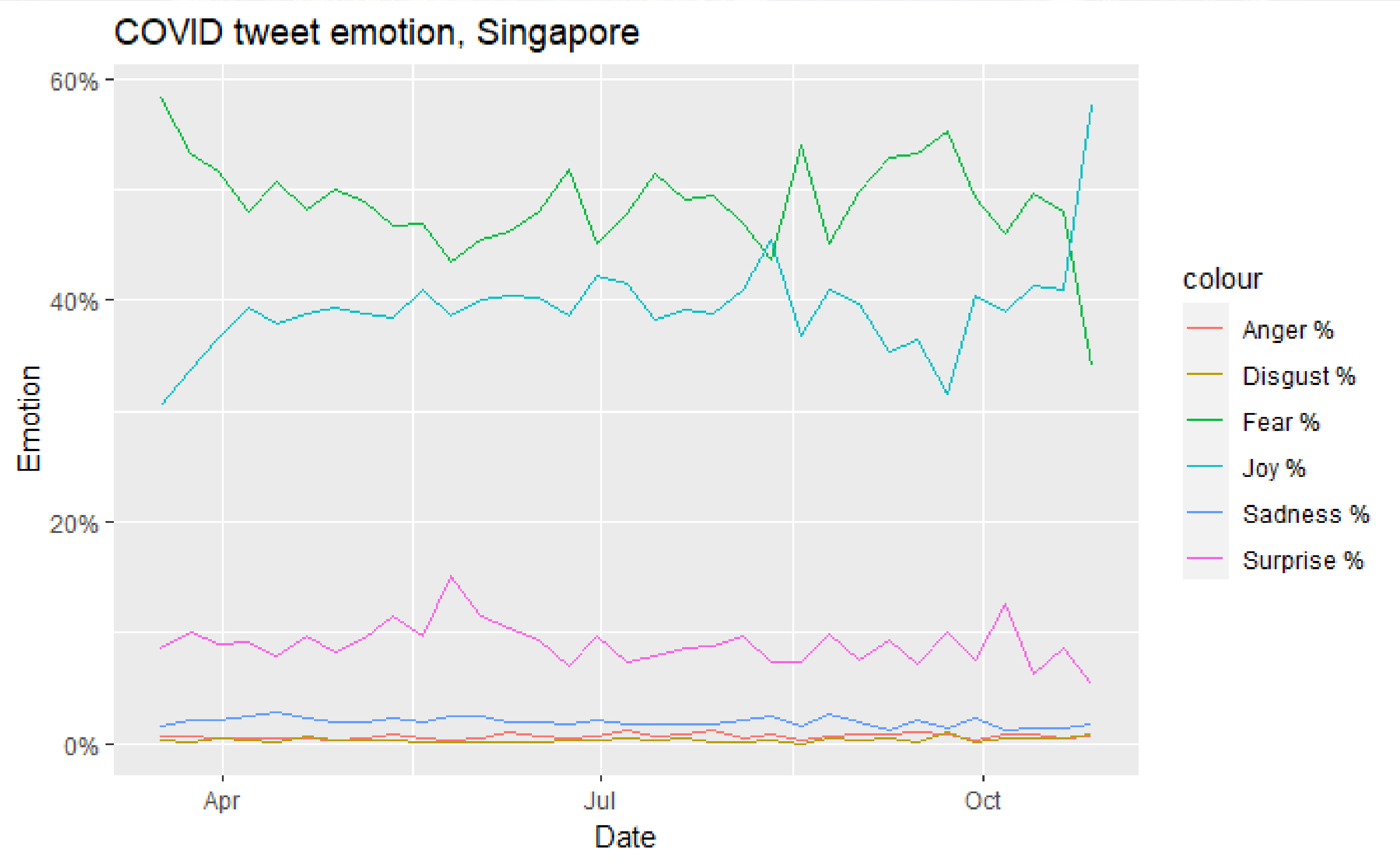
- Note:
  1. It’s a bit slow to run because the neural network it’s using is quite a large file
  2. You can try your own tweets by replacing the list `tweets` in cell number 4

Fear around COVID across the US, 2020 Mar to Oct



- May 27: COVID-19 deaths in the U.S. passed 100,000.
- Oct 2: US President tests positive for COVID-19

Singapore emotion and COVID-19



Personality Recognizer

Personality: The Big 5

1. Extraversion
2. Emotional stability
3. Agreeableness
4. Conscientiousness
5. Openness to experience

> The idea is to use cues from text (either written or transcribed) to identify a person's personality

- There are a lot of documented differences in speech across personality types, so the hope is to learn these from text and build it all into a model

The algorithm

1. Run a psychology experiment to collect text corpora and administer personality tests
    - Already done in Pennebaker and King 1999 (written text) and Mehl et al. 2001 (transcribed conversations)
2. Process the corpora
    - Examine word counts in a number of word lists from LIWC
    - Examine word counts from the MRC Psycholinguistic database
    - Other linguistic aspects: commands, prompts, questioning. assertion
    - For speech: voice pitch statistics, intensity, time, and speed
3. Try to build a model to determine personalities
    - Only SVM can capture all Big-5 characteristics in a statistically significant manner for text
    - None accomplish this for audio; best would be to use Adaboost for extraversion and SVM for the others

Example workflow

- Following Green et al. (2019 TAR), as used in Crowley, Huang and Lu (2022)

1. Collect all conference call Q&A text from StreetEvents per executive
    - Exact match on executive name + company to Execucomp
        - Leverage genealogy table nickname data from Old Dominion
        - Fuzzy + manual match on the rest
    - 163,099 observations, ~36/executive
2. Apply an SVM model with linear kernel called *Personality Recognizer*
    - From Mairesse et al. (2007)
3. Average across calls per manager
    - Keep only executives with ≥3 call Q&As

Working with the code

> Note: It is likely you will run into missing files using the source code from Mairesse' website. There are repositories that contain the missing files online, e.g., on github.

```
# Make sure your preferences are properly set in PersonalityRecognizer.properties first!

# -d : process a directory
# -i : input data
# -m : model to use, 1=OLS, 2=M5 Model Tree, 3=M5 Reg Tree, 4=SVM
# -a : output to arff file (can be converted into a csv later using Python)
./PersonalityRecognizer -i ../../Proc/Sentic -d -m 4 -a ../../{your file}.arff
```

> Output is in the `arff` file format – there is an arff package for python that can handle this

Speeding it up a bit

- Since the script is single threaded, it is faster to split your data up into multiple folders and process multiple times. E.g., with 12 folders and 4 threads, consider something like the following:

> Saved in a script named parallel.sh:

```
./PersonalityRecognizer -i ../../Proc/SE_QAs-1 -d -m 4 -a ../../SE_QAs_Mairesse-1.arff
./PersonalityRecognizer -i ../../Proc/SE_QAs-2 -d -m 4 -a ../../SE_QAs_Mairesse-2.arff
./PersonalityRecognizer -i ../../Proc/SE_QAs-3 -d -m 4 -a ../../SE_QAs_Mairesse-3.arff
./PersonalityRecognizer -i ../../Proc/SE_QAs-4 -d -m 4 -a ../../SE_QAs_Mairesse-4.arff
./PersonalityRecognizer -i ../../Proc/SE_QAs-5 -d -m 4 -a ../../SE_QAs_Mairesse-5.arff
./PersonalityRecognizer -i ../../Proc/SE_QAs-6 -d -m 4 -a ../../SE_QAs_Mairesse-6.arff
./PersonalityRecognizer -i ../../Proc/SE_QAs-7 -d -m 4 -a ../../SE_QAs_Mairesse-7.arff
./PersonalityRecognizer -i ../../Proc/SE_QAs-8 -d -m 4 -a ../../SE_QAs_Mairesse-8.arff
./PersonalityRecognizer -i ../../Proc/SE_QAs-9 -d -m 4 -a ../../SE_QAs_Mairesse-9.arff
./PersonalityRecognizer -i ../../Proc/SE_QAs-10 -d -m 4 -a ../../SE_QAs_Mairesse-10.arff
./PersonalityRecognizer -i ../../Proc/SE_QAs-11 -d -m 4 -a ../../SE_QAs_Mairesse-11.arff
./PersonalityRecognizer -i ../../Proc/SE_QAs-12 -d -m 4 -a ../../SE_QAs_Mairesse-12.arff
```

> To execute in parallel (on *nix systems)

```
parallel --delay 2 --jobs 12 --no-notice < parallel.sh
```

Output

```
@relation features_/media/Data/Research/T013_TwitterMgmt_py3/Libraries/PersonalityRecognizer-master/../../

@attribute filename string
@attribute AOA numeric
@attribute BROWN-FREQ numeric
@attribute CONC numeric
@attribute FAM numeric
@attribute IMAG numeric
@attribute K-F-FREQ numeric
@attribute K-F-NCATS numeric
@attribute K-F-NSAMP numeric
@attribute MEANC numeric
@attribute MEANP numeric
@attribute NLET numeric
@attribute NPHON numeric
@attribute NSYL numeric
@attribute T-L-FREQ numeric
@attribute WC numeric
```

> The above is an `arff` file. It's essentially a `csv` file but where the header is a list of attributes instead of a comma separated line.

