Text Analysis

COGS 108 Fall 2025
Jason Chen
Week 8

xic007@ucsd.edu

OH: Tue 3-5 pm

Due dates

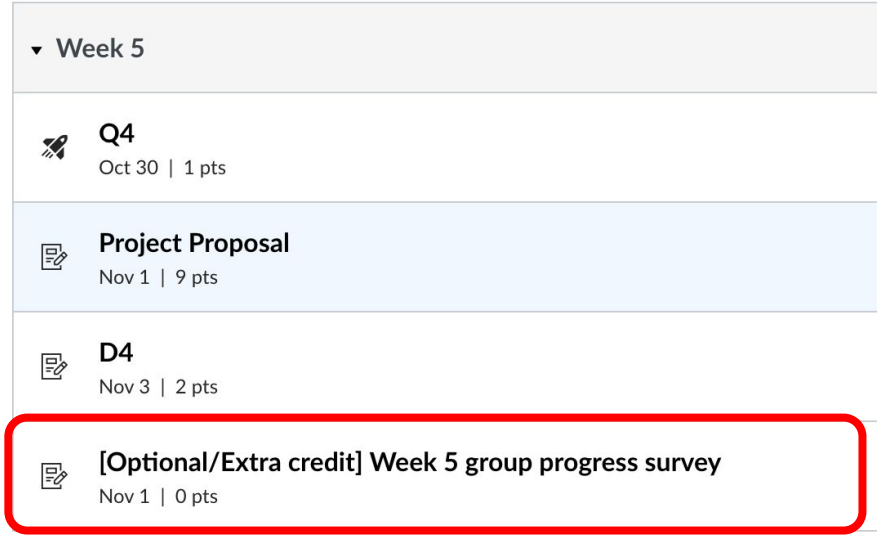
- A3: TODAY (11/19)
- D7: Friday (11/21)

Announcement

- Your Data Checkpoints should be graded
  - There might be some additional feedback for the proposal stuff, we've updated your grade in your proposal issue if we see that you fixed it
  - We gave feedback on the Data part of the checkpoint
  - If there are still things you need to fix from the proposal, we put those comments into the checkpoint issue

Project - Weekly Check-ins

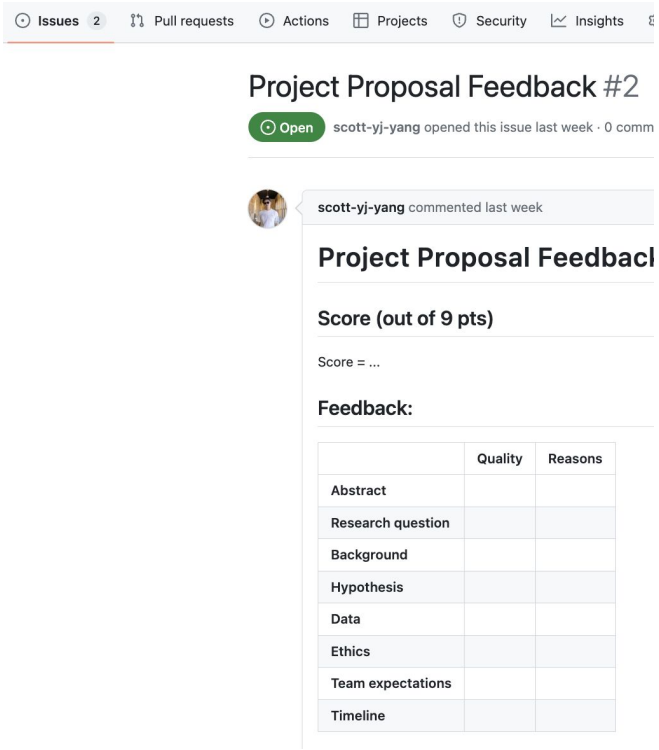
- Every week you can fill out the weekly group progress survey
- If you fill them all out you get <u>Extra Credit!!!</u>
- It's a chance for you to let us know how your project is going
  - Questions?
  - Concerns about groupmates?
  - Challenges you're facing



Checkpoint Feedback

- Feedback will be released on Issues!
- There is an issue in your repo with your assigned TA/IA ⇒ Reach out to them with any questions.

