Übung 11: NLTK

1. Starten Sie eine Python Interpreter-Session (`$ python` oder `$ ipython`) und importieren Sie zunächst nur das `corpus` Modul.

2. Wir wollen genauer wissen, was für Korpora uns das NLTK bietet. Sie sind alle im Modul `corpus` enthalten. Versuchen Sie, herauszufinden, welche Korpora enthalten sind und finden Sie eines, das in getaggter Form vorliegt. Tipp:

   - Die Korpora sind unterschiedlich formatiert und annotiert. Welche Formate und Annotationen ein Korpus bietet, erkennt man schnell daran, welche Methoden sein Reader enthält (z.B. in der API oder über die Onlinehilfe und Autovervollständigung in iPython). Eine Übersicht gibt es im Appendix des NLTK-Buches.

   a) Lassen Sie sich die Anzahl der im Corpus enthaltenen Wörter ausgeben.

   b) Lassen Sie sich die Anzahl der im Corpus enthaltenen Sätze ausgeben.

3. Wir wollen jetzt sehen, welche Möglichkeiten das NLTK für den Umgang mit CFGs bietet und dazu die Module `tree` und `grammar` näher betrachten. Importieren Sie einfach das gesamte NLTK in Ihre Python Session. Nehmen wir an, wir wollen aus einem geparsten Korpus eine Grammatik induzieren.

   a) Finden Sie heraus, wie ein Parsebaum als String aussehen muss, damit das `nltk.tree` Modul ihn in ein Objekt der `Tree`-Klasse umwandeln zu können.

   b) Erstellen Sie einen Parsebaum in diesem Format für einen Beispielsatz. Nehmen Sie dazu einen der Sätze des Korpus oder wählen Sie diesen: `"Einstein war ein Physiker."`

   c) Studieren Sie die API-Dokumentation oder die Onlinehilfe, um herauszufinden, mit welcher Funktion von `nltk.tree.Tree` Sie diese Stringrepräsentation in ein `Tree`-Objekt überführen können. Erzeugen Sie so ein Objekt für den Beispielsatz und weisen Sie ihm einen Namen zu, z.B. `my_tree`.

   d) Lassen Sie sich `my_tree` graphisch darstellen. Dazu hat jede `Tree` Instanz eine eingebaute Methode. Sie können ihre Analyse so leicht überprüfen.

   e) Praktisch an der Klasse `Tree` ist, dass sie gleich die benutzten Regeln im Baum ermittelt. Lassen Sie sich die Produktionsregeln von Ihrem Baum ausgeben. Auch das geht mit einer Instanz-Methode.

   f) An dieser Repräsentation ist wiederum praktisch, dass man sie direkt weiter verwenden kann, um eine Grammatik zu erstellen. Finden Sie über die API des `grammar` Moduls dazu die richtige Klasse. Erzeugen Sie die aus Ihrem Satz induzierte (sehr kleine) kontextfreie Grammatik.

   g) Falls Sie den obigen Beispielsatz verwendet haben, sollte die Grammatik jetzt die Tokensequenz `` `ein Physiker' `` abdecken. Wie finden Sie heraus, ob sie das tut?

4. Um nun eine etwas umfangreichere Grammatik zu erzeugen, benutzen wir die Penn Treebank. Das `corpus treebank` liegt u.A. als Liste von Parse-Bäumen vor.

   a) Schreiben Sie eine Funktion `getCFG()`, die eine Liste aller in der `treebank` benutzten Produktionsregeln erzeugt (behandeln Sie hierbei Mehrfachvorkommen sinnvoll) und anschließend daraus eine CFG erstellt und diese zurückliefert.

   b) Sehen Sie sich das `parse` Modul an. Es bietet Implementationen verschiedener CFG-und PCFG-Parser. Wählen Sie einen, der mit einer CFG arbeitet. Schreiben Sie eine Funktion `drawParses()`, die eine CFG und einen Satz als Argumente nimmt. Sie soll dann einen entsprechenden Parser erzeugen, den Satz parsen und die drei besten Bäume grafisch ausgeben. Probieren Sie zunächste eine kurze Phrase wie `"British ships"`, da je nach Parser bei längeren Sätzen die Gefahr besteht, in eine Endlosschleife zu geraten. Hinweis:

      - Manche Parser benötigen die Angabe einer Strategie. Dazu am besten die Variable `nltk.parse.chart.BU_STRATEGY` übergeben. Es ist eine Sammlung von Regeln, nach denen die verschiedenen Expansionen durchprobiert werden.