This notebook explores dependency parsing by identifying the actions and objects that are characteristically associated with male and female characters.

In [ ]:
```
import spacy, math
from collections import Counter
import operator
```

In [ ]:
```
nlp = spacy.load('en_core_web_sm')
```

We'll run seven novels by Jane Austen through spacy (this will take a few minutes).

In [ ]:
```
filenames=["../data/fiction/emma.txt", "../data/fiction/lady_susan.txt", "../
all_tokens=[]
for filename in filenames:
    print(filename)
    data=open(filename, encoding="utf-8").read()
    tokens=nlp(data)
    all_tokens.extend(tokens)
```

```
../data/fiction/emma.txt
../data/fiction/lady_susan.txt
../data/fiction/mansfield_park.txt
../data/fiction/northanger_abbey.txt
../data/fiction/persuasion.txt
../data/fiction/pride.txt
../data/fiction/sense_and_sensibility.txt
```

In [ ]:
```
print (len(all_tokens))
```

```
972808
```

In [ ]:
```
def test(maleCounter, femaleCounter, display=25):

    """ Function that takes two Counter objects as inputs and prints out a ra
    more characteristic of the first counter than the second.  Here we'll use
    with an uninformative prior (from Monroe et al 2008, "Fightin Words", eqn

    """

    vocab=dict(maleCounter)
    vocab.update(dict(femaleCounter))
    maleSum=sum(maleCounter.values())
    femaleSum=sum(femaleCounter.values())

    ranks={}
    alpha=0.01
    alphaV=len(vocab)*alpha

    for word in vocab:

        log_odds_ratio=math.log( (maleCounter[word] + alpha) / (maleSum+alpha
        variance=1./(maleCounter[word] + alpha) + 1./(femaleCounter[word] + a

        ranks[word]=log_odds_ratio/math.sqrt(variance)

    sorted_x = sorted(ranks.items(), key=operator.itemgetter(1), reverse=True

    print("Most male:")
    for k,v in sorted_x[:display]:
        print("%.3f\t%s" % (v,k))

    print("\nMost female:")
    for k,v in reversed(sorted_x[-display:]):
        print("%.3f\t%s" % (v,k))
```

Spacy uses the ClearNLP dependency labels, which are very close to the Stanford typed dependencies. See the Stanford dependencies manual for more information about each tag. Parse information is contained in the spacy token object; see the following for which attributes encode the token text, idx (position in sentence), part of speech, and dependency relation. The syntactic head for a token is another token given in `token.head` (where all of those same token attributes are accessible).

In [ ]:
```
testDoc=nlp("He started his car.")
for token in testDoc:
    print("%s\t%s\t%s\t%s\t%s\t%s" % (token.text, token.idx, token.tag_,
```

```
He      0       PRP     nsubj   started 3       VBD
started 3       VBD     ROOT    started 3       VBD
his     11      PRP$    poss    car     15      NN
car     15      NN      dobj    started 3       VBD
.       18      .       punct   started 3       VBD
```

Q1: Find the verbs that men are more characteristically the *subject* of than women. Feel free to only consider subjects that are "he" and "she" pronouns. This function should return two Counter objects (`maleCounter` and `femaleCounter`) which counts the number of times a given verb has "he" (`maleCounter`) and "she" (`femaleCounter`) as its syntactic subject.

In [ ]:

```
def count_subjects():
    maleCounter=Counter()
    femaleCounter=Counter()

    for token in all_tokens:
        if token.text.lower() == 'he' and token.head.tag_ == 'VBD' and token.
            maleCounter[token.head.text] +=1
        if token.text.lower() == 'she' and token.head.tag_ == 'VBD' and token
            femaleCounter[token.head.text] +=1

    return maleCounter, femaleCounter
```

In [ ]:

```
male, female=count_subjects()
test(male, female, display=10)
```

```
Most male:
6.087   said
5.822   replied
5.380   came
4.577   seemed
3.484   told
2.747   took
2.707   continued
2.353   talked
2.340   left
2.292   asked

Most female:
-6.983  felt
-4.672  saw
-3.975  found
-3.706  knew
-3.694  heard
-3.212  cried
-3.152  thought
-2.504  read
-2.359  feared
-2.358  dared
```

Q2: Find the verbs that men are more characteristically the *object* of than women. Feel free to only consider objects that are "him" and "her" pronouns. This function should return two Counter objects (`maleCounter` and `femaleCounter`) which counts the number of times a given verb has "he" (`maleCounter`) and "she" (`femaleCounter`) as its syntactic direct object.

