COMPUTATIONAL LINGUISTICS WORKSHOP

Data Science Club - Isaiah Stapleton

TABLE OF CONTENTS

01 Introduction

02 Techniques

03 Source Code Summarization

04 Code Demonstration

05 Conclusion

01

INTRODUCTION

WHAT IS COMPUTATIONAL LINGUISTICS?

- Interdisciplinary field
  - Linguistics & Computer Science
- Goal: Use computational methods to study human language
- Natural Language Processing (NLP)
- Goal: The implementation of algorithms for processing and analyzing human language

CHALLENGES WITH PROCESSING NATURAL LANGUAGE

- **Ambiguity:** Natural language inherently ambiguous
- **Variation**: Variation in language use
- **Pragmatics**: Context in which language is used
- **Human bias**

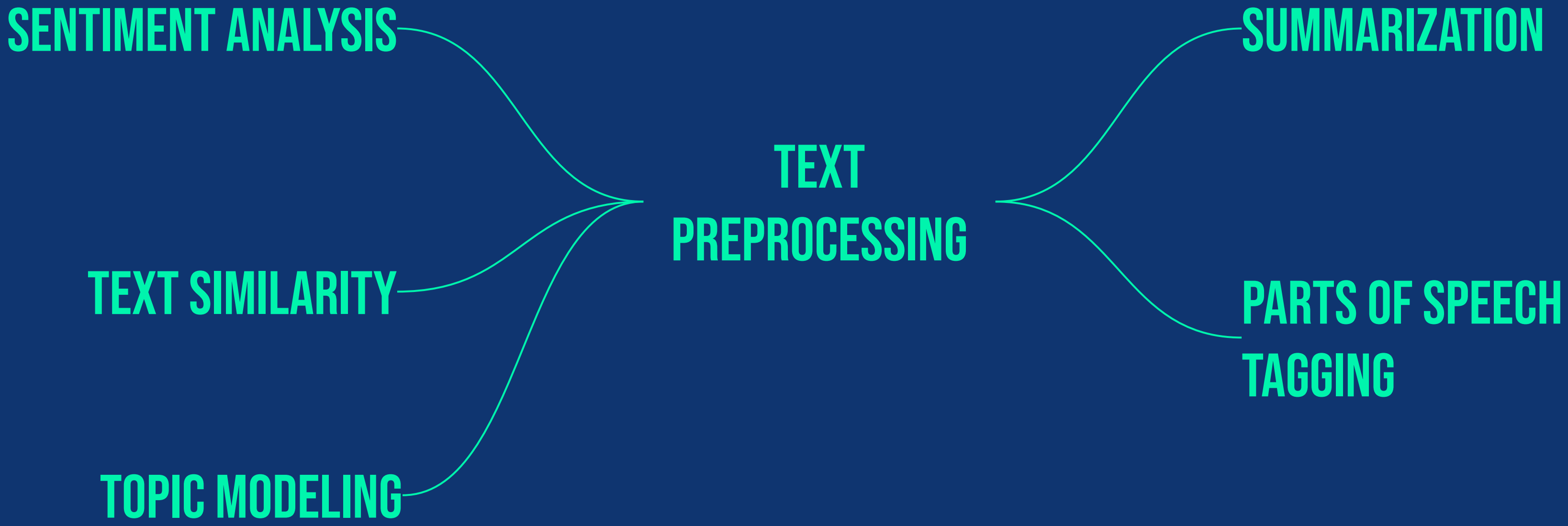
CAREERS INVOLVING COMPUTATIONAL LINGUISTICS / NLP

- NLP / Machine Learning Engineer
- Data Scientist
- Linguistics Researcher
- Software Developer

02

NLP TECHNIQUES

NLP TECHNIQUES



TEXT PREPROCESSING

SENTIMENT ANALYSIS

TEXT SIMILARITY

TOPIC MODELING

SUMMARIZATION

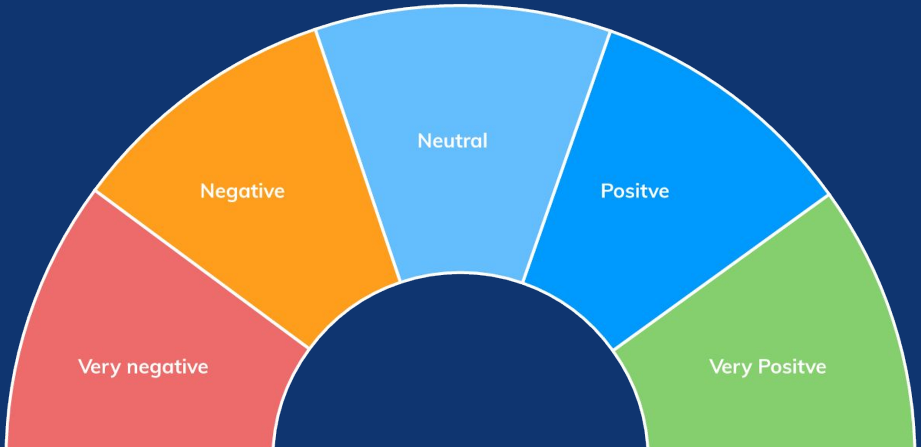
PARTS OF SPEECH TAGGING

TEXT PREPROCESSING

- Remove
  - Punctuation
  - Stop words
    - Commonly used words that do not carry much meaning
- Lower case
- Tokenization
  - Splitting text into smaller units, “tokens”.
- Stemming
  - Removing suffixes to obtain root of the word
- Lemmatization
  - Reducing a word to its base or dictionary form, “lemma”

SENTIMENT ANALYSIS

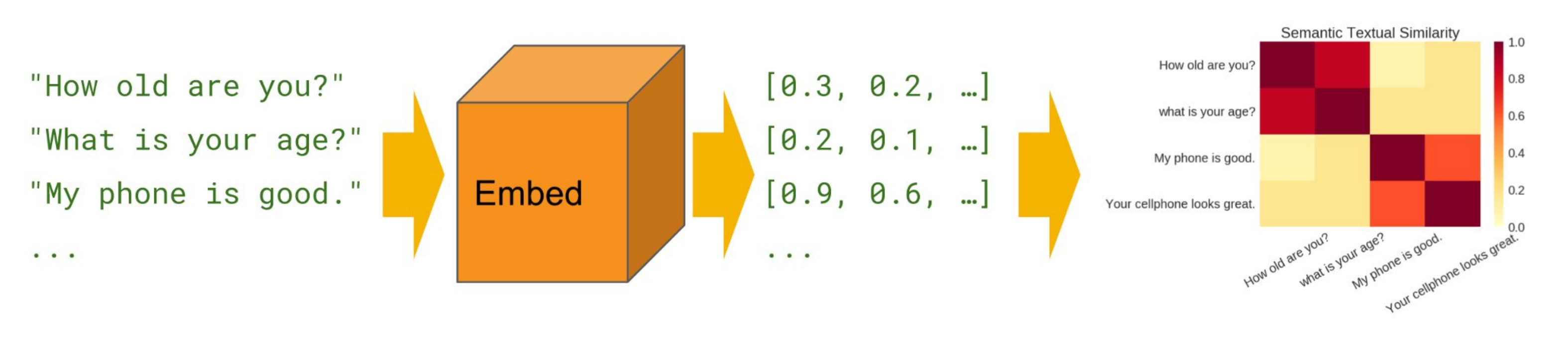
- Classify the emotional tone / sentiment of a piece of text
- You own a restaurant and you want to improve customer satisfaction by identifying areas where your guests are most satisfied and dissatisfied. You have a lot of customer feedback and reviews on your website, but it's difficult to read and analyze them all manually
- Specific keywords / phrases

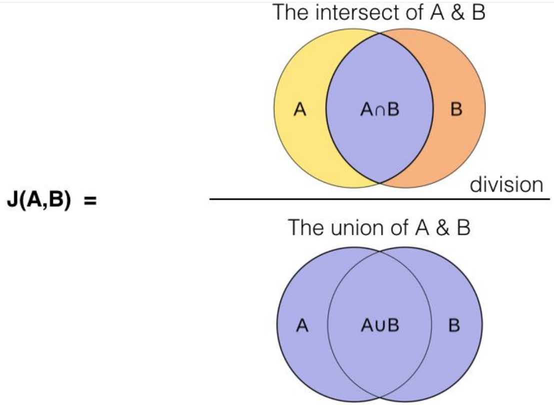


TEXT SIMILARITY

- Quantify the similarity between two pieces of text
- Lexical similarity & semantic similarity
- Jaccard similarity, cosine similarity, kmeans clustering
- Convert words to vectors

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2}\sqrt{\sum\limits_{i=1}^{n} B_i^2}},$$





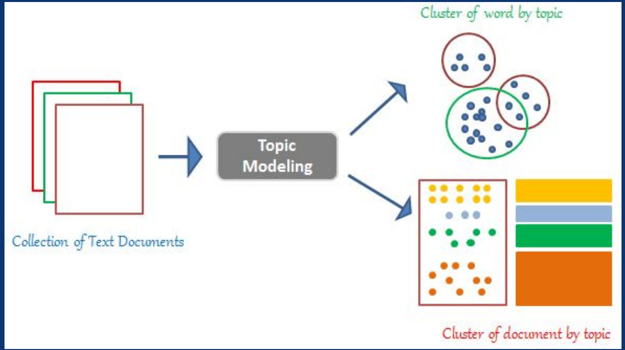
$$J(doc_1, doc_2) = \frac{\{\text{'data', 'is', 'the', 'new', 'oil', 'of', 'digital', 'economy'}\} \bigcap \{\text{'data', 'is', 'a', 'new', 'oil'}\}}{\{\text{'data', 'is', 'the', 'new', 'oil', 'of', 'digital', 'economy'}\} \bigcup \{\text{'data', 'is', 'a', 'new', 'oil'}\}}$$

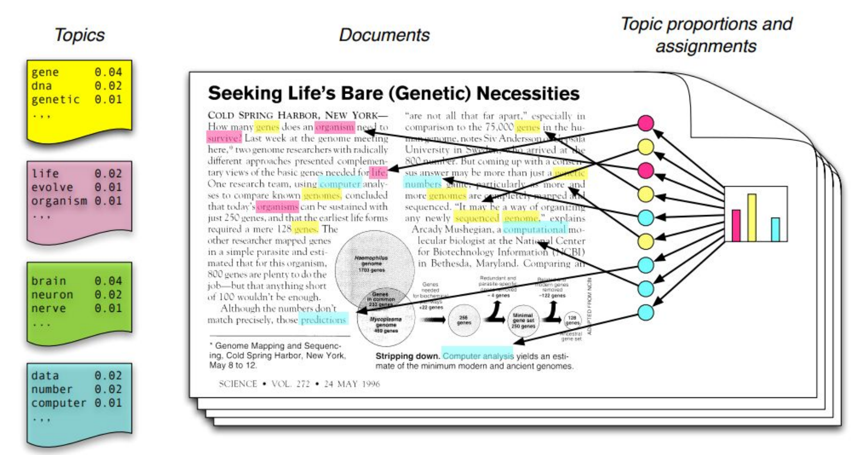
$$= \frac{\{\text{'data', 'is', 'new', 'oil'}\}}{\{\text{'data', 'a', 'of', 'is', 'economy', 'the', 'new', 'digital', 'oil'}\}}$$