GPT models

What is a GPT model?

> A GPT model is a type of *Large Language Model* (**LLM**)

- Large: many parameters in the model (usually >1 billion)
- Language: the models are trained by seeing a large amount of written text
  - They infer everything from language
- Model: It's just an algorithm like everything else

> What does GPT mean? Generative Pre-trained Transformers

- Generative: It provides answers by generating an answer based on some latent space, as opposed to selecting answers it has previously seen
- Pre-trained: It's seen a lot of data already. That does not preclude it from seeing more.
- Transformer: A specific neural network architecture (which will talk about in Session 11)

What can ____-GPT do?

What can they do

- Classify data based on a small number of examples
  - “Few shot learning”
- Provide answers in flexible/trainable formats
- Encode and decode language
- Pattern matching
- Images as language

What can they not do

- Unless you train it yourself, it won’t have much domain-specific knowledge
- Beat single-purpose SOTA algorithms on most tasks
  - Validation is always needed to show the performance

How do different GPT models vary?

> Context length

- GPT-2: 2,048 tokens
- GPT-3: 4,096 tokens
- GPT-3.5: 4,096
- Chat-GPT: 4,096 tokens
- GPT-4: 8,096 or 32,384 tokens

Let’s build one!

> Go to: rmc.link/colab_gpt

- A simple one
  - 12,656 parameters
  - 2 possible tokens
  - A context length of 3
- As a comparison, GPT-2 has:
  - 1.5 billion parameters
  - 50,257 possible tokens
  - a context length of 2,048

How to interpret the network

The arrows show transition from a set of 3 characters to the next. In this process, the left-most character is dropped, the remaining two characters shift left, and a new character is added to the right side.

What to look for in the GPT Colab

1. We can see that it encodes simple patterns in the data well
2. We can see that answers are effectively probabilistic
3. We can see why hallucination occurs

> Additionally, things you can play around with:

1. How does adding more training iterations (epochs) change the output of the model?
2. How does the length of the input data (`seq` in the file) change the output of the model?

Conclusion

Wrap-up

> Many ways to approach these types of problems

- Much more approachable when there is an open source implementation

> Emotion recognition can work, but it can be noisy

- Better for some emotions than others.

> Personality recognition can work, but it is noisy when automated

- Need enough data that the noise isn’t a concern

> GPT is pretty general, with many use cases

Packages used for these slides

Python

- `arff`
- `numpy`
- `pandas`
- `pytorch`
- `theano`

R

- kableExtra
- knitr
- quarto
- reticulate
- revealjs

References

- Colnerič, Niko, and Janez Demšar. “Emotion recognition on twitter: Comparative study and training a unison model.” IEEE transactions on affective computing 11, no. 3 (2018): 433-446.
- De Choudhury, Munmun, Michael Gamon, Scott Counts, and Eric Horvitz. “Predicting depression via social media.” In Proceedings of the International AAAI Conference on Web and Social Media, vol. 7, no. 1. 2013.
- Eichstaedt, Johannes C., Hansen Andrew Schwartz, Margaret L. Kern, Gregory Park, Darwin R. Labarthe, Raina M. Merchant, Sneha Jha et al. “Psychological language on Twitter predicts county-level heart disease mortality.” Psychological science 26, no. 2 (2015): 159-169.
- Gentzkow, Matthew, Jesse M. Shapiro, and Matt Taddy. “Measuring group differences in high-dimensional choices: method and application to congressional speech.” Econometrica 87, no. 4 (2019): 1307-1340.
- Kim, Alex G., Maximilian Muhn, and Valeri V. Nikolaev. “Bloated Disclosures: Can ChatGPT Help Investors Process Information?.” Chicago Booth Research Paper 23-07 (2023).
- Majumder, Navonil, Soujanya Poria, Alexander Gelbukh, and Erik Cambria. “Deep learning-based document modeling for personality detection from text.” IEEE Intelligent Systems 32, no. 2 (2017): 74-79.
- Mehl, Matthias R., James W. Pennebaker, D. Michael Crow, James Dabbs, and John H. Price. “The Electronically Activated Recorder (EAR): A device for sampling naturalistic daily activities and conversations.” Behavior research methods, instruments, & computers 33, no. 4 (2001): 517-523.
- Pennebaker, James W., and Laura A. King. “Linguistic styles: language use as an individual difference.” Journal of personality and social psychology 77, no. 6 (1999): 1296.