In [ ]:
```
def count_objects():
    maleCounter=Counter()
    femaleCounter=Counter()

    for token in all_tokens:
        if token.text.lower() == 'him' and token.head.tag_ == 'VBD' and token
            maleCounter[token.head.text] +=1
        if token.text.lower() == 'her' and token.head.tag_ == 'VBD' and token
            femaleCounter[token.head.text] +=1

    return maleCounter, femaleCounter
```

In [ ]:
```
male, female=count_objects()
test(male, female, display=10)
```

```
Most male:
3.039   saw
2.416   thanked
2.359   liked
1.812   begged
1.776   did
1.776   recommended
1.600   brought
1.518   understood
1.490   wished
1.468   observed

Most female:
-2.657  left
-1.996  attended
-1.954  struck
-1.892  convinced
-1.587  gave
-1.587  obliged
-1.533  joined
-1.115  enabled
-1.115  pleased
-0.932  advised
```

Q3: Find the objects that are *possessed* more frequently by men than women. Feel free to only consider possessors that are "his" and "her" pronouns. This function should return two Counter objects (`maleCounter` and `femaleCounter`) which counts the number of times a given term is possessed by "he" (`maleCounter`) and "she" (`femaleCounter`).

In [ ]:
```
def count_possessions():
    maleCounter=Counter()
    femaleCounter=Counter()

    for token in all_tokens:
        if token.text.lower() == 'his' and token.head.tag_ == 'NN' and token.
            maleCounter[token.head.text] +=1
        if token.text.lower() == 'her' and token.head.tag_ == 'NN' and token.
            femaleCounter[token.head.text] +=1

    return maleCounter, femaleCounter
```

In [ ]:
```
male, female=count_possessions()
test(male, female, display=10)
```

```
Most male:
4.284   return
4.230   house
3.978   name
3.570   horse
3.525   son
3.515   attachment
3.416   character
3.410   behaviour
3.265   business
3.248   pride

Most female:
-7.071  mother
-6.112  sister
-4.946  aunt
-4.079  uncle
-3.630  heart
-3.528  room
-3.068  hand
-2.990  brother
-2.927  mind
-2.686  fancy
```

Q4: Find the actions that are men do *to women* more frequently than women do *to men*. Feel free to only consider subjects and objects that are "she"/"he"/"her"/"him" pronouns. This function should return two Counter objects (`maleCounter` and `femaleCounter`) which counts the number of times a given verb has "he" as the subject and "her" as the object (`maleCounter`) and "she" as the subject and "him" as the object (`femaleCounter`).

In [ ]:
```
def count_SVO_tuples():
    maleCounter=Counter()
    femaleCounter=Counter()

    for i, token in enumerate(all_tokens):
        if token.text.lower() == 'he' and token.head.tag_ == 'VBD' and token.
            for token_2 in all_tokens[i-20:i+20]:
                if token_2.text.lower() == 'her' and token_2.head.idx == toke
                    maleCounter[token.head.text] +=1

        if token.text.lower() == 'she' and token.head.tag_ == 'VBD' and token
            for token_2 in all_tokens[i-20:i+20]:
                if token_2.text.lower() == 'him' and token_2.head.idx == toke
                    femaleCounter[token.head.text] +=1

    return maleCounter, femaleCounter
```

In [ ]:
```
male, female=count_SVO_tuples()
test(male, female, display=10)
```

```
Most male:
1.464   loved
1.229   joined
1.229   asked
1.131   told
0.931   left
0.569   handed
0.569   heard
0.538   called
0.538   distinguished
0.527   gave

Most female:
-1.518  saw
-1.006  found
-0.789  thanked
-0.667  liked
-0.608  answered
-0.608  watched
-0.607  refused
-0.535  perceived
-0.535  understood
-0.535  received
```

In [ ]: