project3

December 5, 2024

[157]:
```
# Initialize Otter
import otter
grader = otter.Notebook("project3.ipynb")
```

1 Project 3: Movie Classification

Welcome to the third and final project of Data 8! You will build a classification model that guesses whether a movie is a comedy or a thriller by using only the number of times chosen words appear in the movie's screenplay. By the end of the project, you should know how to:

1. Build a k-nearest-neighbors classifier.
2. Test a classifier on data.

1.0.1 Logistics

Deadline. This project is due at **5:00pm PT on Friday, 12/6**. You can receive 5 bonus points for submitting the project by **5:00pm PT on Thursday, 12/5.** Projects will be accepted up to 1 day (24 hours) late. Projects submitted fewer than 24 hours after the deadline will receive 80% credit. Any submissions later than 24 hours after the deadline will not be accepted. It's **much** better to be early than late, so start working now.

Checkpoint. For full credit on the checkpoint, you must complete the questions up to the checkpoint, **pass all *public* autograder tests** for those sections, and submit to the Gradescope Project 3 Checkpoint assignment by **5:00pm PT on Friday, 11/22**. **The checkpoint is worth 5% of your entire project grade**. After you've submitted the checkpoint, you may still change your project answers before the final project deadline - **only your final submission, to the "Project 3" assignment, will be graded for correctness** (including questions from before the checkpoint). You will have some lab time to work on these questions, but we recommend that you start the project before lab and leave time to finish the checkpoint afterward.

Partners. You may work with one other partner; your partner must be from your assigned lab section. **Only one partner should submit the project notebook to Gradescope. If both partners submit, you will be docked 10% of your project grade. On Gradescope, the person who submits should also designate their partner so that both of you receive credit.** Once you submit, click into your submission, and there will be an option to Add Group Member in the top right corner. You may also reference this walkthrough video on how to add partners on Gradescope.

Rules. Don't share your code with anybody but your partner. You are welcome to discuss questions

with other students, but don't share the answers. The experience of solving the problems in this project will prepare you for exams (and life). If someone asks you for the answer, resist! Instead, you can demonstrate how you would solve a similar problem.

Support. You are not alone! Come to office hours, post on Ed, and talk to your classmates. If you want to ask about the details of your solution to a problem, make a private Ed post and the staff will respond. If you're ever feeling overwhelmed or don't know how to make progress, email your TA or tutor for help. You can find contact information for the staff on the course website.

Tests. The tests that are given are **not comprehensive** and passing the tests for a question **does not** mean that you answered the question correctly. Tests usually only check that your table has the correct column labels. However, more tests will be applied to verify the correctness of your submission in order to assign your final score, so be careful and check your work! You might want to create your own checks along the way to see if your answers make sense. Additionally, before you submit, make sure that none of your cells take a very long time to run (several minutes).

Free Response Questions: Make sure that you put the answers to the written questions in the indicated cell we provide. **Every free response question should include an explanation** that adequately answers the question.

Advice. Develop your answers incrementally. To perform a complicated task, break it up into steps, perform each step on a different line, give a new name to each result, and check that each intermediate result is what you expect. You can add any additional names or functions you want to the provided cells. Make sure that you are using distinct and meaningful variable names throughout the notebook. Along that line, **DO NOT** reuse the variable names that we use when we grade your answers.

You **never** have to use just one line in this project or any others. Use intermediate variables and multiple lines as much as you would like!

All of the concepts necessary for this project are found in the textbook. If you are stuck on a particular problem, reading through the relevant textbook section will often help to clarify concepts.

The point breakdown for this assignment is given in the table below: | Category | Points | | — | — | | Autograder (Coding questions) | 82 | | Written | 13 | | Checkpoint | 5 | | **Total** | 100 |

To get started, load `datascience`, `numpy`, and `matplotlib`. Make sure to also run the first cell of this notebook to load `otter`.

[158]:
```
# Run this cell to set up the notebook, but please don't change it.
import numpy as np
import math
import datascience
from datascience import *

# These lines set up the plotting functionality and formatting.
import matplotlib
%matplotlib inline
```

```
import matplotlib.pyplot as plots
plots.style.use('fivethirtyeight')
import warnings
warnings.simplefilter("ignore")
```

2 Part 1: The Dataset

In this project, we are exploring movie screenplays. We'll be trying to predict each movie's genre from the text of its screenplay. In particular, we have compiled a list of 5,000 words that occur in conversations between movie characters. For each movie, our dataset tells us the frequency with which each of these words occurs in certain conversations in its screenplay. All words have been converted to lowercase.

Run the cell below to read the `movies` table. **It may take up to a minute to load.**

[159]:
```
movies = Table.read_table('movies.csv')
```

Here is one row of the table and some of the frequencies of words that were said in the movie.

[160]:
```
movies.where("Title", "runaway bride").select(0, 1, 2, 3, 4, 14, 49, 1042, 4004)
```

[160]:
```
Title         | Year | Rating | Genre  | # Words | breez | england | it        |
bravo
runaway bride | 1999 | 5.2    | comedy | 4895    | 0     | 0       | 0.0234092 |
0
```

The above cell prints a few columns of the row for the comedy movie *Runaway Bride*. The movie contains 4895 words. The word "it" appears 115 times, as it makes up $\frac{115}{4895} \approx 0.0234092$ of the words in the movie. The word "england" doesn't appear at all.

Additional context: This numerical representation of a body of text, one that describes only the frequencies of individual words, is called a bag-of-words representation. This is a model that is often used in NLP. A lot of information is discarded in this representation: the order of the words, the context of each word, who said what, the cast of characters and actors, etc. However, a bag-of-words representation is often used for machine learning applications as a reasonable starting point, because a great deal of information is also retained and expressed in a convenient and compact format.

In this project, we will investigate whether this representation is sufficient to build an accurate genre classifier.

All movie titles are unique. The `row_for_title` function provides fast access to the one row for each title.

Note: All movies in our dataset have their titles lower-cased.

[161]:
```
title_index = movies.index_by('Title')
def row_for_title(title):
    """Return the row for a title, similar to the following expression (but␣
 ↪faster)
```

```
    movies.where('Title', title).row(0)
    """
    return title_index.get(title)[0]

row_for_title('toy story')
```

[161]:
```
Row(Title='toy story', Year='1995', Rating=8.1999999999999993, Genre='comedy', #
Words=3016, she=0.0017427675148135001, decid=0.00034855350296270001,
talk=0.0017427675148135001, wit=0.0, razor=0.0, slam=0.0, credit=0.0, rai=0.0,
hugh=0.0, breez=0.0, conscienc=0.0, audienc=0.0, cathi=0.0, log=0.0, met=0.0,
chosen=0.0, grip=0.0, booz=0.0, bianca=0.0, doubl=0.00034855350296270001,
agent=0.0, exit=0.0, carpent=0.0, underground=0.0, clemenza=0.0, gain=0.0,
neg=0.00069710700592540001, majesti=0.0, studio=0.0, chri=0.0, spin=0.0,
greater=0.0, eaten=0.0, vibrat=0.0, stupid=0.0010456605088881, cigarett=0.0,
jesu=0.0, mani=0.0, violin=0.0, financi=0.00034855350296270001, bai=0.0,
cop=0.0, neighbor=0.0, cd=0.0, england=0.0, made=0.00034855350296270001,
conni=0.0, instinct=0.0, took=0.0, jacquelin=0.0, mace=0.0, disappear=0.0,
waltz=0.0, behind=0.00034855350296270001, bourbon=0.0,
favorit=0.00069710700592540001, benni=0.0, manhattan=0.0, nixon=0.0, lunch=0.0,
principl=0.0, tradit=0.0, counterfeit=0.0, sophi=0.0, third=0.0, exist=0.0,
wouldv=0.00034855350296270001, hero=0.0, theyr=0.0024398745207389002,
anytim=0.0, christin=0.0, vallei=0.0, chess=0.0, paid=0.0, burglar=0.0,
nostril=0.0, rubber=0.0, human=0.0, british=0.0, plissken=0.0, eddi=0.0,
gee=0.0, offend=0.0, rebecca=0.0, anger=0.0, plant=0.0, famou=0.0, repres=0.0,
latest=0.0, rent=0.0, dip=0.0, bell=0.0, andi=0.0069710700592541001,
so=0.0017427675148135001, london=0.0, cooler=0.0, keaton=0.0, portland=0.0,
headlin=0.0, whatta=0.0, fatal=0.0, sew=0.0, cheer=0.0, davi=0.0, feed=0.0,
hudson=0.0, ambros=0.0, digest=0.0, redi=0.0, fri=0.0,
staff=0.00069710700592540001, casino=0.0, occasion=0.0, shadow=0.0,
work=0.00069710700592540001, restrain=0.00034855350296270001, face=0.0,
exercis=0.0, sidnei=0.0, pile=0.0, whyd=0.0, teenag=0.0, her=0.0013942140118508,
retir=0.0, hazard=0.0, roth=0.0, hurrican=0.0, impuls=0.0,
ranger=0.0020913210177761999, pour=0.0, lester=0.0, slash=0.0, deer=0.0,
could=0.0013942140118508, vital=0.0, qualiti=0.0, coma=0.0, incred=0.0,
hank=0.0, famili=0.0, duchess=0.0, global=0.0, virgin=0.0, scientif=0.0,
between=0.0, holidai=0.0, qualifi=0.0, moor=0.0, happili=0.0, arizona=0.0,
non=0.0, bruce=0.0, ankl=0.0, constant=0.0, buzz=0.0163820146392471, harder=0.0,
ing=0.0, christian=0.0, palmer=0.0, tent=0.0, sunset=0.0, damour=0.0,
cohaagen=0.0, advertis=0.0, sensat=0.0, local=0.0, there=0.0080167305681421996,
terri=0.0, sedat=0.0, rotten=0.0, struck=0.0, deck=0.0, past=0.0, bro=0.0,
ann=0.0, dump=0.0, kane=0.0, slot=0.0, immun=0.0, block=0.00034855350296270001,
lil=0.0, technic=0.0, tactic=0.0, pencil=0.0, outsid=0.0, laboratori=0.0,
easi=0.0, nephew=0.0, coffin=0.0, pretti=0.0, coward=0.0, verbal=0.0,
permiss=0.0, bartend=0.0, wont=0.0, watch=0.0013942140118508, lindenmey=0.0,
cosmo=0.0, capabl=0.00034855350296270001, flirt=0.0, huge=0.0, berkelei=0.0,
max=0.0, walter=0.0, lime=0.0, rico=0.0, marvin=0.0, aboard=0.0, bacon=0.0,
```

account=0.0, kirk=0.0, quaid=0.0, stunt=0.00034855350296270001, closet=0.0,
due=0.0, nuclear=0.0, blind=0.0, pussi=0.0, howdi=0.00034855350296270001,
snuff=0.0, eas=0.00034855350296270001, now=0.0076681770651794998, leak=0.0,
underwear=0.0, westlei=0.0, mayb=0.0, theo=0.0, limo=0.0, cousin=0.0, illeg=0.0,
silli=0.0, against=0.0, done=0.00034855350296270001, district=0.0, invad=0.0,
ryan=0.0, wait=0.0034855350296270002, grudg=0.0, charact=0.0, hick=0.0,
jami=0.0, lifetim=0.0, lecktor=0.0, and=0.0083652840711049004, republican=0.0,
life=0.0, hidden=0.0, wire=0.0, paranoia=0.0, network=0.0, messi=0.0,
uthatu=0.0, effort=0.0, carri=0.0, windham=0.0, fun=0.00034855350296270001,
psychologist=0.0, sean=0.0, scent=0.0, answer=0.0, mom=0.0031369815266642999,
wake=0.0, sign=0.0, ho=0.0, relat=0.0, jame=0.0, fat=0.0, myself=0.0,
disrupt=0.0, scan=0.0, vagu=0.0, basket=0.0, christma=0.00069710700592540001,
estim=0.0, em=0.0013942140118508, union=0.0, involv=0.0, norman=0.0,
suspici=0.0, becom=0.0, shoe=0.0, librari=0.0, administr=0.0, ford=0.0,
complic=0.0, stuck=0.00034855350296270001, justic=0.0,
attack=0.00034855350296270001, releas=0.0, econom=0.0, hesit=0.0, autopsi=0.0,
jurisdict=0.0, four=0.0, factor=0.0, inquiri=0.0, lion=0.0, meanwhil=0.0,
prison=0.0, blair=0.0, seri=0.0, groceri=0.0, surgeri=0.0, season=0.0,
christi=0.0, clean=0.0, ow=0.00034855350296270001, wrestl=0.0, en=0.0,
moral=0.0, hungri=0.00034855350296270001, cole=0.0, surfer=0.0,
sixteen=0.00034855350296270001, angl=0.0, shame=0.0,
barrel=0.00034855350296270001, major=0.0, ago=0.0, lott=0.0, airplan=0.0,
worth=0.0, train=0.0, easili=0.0, feller=0.0, valentin=0.0, harvei=0.0,
wherev=0.0, francisco=0.0, true=0.00034855350296270001, dramat=0.0, boston=0.0,
besid=0.0, inspector=0.0, orlean=0.0, opportun=0.0, nearli=0.0, lindsei=0.0,
photograph=0.0, frame=0.0, at=0.0027884280237016001, psychopath=0.0, press=0.0,
youyou=0.00034855350296270001, havana=0.0, australia=0.0,
plai=0.00034855350296270001, mayfield=0.0, chick=0.0, stewart=0.0, seven=0.0,
reflect=0.0, outer=0.0, vega=0.0, anywai=0.00034855350296270001, prime=0.0,
farmer=0.0, backyard=0.0, joe=0.0, otherwis=0.0, cowgirl=0.0, grate=0.0,
clerk=0.0, dispos=0.0, tow=0.0, mari=0.00034855350296270001, certifi=0.0,
thi=0.0108051585918438, wheel=0.0, privaci=0.0, todai=0.00069710700592540001,
nathan=0.0, teller=0.0, plot=0.0, correct=0.0, couch=0.0,
job=0.00069710700592540001, hurt=0.0, inject=0.0, chocol=0.0, session=0.0,
outrag=0.0, reduc=0.0, knew=0.00034855350296270001, jd=0.0, perfum=0.0,
fabric=0.0, bodyguard=0.0, think=0.0041826420355523999, il=0.0, yesterdai=0.0,
side=0.0, doesnt=0.00069710700592540001, ronni=0.0, blank=0.0, jess=0.0,
push=0.00034855350296270001, ahh=0.0, jealou=0.0, alter=0.0, blew=0.0, bu=0.0,
off=0.00034855350296270001, sweetheart=0.0, abl=0.00034855350296270001,
angelo=0.0, nicer=0.0, coupla=0.0, resum=0.00034855350296270001, coke=0.0,
strangl=0.0, gut=0.0, morn=0.00034855350296270001, miracl=0.0,
bit=0.00034855350296270001, intimid=0.0, pipelin=0.0, sour=0.0, shep=0.0,
vivian=0.0, grave=0.0, chemic=0.0, czech=0.0, scholarship=0.0, oldfashion=0.0,
accent=0.0, spitz=0.0, dirti=0.0, shot=0.0, lit=0.0, cedar=0.0, pirat=0.0,
weather=0.0, stun=0.00034855350296270001, learn=0.0, wick=0.0,
bring=0.00069710700592540001, slack=0.0, brave=0.0, shakespear=0.0,
monkei=0.00034855350296270001, presum=0.0, vacat=0.0, faint=0.0,

strap=0.00034855350296270001, stephen=0.0, maggi=0.0, indic=0.0, sundai=0.0,
nois=0.0, organ=0.0, terranc=0.0, foundat=0.0, littl=0.0020913210177761999,
perman=0.00034855350296270001, insid=0.0, stabl=0.0, sharp=0.0, uptight=0.0,
wholl=0.0, jeffrei=0.0, root=0.0, thy=0.0, josi=0.0, woman=0.0,
post=0.00034855350296270001, judg=0.0, ralph=0.0, amaz=0.0, surf=0.0,
naughti=0.0, norm=0.0, glove=0.0, cigar=0.0, wendi=0.0,
corpor=0.00034855350296270001, statement=0.0, defin=0.0, drawn=0.0,
progress=0.0, year.1=0.00034855350296270001, shovel=0.0, sequenc=0.0,
andand=0.0, reel=0.0, held=0.0, youv=0.0013942140118508, trick=0.0,
horseman=0.0, whoa=0.00034855350296270001, emploi=0.0, chain=0.0,
cmon=0.0024398745207389002, brief=0.0, creativ=0.0, moscow=0.0, challeng=0.0,
walli=0.0, golf=0.0, abort=0.0, aha=0.0, bent=0.0, exclus=0.0, amber=0.0,
figur=0.0, healthi=0.0, ransom=0.0, steer=0.0, blow=0.00069710700592540001,
bark=0.0, imbecil=0.0, mother=0.00069710700592540001,
had=0.00069710700592540001, whatev=0.00034855350296270001, donit=0.0,
goodby=0.0, terrifi=0.0, cash=0.0, descript=0.0, spell=0.0, west=0.0,
shoulda=0.0, when=0.0010456605088881, wear=0.0, crop=0.0, trapper=0.0,
donat=0.0, breath=0.0, bracelet=0.0, lover=0.0, afraid=0.0, vice=0.0, ms=0.0,
didnt=0.0, kill=0.00069710700592540001, depth=0.0, si=0.0, experiment=0.0,
dna=0.0, desmond=0.0, coconut=0.0, dil=0.0, llewelyn=0.0, spoil=0.0, lung=0.0,
attent=0.00034855350296270001, offens=0.0, babi=0.0, havin=0.0, mustnt=0.0,
creepi=0.0, daniel=0.0, abandon=0.0, less=0.00034855350296270001,
go=0.0038340885325897, negoti=0.0, butcher=0.0, sudden=0.0, templ=0.0, ii=0.0,
alex=0.0, deal=0.0, rememb=0.0, polic=0.0, gasolin=0.0, luggag=0.0, smooth=0.0,
declar=0.0, chase=0.0, host=0.0, uptown=0.0, heavi=0.0, tenni=0.0, picard=0.0,
success=0.0, as=0.0010456605088881, heroin=0.0, hi=0.0024398745207389002,
seduc=0.0, den=0.0, accur=0.0, parasit=0.0, fiddl=0.0, altern=0.0, chees=0.0,
flatter=0.0, lloyd=0.0, collector=0.0, athlet=0.0, useless=0.0, yeh=0.0,
lawsuit=0.0, guitar=0.0, apart=0.0, strong=0.0, ditch=0.0, doc=0.0,
wors=0.00034855350296270001, trigger=0.0, mister=0.0, request=0.0, direct=0.0,
telephon=0.0, expir=0.0, blond=0.0, energi=0.0, eh=0.00034855350296270001,
damag=0.0, across=0.0, anni=0.0, eleven=0.0, episod=0.0, reliant=0.0, jason=0.0,
get=0.0083652840711049004, duh=0.0, victori=0.0, speci=0.0, includ=0.0,
time=0.0031369815266642999, imagin=0.0, hang=0.00034855350296270001, estat=0.0,
full=0.0, sleep=0.0, receiv=0.0, grei=0.0, belief=0.0, itali=0.0, separ=0.0,
claw=0.0, circuit=0.0, memo=0.0, monei=0.0, anna=0.0, term=0.0, drunk=0.0,
everydai=0.0, haldeman=0.0, frozen=0.0, castro=0.0, soup=0.0, mornin=0.0,
starship=0.0, dawn=0.0, curtain=0.0, pipe=0.0, twist=0.0, downtown=0.0,
drive=0.00034855350296270001, tank=0.0, servant=0.0, circumst=0.0,
town=0.00034855350296270001, beat=0.0, wooden=0.0, thursdai=0.0, ordinari=0.0,
acid=0.0, must=0.00034855350296270001, literatur=0.0, carol=0.0, homework=0.0,
present=0.0017427675148135001, lunat=0.0, lifestyl=0.0, rat=0.0, satisfi=0.0,
funni=0.00069710700592540001, gang=0.0, jewelri=0.0, develop=0.0, yell=0.0,
stash=0.0, ab=0.0, mm=0.0, thumb=0.0, to=0.017079121645172599, lift=0.0, re=0.0,
mud=0.0, taranski=0.0, camel=0.0, cancer=0.0, valiant=0.0, leavin=0.0, sand=0.0,
themselv=0.0, stai=0.0010456605088881, junki=0.0, dinner=0.00069710700592540001,
subject=0.0, ax=0.0, thankyou=0.0, squad=0.0, charl=0.0, pretend=0.0,

mumford=0.0, player=0.0, sorri=0.0013942140118508, feet=0.0, dee=0.0,
casual=0.0, check=0.0, fugit=0.0, garden=0.0, million=0.0,
suppos=0.00034855350296270001, observ=0.0, low=0.0, cathol=0.0, parad=0.0,
confus=0.0, appli=0.0, uhura=0.0, sane=0.0, beverli=0.0, princip=0.0, helen=0.0,
drain=0.0, want=0.0031369815266642999, maam=0.0, care=0.00034855350296270001,
tomorrow=0.00034855350296270001, abus=0.0, parent=0.0, diner=0.0, proud=0.0,
squeez=0.0, allerg=0.0, displai=0.0, rave=0.0, da=0.0, tabl=0.0, kip=0.0,
string=0.00069710700592540001, ik=0.0, microwav=0.0, tongu=0.0, miner=0.0,
montana=0.0, ident=0.0, gaston=0.0, individu=0.0, treasur=0.0, brown=0.0,
delic=0.0, forbidden=0.0, blah=0.0, although=0.0, cold=0.0, strategi=0.0,
psycholog=0.0, fill=0.0, profession=0.00034855350296270001, rig=0.0, theori=0.0,
psychiatr=0.0, raw=0.0, critic=0.0, contribut=0.0, fart=0.0, bitter=0.0,
lemm=0.0, isol=0.0, drug=0.0, also=0.0, type=0.0, oblig=0.0, jen=0.0, feder=0.0,
note=0.0, perhap=0.0, dwayn=0.0, wolf=0.0, central=0.0, annoi=0.0, cycl=0.0,
accid=0.00034855350296270001, puzzl=0.0, invis=0.0, ronald=0.0, mercuri=0.0,
escap=0.0, damon=0.0, acr=0.0, spy=0.0, gig=0.0, armor=0.0,
gotten=0.00034855350296270001, swana=0.0, scene=0.0, marla=0.0, penetr=0.0,
shock=0.0, sunk=0.00034855350296270001, iv=0.00069710700592540001, diari=0.0,
atlant=0.0, absenc=0.0, corps=0.0, relai=0.0, rip=0.0, bull=0.0, requir=0.0,
buck=0.0, complain=0.0, russia=0.0, arctic=0.0, schedul=0.0, casei=0.0,
temperatur=0.0, tree=0.0, unbeliev=0.0, graduat=0.0, place=0.0013942140118508,
communist=0.0, nine=0.0, vulner=0.0, hike=0.0, raymond=0.0, laura=0.0,
manifest=0.0, cellular=0.0, repress=0.0, divid=0.0, lesson=0.0, crate=0.0,
coin=0.0, bachelor=0.0, carl=0.00069710700592540001, sport=0.0, simpli=0.0,
politician=0.0, destroi=0.0, juno=0.0, castor=0.0, liter=0.0, dwight=0.0,
malkovich=0.0, cord=0.0, edmund=0.0, walk=0.0, cake=0.00034855350296270001,
protect=0.0, nearest=0.0, takin=0.0, comrad=0.0, dai=0.00069710700592540001,
tour=0.0, whose=0.0, partial=0.0, vada=0.0, delight=0.0, godfath=0.0, cheat=0.0,
harmless=0.0, romant=0.0, pound=0.0, im=0.0080167305681421996, violat=0.0,
chief=0.0, onlin=0.0, pleasur=0.0, crystal=0.00034855350296270001, luther=0.0,
festiv=0.0, along=0.00034855350296270001, admir=0.0, owner=0.0,
prove=0.00034855350296270001, giron=0.0, lydia=0.0, dad=0.0, ultim=0.0,
hawkin=0.0, hei=0.0059254095503659997, father=0.0, rot=0.0, doll=0.0,
fountain=0.0, williamson=0.0, horror=0.0, ell=0.0, media=0.0, russian=0.0,
carefulli=0.0, verifi=0.0, prepar=0.0, cia=0.0, flew=0.0, logic=0.0, affect=0.0,
skirt=0.0, nineteen=0.0, breakdown=0.0, william=0.0,
batteri=0.00034855350296270001, shore=0.0, project=0.0, strength=0.0, bail=0.0,
piti=0.0, harbor=0.0, oper=0.0, yanke=0.0, orphan=0.0, squar=0.0, gari=0.0,
driven=0.0, nanci=0.0, leonard=0.0, nonsens=0.0, anybodi=0.0, strict=0.0,
riddl=0.0, boulevard=0.0, articl=0.0, shop=0.0, mexico=0.0, ruin=0.0,
did=0.0027884280237016001, annett=0.0, piano=0.0, chest=0.0, becker=0.0,
strip=0.00034855350296270001, stroll=0.0, philosophi=0.0, footbal=0.0, whom=0.0,
flight=0.0, refriger=0.0, loretta=0.0, geniu=0.00069710700592540001, condit=0.0,
for=0.0066225165562914003, anyon=0.00069710700592540001, ideal=0.0, terribl=0.0,
otho=0.0, self=0.0, swore=0.0, ring=0.0, baron=0.0, civilian=0.0, panick=0.0,
settl=0.0, spread=0.0, turn=0.00069710700592540001, blast=0.0, growth=0.0,
seal=0.0, box=0.0, locker=0.0, help=0.0020913210177761999, cast=0.0, jennif=0.0,

regan=0.0, stole=0.0, yessir=0.0, randi=0.0, left=0.0013942140118508,
explain=0.00034855350296270001, inspect=0.0, tribun=0.0, pop=0.0,
bed=0.00069710700592540001, encourag=0.0, bubbl=0.0, contact=0.0, patrick=0.0,
domino=0.0, reliev=0.0, superman=0.0, sector=0.0, entranc=0.0, corner=0.0,
mama=0.0, supposedli=0.0, yo=0.0, role=0.0, wai=0.0024398745207389002,
vicki=0.0, buddi=0.00069710700592540001, wizard=0.0, heal=0.0, element=0.0,
here=0.0069710700592541001, leather=0.0, hardli=0.0,
their=0.00034855350296270001, sworn=0.0, africa=0.0, yearold=0.0, sphere=0.0,
psych=0.0, behalf=0.0, moren=0.0, shelter=0.0, dr=0.0, weight=0.0, ami=0.0,
pike=0.0, gitt=0.0, kinda=0.0, liber=0.0, badg=0.0, last=0.0010456605088881,
from=0.0034855350296270002, uhhuh=0.00034855350296270001, afford=0.0, shag=0.0,
presenc=0.0, jai=0.0, hall=0.0, worst=0.0, agenc=0.0, threat=0.0, jaw=0.0,
hold=0.0010456605088881, someplac=0.0, birth=0.0, junk=0.0, natali=0.0,
graveyard=0.0, march=0.0, focu=0.0, graviti=0.0, smoke=0.0, normal=0.0,
congress=0.0, ash=0.0, predict=0.0, copper=0.0, court=0.0, wealth=0.0,
it=0.0226559776925758, um=0.0, issu=0.0, misunderstand=0.0, berlin=0.0,
structur=0.0, hardwar=0.0, casanova=0.0, worship=0.0, unhappi=0.0, spit=0.0,
quadrant=0.0, downstair=0.00034855350296270001, see=0.0052283025444405003,
gentlemen=0.0, intern=0.0, conrad=0.0, flame=0.0, within=0.0, drum=0.0, yet=0.0,
differ=0.00034855350296270001, jacket=0.0, onto=0.0, prospect=0.0, nation=0.0,
soft=0.0, girl=0.0, counter=0.0, rug=0.0, neutral=0.0, elli=0.0, transmiss=0.0,
integr=0.0, cadillac=0.0, hose=0.0, lillian=0.0, clip=0.0, stanlei=0.0,
marriag=0.0, built=0.0, paso=0.0, subwai=0.0, whiskei=0.0, kyle=0.0,
sai=0.0024398745207389002, barzini=0.0, virginia=0.0, lou=0.0, slip=0.0,
pageant=0.0, tellin=0.0, taylor=0.0, proof=0.0, yank=0.0, yacht=0.0, bless=0.0,
brooklyn=0.0, argu=0.0, pistol=0.0, bo=0.0010456605088881, unnecessari=0.0,
share=0.0, th=0.0, sabotag=0.0, china=0.0, hojon=0.0, intim=0.0,
chop=0.00034855350296270001, comic=0.0, compuls=0.0, bingo=0.0,
make=0.0010456605088881, vulcan=0.0, parri=0.0, shouldnt=0.00069710700592540001,
disconnect=0.0, toon=0.0, forc=0.0, annabel=0.0, clown=0.0, sentenc=0.0,
with=0.0041826420355523999, cartoon=0.0, suspicion=0.0, willi=0.0, brook=0.0,
revers=0.0, luci=0.0, visitor=0.0, wave=0.0, stick=0.0, cuban=0.0, sweep=0.0,
comment=0.0, elimin=0.0, spencer=0.0, monica=0.0, debat=0.0, coron=0.0, lie=0.0,
ourselv=0.0, weapon=0.0, kate=0.0, distant=0.0, grade=0.0, atf=0.0, balloon=0.0,
southern=0.0, chang=0.0, oppos=0.0, indict=0.0, respect=0.0,
sure=0.0038340885325897, jane=0.0, doubt=0.0, princ=0.0, admiss=0.0, ador=0.0,
jungl=0.0, paint=0.0, south=0.0, event=0.0, taxi=0.0, voic=0.0, video=0.0,
convert=0.0, couldnt=0.00034855350296270001, citi=0.0, motiv=0.0, rel=0.0,
herself=0.0, curios=0.0, buffalo=0.0, terrorist=0.0, write=0.0, school=0.0,
wing=0.00034855350296270001, smack=0.0, marti=0.0, counti=0.0, pud=0.0,
thatd=0.0, readi=0.00034855350296270001, manrai=0.0, airlin=0.0, romeo=0.0,
weekend=0.0, apolog=0.0, ladi=0.0, emili=0.0, maker=0.0, dian=0.0, barf=0.0,
quest=0.0, crown=0.0, expect=0.0, gull=0.0, pump=0.00034855350296270001,
depress=0.0, interrog=0.0, instant=0.0, barri=0.0, book=0.0, ahm=0.0,
shutup=0.0, larri=0.0, seriou=0.00034855350296270001, hobbi=0.0, unotu=0.0,
kingdom=0.0, on=0.0101080515859184, quarter=0.0, nicki=0.0, pierr=0.0,
dream=0.0, clarenc=0.0, buffi=0.0, winner=0.0, scatter=0.0, erik=0.0, bike=0.0,

bean=0.0, unabl=0.0, women=0.0, maya=0.0, tommi=0.0, seventyf=0.0, curs=0.0,
maintain=0.0, genesi=0.0, background=0.0, dude=0.0, satellit=0.0,
hour=0.00034855350296270001, extend=0.0, transport=0.00034855350296270001,
hull=0.0, top=0.0, grief=0.0, evid=0.0, gradi=0.0, onc=0.00034855350296270001,
philadelphia=0.0, honesti=0.0, super=0.0, florida=0.0, gentleman=0.0,
cannot=0.0, too=0.0010456605088881, fuss=0.0, hasnt=0.0, lookin=0.0, arrang=0.0,
brill=0.0, sale=0.0, month=0.0, intact=0.0, foolish=0.0, newspap=0.0,
transit=0.0, roi=0.0, whew=0.0, uwhatu=0.0, main=0.0, bat=0.0, line=0.0,
snake=0.0, bald=0.0, cage=0.0, toward=0.0, sold=0.0, till=0.0, mood=0.0,
warn=0.0, johnni=0.0, mustv=0.0, scream=0.0, undress=0.0, absolut=0.0, flaw=0.0,
bake=0.0, mmm=0.0, jerri=0.0, employe=0.0, quot=0.0, waitin=0.0, benefit=0.0,
werent=0.0, fulfil=0.0, plastic=0.00034855350296270001, tribe=0.0, yknow=0.0,
hope=0.00069710700592540001, layer=0.0, tom=0.0, reaction=0.0, san=0.0,
katrina=0.0, identifi=0.0, red=0.0010456605088881, exorc=0.0, iron=0.0,
beer=0.0, santo=0.0, bruis=0.0, stab=0.0, louis=0.0, agreement=0.0, coulda=0.0,
film=0.0, rufu=0.0, soil=0.0, playin=0.0, millionair=0.0, middl=0.0, closer=0.0,
spirit=0.0, accident=0.0, yer=0.0, traffic=0.0, what=0.0139421401185082,
convict=0.0, sack=0.0, examin=0.0, compound=0.0, squid=0.0,
gimm=0.0010456605088881, fault=0.00069710700592540001, mysteri=0.0, sea=0.0,
barbara=0.0, surpris=0.0, rm=0.0, hid=0.0, sewer=0.0, kilomet=0.0, lisa=0.0,
workshop=0.0, safeti=0.0, touch=0.00069710700592540001, jersei=0.0,
frequent=0.0, enjoi=0.0, loos=0.0, homeless=0.0, ah=0.00034855350296270001,
extens=0.0, termin=0.00034855350296270001, level=0.0, violent=0.0, rush=0.0,
coordin=0.0, earli=0.0, wa=0.0055768560474032001, dizzi=0.0, privat=0.0,
gwen=0.0, suicid=0.0, headquart=0.0, educ=0.0, sort=0.0, handsom=0.0,
stella=0.0, ac=0.0, audit=0.0, antiqu=0.0, dot=0.0, technolog=0.0, motion=0.0,
hairi=0.00034855350296270001, site=0.0, student=0.0, up=0.0066225165562914003,
either=0.0, pry=0.0, conspiraci=0.0, basi=0.0, timer=0.0, heat=0.0, lawson=0.0,
hear=0.00034855350296270001, ui=0.0, fool=0.0, mere=0.0,
huh=0.0017427675148135001, donni=0.0, none=0.0, fifth=0.0,
becaus=0.0017427675148135001, lamb=0.0, interpret=0.0, increas=0.0, tower=0.0,
mickei=0.0, colleagu=0.0, confer=0.0, hollow=0.0, leon=0.0, thou=0.0, fran=0.0,
percent=0.0, six=0.0, limp=0.0, arrowai=0.0, explod=0.0, friendli=0.0,
breakfast=0.0, greek=0.0, need=0.0017427675148135001, rome=0.0, beast=0.0,
rehab=0.0, ben=0.0, land=0.0, river=0.0, frank=0.0, smash=0.0, quicker=0.0,
former=0.0, lower=0.0, rap=0.0, nick=0.0, came=0.0, hopeless=0.0, comedian=0.0,
right=0.0045311955385150997, yall=0.0, truli=0.0, not=0.0090623910770303,
soze=0.0, forgotten=0.00034855350296270001, tight=0.0, pro=0.0, sona=0.0,
under=0.00069710700592540001, precis=0.0, center=0.0, stiff=0.0, virtual=0.0,
author=0.0, dry=0.0, theyll=0.00069710700592540001, golden=0.0, final=0.0,
properli=0.0, paper=0.0, older=0.0, serv=0.0, dant=0.0, whistl=0.0, suffer=0.0,
confid=0.0, fraud=0.0, brain=0.0, minu=0.00034855350296270001, twentyf=0.0,
cut=0.0, atmospher=0.0, bid=0.0, curv=0.0, pizza=0.0010456605088881, bench=0.0,
tattoo=0.0, poor=0.0, enid=0.0, pink=0.0, bathroom=0.0,
cramp=0.00034855350296270001, hill=0.0, sight=0.0, patrol=0.0, niec=0.0,
calib=0.0, hafta=0.0, journei=0.0, poster=0.0, thruster=0.0, dela=0.0,
celebr=0.0, myer=0.0, ruth=0.0, suzi=0.0, bunni=0.0, male=0.0, margi=0.0,

bate=0.0, naw=0.0, than=0.00069710700592540001, msieu=0.0, lone=0.0, mole=0.0,
briefcas=0.0, rudi=0.0, excel=0.0, madman=0.0, nazi=0.0, flop=0.0, invent=0.0,
signor=0.0, suggest=0.0, edward=0.0, station=0.0, senat=0.0, amen=0.0, hip=0.0,
price=0.0, awai=0.0017427675148135001, randal=0.0, high=0.0, field=0.0,
spiritu=0.0, tone=0.0, citizen=0.0, stair=0.0, equal=0.0, nor=0.0, sixth=0.0,
gettin=0.00034855350296270001, ground=0.00034855350296270001,
control=0.00034855350296270001, awak=0.0, oak=0.0, enterpris=0.0, slightli=0.0,
lee=0.0, scope=0.0, holli=0.0, hunch=0.0, ethic=0.0, nasti=0.0,
fall=0.0010456605088881, wednesdai=0.0, gulf=0.0, dont=0.0073196235622168,
pictur=0.0, awar=0.0, sponsor=0.0, seattl=0.0, english=0.0, introduc=0.0,
health=0.0, hallucin=0.0, quickli=0.0, jacob=0.0, crew=0.0, cuervo=0.0,
alabama=0.0, teresa=0.0, bain=0.0, precog=0.0, brake=0.0, professor=0.0,
somethin=0.0, formal=0.0, unload=0.0, curiou=0.00034855350296270001, daddi=0.0,
porch=0.0, model=0.0, vehicl=0.0, wisdom=0.0, find=0.0024398745207389002,
ruben=0.0, both=0.0, report=0.0, denver=0.0, helicopt=0.0, complex=0.0,
neednt=0.0, be=0.0052283025444405003, greet=0.0, spock=0.0, song=0.0, bibl=0.0,
lipstick=0.0, stroke=0.00034855350296270001, persuad=0.0, recommend=0.0,
deliveri=0.0, hors=0.0, shut=0.0010456605088881, zone=0.0, bomb=0.0, linda=0.0,
tube=0.0, gross=0.0, order=0.0, dillon=0.0, action=0.00034855350296270001,
spot=0.00069710700592540001, resign=0.0, barn=0.0, bush=0.0, itu=0.0, cry=0.0,
decemb=0.0, manufactur=0.0, satisfact=0.0, fairi=0.0,
excit=0.00069710700592540001, extraordinari=0.0, thoma=0.0, musician=0.0,
black=0.0, mmmmm=0.0, fuel=0.00034855350296270001, board=0.0, joei=0.0,
detail=0.0, stewardess=0.0, hunt=0.0, mum=0.0, preacher=0.0,
shown=0.00034855350296270001, bread=0.0, mind=0.0, saint=0.0,
oclock=0.00034855350296270001, sail=0.0, stranger=0.0, smile=0.0, product=0.0,
rubi=0.0, disabl=0.0, question=0.0, philip=0.0, exampl=0.0, freddi=0.0,
stori=0.00034855350296270001, chariti=0.0, franklin=0.0, routin=0.0, engin=0.0,
lax=0.0, blade=0.0, chew=0.0, buljanoff=0.0, counselor=0.0, houston=0.0,
alright=0.0013942140118508, support=0.0, wheelchair=0.0, goe=0.0, rachel=0.0,
stuff=0.0, neck=0.0, have=0.0031369815266642999, divis=0.0, anniversari=0.0,
diamond=0.0, sparazza=0.0, try=0.00069710700592540001, appar=0.0, joker=0.0,
dentist=0.0, section=0.0, gallagh=0.0, slept=0.0, bank=0.0,
hurri=0.00034855350296270001, dure=0.0, sunni=0.0, ink=0.0, vein=0.0,
immigr=0.0, concentr=0.0, kat=0.0, eager=0.0, don=0.0, somewher=0.0, religi=0.0,
pierc=0.0, bureau=0.0, theyd=0.0, mueller=0.0, familiar=0.0, bonu=0.0,
austrian=0.0, violenc=0.0, produc=0.0, tonight=0.00034855350296270001,
royal=0.0, breakin=0.0, map=0.0, sayin=0.0, discuss=0.0, hed=0.0, strictli=0.0,
led=0.0, mile=0.0, thisll=0.0, dammit=0.0, leav=0.0020913210177761999,
dummi=0.0, reactor=0.0, sauc=0.0, rice=0.0, coupl=0.00034855350296270001,
clever=0.0, choic=0.0, chrissak=0.0, director=0.0, kastl=0.0, borg=0.0, fax=0.0,
brought=0.0, disturb=0.0, poison=0.00034855350296270001, grissom=0.0,
shake=0.00034855350296270001, corrupt=0.0, stall=0.0, sarah=0.0, starter=0.0,
alik=0.0, quiet=0.0013942140118508, entertain=0.0, demonstr=0.0, oxygen=0.0,
asham=0.0, undercov=0.0, beef=0.0, nice=0.0010456605088881, televis=0.0,
oscar=0.0, tourist=0.0, practic=0.0, mount=0.0, swedish=0.0,
compani=0.00034855350296270001, twenti=0.0, shall=0.0, sherman=0.0, daryl=0.0,