$$= \frac{4}{9} = 0.444$$

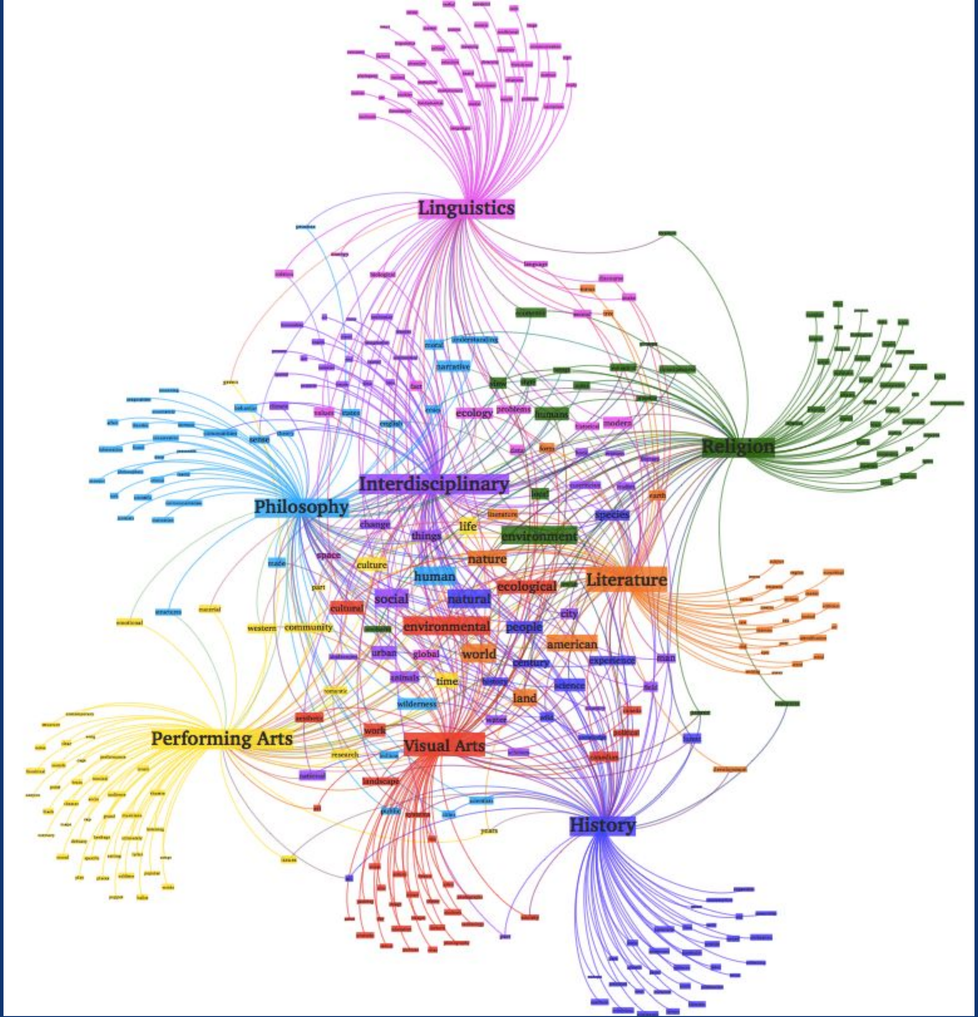
TOPIC MODELING

- Discovering the underlying topics present in a collection of text documents



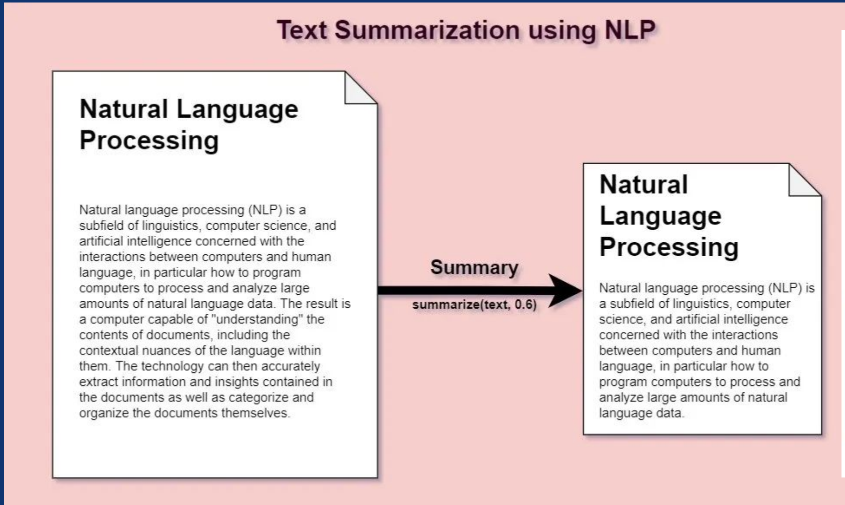


- Each **topic** is a distribution over words
- Each **document** is a mixture of corpus-wide topics
- Each **word** is drawn from one of those topics


Linguistics
Religion
Interdisciplinary
Philosophy
Literature
Performing Arts
Visual Arts
History
understanding
narrative
ecology
problems
humans
modern
life
change
things
environment
species
space
culture
nature
human
ecological
social
natural
cultural
city
western
community
environmental
people
american
urban
global
world
century
experience
man
animals
time
history
science
land
wilderness
water
work
research
landscape
national
years
literature
sense
values
english
earth
issues
emotional
material

SUMMARIZATION

- Generating a short summary of a longer piece of text while retaining the most important information from the original (longer) text



Text Summarization using NLP

Natural Language Processing

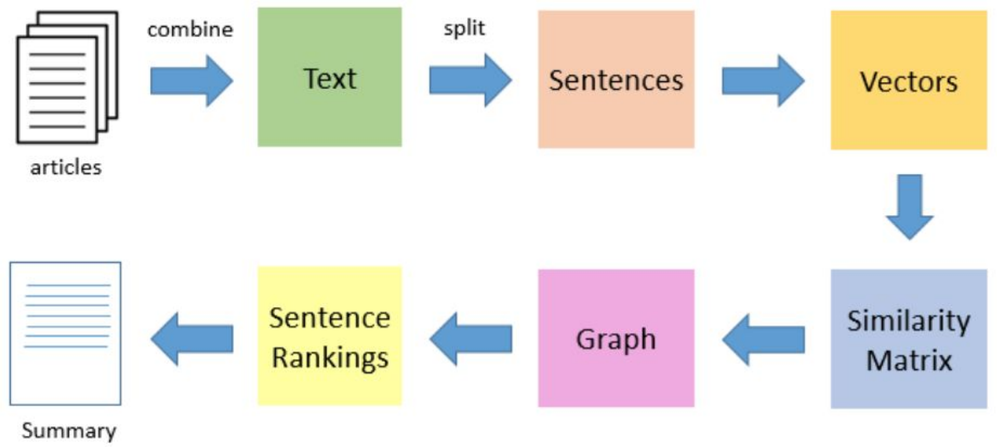
Natural language processing (NLP) is a subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language, in particular how to program computers to process and analyze large amounts of natural language data. The result is a computer capable of "understanding" the contents of documents, including the contextual nuances of the language within them. The technology can then accurately extract information and insights contained in the documents as well as categorize and organize the documents themselves.

