Lect 15 – Data Analysis Examples

Rob Capra

INLS 490-172

usa.gov bit.ly data

- Every time someone created a shortened URL using bit.ly of a US gov website (.gov, .mil), a record is logged:

```
print records[0]
{u'a': u'Mozilla/5.0 (Windows NT 6.1; WOW64)
AppleWebKit/535.11 (KHTML, like Gecko) Chrome/17.0.963.78
Safari/535.11', u'c': u'US', u'nk': 1, u'tz':
u'America/New_York', u'gr': u'MA', u'g': u'A6qOVH', u'h':
u'wfLQtf', u'cy': u'Danvers', u'l': u'orofrog', u'al': u'en-
US,en;q=0.8', u'hh': u'1.usa.gov', u'r':
u'http://www.facebook.com/l/7AQEFzjSi/1.usa.gov/wfLQtf', u'u':
u'http://www.ncbi.nlm.nih.gov/pubmed/22415991', u't':
1331923247, u'hc': 1331822918, u'll': [42.576698, -70.954903]}
```

- You can download this dataset from:
  - https://raw.githubusercontent.com/pydata/pydata-book/master/ch02/usagov_bitly_data2012-03-16-1331923249.txt

usa.gov data

- Read the JSON data into a ***list of dicts***
- Longer version with a for loop:

```
fp = open('usagov_ex1.txt')
records = []
for line in fp:
    records.append(json.loads(line))
fp.close()
```

- Shorter version using list comprehension

```
fp = open('usagov_ex1.txt')
records = [json.loads(line) for line in fp]
fp.close()
```

usa.gov data

- Pull out the time zones; key = ‘tz’
- First try:

```
tzs = []
for rec in records:
    tzs.append(rec['tz'])
```

- Longer version with a for loop:

```
tzs = []
for rec in records:
    if 'tz' in rec:
        tzs.append(rec['tz'])
```

- Shorter version using list comprehension:

```
tzs = [rec['tz'] for rec in records if 'tz' in rec]
```

usa.gov data

- Count the time zones; three equivalent methods:

```
tz_counts = {}
for tz in tzs:
    if tz in tz_counts:
        tz_counts[tz] += 1
    else:
        tz_counts[tz] = 1
print tz_counts['America/New_York']
```

```
tz_counts2 = {}
for tz in tzs:
    tz_counts2[tz] = tz_counts2.get(tz,0) + 1
print tz_counts2['America/New_York']
```

```
from collections import defaultdict
tz_counts3 = defaultdict(int)
for tz in tzs:
    tz_counts3[tz] += 1
print tz_counts3['America/New_York']
```

This is the first time we have seen defaultdict.

defaultdict (part of collections)

- Like a dict, except will automatically initialize new keys the first time they are seen.
- "When each key is encountered for the first time, it is not already in the mapping; so an entry is automatically created using the *default_factory* function..."

```
from collections import defaultdict
tz_counts3 = defaultdict(int)
for tz in tzs:
    tz_counts3[tz] += 1
print tz_counts3['America/New_York']
```

Here, we use "int" for the default_factory. This will initialize new the value of new entries to be an integer set to 0.

Other objects can be used for the default_factory. For example, **`list`**, would initialize the new value to be an empty list.

usa.gov data

- Top 10 time zones based on counts
- tz_counts is a dict mapping time zone to count
- We need to sort based on count
- We know options for how to sort dicts by value!
- We could use sorted:

```
sorted_tzc = sorted(tz_counts.items(), key=itemgetter(1),
reverse=True)
for tz, c in sorted_tzc[:10]:
    print tz, c
```

usa.gov data

- We could use sorted:

```
sorted_tzc = sorted(tz_counts.items(), key=itemgetter(1),
reverse=True)
for tz, c in sorted_tzc[:10]:
    print tz, c
```

- Or we could do the acrobatics from the book:

```
ctz = [(c,tz) for tz,c in tz_counts.items()]
sorted_ctz = ctz[:]
sorted_ctz.sort(reverse=True)
for c, tz in sorted_ctz[:10]:
    print c,tz
```

collections

- Python has some very powerful datatypes in the collections library
- We've already seen collections.defaultdict
- Here, we will look at collections.Counter

- But there are others:
  - namedtuple
  - deque
  - ordereddict

  - http://docs.python.org/2/library/collections.html

collections.Couter

- Top 10 time zones based on counts
- tz_counts is a dict mapping time zone to count
- We need to sort based on count
- We know options for how to sort dicts by value!
- We could use sorted:

```
tzs2 = [rec['tz'] for rec in records if 'tz' in rec]
cc = Counter(tzs2)
cc.most_common(10)
```

pandas

- pandas is a powerful Python library
- For now, we will just get a “peek”
- DataFrame – represents a table of data
  - Can think of a DataFrame like a spreadsheet

```
frame = DataFrame(records)
ftzc = frame['tz'].value_counts()
print ftzc[:10]
ftzc[:10].plot(kind='barh', rot=0)
```