forev=0.0, crowd=0.0, tap=0.0, fix=0.0, store=0.0, grail=0.0, vietnam=0.0,
candl=0.0, whoop=0.0, taken=0.00034855350296270001, end=0.0, eleph=0.0,
prefer=0.0, brian=0.0, lamar=0.0, ma=0.0, victoria=0.0, surviv=0.0,
three=0.00069710700592540001, bobbi=0.0, stage=0.0, steed=0.0, macfarlan=0.0,
spider=0.0, trial=0.0, suprem=0.0, awfulli=0.0, foot=0.0,
ar=0.0087138375740675993, thrown=0.0, ask=0.0, cap=0.0, strang=0.0, phoenix=0.0,
boot=0.0, georgia=0.0, replac=0.00069710700592540001, reckon=0.0, divorc=0.0,
arrest=0.0, horni=0.0, grandfath=0.0, problem=0.0, bust=0.00034855350296270001,
pari=0.0, roommat=0.0, consum=0.0, ag=0.0, been=0.00069710700592540001,
spaghetti=0.0, minimum=0.0, car=0.00034855350296270001, recruit=0.0, farm=0.0,
dave=0.0, tape=0.0, regular=0.0, decor=0.0, shirt=0.0, multipl=0.0, mechan=0.0,
effici=0.0, all=0.0048797490414778004, thiev=0.0, pose=0.0, silenc=0.0,
lenni=0.00034855350296270001, radiat=0.0, doesn=0.0,
worri=0.0020913210177761999, garrison=0.0, bound=0.0, superior=0.0, cure=0.0,
belli=0.0, mmmm=0.0, of=0.0080167305681421996, bride=0.0,
hole=0.00034855350296270001, toler=0.0, content=0.0, applic=0.0, frequenc=0.0,
sore=0.0, ranch=0.0, fair=0.0, nowher=0.0, monitor=0.0, peanut=0.0, presid=0.0,
cooper=0.0, speech=0.0, canyon=0.0, humili=0.0, primari=0.0, anchor=0.0,
everi=0.00034855350296270001, fiance=0.0, temporari=0.0, nyah=0.0,
greenleaf=0.0, marvel=0.0, enough=0.0013942140118508, extra=0.0, breast=0.0,
properti=0.0, social=0.0, hug=0.00034855350296270001, tempt=0.0, dracula=0.0,
richard=0.0, rob=0.0, guinea=0.0, wanna=0.0, hate=0.00034855350296270001,
dash=0.0, pull=0.0010456605088881, lab=0.0, disast=0.0, lobbi=0.0, plug=0.0,
rear=0.0, conveni=0.0, bounti=0.0, auggi=0.0, thee=0.0, proposit=0.0, jeep=0.0,
pee=0.0, josephin=0.0, gestur=0.0, profil=0.0, mimi=0.0, infect=0.0,
coincid=0.0, mel=0.0, knock=0.00034855350296270001,
search=0.00069710700592540001, pan=0.0, if=0.0024398745207389002, odd=0.0,
jenni=0.0, howd=0.0, simon=0.0, dodg=0.0, outfit=0.0, floor=0.0, uhh=0.0,
nah=0.0, jimmi=0.0, chuck=0.0, rich=0.0, emot=0.0, destruct=0.0, blue=0.0,
game=0.00034855350296270001, peculiar=0.0, internet=0.0, paranoid=0.0,
paradis=0.0, deposit=0.0, look=0.0062739630533287004, count=0.0, ooz=0.0,
gale=0.0, plumb=0.0, shouldv=0.0, giant=0.0, sooz=0.0, gino=0.0, waitress=0.0,
conceiv=0.0, messag=0.0, door=0.00034855350296270001, mondai=0.0, mayor=0.0,
fanci=0.0, cuba=0.0, most=0.0, barrier=0.0, eject=0.0, grew=0.0, alic=0.0,
jazz=0.0, thread=0.0, commun=0.0, treati=0.0, lieuten=0.0, tran=0.0,
mackelwai=0.0, colonel=0.0, vincent=0.0, pant=0.0, simpl=0.0, shave=0.0,
snoop=0.0, flower=0.0, barton=0.0, trailer=0.0, towel=0.0, sheldon=0.0,
comput=0.0, medic=0.0, hit=0.0, grow=0.0, ventur=0.0, mistaken=0.0,
difficulti=0.0, thirtyf=0.0, pm=0.0, peter=0.0, orang=0.0, kei=0.0, earl=0.0,
ex=0.0, vault=0.0, doe=0.00069710700592540001, music=0.0, romanc=0.0,
blanket=0.0, colleg=0.0, fog=0.0, bad=0.00069710700592540001, novel=0.0,
kidnap=0.0, consult=0.0, recogn=0.0, laugh=0.0, shooter=0.0,
galaxi=0.00034855350296270001, paulin=0.0, procedur=0.0, seed=0.0, radioact=0.0,
ars=0.0, total=0.0, theater=0.0, cute=0.0, iraq=0.0, net=0.0, laplant=0.0,
waiter=0.0, print=0.0, milo=0.0, judgment=0.0, act=0.00034855350296270001,
gone=0.00069710700592540001, lecter=0.0, draft=0.0,
build=0.00034855350296270001, vow=0.0, steam=0.0, veteran=0.0, eighth=0.0,

insect=0.0, entir=0.00034855350296270001, hon=0.0, okai=0.0024398745207389002,
fourteen=0.0, meet=0.00069710700592540001, tens=0.00034855350296270001,
doin=0.00034855350296270001, hat=0.0017427675148135001, sub=0.0, counsel=0.0,
channel=0.0, close=0.00069710700592540001, daylight=0.0, loser=0.0, tend=0.0,
judgement=0.0, accus=0.0, coast=0.0, everywher=0.00034855350296270001,
welli=0.0, yep=0.0, label=0.0, chines=0.0, kim=0.0, rather=0.0, publish=0.0,
neat=0.0, askin=0.0, pearl=0.0, bond=0.00034855350296270001, labor=0.0,
cover=0.0, forget=0.0, warrior=0.0, asid=0.0, favor=0.0,
sheep=0.00034855350296270001, crook=0.0, imag=0.0, wimp=0.0, coach=0.0,
seller=0.0, impact=0.0, shoot=0.00034855350296270001, soda=0.0, mitch=0.0,
western=0.0, gift=0.0, shred=0.0, compromis=0.0, fridai=0.0, scout=0.0,
steve=0.0, solar=0.0, evan=0.0, rule=0.0, dylan=0.0, han=0.0, faith=0.0,
umyu=0.0, wise=0.0, tortur=0.00034855350296270001, girlfriend=0.0, anywher=0.0,
air=0.0, murder=0.0010456605088881, younger=0.0, handl=0.0, albanian=0.0,
laser=0.0010456605088881, dull=0.0, guidanc=0.0, live=0.0, dish=0.0, dancer=0.0,
heh=0.0020913210177761999, samuel=0.0, lawn=0.0, defeat=0.0, photon=0.0,
mason=0.0, goal=0.0, prescott=0.0, insur=0.0, deep=0.0, lugosi=0.0, singl=0.0,
chancellor=0.0, avail=0.0, acknowledg=0.0, dread=0.0, manual=0.0, ll=0.0,
stew=0.0, pai=0.00034855350296270001, crawford=0.0, janet=0.0, loan=0.0,
rib=0.0, peel=0.0, hormon=0.0, about=0.0027884280237016001, ohio=0.0,
advisor=0.0, crusher=0.0, centuri=0.0, govern=0.0, probabl=0.0, slow=0.0,
homicid=0.0, shift=0.0, sentiment=0.0, sophist=0.0, mock=0.00034855350296270001,
supper=0.0, washington=0.0, inn=0.0, polici=0.0, call=0.0, surgic=0.0,
talent=0.0, futur=0.0, appl=0.0, hiya=0.0, should=0.00034855350296270001,
hoop=0.0, joseph=0.0, uisu=0.0, unit=0.0, radar=0.0, magnific=0.0, rendezv=0.0,
anxiou=0.0, featur=0.0, discov=0.00034855350296270001, dental=0.0, legal=0.0,
shuttl=0.0, word=0.0010456605088881, tool=0.0, deceas=0.0, consider=0.0,
noon=0.0, york=0.0, lauri=0.0, fell=0.0, reject=0.00034855350296270001, ton=0.0,
lead=0.0, marin=0.0, name=0.0, commiss=0.0, windshield=0.0, captain=0.0,
winter=0.0, slap=0.0, burst=0.0, mallori=0.0, goodnight=0.0, polish=0.0,
letter=0.0, voyag=0.0, sandi=0.0, ta=0.0, commission=0.0, joke=0.0,
idea=0.00069710700592540001, deserv=0.0, shove=0.0, rd=0.0,
mistak=0.0010456605088881, earthquak=0.0, worm=0.0, abil=0.0, guarante=0.0,
outta=0.00069710700592540001, mutual=0.0, prize=0.00034855350296270001,
welcom=0.0, seventi=0.0, trunk=0.0, teas=0.0, access=0.0, jacki=0.0,
refresh=0.0, affirm=0.0, ill=0.0017427675148135001, gregor=0.0, chet=0.0,
complet=0.0, mill=0.0, speak=0.0, whatiya=0.0, prei=0.0, fed=0.0, flash=0.0,
crab=0.0, wet=0.0, warm=0.0, hoover=0.0, flood=0.0, regard=0.0, hamburg=0.0,
anim=0.0, random=0.0, lick=0.0, liabl=0.0, unknown=0.0, histori=0.0, highli=0.0,
fred=0.0, peggi=0.0, emerg=0.0, maid=0.0, relax=0.0, whaddya=0.0,
give=0.0013942140118508, starfleet=0.0, rout=0.0, resourc=0.0, night=0.0,
statu=0.0, spare=0.0, abov=0.0, save=0.0010456605088881, fett=0.0, goat=0.0,
career=0.0, van=0.00069710700592540001, puke=0.0, disagre=0.0,
reveng=0.00034855350296270001, femal=0.0, penni=0.0, better=0.0010456605088881,
rex=0.00069710700592540001, marg=0.0, but=0.0041826420355523999, border=0.0,
phil=0.0, twentyseven=0.0, sixti=0.0, delai=0.0, jewel=0.0, uiu=0.0, leash=0.0,
berni=0.0, nickel=0.0, aim=0.00034855350296270001, uyouu=0.0, advic=0.0,

