BIS 420 PROGRAMMING FOR DATA SCIENCE

PRAJAKTA POHARE
CHAPTER 18 EXERCISE 18.5
ILLINOIS STATE UNIVERSITY

Download my code from Section 13.8 (http: // thinkpython. com/ code/markov. py), and follow the steps described above to encapsulate the global variables as attributes of a new class called Markov. Solution: http: // thinkpython. com/ code/ Markov. py (note the capital M).

```
import random


class Markov:
    def __init__(self):
        self.markov = {}


    def process_file(self, filename, skip_header):
        with open(filename, 'r') as f:
            if skip_header:
                self.skip_gutenberg_header(f)


            for line in f:
                self.process_line(line)


    def skip_gutenberg_header(self, fp):
        for line in fp:
            if line.startswith('*** START OF'):
                break


    def process_line(self, line):
```

```
    for word in line.replace('-', ' ').split():
        self.process_word(word.lower())

def process_word(self, word):
    if hasattr(self, 'prefix'):
        prefix = self.prefix
    else:
        prefix = ()

    if len(prefix) < 2:
        self.prefix = prefix + (word,)
        return

    if prefix not in self.markov:
        self.markov[prefix] = []
    self.markov[prefix].append(word)

    self.prefix = (prefix[1], word)

def generate_text(self, n=100):
    start = random.choice(list(self.markov.keys()))
    result = list(start)
    for _ in range(n):
        suffixes = self.markov.get(start)
        if not suffixes:
            break
        word = random.choice(suffixes)
```

```
            result.append(word)
            start = (start[1], word)
        return ' '.join(result)

# Example usage
if __name__ == '__main__':
    m = Markov()
    m.process_file('/Users/prajaktapohare/Library/CloudStorage/OneDrive-ILStateUniversity/BIS420/Week 18/emma.txt', skip_header=True)
    print(m.generate_text(100))
```

```python
import random

class Markov:
    def __init__(self):
        self.markov = {}

    def process_file(self, filename, skip_header):
        with open(filename, 'r') as f:
            if skip_header:
                self.skip_gutenberg_header(f)

            for line in f:
                self.process_line(line)

    def skip_gutenberg_header(self, fp):
        for line in fp:
            if line.startswith('*** START OF'):
                break

    def process_line(self, line):
        for word in line.replace('-', ' ').split():
            self.process_word(word.lower())

    def process_word(self, word):
        if hasattr(self, 'prefix'):
            prefix = self.prefix
        else:
            prefix = ()
```

```
        if len(prefix) < 2:
            self.prefix = prefix + (word,)
            return

        if prefix not in self.markov:
            self.markov[prefix] = []
        self.markov[prefix].append(word)

        self.prefix = (prefix[1], word)

    def generate_text(self, n=100):
        start = random.choice(list(self.markov.keys()))
        result = list(start)
        for _ in range(n):
            suffixes = self.markov.get(start)
            if not suffixes:
                break
            word = random.choice(suffixes)
            result.append(word)
            start = (start[1], word)
        return ' '.join(result)

# Example usage
if __name__ == '__main__':
    m = Markov()
    m.process_file('/Users/prajaktapohare/Library/CloudStorage/OneDrive-
ILStateUniversity/BIS420/Week 18/emma.txt', skip_header=True)
    print(m.generate_text(100))
```