Tutorial: Data visualization and analysis

COMP3222/COMP6246 Machine Learning Technologies

1 Introduction

Read course text: Chapter 2

Follow each section below at your own pace, you can have a look at the book, tutorial links or the example answers if you find something confusing.

2 Manual inspection of data with a text editor [format, volume, quality]

The quickest and easiest way to visualize data is with a text editor or Excel. It is surprising how far you can go with manual inspection, especially when looking at data format, volume and quality.

Exercise 2.1: Load 20_newsgroups_corpus.json into a text editor. Use a JSON formatter to make it easier to read (e.g. https://jsonformatter.org/) if needed. What is the format? What is the volume of posts? Does the text need pre-processing (e.g. cleaning up)?

Solution 2.1

Format

JSON formatted UTF-8 encoded, newline per JSON entry in file

"post:file" -> file ID

"post:group" -> group name

"header:XXX": -> post header metadata field

"body:quote": -> post quoted text (if a reply)

"body:text": -> post text,

"NER:XXX": -> named entity recognition (NER) type

Volume

19,997 posts, newline delimited

Post text quality

This text will probably need some pre-processing.

Details:

There are newlines everywhere

Fragments of quoted text (in wrong place)

>>> indent spam

emoji's and symbolic characters mix of ASCII expansion and Unicode

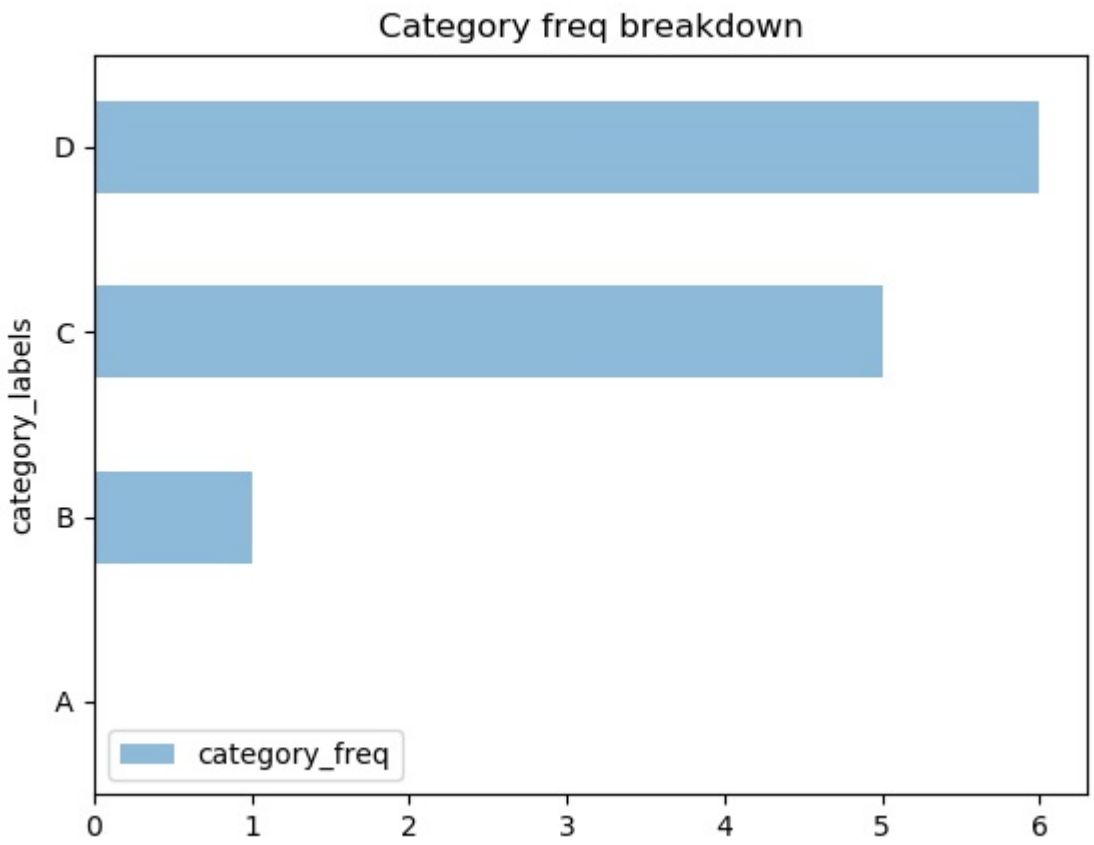
3 Data segmentation [bias, skew]