gold=0.0, actress=0.0, whisper=0.0, goin=0.0, particularli=0.0, gossip=0.0,
runnin=0.0, peni=0.0, minut=0.0020913210177761999, scandal=0.0, awkward=0.0,
crank=0.0, ev=0.0, owen=0.0, diet=0.0, cargo=0.00034855350296270001,
fashion=0.0, tale=0.0, quietli=0.0, son=0.00034855350296270001, geek=0.0,
devot=0.0, madelein=0.0, luke=0.0, laval=0.0, seventeen=0.0, yourselv=0.0,
sleev=0.0, robberi=0.0, debt=0.0, flow=0.0, mark=0.0, subtl=0.0, symbol=0.0,
prank=0.0, illus=0.0, restor=0.0, catch=0.00034855350296270001, predat=0.0,
temper=0.0, globe=0.0, pacif=0.0, ethel=0.0, phaser=0.0, midnight=0.0, lock=0.0,
adult=0.0, brandon=0.0, kitchen=0.0, noi=0.0, toni=0.0, mo=0.0, awhil=0.0,
actor=0.0, length=0.0, pinch=0.0, throw=0.0, tunnel=0.0, materi=0.0,
function=0.0, zip=0.0, visa=0.0, josh=0.0, shed=0.0, crash=0.0, craig=0.0,
he=0.0066225165562914003, perri=0.0, pathet=0.0, eastern=0.0, demand=0.0,
sheet=0.0, data=0.0, fbi=0.0, insist=0.0, regist=0.0, fame=0.0, particular=0.0,
upon=0.0, screen=0.0, becki=0.0, ya=0.0031369815266642999, attornei=0.0,
clai=0.0, mate=0.0, interview=0.0, yard=0.0, duke=0.0, footprint=0.0, user=0.0,
influenc=0.0, swallow=0.0, lula=0.0, plate=0.0, queen=0.0, until=0.0, rode=0.0,
creat=0.0, disappoint=0.0, region=0.0, marcia=0.0, defens=0.0, sensit=0.0,
defend=0.00034855350296270001, miseri=0.0, take=0.0024398745207389002, oil=0.0,
holi=0.0, polit=0.0, crazi=0.0, interrupt=0.0, luca=0.0, franc=0.0, pete=0.0,
afternoon=0.0, distress=0.0, finder=0.0, around=0.0017427675148135001,
europ=0.0, sit=0.00034855350296270001, follow=0.0, intellectu=0.0, tee=0.0,
planet=0.0017427675148135001, perk=0.0, dame=0.0, wife=0.0, distanc=0.0,
breed=0.0, height=0.0, compens=0.0, storm=0.0, appoint=0.0, daphn=0.0,
salesman=0.0, scoobi=0.0, distinct=0.0, alon=0.0, remot=0.0, boss=0.0, pawn=0.0,
wonder=0.0, easier=0.0, lt=0.0, zero=0.0, stake=0.0,
food=0.00034855350296270001, forest=0.0, june=0.0, pickup=0.0, sensor=0.0,
choke=0.0, situat=0.0, david=0.0, blackmail=0.0, atom=0.0, necessarili=0.0,
albert=0.0, advantag=0.0, fli=0.0, horn=0.0, metal=0.0, consequ=0.0, sloan=0.0,
tragic=0.0, determin=0.0, meter=0.0, toast=0.0, disguis=0.0, loud=0.0,
employ=0.0, xxxxxx=0.0, cat=0.0, math=0.0, sue=0.0, foul=0.0,
like=0.0013942140118508, ferri=0.0, praetor=0.0, marri=0.0, stamp=0.0,
lainei=0.0, gai=0.0, potato=0.0013942140118508, licens=0.0,
some=0.0010456605088881, sylvia=0.0, tag=0.0, caught=0.0, runner=0.0,
unpleas=0.0, typic=0.0, dive=0.0, devic=0.00034855350296270001,
alien=0.00034855350296270001, deadlin=0.0, landlord=0.0, bridg=0.0, sink=0.0,
affair=0.0, record=0.0, thinkin=0.0, gate=0.0, austin=0.0, poker=0.0,
liquor=0.0, cow=0.0, run=0.0, yourself=0.00069710700592540001,
tie=0.00034855350296270001, slightest=0.0, businessman=0.0, martini=0.0,
dell=0.0, stone=0.0, iti=0.0, dri=0.0, scanner=0.0, effect=0.0, appreci=0.0,
shark=0.0, child=0.00069710700592540001, bend=0.0, slice=0.0, hair=0.0,
patienc=0.0, ga=0.00034855350296270001, disco=0.0, expert=0.0, implant=0.0,
dealer=0.0, oooh=0.0, beavi=0.0, troubl=0.00069710700592540001, lesli=0.0,
lectur=0.0, youd=0.00069710700592540001, baldwin=0.0, scienc=0.0, surg=0.0,
uallu=0.0, ooh=0.0, harvard=0.0, smart=0.0, statist=0.0, sheila=0.0, haul=0.0,
livingston=0.0, exhaust=0.0, jail=0.0, biggest=0.0, hudsuck=0.0,
rocket=0.00069710700592540001, trace=0.0, punish=0.0, contain=0.0, twomblei=0.0,
intend=0.0, flynn=0.0, describ=0.0, contract=0.0, straighten=0.0, reilli=0.0,

regula=0.0, tremend=0.0, love=0.0010456605088881, setup=0.0, enforc=0.0,
castl=0.0, lousi=0.0, bloke=0.0, marietta=0.0, cooki=0.0,
someon=0.00034855350296270001, design=0.0, address=0.0, knowi=0.0, award=0.0,
tick=0.0, unconsci=0.0, weve=0.0020913210177761999, cleveland=0.0, sweeti=0.0,
intellect=0.0, beard=0.0, surrend=0.0, vinci=0.0, hildi=0.0, tail=0.0,
dignan=0.0, moss=0.0, photo=0.0, myth=0.0, tuesdai=0.0, stress=0.0, limb=0.0,
wallet=0.0, tooth=0.0, bare=0.0, ten=0.0, guilti=0.00034855350296270001,
inform=0.0, list=0.0, husband=0.0, visual=0.0, ross=0.0, jabez=0.0, cape=0.0,
atlanta=0.0, champion=0.0, valuabl=0.0, respons=0.0, addit=0.0, spike=0.0,
lighten=0.0, wagon=0.0, owl=0.0, dog=0.00034855350296270001, santa=0.0,
fantasi=0.0, belt=0.0, unowu=0.0, enorm=0.0, kind=0.00069710700592540001,
light=0.00034855350296270001, investig=0.0, lila=0.0, laundri=0.0, jessica=0.0,
jodi=0.0, masturb=0.0, porter=0.0, la=0.0, couldn=0.0, canada=0.0, dress=0.0,
port=0.00034855350296270001, alwai=0.00034855350296270001, explor=0.0,
bateman=0.0, governor=0.0, sat=0.0, fought=0.0, perimet=0.0,
wasnt=0.00034855350296270001, testimoni=0.0, conklin=0.0, cart=0.0, caus=0.0,
number=0.0, crane=0.0, loyalti=0.0, fake=0.0, puff=0.0, accompani=0.0,
cultur=0.0, fenc=0.0, outa=0.0, cal=0.0, whenev=0.0, roach=0.0, transmitt=0.0,
fingerprint=0.0, mail=0.00034855350296270001, kaufman=0.0, pole=0.0, engag=0.0,
plain=0.0, suppli=0.00034855350296270001, reserv=0.0, few=0.0, sandwich=0.0,
fredo=0.0, safe=0.0, hockei=0.0, hous=0.0027884280237016001, elain=0.0,
knox=0.0, oath=0.0, wander=0.0, latin=0.0, same=0.0, git=0.0, garbag=0.0,
charg=0.0, knowledg=0.0, uknowu=0.0, veronica=0.0, john=0.0, connel=0.0,
anticip=0.0, drown=0.0, rose=0.0, carpet=0.0, circu=0.0, sebastian=0.0,
cmere=0.0, irish=0.0, robert=0.0, fork=0.0, lazi=0.0, flare=0.0, bobo=0.0,
guess=0.0, hippi=0.0, the=0.0240501917044267, specif=0.0, offici=0.0, god=0.0,
exhibit=0.0, madam=0.0, privileg=0.0, scrambl=0.0, con=0.0, handi=0.0, rifl=0.0,
peep=0.0, moron=0.0, resent=0.0, lai=0.0, scotch=0.0, brenner=0.0, lake=0.0,
therel=0.0, gordo=0.0, everett=0.0, neural=0.0, madison=0.0, realiz=0.0,
georg=0.0, skate=0.0, drink=0.0, goddam=0.0, anyhow=0.0, jew=0.0, popul=0.0,
freedom=0.0, togeth=0.0, attempt=0.0, adel=0.0, coat=0.0, mob=0.0, higher=0.0,
cowboi=0.00069710700592540001, heel=0.0, inconveni=0.0, respond=0.0, patch=0.0,
mostli=0.0, fan=0.0, highwai=0.0, reason=0.0, soon=0.0,
helpless=0.00034855350296270001, then=0.0020913210177761999, cobb=0.0,
weird=0.0, parker=0.0, seen=0.0017427675148135001, happier=0.0, roast=0.0,
tip=0.0, legend=0.0, behavior=0.0, secur=0.00034855350296270001, tatum=0.0,
especi=0.0, jeez=0.0, your=0.011850819100731999, dispatch=0.0, heart=0.0,
test=0.0, webster=0.0, categori=0.0, chicken=0.0, alibi=0.0, wouldn=0.0,
hyster=0.0, mankind=0.0, north=0.0, kiss=0.0, mighti=0.0, levi=0.0, fond=0.0,
rain=0.0, obsess=0.0, michael=0.0, corleon=0.0, vision=0.0, edg=0.0,
command=0.00069710700592540001, nineti=0.0, gin=0.0, nix=0.0, amus=0.0,
afterward=0.0, wanta=0.0, thorwald=0.0, by=0.00069710700592540001, rag=0.0,
jack=0.0, desert=0.0, maureen=0.0, own=0.0, heck=0.00034855350296270001,
walker=0.0, empti=0.0, grunemann=0.0, relief=0.0, narrow=0.0, saavik=0.0,
seymour=0.0, anonym=0.0, footag=0.0, territori=0.0, maniac=0.0, envelop=0.0,
despis=0.0, definit=0.0, return=0.00034855350296270001, nobodi=0.0, eventu=0.0,
rita=0.0, pepper=0.0, prep=0.0, stand=0.00034855350296270001, deepli=0.0,

park=0.0, aunt=0.0, youth=0.0, water=0.00034855350296270001,
els=0.00034855350296270001, starl=0.0, allei=0.0, snow=0.0, pope=0.0, wax=0.0,
object=0.0, basic=0.0, file=0.0, dismiss=0.0, much=0.0017427675148135001,
sometim=0.0, warrant=0.0, drama=0.0, android=0.0, wast=0.0, brick=0.0,
restless=0.0, unusu=0.0, orbit=0.0, headach=0.0, ambul=0.0, pier=0.0,
talkin=0.0, instal=0.0, toss=0.00034855350296270001, pure=0.0, maxin=0.0,
canadian=0.0, amount=0.0, shatter=0.0, spoke=0.0, danc=0.0, ir=0.0, proven=0.0,
broke=0.0, ninotchka=0.0, lili=0.0, dat=0.0, scum=0.0, psychic=0.0,
guest=0.00034855350296270001, architect=0.0, vancouv=0.0, balconi=0.0,
my=0.0062739630533287004, resist=0.0, juic=0.0, reliabl=0.0,
will=0.0024398745207389002, launch=0.0, tire=0.00034855350296270001, villag=0.0,
ordel=0.0, broad=0.0, juri=0.00034855350296270001, mccoi=0.0, italian=0.0,
item=0.0, process=0.0, sake=0.0, diseas=0.0, kelli=0.0, contest=0.0,
champagn=0.0, cabinet=0.0, plead=0.0, era=0.0, sergeant=0.0,
nervou=0.00034855350296270001, imposs=0.0, cruel=0.0, claud=0.0, movi=0.0,
cruis=0.0, daughter=0.0, premier=0.0, law=0.0, intent=0.0, liz=0.0, nake=0.0,
petti=0.0, mirror=0.0, ha=0.0034855350296270002, each=0.0,
never=0.00069710700592540001, do=0.0059254095503659997, exact=0.0, rescu=0.0,
drivin=0.0, limit=0.0, ti=0.0, manag=0.0, thirsti=0.0, bargain=0.0,
origin=0.00034855350296270001, acceler=0.0, pilot=0.0, guid=0.0, mummi=0.0,
mad=0.0, frighten=0.0, factori=0.0, cotton=0.0, rape=0.0, chef=0.0,
leg=0.00034855350296270001, calm=0.00069710700592540001, fail=0.0, interior=0.0,
germ=0.0, inch=0.0, donald=0.0, smear=0.0, berserk=0.0, finest=0.0,
hard=0.00069710700592540001, bourn=0.0, brilliant=0.0, solv=0.0, fireman=0.0,
electr=0.0, moment=0.0, wound=0.0, begun=0.0, while=0.0, militari=0.0,
scoop=0.0, saw=0.00034855350296270001, machin=0.0, german=0.0, enemi=0.0,
creatur=0.0, throat=0.0, stuf=0.0, impli=0.0, burnt=0.0, mall=0.0, navi=0.0,
rude=0.0, lean=0.0, bullet=0.0, sissi=0.0, dough=0.0, reynold=0.0,
promot=0.00034855350296270001, burn=0.0, rise=0.0, guard=0.0, deton=0.0,
gloriou=0.0, confidenti=0.0, lost=0.00034855350296270001, desk=0.0, thin=0.0,
superhero=0.0, wive=0.0, pooch=0.0, col=0.0, gambl=0.0, crimin=0.0,
can=0.0048797490414778004, tiger=0.0, pair=0.0, below=0.0,
him=0.0045311955385150997, un=0.0, mortgag=0.0, toto=0.0, dine=0.0, gun=0.0,
thirti=0.0, warp=0.0, sid=0.0024398745207389002, standard=0.0, deliv=0.0,
couldv=0.0, sandra=0.0, wide=0.0, daydai=0.0, nail=0.0, novemb=0.0, speck=0.0,
broken=0.0, fulli=0.0, di=0.0, concept=0.0, plenti=0.0, sara=0.0, toilet=0.0,
smyth=0.0, dirt=0.0, skull=0.0, match=0.00034855350296270001, win=0.0,
abduct=0.0, chamber=0.0, putter=0.0, advanc=0.0, baxter=0.0, group=0.0,
onli=0.0, commerci=0.0, uniqu=0.0, bet=0.00034855350296270001,
back=0.0041826420355523999, aisl=0.0, darlin=0.0, creation=0.0, target=0.0,
sip=0.0, sam=0.0, worker=0.0, five=0.00069710700592540001, ladder=0.0,
capit=0.0, smaller=0.00034855350296270001, merri=0.00034855350296270001,
won=0.0, howard=0.0, eleg=0.0, bee=0.0, digit=0.0, lowel=0.0, outstand=0.0,
meant=0.00034855350296270001, precaut=0.0, degre=0.0, suspend=0.0, week=0.0,
evacu=0.0, wow=0.00069710700592540001, loui=0.0, iii=0.0, per=0.0, tenth=0.0,
open=0.0010456605088881, possibl=0.00034855350296270001,
old=0.00034855350296270001, primit=0.0, keyser=0.0, fianc=0.0, point=0.0,

