Python: Data Analysis and Data Visualization

David Pinezich

Learning Targets

- Import data and pre process
  - formats, parse data, use suited data structures
- Aggregate
  - Perform basic analysis
  - descriptive statistics
  - text analysis
- Visualization types
  - Tables, x-y Plot, Normal distribution, Pie chart, Spider diagram, Word cluster, Histogram, 3D Plot

Sources

- All updated source files are found here:
  - https://github.com/dpinezich/apyd_19/archive/master.zip

- Additionally you need to unzip the archive reviews.zip in /exercises/amazon_exercise/review_files/

Amazon dataset

- 34.5 Mio reviews
- 18+ years of data
- By categories (Jewellery, Music, Watches, etc.)
- Available here: https://snap.stanford.edu/data/web-Amazon-links.html

Amazon dataset

Example Entry

```
product/productId: B000PC6A84
product/title: Listener Supported: The Culture and History of Public Radio
product/price: unknown
review/userId: A3UZO66MKBBLL8
review/profileName: Sarah
review/helpfulness: 3/4
review/score: 4.0
review/time: 1353283200
review/summary: Good book
review/text: Good Book. Easy reading. I learned thinkg I never knew about the
   public broadcasting industry that I needed to learn for a paper I was
   writing. interesting.
```

Amazon Exercise – Part 1

- Complete the function read_review_file in the file_reader.py file
  - Try to solve this without considering the max_reviews parameter

Amazon Exercise – Part 1

Clues

- An entry in the text file has a similar structure as a python dictionary
- You need to parse every line of the file
- In each row, a colon is separating the key from the value
  - → How to find the index of the colon to store the key and the value?
  - → How to handle lines with no colon?

Amazon Exercise – Part 1

Write

- All the numerical values should be converted into a number before they are put into the python dictionary
- This way, numerical calculations are allowed

```
try:
    entry_content = float(entry_content)
except: ValueError:
    pass
```

Amazon Exercise – Part 2

- Calculate, the average, the mode (most frequent value) and the std deviation of the score and the review length (number of words) of each product
- Implement your solution in the print_review_statistics function within the stats.py file

Amazon Exercise – Part 2

Clues

- Work with the statistics package
- Hava a look at the manual:
  - https://docs.python.org/3/library/statistics.html

Amazon Exercise – Part 3

- Complete the create_score_barchart function
  - The function creates a bar chart, which shows the frequency of the scores

- Complete the create_review_length_boxplots function
  - The function creates a chart with 5 boxplots which shows the length distribution of the reviews (number of words)

Amazon Exercise – Part 3

Clues

- Go through the pygal documentation:
  - http://www.pygal.org/en/latest/documentation/index.html
  - Documentation of bar charts:
    - http://www.pygal.org/en/latest/documentation/types/bar.html
  - Documentation of box charts:
    - http://www.pygal.org/en/latest/documentation/types/box.html
  - Use the render_to_file function, to save the chart to file:

```
line_chart.render_to_file("test.svg")
```

Text processing

Natural Language Toolkit (NLTK)

- Very extensive library for text processing
  - Tokenization
  - Part-of-Speech Tagging (PoS Tagging)
  - Lemmatizing

- http://www.nltk.org

NLTK

Tokenization

- Split sentence into words
- Punctuation is also treated as tokens!

```
import nltk

sentence = "At eight o’clock on Thursday morning Arthur didn’t feel very good."

tokens = nltk.word_tokenize(sentence)

# ['At', 'eight', "o'clock", 'on', 'Thursday', 'morning', 'Arthur', 'did', "n't", 'feel', 'very', 'good']
```

NLTK

Part-of-Speech (POS) Tagging

- Categorizing the word types (Verb, Adverb, Nomen, etc)

```
import nltk

sentence = "At eight o’clock on Thursday morning Arthur didn’t feel very good."

tokens = nltk.word_tokenize(sentence)
pos_tags = ntlk.pos_tag(tokens, tagset="universal")

# [('At', 'ADP'), ('eight', 'NUM'), ('o', 'NOUN'), ('’', 'NOUN'), ('clock', 'NOUN'), ('on', 'ADP'), ('Thursday', 'NOUN'), ('morning', 'NOUN'), ('Arthur', 'NOUN'), ('didn', 'NOUN'), ('’', 'NOUN'), ('t', 'NOUN'), ('feel', 'VERB'), ('very', 'ADV'), ('good', 'ADJ'), ('.', '.')]
```

NLTK

Lemmatizing

- Getting the verbs basic form

```
import nltk

wordnet_lemmatizer = nltk.WordNetLemmatizer()

wordnet_lemmatizer.lemmatize("finds", "v") # find

wordnet_lemmatizer.lemmatize(”found”, "v") # find

wordnet_lemmatizer.lemmatize(”finding”, "v") # find

wordnet_lemmatizer.lemmatize("finding", "n") # finding
```

Amazon Exercise – Part 4

- Complete the get_adjectives function
  - The function adds the adjectives to a list and returns the list
  - The list can contain the same word multiple times
  - Words of the file custom_stopwords.txt should be excluded
  - Use only the reviews where a score has been given
  - Use the nltk.pos_tag function, to determine the word category

Amazon Exercise – Part 5

- Complete the create_wordcloud function
  - The function creates a “Wordcloud” and saves it under the given file name
  - Have a look at an example of how to build a “Wordcloud”
    - https://github.com/amueller/word_cloud

Amazon Exercise – Part 6

- Complete the get_lemmas function
  - The function turns each adjective, noun or verb into a lemma and adds it to a list which will be returned
  - Use the lemmatize function of the WordNetLemmatizer, to get the lemma of the word
  - Other conditions are the same as in part 4