Dennis Nikolay 2544912
Julia Schulz 2553760
Mark Schuegraf 2543591

Einführung CoLi: Blatt 9

Aufgabe 9.1

a. Zwei, ADJA und NADJA.
b. Drei Merkmale mit binärem Wertebereich.
c. Ereignisraum = 2*2*2 = 8 Elemente
   Ereignisse: f = falsch, w = wahr
   <f,f,f>,<f,f,w>,<<f,w,f>,<f,w,w>,<w,f,f>,<w,f,w>,<w,w,f>,<w,w,w>

   <f,f,f> Er geht mit <NADJA> ihm weg. Der Tag ist schön <ADJA> und heiß.

   <f,f,w> Er ist mit anderen <NADJA> kurz weg gegangen. Er mag seinen schönen <ADJA> und gepflegten Garten.

   <f,w,f> Kein <NADJA> Fahrrad steht in der Ecke. Sei affig <ADJA>, Gorilla.

   <f,w,w> Auf meinem <NADJA> Bus ist ein Vogel. Auf alten <ADJA> Regalen ist Staub.

   <w,f,f> Die Frau, die <NADJA> kommt. Das Wort w ist nach Merkmalsforderung ein Artikel und kann somit kein Adjektiv sein.

   <w,f,w> Der Mann, der <NADJA> muss. Das Wort w ist nach Merkmalsforderung ein Artikel und kann somit kein Adjektiv sein.

   <w,w,f> Der Mann, der ein <NADJA> Kind sucht. Das Wort w ist nach Merkmalsforderung ein Artikel und kann somit kein Adjektiv sein.

   <w,w,w> Der Mann, der den <NADJA> Kuchen isst. Das Wort w ist nach Merkmalsforderung ein Artikel und kann somit kein Adjektiv sein.
d. Von acht möglichen Ereignissen, wurden für sieben entsprechende Vorkommen im Trainingskorpus entdeckt. Zum Ereignis <w,f,f> lagen keine Daten vor. Die Abdeckung ist daher an dieser Stelle lückenhaft. Deswegen kann das System in diesem Fall keine Entscheidung treffen.

e. <f,f,f> => NADJA (100%)
<f,f,w> => NADJA (4,5%), ADJA (95,5%)
<f,w,f> => NADJA (0,6%), ADJA (99,4%)
<f,w,w> => NADJA (32,3%), ADJA (66,7%)
<w,f,f> => keine Entscheidung möglich
<w,f,w> => NADJA (100%)
<w,w,f> => NADJA (100%)
<w,w,w> => NADJA (100%)

Aufgabe 9.2

a. Recall: 880/(880+80) = 11/12
Precision: 880/(880+20) = 44/45
b. Recall und Precision sind beide deutlich höher als bei ADJA. Das Modell für nicht adjektivische Wortarten ist folglich besser, da mehr Instanzen richtig klassifiziert wurden und von den klassifizierten Instanzen weniger falsch eingestuft wurden. Das widerum deutet auf größere Relevanz der gewählten Merkmale in Hinblick auf die Bestimmung anderer Wortarten hin.
c. Die Precision, da ein vorsichtiges Klassifikationsverhalten ambige Instanzen unberührt lässt. Dadurch kann im Nachhinein manuell oder aber durch andere Verfahren eine Resolution stattfinden, anstatt dem Nutzer eine mehr oder weniger zufällige Wortart zu präsentieren.

Aufgabe 9.3

(a)

Zunächst konstruiere man reguläre Ausdrücke um die drei Merkmale zu charakterisieren:

Merkmal	Regex
Wort w Artikel?	"\sART"
Wort w+1 großgeschrieben?	"^\w+\t[A-Z]"
Wort w hat -er/-es/-e/-en/-em Endung?	"^\w+(er|es|e|en|em)\t"

Nun gilt es diese Ausdrücke geeignet durch Pipelining den Merkmalen eines Ereignisses entsprechend zu verketten:

Merkmalsmuster	Befehl
<-,-,->	grep -v -E "\sART" tiger_bigram.txt | grep -v -E "^\w+\t[A-Z]" | grep -v -E "^\w+(er|es|e|en|em)\t"
<-,-,+>	grep -v -E "\sART" tiger_bigram.txt | grep -v -E "^\w+\t[A-Z]" | grep -E "^\w+(er|es|e|en|em)\t"
<-,+,->	grep -v -E "\sART" tiger_bigram.txt | grep -E "^\w+\t[A-Z]" | grep -v -E "^\w+(er|es|e|en|em)\t"
<-,+,+>	grep -v -E "\sART" tiger_bigram.txt | grep -E "^\w+\t[A-Z]" | grep -E "^\w+(er|es|e|en|em)\t"
<+,-,->	grep -E "\sART" tiger_bigram.txt | grep -v -E "^\w+\t[A-Z]" | grep -v -E "^\w+(er|es|e|en|em)\t"
<+,-,+>	grep -E "\sART" tiger_bigram.txt | grep -v -E "^\w+\t[A-Z]" | grep -E "^\w+(er|es|e|en|em)\t"
<+,+,->	grep -E "\sART" tiger_bigram.txt | grep -E "^\w+\t[A-Z]" | grep -v -E "^\w+(er|es|e|en|em)\t"
<+,+,+>	grep -E "\sART" tiger_bigram.txt | grep -E "^\w+\t[A-Z]" | grep -E "^\w+(er|es|e|en|em)\t"

(b)

Man kann sed mit dem "s" Befehl verwenden, um das Ende einer Zeile $ durch eine entsprechende Klassifizierung zu ersetzen.

Es bietet sich somit folgende Vorgehensweise an: Der Reihenfolge nach im Korpus nach den Merkmalsmustern suchen, eine Klassifikation vornehmen und die Ergebniszeilen in einer Datei akkumulieren. Realisiert wird diese durch die Befehlsketten:

Merkmalsmuster	Befehl
<-,-,->	grep -v -E "\sART" tiger_bigram.txt | grep -v -E "^\w+\t[A-Z]" | grep -v -E "^\w+(er|es|e|en|em)\t" | sed "s/.*/& NADJA/g" >> tiger_bigram_ADJA.txt
<-,-,+>	grep -v -E "\sART" tiger_bigram.txt | grep -v -E "^\w+\t[A-Z]" | grep -E "^\w+(er|es|e|en|em)\t" | sed "s/.*/& NADJA/g" >> tiger_bigram_ADJA.txt
<-,+,->	grep -v -E "\sART" tiger_bigram.txt | grep -E "^\w+\t[A-Z]" | grep -v -E "^\w+(er|es|e|en|em)\t" | sed "s/.*/& NADJA/g" >> tiger_bigram_ADJA.txt

<-,+,+>	grep -v -E "\sART" tiger_bigram.txt | grep -E "^\w+\t[A-Z]" | grep -E "^\w+(er|es|e|en|em)\t" | sed "s/.*/& ADJA/g" >> tiger_bigram_ADJA.txt
<+,-,->	grep -E "\sART" tiger_bigram.txt | grep -v -E "^\w+\t[A-Z]" | grep -v -E "^\w+(er|es|e|en|em)\t" | sed "s/.*/& NichtKlassifizierbar/g" >> tiger_bigram_ADJA.txt
<+,-,+>	grep -E "\sART" tiger_bigram.txt | grep -v -E "^\w+\t[A-Z]" | grep -E "^\w+(er|es|e|en|em)\t" | sed "s/.*/& NADJA/g" >> tiger_bigram_ADJA.txt
<+,+,->	grep -E "\sART" tiger_bigram.txt | grep -E "^\w+\t[A-Z]" | grep -v -E "^\w+(er|es|e|en|em)\t" | sed "s/.*/& NADJA/g" >> tiger_bigram_ADJA.txt
<+,+,+>	grep -E "\sART" tiger_bigram.txt | grep -E "^\w+\t[A-Z]" | grep -E "^\w+(er|es|e|en|em)\t" | sed "s/.*/& NADJA/g" >> tiger_bigram_ADJA.txt

Anmerkung zum sed Befehl: leider unterstützt sed keine Tabs in der Form \t. Deswegen wurde ein Tab über die entsprechende Keyboard Tastenkombination im Terminal erzeugt.

Anmerkung zum Fall <+,-,->: aufgrund des Sparse-Data Problems welches in diesem Fall vorliegt kann der Klassifikator keine Entscheidung treffen.

Die Datei tiger_bigram_ADJA.txt enthält nun den um eine zusätzliche Spalte erweiterten Korpus.

(c)

```
cut -f 3,4 tiger_bigram_ADJA.txt > tiger_bigram_ADJA_cut.txt;

grep -v -E "^(ADJA)\t" tiger_bigram_ADJA_cut.txt | while read -r line; do echo "NADJA\t" >> tiger_bigram_ADJA_simple.txt; done;

cut -d '\s' -f 2 tiger_bigram_ADJA_cut.txt > tmp.txt;

paste tmp.txt tiger_bigram_ADJA_simple.txt > tiger_bigram_ADJA_res.txt;
```

tiger_bigram_ADJA_res.txt enthält nun die Paare aus Goldstandard und Klassifikation. Die statistische Analyse ist trivial.