bone=0.0, alreadi=0.0, straight=0.0, henri=0.0, hack=0.0, rank=0.0, scott=0.0,
fold=0.0, nothin=0.0, kendal=0.0, bar=0.0, ant=0.0, meredith=0.0, musta=0.0,
dollar=0.0, retard=0.0, volum=0.0, ocean=0.0, acquaint=0.0, jon=0.0, dozen=0.0,
treat=0.0, fund=0.0, crush=0.0, accomplish=0.00034855350296270001, cabl=0.0,
stock=0.0, pig=0.0, recent=0.0, ed=0.0, escort=0.0, east=0.0, vodka=0.0,
natur=0.0, fella=0.00034855350296270001, missil=0.0, ancient=0.0,
how=0.0020913210177761999, rumor=0.0, torn=0.0, palei=0.0,
well=0.0059254095503659997, fear=0.0, detroit=0.0, civil=0.0, current=0.0,
advis=0.0, elect=0.0, pill=0.00034855350296270001, artist=0.0, bastaldi=0.0,
market=0.0, symptom=0.0, therapi=0.0, toi=0.0080167305681421996, scumbag=0.0,
satan=0.0, ouch=0.0, jeff=0.0, cemeteri=0.0, sorta=0.0, deni=0.0,
sister=0.00034855350296270001, givin=0.0, boi=0.00069710700592540001,
closest=0.0, mission=0.00034855350296270001, stabil=0.0, identif=0.0,
set=0.00034855350296270001, rebel=0.0, salli=0.0, joint=0.0, zoo=0.0,
version=0.0, camper=0.0, directli=0.0, kid=0.00069710700592540001, jim=0.0,
viru=0.0, desir=0.0, tide=0.0, cell=0.0, mulwrai=0.0, sod=0.0, domini=0.0,
dealt=0.0, pane=0.0, trade=0.0, spose=0.0, eleanor=0.0,
someth=0.0010456605088881, milk=0.0, honestli=0.00034855350296270001, sap=0.0,
graham=0.0, tune=0.0, secretari=0.0, chao=0.0, backup=0.0, freez=0.0,
obvious=0.00034855350296270001, believ=0.00069710700592540001, delmar=0.0,
sting=0.0, happiest=0.0, broadcast=0.0, haunt=0.0, haven=0.0, kong=0.0,
gonna=0.00069710700592540001, without=0.00034855350296270001, lip=0.0,
spill=0.0, psychiatrist=0.0, suddenli=0.00034855350296270001,
realli=0.00069710700592540001, byeby=0.0, coffe=0.0, opinion=0.0, borrow=0.0,
lechter=0.0, cuff=0.0, overload=0.0, attract=0.0, rid=0.0, bertrand=0.0,
mordechai=0.0, amanda=0.0, nicknam=0.0, gather=0.00034855350296270001,
mr=0.0013942140118508, miss=0.00069710700592540001, fascin=0.0, hound=0.0,
lid=0.0, joi=0.0, weak=0.0, though=0.00034855350296270001, alfr=0.0,
comfort=0.0, ich=0.0, gallon=0.0, narcot=0.0, evelyn=0.0, duck=0.0, bucket=0.0,
cabin=0.0, settlement=0.0, send=0.00034855350296270001, milli=0.0, israel=0.0,
alcohol=0.0, suspect=0.0, robinson=0.0, doug=0.0, alan=0.0, sleepi=0.0,
adventur=0.0, ridicul=0.0, crabtre=0.0, lawyer=0.0, reward=0.0, cloth=0.0,
riot=0.0, seek=0.0, other=0.00069710700592540001, pet=0.0,
everyth=0.0010456605088881, found=0.00034855350296270001, basketbal=0.0,
liar=0.00034855350296270001, wouldnt=0.0, fit=0.0, stop=0.0020913210177761999,
cleaner=0.0, paul=0.0, execut=0.0, alpha=0.00034855350296270001, movement=0.0,
cheek=0.0, recov=0.0, phillip=0.0, seem=0.0, leo=0.0, rand=0.0, bright=0.0,
moon=0.0, fallen=0.0, prayer=0.0, betti=0.0, fantast=0.0,
theyv=0.00034855350296270001, drove=0.0, turkei=0.0, modern=0.0, edi=0.0,
jonah=0.0, exagger=0.0, mine=0.00034855350296270001, envi=0.0,
big=0.0013942140118508, krueger=0.0, faster=0.0, alli=0.0,
fine=0.00034855350296270001, iraqi=0.0, appear=0.0, independ=0.0, poni=0.0,
invest=0.0, hooker=0.0, dictat=0.0, medicin=0.0, honei=0.0013942140118508,
shine=0.0, comedi=0.0, unou=0.0, who=0.0024398745207389002, late=0.0, boost=0.0,
gentli=0.0, sun=0.0, dunno=0.00034855350296270001, improv=0.0, chairman=0.0,
honest=0.0, shout=0.0, invas=0.0, hadnt=0.00069710700592540001, fox=0.0,
powder=0.0, quick=0.00034855350296270001, breaker=0.0, breach=0.0, dope=0.0,

frustrat=0.0, cent=0.0, long=0.00069710700592540001, interest=0.0, freak=0.0,
marsh=0.0, maria=0.0, shhhh=0.0, cloak=0.0, solid=0.0,
id.1=0.00069710700592540001, nobl=0.0, swing=0.0, evalu=0.0, member=0.0,
skunk=0.0, zira=0.0, conduct=0.0, buyer=0.0, instanc=0.0, financ=0.0,
protest=0.0, karl=0.0, dent=0.0, jake=0.0, goofi=0.0, au=0.0,
plu=0.00034855350296270001, ahead=0.0, possess=0.0, prototyp=0.0, whatd=0.0,
road=0.0, fifti=0.0, down=0.0024398745207389002, viciou=0.0, destin=0.0,
ride=0.00034855350296270001, hmm=0.0, democrat=0.0, daili=0.0, remov=0.0,
catherin=0.0, unlock=0.0, hot=0.0, yuh=0.0, lewi=0.0,
concern=0.00034855350296270001, mouth=0.0, didn=0.0, merci=0.0, lotta=0.0,
sacrific=0.0, chekov=0.0, rock=0.0, instead=0.0, egon=0.0, sank=0.0, probe=0.0,
pit=0.0, rest=0.0, confront=0.00034855350296270001, magic=0.0,
mean=0.0017427675148135001, similar=0.0, itd=0.0, flesh=0.0, bunch=0.0,
donut=0.0, soldier=0.00034855350296270001, dork=0.0,
short=0.00034855350296270001, innoc=0.0, fight=0.00034855350296270001, grai=0.0,
bout=0.00034855350296270001, befor=0.0017427675148135001, encount=0.0,
lombardo=0.0, arriv=0.0, date=0.0, worn=0.0, frankenstein=0.0, budget=0.0,
gag=0.0, part=0.00069710700592540001, forgot=0.0, blake=0.0, himself=0.0,
facil=0.0, thousand=0.0, frankli=0.0, whatcha=0.0, struggl=0.0, lad=0.0,
a=0.022307424189613099, resid=0.0, salad=0.0, told=0.00034855350296270001,
brother=0.0, rmph=0.0, which=0.00034855350296270001, philosoph=0.0,
clear=0.00034855350296270001, cracker=0.0, zavitz=0.0,
tell=0.0034855350296270002, conclus=0.0, collin=0.0,
shhh=0.00034855350296270001, tickl=0.0, delus=0.0, ship=0.00034855350296270001,
band=0.0, naiv=0.0, sever=0.0, wreck=0.0, torch=0.0, gym=0.0, bang=0.0,
man=0.00034855350296270001, aid=0.0, purchas=0.00034855350296270001, review=0.0,
lantern=0.0, new=0.0017427675148135001, nunez=0.0, wish=0.0, nest=0.0,
fortun=0.0, bud=0.0, excus=0.00069710700592540001, rattl=0.0, knight=0.0,
broadwai=0.0, destini=0.0, smell=0.0, rick=0.0, mommi=0.0, klingon=0.0,
staci=0.0, calcul=0.0, signific=0.0, arrog=0.0, mac=0.0, offic=0.0, pitch=0.0,
detect=0.0, hood=0.0, relationship=0.0, sunshin=0.0, arlyn=0.0, gentl=0.0,
heard=0.0, babe=0.0, beg=0.0, princess=0.0, memphi=0.0, addict=0.0, incid=0.0,
reput=0.0, forth=0.0, club=0.0, theme=0.0, anthoni=0.0, corn=0.0, suitcas=0.0,
went=0.0, worthi=0.0, jessi=0.0, announc=0.0, panic=0.00069710700592540001,
pleasant=0.0, bath=0.0, pauli=0.0, sampl=0.0, policeman=0.0, co=0.0, sonni=0.0,
gave=0.0, weigh=0.0, attic=0.00034855350296270001, incident=0.0, quarantin=0.0,
got=0.0045311955385150997, napkin=0.0, chair=0.0, grandmoth=0.0, theft=0.0,
disord=0.0, blame=0.0, stronger=0.0, perform=0.00034855350296270001, absurd=0.0,
mitchel=0.0, cab=0.0, mutant=0.0, corbett=0.0, travi=0.0, erica=0.0,
pressur=0.0, larg=0.0, toddi=0.0, gum=0.0, trap=0.00034855350296270001,
twelv=0.0, treatment=0.0, xrai=0.0, callin=0.0, rope=0.0, japan=0.0, pot=0.0,
method=0.0, enter=0.0, confess=0.0, killer=0.0, constantli=0.0, scar=0.0,
concert=0.0, thirteen=0.0, debbi=0.0, knot=0.0, pension=0.0,
bob=0.00034855350296270001, home=0.00034855350296270001, walt=0.0, troi=0.0,
argument=0.0, desper=0.0, jeremi=0.0, epp=0.0, havent=0.00034855350296270001,
ugh=0.0, dewei=0.0, dumb=0.0, deputi=0.0, phone=0.0, worf=0.0, spook=0.0,
avenu=0.0, hm=0.0, keep=0.0, happen=0.00034855350296270001, plane=0.0,