Summary

summarize(text, 0.6)

Natural Language Processing

Natural language processing (NLP) is a subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language, in particular how to program computers to process and analyze large amounts of natural language data.



articles

combine

Text

split

Sentences

Vectors

Similarity Matrix

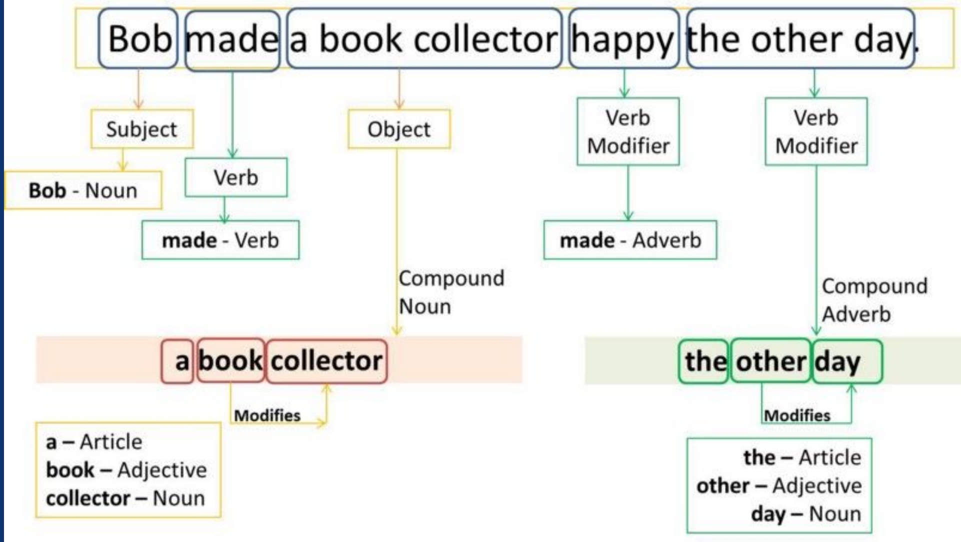
Graph

Sentence Rankings

Summary

PARTS OF SPEECH TAGGING

- Assigning parts of speech to each word in a given text
  - Noun, verb, adjective, adverb, pronoun, etc.
- Useful for supervised machine learning algorithms
  - Using parts of speech as a feature in a ML algorithm
- Useful for other NLP tasks
  - Sentiment analysis, text similarity, topic modeling, etc.



Bob made a book collector happy the other day.

Subject

Bob - Noun

Verb

made - Verb

Object

Compound Noun

a book collector

Modifies

a – Article
book – Adjective
collector – Noun

Verb Modifier

made - Adverb

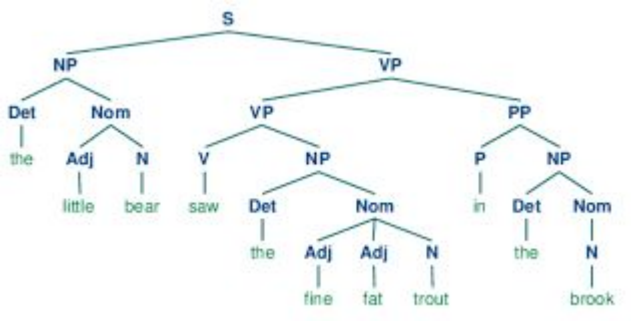
Verb Modifier

Compound Adverb

the other day

Modifies

the – Article
other – Adjective
day – Noun



03

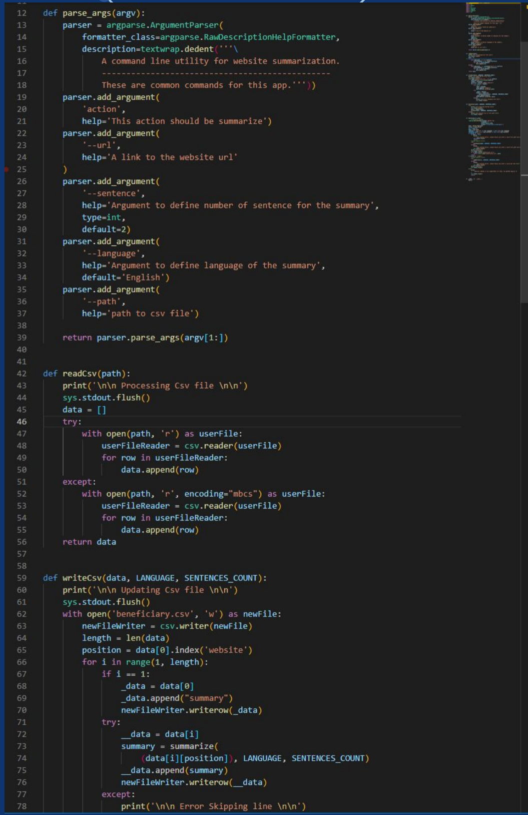
SOURCE CODE SUMMARIZATION

SOURCE CODE SUMMARIZATION

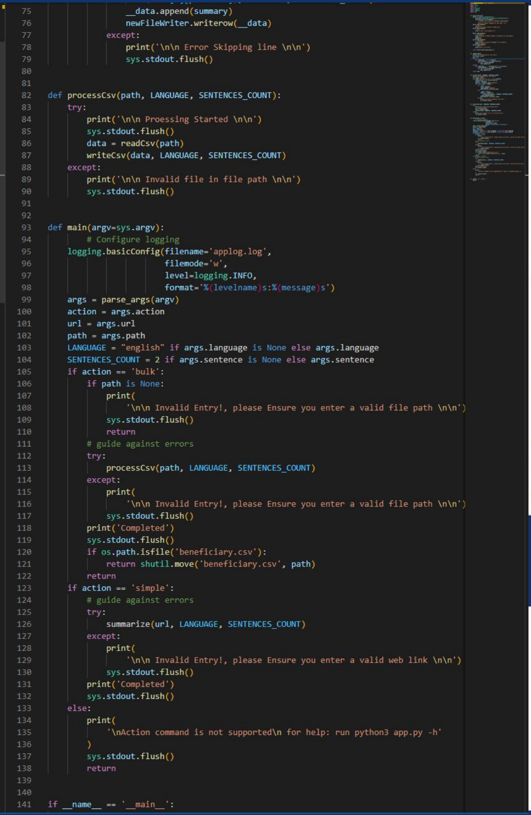
- Generate descriptions for functions, class methods, packages, etc.
- Why is it important?
  - Quicker understanding of source code
  - Missing / incomplete documentation
- State of the art technique
  - Using deep learning methods/ neural networks
  - Transformer based approach

MY (NAIVE) SOLUTION

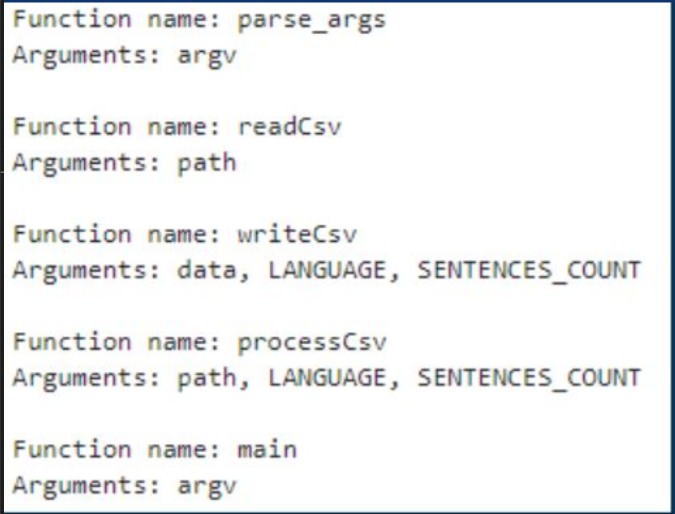
- Classification
- Take a piece of source code and extract the function names and class method names, as well as the arguments associated with each
- Create a dataset containing function names and their use cases
  - Prints something
  - Performs a calculation
  - Operates on a file
  - Sorts something
- Use the dataset to train a Naïve Bayes Classifier in order to predict function or class methods use
  - 80% Training 20% Testing



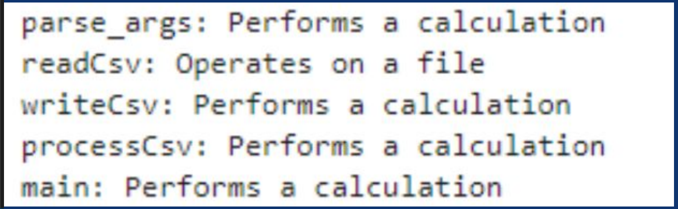
```
def parse_args(argv):
    parser = argparse.ArgumentParser(
        formatter_class=argparse.RawDescriptionHelpFormatter,
        description=textwrap.dedent('''\
            A command line utility for website summarization.
            ---------------------------------------------
            These are common commands for this app.'''))
    parser.add_argument(
        'action',
        help='This action should be summarize')
    parser.add_argument(
        '--url',
        help='A link to the website url'
    )
    parser.add_argument(
        '--sentence',
        help='Argument to define number of sentence for the summary',
        type=int,
        default=2)
    parser.add_argument(
        '--language',
        help='Argument to define language of the summary',
        default='English')
    parser.add_argument(
        '--path',
        help='path to csv file')

    return parser.parse_args(argv[1:])


def readCsv(path):
    print('\n\n Processing Csv file \n\n')
    sys.stdout.flush()
    data = []
    try:
        with open(path, 'r') as userFile:
            userFileReader = csv.reader(userFile)
            for row in userFileReader:
                data.append(row)
    except:
        with open(path, 'r', encoding="mbcs") as userFile:
            userFileReader = csv.reader(userFile)
            for row in userFileReader:
                data.append(row)
    return data


def writeCsv(data, LANGUAGE, SENTENCES_COUNT):
    print('\n\n Updating Csv file \n\n')
    sys.stdout.flush()
    with open('beneficiary.csv', 'w') as newFile:
        newFileWriter = csv.writer(newFile)
        length = len(data)
        position = data[0].index('website')
        for i in range(1, length):
            if i == 1:
                _data = data[0]
                _data.append("summary")
                newFileWriter.writerow(_data)
            try:
                __data = data[i]
                summary = summarize(
                    (data[i][position]), LANGUAGE, SENTENCES_COUNT)
                __data.append(summary)
                newFileWriter.writerow(__data)
            except:
                print('\n\n Error Skipping line \n\n')
```



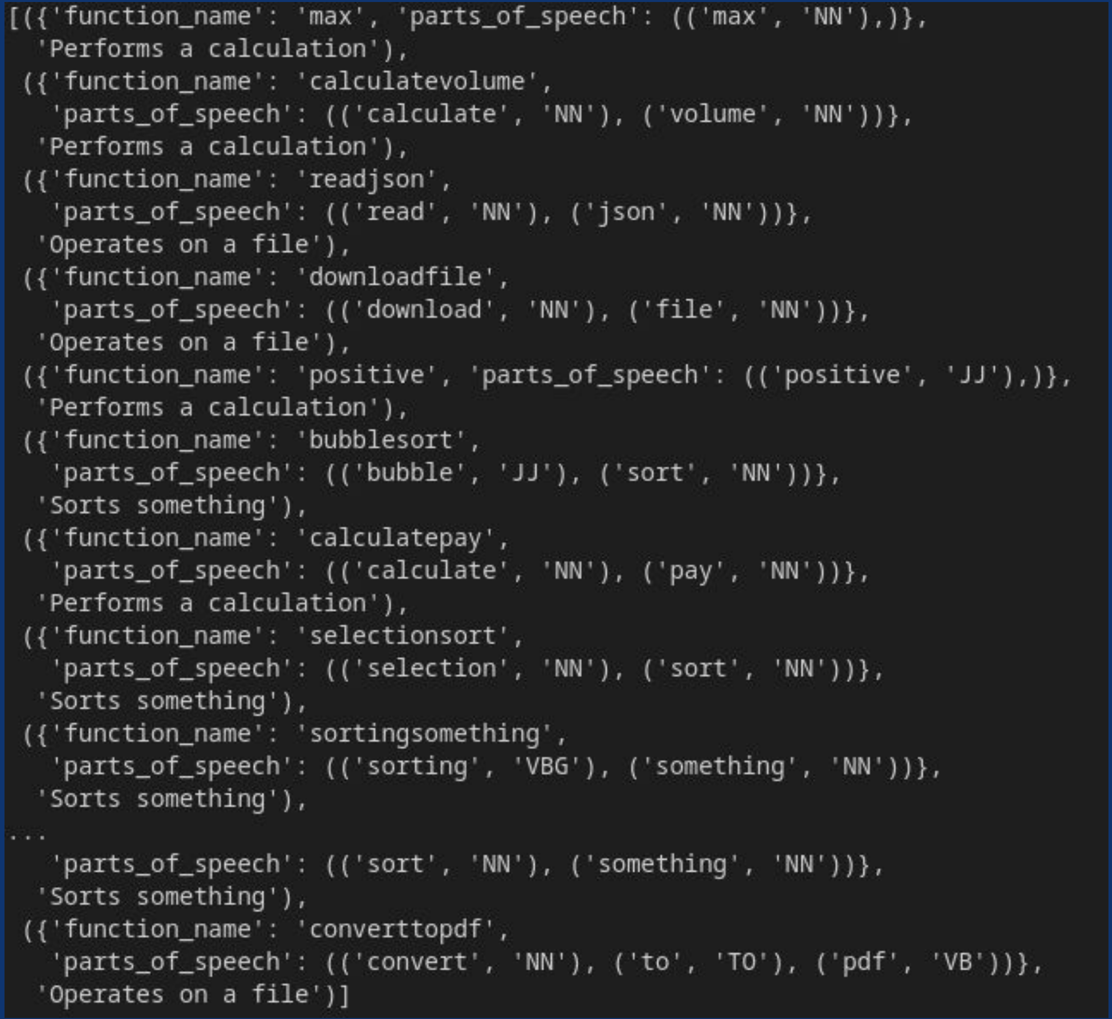
```
                __data.append(summary)
                newFileWriter.writerow(__data)
            except:
                print('\n\n Error Skipping line \n\n')
                sys.stdout.flush()


def processCsv(path, LANGUAGE, SENTENCES_COUNT):
    try:
        print('\n\n Proessing Started \n\n')
        sys.stdout.flush()
        data = readCsv(path)
        writeCsv(data, LANGUAGE, SENTENCES_COUNT)
    except:
        print('\n\n Invalid file in file path \n\n')
        sys.stdout.flush()


def main(argv=sys.argv):
        # Configure logging
    logging.basicConfig(filename='applog.log',
                        filemode='w',
                        level=logging.INFO,
                        format='%(levelname)s:%(message)s')
    args = parse_args(argv)
    action = args.action
    url = args.url
    path = args.path
    LANGUAGE = "english" if args.language is None else args.language
    SENTENCES_COUNT = 2 if args.sentence is None else args.sentence
    if action == 'bulk':
        if path is None:
            print(
                '\n\n Invalid Entry!, please Ensure you enter a valid file path \n\n')
            sys.stdout.flush()
            return
        # guide against errors
        try:
            processCsv(path, LANGUAGE, SENTENCES_COUNT)
        except:
            print(
                '\n\n Invalid Entry!, please Ensure you enter a valid file path \n\n')
            sys.stdout.flush()
        print('Completed')
        sys.stdout.flush()
        if os.path.isfile('beneficiary.csv'):
            return shutil.move('beneficiary.csv', path)
        return
    if action == 'simple':
        # guide against errors
        try:
            summarize(url, LANGUAGE, SENTENCES_COUNT)
        except:
            print(
                '\n\n Invalid Entry!, please Ensure you enter a valid web link \n\n')
            sys.stdout.flush()
        print('Completed')
        sys.stdout.flush()
    else:
        print(
            '\nAction command is not supported\n for help: run python3 app.py -h'
        )
        sys.stdout.flush()
        return


if __name__ == '__main__':
```



```
Function name: parse_args
Arguments: argv

Function name: readCsv
Arguments: path

Function name: writeCsv
Arguments: data, LANGUAGE, SENTENCES_COUNT

Function name: processCsv
Arguments: path, LANGUAGE, SENTENCES_COUNT

Function name: main
Arguments: argv
```



```
parse_args: Performs a calculation
readCsv: Operates on a file
writeCsv: Performs a calculation
processCsv: Performs a calculation
main: Performs a calculation
```



```
[({'function_name': 'max', 'parts_of_speech': (('max', 'NN'),)},
  'Performs a calculation'),
 ({'function_name': 'calculatevolume',
   'parts_of_speech': (('calculate', 'NN'), ('volume', 'NN'))},
  'Performs a calculation'),
 ({'function_name': 'readjson',
   'parts_of_speech': (('read', 'NN'), ('json', 'NN'))},
  'Operates on a file'),
 ({'function_name': 'downloadfile',
   'parts_of_speech': (('download', 'NN'), ('file', 'NN'))},
  'Operates on a file'),
 ({'function_name': 'positive', 'parts_of_speech': (('positive', 'JJ'),)},
  'Performs a calculation'),
 ({'function_name': 'bubblesort',
   'parts_of_speech': (('bubble', 'JJ'), ('sort', 'NN'))},
  'Sorts something'),
 ({'function_name': 'calculatepay',
   'parts_of_speech': (('calculate', 'NN'), ('pay', 'NN'))},
  'Performs a calculation'),
 ({'function_name': 'selectionsort',
   'parts_of_speech': (('selection', 'NN'), ('sort', 'NN'))},
  'Sorts something'),
 ({'function_name': 'sortingsomething',
   'parts_of_speech': (('sorting', 'VBG'), ('something', 'NN'))},
  'Sorts something'),
...
   'parts_of_speech': (('sort', 'NN'), ('something', 'NN'))},
  'Sorts something'),
 ({'function_name': 'converttopdf',
   'parts_of_speech': (('convert', 'NN'), ('to', 'TO'), ('pdf', 'VB'))},
  'Operates on a file')]
```

FINDINGS

- Not an effective approach
  - 30-40% accuracy
- Selecting better features
  - Parts of speech as a feature did not have much effect on model accuracy due to small dataset containing mostly nouns
- Further work
  - Translation into other languages

04

CODE DEMONSTRATION

05

CONCLUSION

THANK YOU!

Questions?

EMAIL

stapletonin@g.cofc.edu

LINKEDIN

Isaiah Stapleton