CS 5785 Homework 2

1 Programming Exercises

2.

1. ['not as advertised product arrived labeled as jumbo salted peanuts the peanuts were actually small sized unsalted not sure if this was an error or if the vendor intended to represent the product as jumbo', 0]. Positive reviews tend to be longer and use more formal words, while negative reviews tend to use more insulting words and do not always follow grammar rules.
2. We can’t get meaningful information from the unigrams not excluding stopwords. Using sort and uniq -u on the data below we can discover that there are only six words that positive and negative unigrams do not have in common, which are: *be, good, great, like, taste, these*; within which only four are not stopwords, way too uninformative.

30 most popular unigrams among all reviews and their frequencies:

the 1928184

i 1717187

and 1325118

a 1276976

it 1106226

to 1036816

of 822668

is 753238

this 699305

br 647122

for 593096

in 562289

my 493105

that 458696

but 401656

you 381556

not 368841

with 363336

have 350361

are 326708

s 320736

was 319120

they 318665

t 314260

as 289605

on 280579

like 269642

so 265314

good 252549

these 247329

30 most popular unigrams among positive reviews and their frequencies:

the 1403403

i 1268739

and 1041617

a 988842

it 827003

to 778328

of 606908

is 583994

this 524404

br 475248

for 464436

in 427834

my 391658

that 330800

you 298452

with 282111

but 276181

have 271712

are 259484

s 244487

they 239904

not 224659

great 219086

as 217169

on 214324

t 213917

was 206673

good 202799

so 202618

these 197166

30 most popular unigrams among negative reviews and their frequencies:

the 524781

i 448448

a 288134

and 283501

it 279223

to 258488

of 215760

this 174901

br 171874

is 169244

not 144182

in 134455

for 128660

that 127896

but 125475

was 112447

my 101447

t 100343

you 83104

with 81225

they 78761

have 78649

s 76249

like 76097

as 72436

are 67224

on 66255

so 62696

be 59167

taste 58095

1. top 30 non stopwords from all reviews and their frequencies:

br 647122

like 269642

good 252549

great 240287

coffee 191977

taste 190799

product 187854

one 182723

flavor 161748

tea 160348

love 155158

food 141829

would 125840

get 111691

best 110358

amazon 109363

really 106742

much 96916

ve 88445

little 87888

also 87277

time 87135

price 87098

use 86056

dog 83550

buy 81924

m 78809

better 78176

tried 77687

even 76682

top 30 non stopwords from positive reviews and their frequencies:

br 475248

great 219086

good 202799

like 193545

coffee 147426

love 138818

one 138624

product 133129

taste 132704

tea 130919

flavor 120831

food 108271

best 101359

amazon 85218

get 84809

would 82506

really 80878

ve 71991

much 70683

price 70352

use 70294

also 70045

little 69421

time 68191

dog 63341

find 63153

well 61320

tried 60077

make 58625

m 58614

top 30 non stopwords from negative reviews and their frequencies:

br 171874

like 76097

taste 58095

product 54725

good 49750

coffee 44551

one 44099

would 43334

flavor 40917

food 33558

tea 29429

get 26882

much 26233

really 25864

amazon 24145

buy 23674

even 22504

great 21201

better 20336

dog 20209

m 20195

time 18944

bad 18866

first 18497

little 18467

box 18086

water 18047

tried 17610

also 17232

eat 17200

1. The entropy function is trivial, while for information gain I simply divided the dataset into two subsets; in one of which the attribute is present, and in the other one its’s not. The sum of their entropy then gets subtracted from the original entropy to get the information gain.
2. The lists positive\_words and negative\_words are merged to the list all\_words, which contains non-duplicate words from these two lists. However I found the way that the decision function was called did not meet my need on going through the decision tree, so I combined its functionality directly into the go function, which recursively calls itself on one of the child.
3. Running the dictionary version of decision\_tree.py and utilizing only 20 levels, we were able to obtain the accuracy, which is 0.82988. For smaller datasets of 1000 or 10,000 reviews, the accuracies obtained using 20 levels are generally greater than those utilizing all levels possible as shallower levels depress overfitting. From the first several selectors (shown below) we can see that scanner did not take into consideration of the tense, plural, third-person singular and abbreviations of the same words. In future versions we may improve on identifying variations of the same word; we may even identify affix, root word, and phrases (e.g., *not sth.*) to achieve higher information gain with less layers.

Top 75 selectors for each split, without level constraint:

great

best

delicious

love

disappointed

excellent

loves

perfect

good

favorite

wonderful

money

yummy

bad

ok

worst

amazing

tasty

awesome

nice

terrible

beware

highly

easy

happy

works

awful

horrible

disappointing

pleased

yum

didn

fantastic

stale

description

okay

would

day

find

loved

smooth

poor

helps

return

nasty

always

china

thank

without

weak

thought

enjoyed

morning

threw

picture

likes

fresh

treat

wrong

label

guess

years

need

box

listed

pieces

bitter

even

buy

delivery

opened

date

expecting

may

healthy

2 Written Exercises

2.

1. Possible values are {0, 1}, each with 0.5 possibility







1. Y is the sum of the values of the two dice rolled















1. 