Pandas data frames are a useful way to handle data for the purposes of visualization and analysis. Segmenting the data in different ways and visualizing histograms of frequency counts can reveal clusters of data which are over or underrepresented compared to what should be expected. This can indicate bias (e.g. bias from over representing a certain demographic in a training set) or skew (e.g. skewed normal distribution suggesting a non-random sampling of training examples).

Unfortunately, real-world datasets will not usually come to you in a nice way to load directly into panads. Therefore, you will often have to load the raw data and manipulate it, so it is in a format suitable for being loaded as a pandas data frame.

We can load JSON data as a Python dict using the json.loads() function. Then you can create a pandas data frame using the pd.DataFrame constructor. Below is an example of loading frequency scores for categorical labels.

```
import pandas as pd, geopandas as gpd

import matplotlib.pyplot as plt

dict_dataset = { 'category_freq' : [0,1,5,6], 'category_labels' : [' A', 'B', 'C', 'D'] }

df = pd.DataFrame( data = dict_dataset )

df.plot.barh( x='category_labels', y='category_freq', alpha=0.5, title = 'Category freq breakdown' )

plt.show()
```



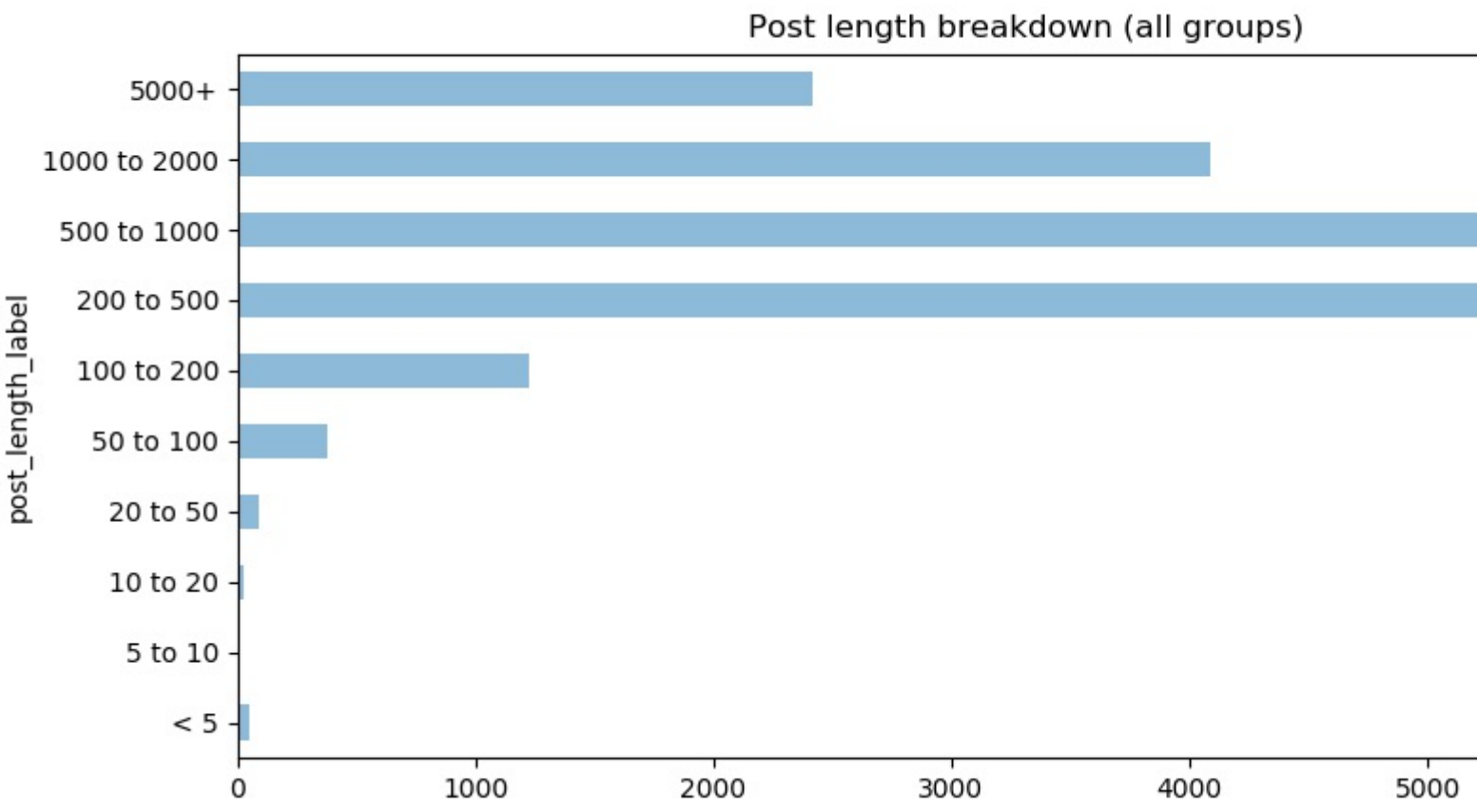
Read more: https://pandas.pydata.org/pandas-docs/version/0.23/dsintro.html