- Start thinking about your EDA Checkpoint!!!



Project Updates

- Remember to check Edstem for all updates on projects

- Note: For very large datasets that cannot be uploaded to GitHub, you may use either Git LFS, or just upload to google drive or another dropbox and paste the link.
  - Your TA MUST be able to see the raw data in some format…

D7: TEXT ANALYSIS

TF-IDF

- What is TF-IDF?
  - term frequency-inverse document frequency.
  - a statistical measure that evaluates how relevant a word is to a document in a collection of documents.

TF-IDF

- How does it work?
  - Multiplies two terms:
  - how many times a word appears in a document
  - the inverse document frequency of the word across a set of documents.
  - So, if the word is very common and appears in many documents, this number will approach 0. Otherwise, it will approach 1.

TF-IDF

- How does it work?

$$\text{TF-IDF}(t, d, D) = \log(1 + \text{frequency of } t \text{ in } d) \quad \log\left(\frac{\text{Total \# of documents}}{\text{\# of documents that contain the word } t}\right)$$

Term frequency

Inverse document frequency

TF-IDF

- Implementing in python

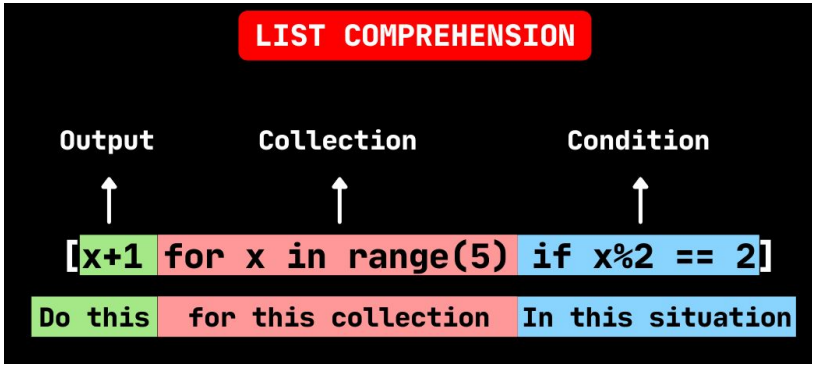
```
tfidf = TfidfVectorizer(sublinear_tf=True,
                        analyzer='word',
                        max_features=2000,
                        tokenizer=word_tokenize,
                        stop_words=stopwords.words("english"))
```

Sublinear_tf: Apply sublinear tf scaling, i.e. replace tf with 1 + log(tf).

max_features: If not None, build a vocabulary that only consider the top max_features ordered by term frequency across the corpus. Otherwise, all features are used.

List Comprehension

- Use List Comprehension in python to get list of items that are a manipulation of another collection
- Ex:
  - years = [____ for speech in inaugural.fileids()]



Plotting

- In general, a linear distribution can be plotted using plt.plot()
  - plt.plot(x = ?, y = ?, label = ?)
  - plt.xlabel(?)
  - plt.ylabel(?)
  - plt.legend(?)

- plt.plot(inaug_tfidf.index.to_numpy(), inaug_tfidf['british'].to_numpy())

K-Nearest Neighbor Classifier

Create 2 column selectors using selector to collect only numerical features and categorical features.

```
numerical_columns_selector =
selector(dtype_exclude=...)

categorical_columns_selector =
selector(dtype_include=...)
```

K-Nearest Neighbor Classifier

Use these selectors to then separate X into numerical columns and categorical columns.

```
numerical_columns = numerical_columns_selector(X)
categorical_columns = categorical_columns_selector(X)
```

K-Nearest Neighbor Classifier

Use preprocessor and ColumnTransformer to chain the one hot encoder and the standard scaler

```
preprocessor = ColumnTransformer([
    ('onehot', OneHotEncoder(), categorical_columns),
    …])
```

Section Materials

Section materials can be accessed at:
https://github.com/JasonC1217/COGS108_FA25_B07-B08

THANKS!

Questions on EdStem or office hours

Office hours: Tue, 3-5 PM