API details: https://pandas.pydata.org/pandas-docs/stable/reference/frame.html

Exercise 3.1: Load the 20newsgroup JSON dataset in and create a pandas data frame to plot the breakdown of all posts by character length. This will show if there is a training bias towards longer or shorter posts.

Solution 3.1

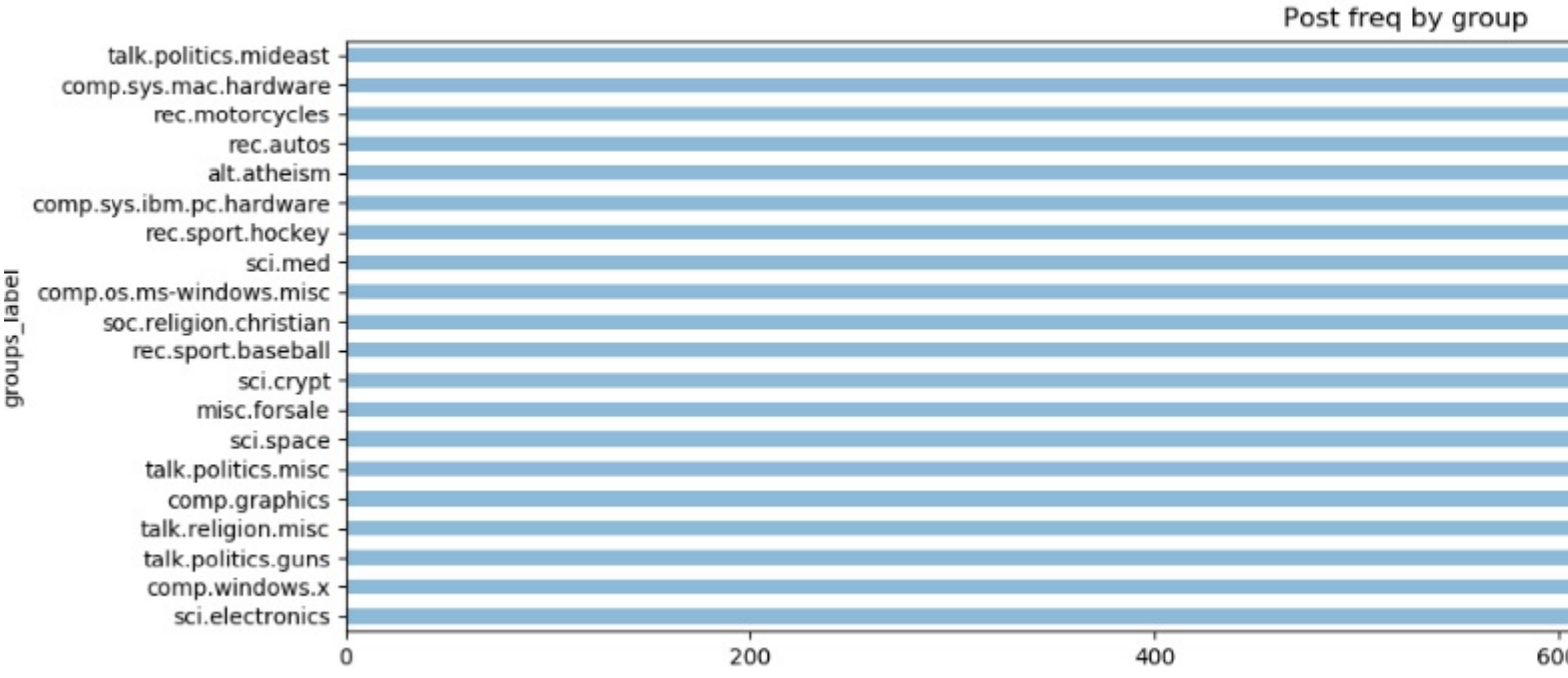
See solution file tutorial-data-viz.py



Exercise 3.2: Load the 20newsgroup JSON dataset in and visualize post freq breakdown for groups. This will show if there is a training bias towards groups with more or less posts.

Solution 3.2

See solution file tutorial-data-viz.py



4 Data analysis & hypothesis formulation

Once you have the basic dataset level characterization done you can go deeper into the data and look at specific data segments, sub-segments and enriched features. Your aim is to test out hypotheses about what patterns might exist in the data that could

suggest some natural classes, clusters or highly discriminating feature sets. This might involve enriching the data through additional annotation techniques (e.g. POS or NER tagging of text data or temporal segmentation of numeric data to look at time-period features) or testing out derived features (e.g. aggregations of features where you have a hypothesis that they are correlated).

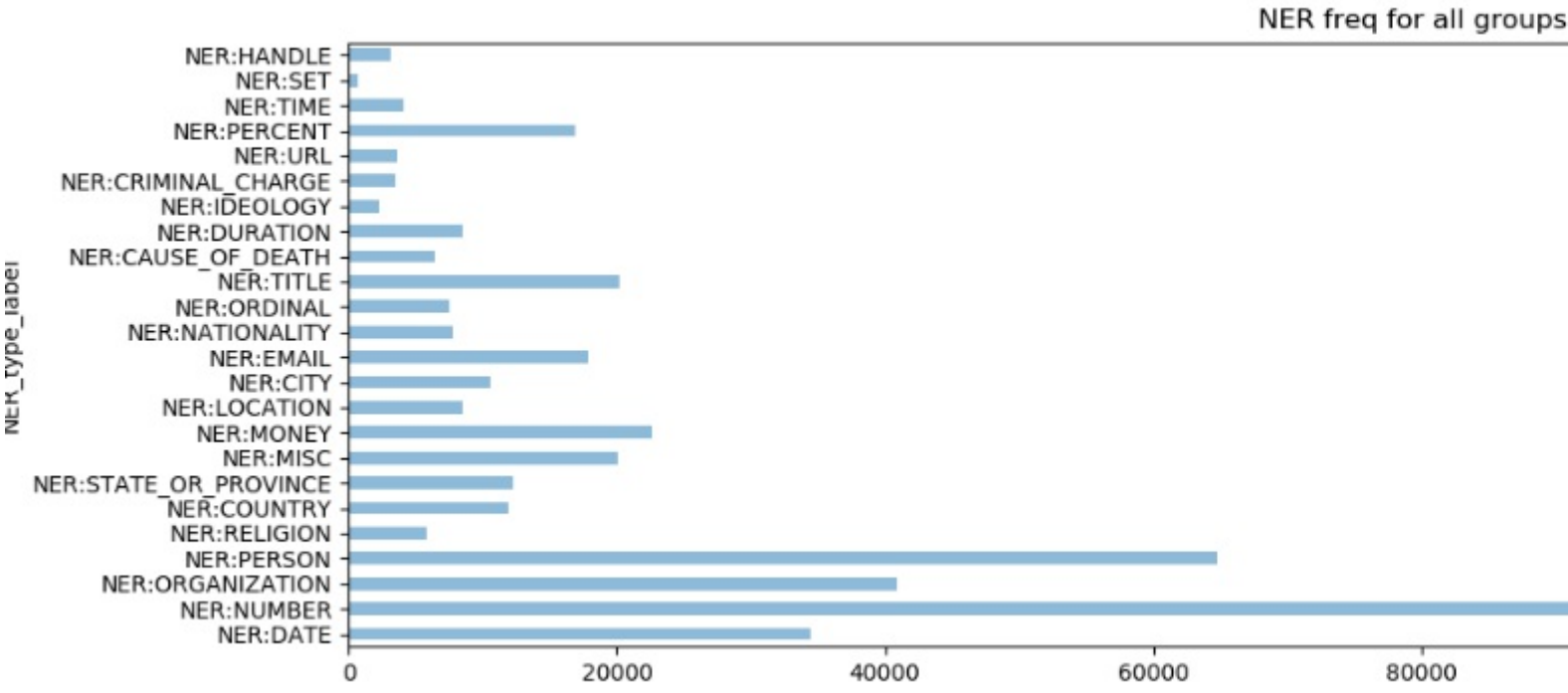
Testing hypotheses early on in the data analysis stage of a project allows better matching of feature characteristics to algorithm choices, allowing more appropriate sets of algorithms to be chosen prior to more time-consuming implementation and evaluation work.

The 20newsgroup JSON dataset has a set of named entity recognition (NER) tags assigned to each post that were generated by the Stanford CoreNLP tools. In the following exercises you will dive deep into these annotations and visualize them broken down into various levels. This is typical of what you would do on any dataset – start by visualizing high level breakdowns of data and then start to explore specific segments, breaking them into sub-segments and looking at patterns around specific values within sub-segments. Your goal is to develop a hypothesis around what features or feature groups might represent useful patterns for a machine learning algorithm to find and thus achieve your problem's end goal.

Exercise 4.1: Load the 20newsgroup JSON dataset in and visualize NER type mention freq breakdown for all groups.

Solution 4.1

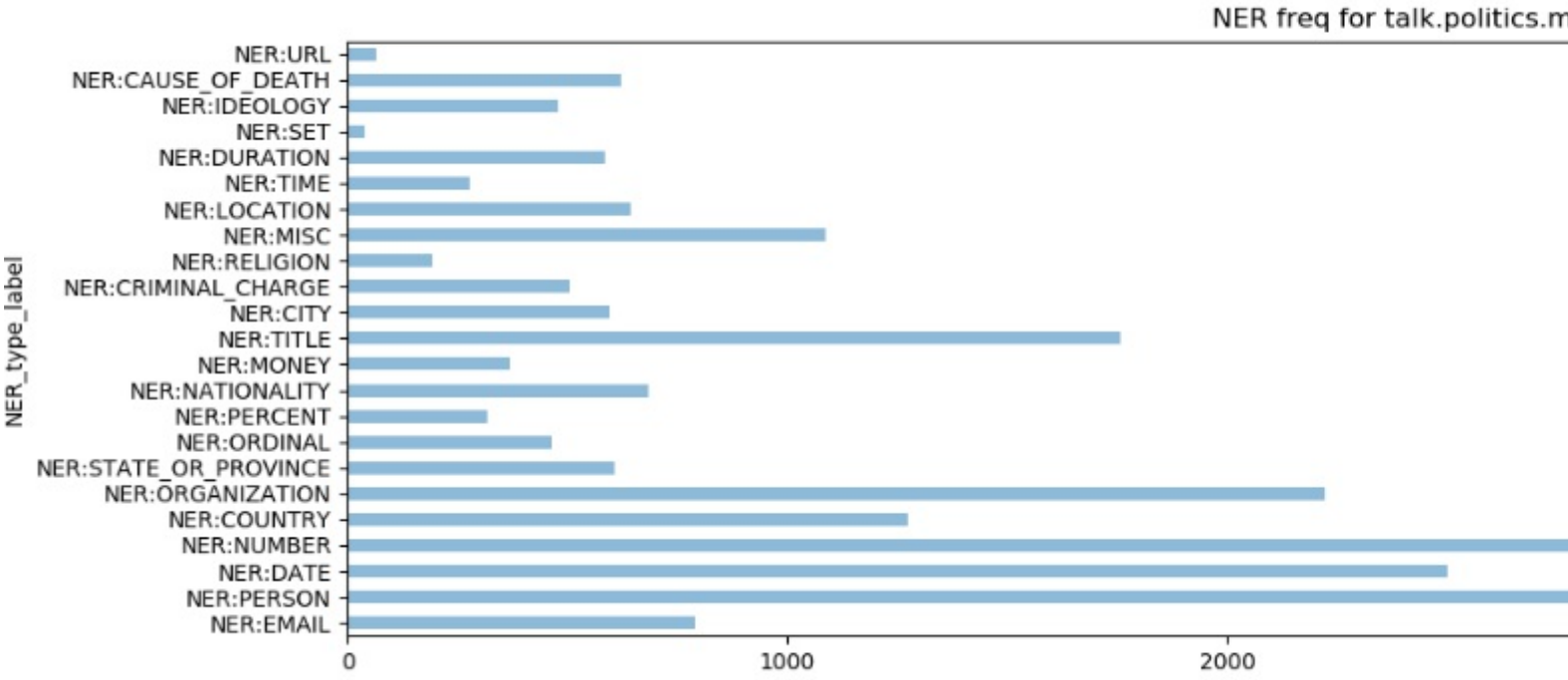
See solution file tutorial-data-viz.py



Exercise 4.2: Load the 20newsgroup JSON dataset in and visualize NER type value mention freq breakdown for talk.politics.misc

Solution 4.2

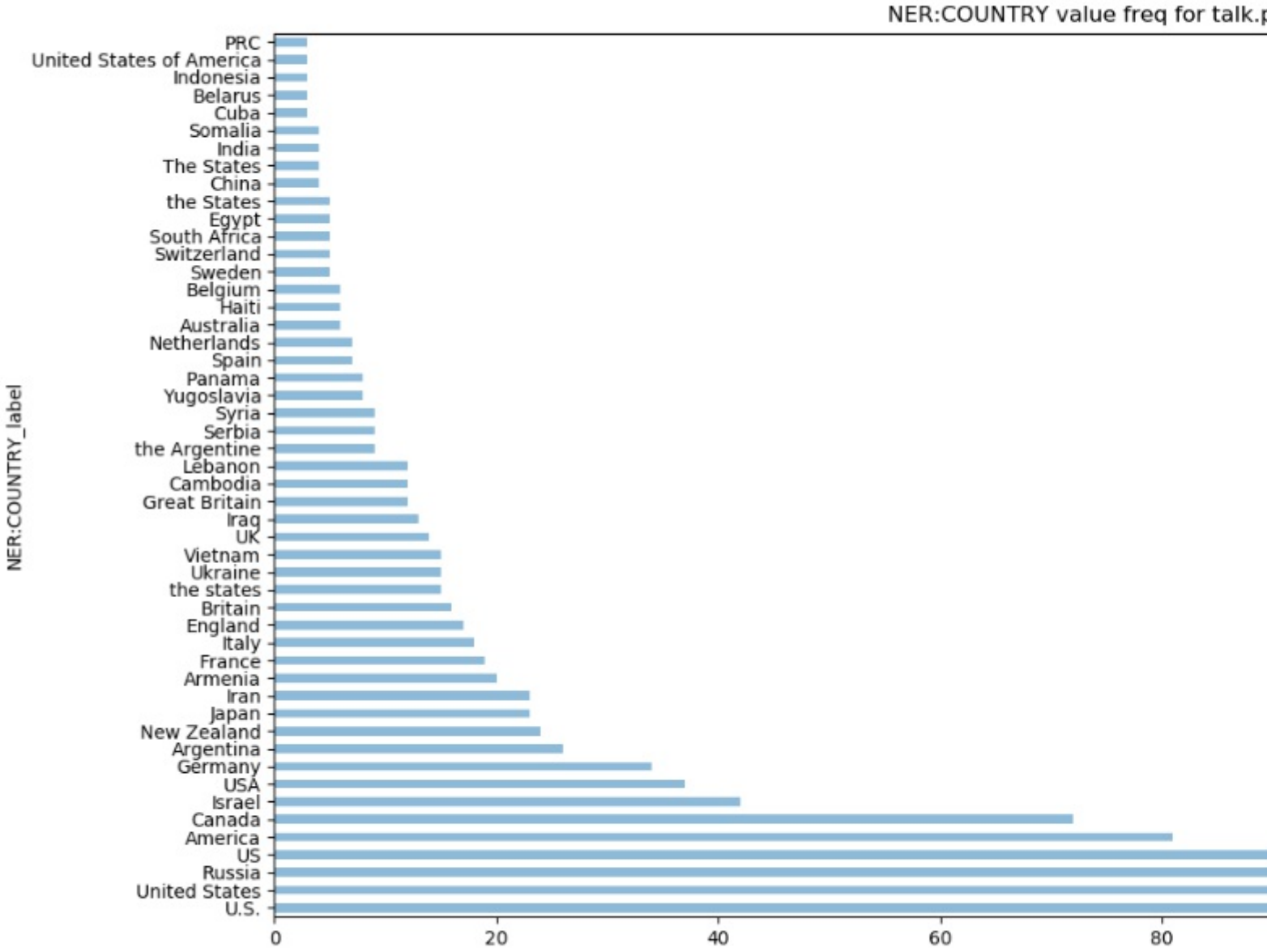
See solution file tutorial-data-viz.py



Exercise 4.3: Load the 20newsgroup JSON dataset in and visualize all NER:COUNTRY value breakdown for talk.politics.misc

Solution 4.3

See solution file tutorial-data-viz.py



5 Geospatial visualization of datasets

When dealing with location data it is often useful to visualize it on a map so you can inspect the data for spatial patterns, trends and/or clusters. The geopandas python library makes this easy to do quickly and can be very helpful.

Installation of geopandas is easy for Linux, but a bit more tricky for Windows. Read the geopandas installation instructions https://geopandas.readthedocs.io/en/latest/install.html and give it a try. When using Windows it is best to use pre-compiled binaries and this avoid having to compile underlying C++ code yourself for your native windows platform.

Linux install (pre-requisite libraries will be installed by pip automatically)

- py -m pip install pandas

- py -m pip install matplotlib

- py -m pip install geopandas

- py -m pip install descartes

- py -m pip install rtree

Windows install (download precompiled binaries from gohlke first)

- py -m pip install pandas

- py -m pip install matplotlib

- py -m pip install Shapely-1.6.4.post2-cp37-cp37m-win_amd64.whl

- py -m pip install GDAL-3.0.1-cp37-cp37m-win_amd64.whl

- py -m pip install Fiona-1.8.9-cp37-cp37m-win_amd64.whl

- py -m pip install geopandas

- py -m pip install descartes

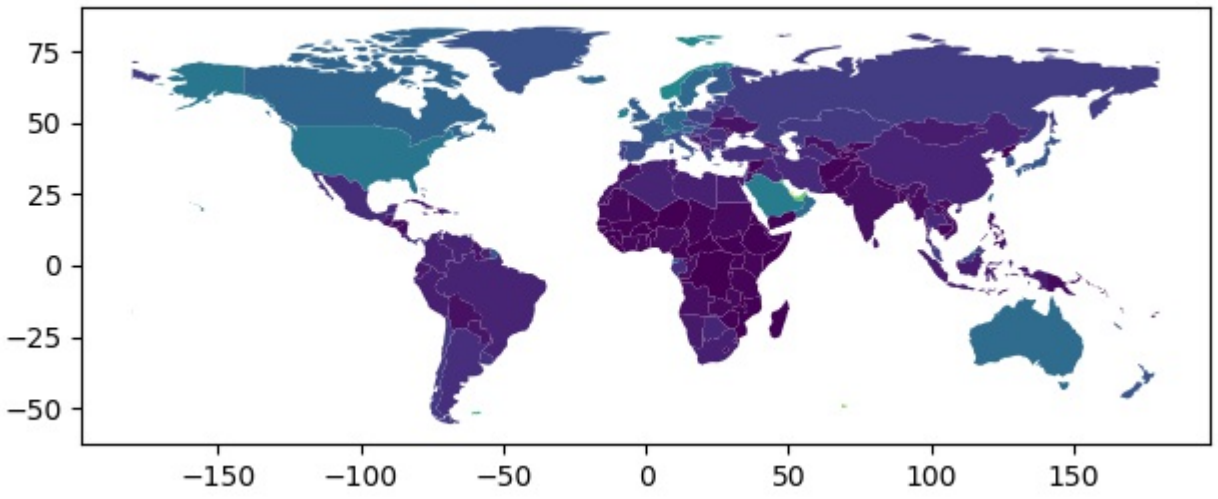
- py -m pip install Rtree-0.8.3-cp37-cp37m-win_amd64.whl

geopandas installation instructions ➔ http://geopandas.org/install.html

Gohlke mirror for pre-compiled binaries ➔ https://www.lfd.uci.edu/~gohlke/pythonlibs/

Read the geopandas mapping tutorial https://geopandas.readthedocs.io/en/latest/mapping.html and try out the code below. These are the basic building blocks to visualize a map.

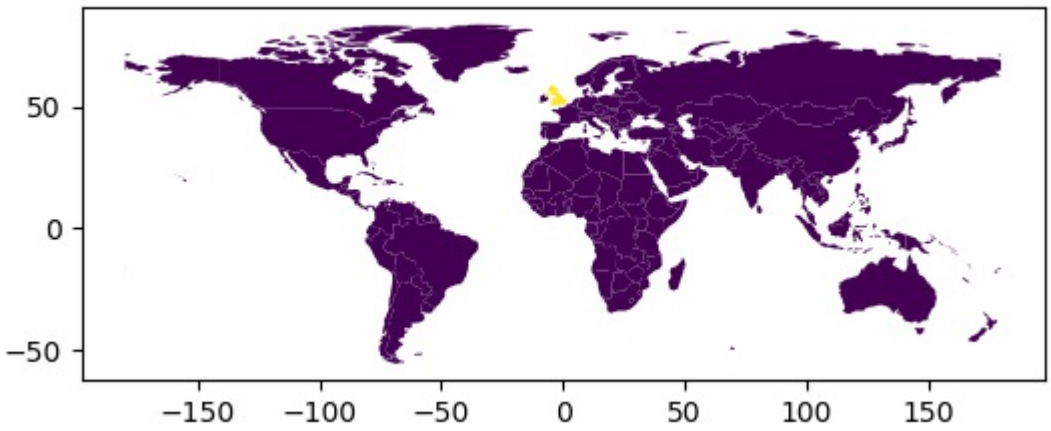
```
import geopandas as gpd

world = gpd.read_file( gpd.datasets.get_path('naturalearth_lowres') )

world = world[(world.pop_est>0) & (world.name!="Antarctica")]

world['gdp_per_cap'] = world.gdp_md_est / world.pop_est

world.plot( column='gdp_per_cap' );

plt.show()
```



You can add your own custom attributes to location data (i.e. beyond attributes like GDP which are shipped with the world dataset) and visualize these.

```
world = gpd.read_file( gpd.datasets.get_path('naturalearth_lowres') )

world = world[(world.pop_est>0) & (world.name!="Antarctica")]

for index, row in world.iterrows():
```

```
str_place = world.loc[ index, 'name' ]

if str_place == 'United Kingdom' :

world.loc[ index,'favourite' ] = 1

else :

world.loc[ index,'favourite' ] = 0

world.plot( column='favourite' );

plt.show()
```



Exercise 5.1: Load the 20newsgroup JSON dataset in and visualize all NER:COUNTRY values on a map (countries). Read up on the geopandas API and find out how to Include a legend for the colour values.

Solution 5.1

See solution file tutorial-data-viz.py