cartel=0.0, brush=0.0, betrai=0.0, geez=0.0, spark=0.0, tryin=0.0, injur=0.0,
lauren=0.0, jam=0.0, jill=0.0, mikei=0.0, cocain=0.0, drop=0.0, harri=0.0,
paus=0.0, space=0.0020913210177761999, sound=0.0, julia=0.0, warehous=0.0,
remark=0.0, rum=0.0, snap=0.0, meantim=0.0, candi=0.0, proper=0.0, realist=0.0,
arthur=0.0, sh=0.0, survivor=0.0, teeth=0.0, de=0.0, sens=0.0, maneuv=0.0,
poetri=0.0, overnight=0.0, bow=0.0, dalla=0.0, discharg=0.0, bill=0.0,
fly=0.0027884280237016001, realiti=0.0, compar=0.0, upstair=0.0, pritchett=0.0,
terrif=0.0, anoth=0.0010456605088881, kiddin=0.0, unfair=0.0, brace=0.0,
chimera=0.0, feather=0.0, regul=0.0, hump=0.0, thank=0.0024398745207389002,
matur=0.0, valu=0.0, even=0.0013942140118508, shoulder=0.0, booth=0.0,
ribbon=0.0, ceas=0.0, base=0.0, dan=0.0, smokei=0.0, coup=0.0, thick=0.0,
bui=0.0, stud=0.0, scenario=0.0, laid=0.0, mexican=0.0, prior=0.0, bra=0.0,
leader=0.0, charli=0.0, fairli=0.0, nurs=0.0, ahhh=0.00034855350296270001,
texa=0.0, lui=0.0, except=0.0, august=0.0, sittin=0.0, bait=0.0, power=0.0,
kansa=0.0, scotti=0.0, swipe=0.0, videotap=0.0, opposit=0.0, expand=0.0,
neighborhood=0.0, out=0.0062739630533287004, melt=0.0, difficult=0.0, crow=0.0,
occup=0.0, swine=0.0, expens=0.0, brad=0.0, balanc=0.0, tension=0.0,
smarter=0.0, team=0.0, salari=0.0, expos=0.0, incom=0.0,
special=0.00069710700592540001, thea=0.0, needl=0.0, aggress=0.0, necessari=0.0,
burk=0.0, uniform=0.0, crawl=0.0, hundr=0.0, radio=0.0,
into=0.00069710700592540001, singer=0.0, explan=0.0, sollozzo=0.0, beauti=0.0,
whole=0.00069710700592540001, attach=0.0, bleed=0.0, card=0.0,
longer=0.00034855350296270001, good=0.0027884280237016001, draw=0.0, rate=0.0,
enlighten=0.0, startin=0.0, demon=0.0, street=0.0, thatll=0.0, convers=0.0,
nerv=0.00034855350296270001, telegram=0.0, therer=0.0, stan=0.0, momma=0.0,
nightmar=0.0, combat=0.00034855350296270001, brand=0.0,
greatest=0.00034855350296270001, color=0.00034855350296270001, sourc=0.0,
swap=0.0, preciou=0.0, greas=0.0, manipul=0.0, blood=0.0, batman=0.0,
hitler=0.0, visit=0.0, india=0.0, poke=0.0, baltimor=0.0, tear=0.0, fate=0.0,
cours=0.00069710700592540001, stu=0.0, vampir=0.0, vessel=0.0,
creep=0.00034855350296270001, margaret=0.0, bigger=0.00034855350296270001,
error=0.0, repair=0.00034855350296270001, grab=0.00034855350296270001, nope=0.0,
period=0.0, construct=0.0, email=0.0, shinzon=0.0, activ=0.0, bradi=0.0,
sprai=0.0, steel=0.0, shack=0.0, til=0.00034855350296270001,
woodi=0.0097594980829556997, roman=0.0, risk=0.0, peach=0.0, instruct=0.0,
plagu=0.0, provid=0.0, grandma=0.0, tough=0.00034855350296270001, schmuck=0.0,
souvenir=0.0, stain=0.0, wine=0.0, carla=0.0, mckenna=0.0, rexroth=0.0,
spend=0.0, i=0.027187173231090999, susi=0.0, brodi=0.0,
far=0.00034855350296270001, sweater=0.0, appeal=0.0, vibe=0.0, poet=0.0, oz=0.0,
finch=0.0, busi=0.00034855350296270001, perfect=0.00034855350296270001,
boyfriend=0.0, lemon=0.0, hand=0.00034855350296270001, charm=0.0, sustain=0.0,
retain=0.0, continu=0.0, hammer=0.0, ripper=0.0, personnel=0.0, beaten=0.0,
remind=0.0, joan=0.0, witch=0.0, second=0.0, impress=0.00069710700592540001,
hint=0.0, avoid=0.0, billi=0.0, wipe=0.0, calvin=0.0, shrink=0.0, darn=0.0,
is=0.014987800627396301, show=0.00034855350296270001, first=0.0013942140118508,
intellig=0.0, kit=0.0, collar=0.0, bundi=0.0, feel=0.0, martha=0.0,
fortyeight=0.0, franki=0.0, rehears=0.0, reunion=0.0, previou=0.0, fire=0.0,

specimen=0.0, exactli=0.0, gotta=0.0013942140118508, claric=0.0,
you=0.041826420355524703, psychot=0.0, liberti=0.0, ever=0.0, indian=0.0,
karen=0.0, ear=0.0013942140118508, uncl=0.0, mar=0.0, surround=0.0, wood=0.0,
airport=0.0, jean=0.0, prescript=0.0, occupi=0.0, slave=0.0, yup=0.0, cri=0.0,
threw=0.0, extort=0.0, cheap=0.0, melvin=0.0, strain=0.0, solut=0.0, skye=0.0,
packag=0.00034855350296270001, young=0.0, agre=0.0, purs=0.0, puppet=0.0,
clearli=0.00069710700592540001, ellen=0.0, serious=0.0, applejack=0.0,
threaten=0.0, swear=0.00034855350296270001, journalist=0.0, beth=0.0,
feelin=0.0, danni=0.0, hmmm=0.0, stomach=0.0, certif=0.0,
them=0.0013942140118508, wigand=0.0, electron=0.0, surfac=0.0, op=0.0, edit=0.0,
backward=0.00034855350296270001, buckaroo=0.0, fare=0.0, mhm=0.0, cynic=0.0,
fee=0.0, blown=0.0, workin=0.0, eric=0.0, gear=0.0, chill=0.0, public=0.0,
elizabeth=0.0, dy=0.0, great=0.0017427675148135001, load=0.0, melani=0.0,
unless=0.0, mora=0.0, nativ=0.0, gene=0.0, wind=0.0,
code=0.00034855350296270001, bribe=0.0, dieter=0.0, boil=0.0, deliber=0.0,
ritual=0.0, dolor=0.0, pride=0.0, molli=0.00069710700592540001, felt=0.0,
oswald=0.0, drank=0.0, el=0.0, soap=0.0, suck=0.0,
psycho=0.00034855350296270001, prom=0.0, dig=0.0, sword=0.0, exchang=0.0,
duti=0.0, parti=0.00034855350296270001, juliet=0.0, seventh=0.0, given=0.0,
hawk=0.0, fry=0.0, thei=0.0010456605088881, vinc=0.0, notion=0.0, consid=0.0,
upset=0.0, ve=0.0, yeah=0.0041826420355523999, compet=0.0, ticket=0.0, doom=0.0,
ski=0.0, courthous=0.0, alert=0.0, tan=0.0, nam=0.0,
posit=0.00069710700592540001, urgent=0.0, tast=0.0, makeup=0.0, parol=0.0,
schwartz=0.0, doctor=0.0, aw=0.0, christ=0.0, scari=0.0, walkin=0.0, wrist=0.0,
cattl=0.0, truck=0.00069710700592540001, eighteen=0.0, ethan=0.0,
idiot=0.0010456605088881, hotel=0.0, hook=0.0, notic=0.0, heali=0.0,
benjamin=0.0, octob=0.0, ought=0.0, isnt=0.0010456605088881, drew=0.0,
whether=0.0, seat=0.0, matter=0.00069710700592540001, er=0.0, int=0.0,
arent=0.0013942140118508, aubrei=0.0, lookout=0.0, gabe=0.0, cliff=0.0,
hail=0.0, horribl=0.0, heaven=0.0, hospit=0.0, whatsoev=0.0, supernatur=0.0,
lot=0.00034855350296270001, establish=0.00034855350296270001, select=0.0,
motor=0.0, brenda=0.0, guilt=0.0, glori=0.0, move=0.0031369815266642999,
rage=0.0, decent=0.0, religion=0.0, pillow=0.0, sincer=0.0, steven=0.0,
winston=0.0, arm=0.0, system=0.0, score=0.0, isn=0.0, wallac=0.0, lane=0.0,
burger=0.0, bowl=0.0, umeu=0.0, journal=0.0, lonnegan=0.0, teddi=0.0, pd=0.0,
trooper=0.0, puppi=0.0, prosecut=0.0, badli=0.0, thought=0.0013942140118508,
troop=0.00034855350296270001, forti=0.0, highest=0.0, dust=0.0, begin=0.0,
lapd=0.0, priest=0.0, turk=0.0, hawaii=0.0, amateur=0.0, honor=0.0, ap=0.0,
lodg=0.00034855350296270001, lombard=0.0, academi=0.00034855350296270001,
gibson=0.0, injuri=0.0, wha=0.0, therefor=0.0, victor=0.0, profess=0.0,
strike=0.0, deed=0.0, grand=0.0, penthous=0.0, yellow=0.0, insult=0.0,
genuin=0.0, ward=0.0, mi=0.0, stolen=0.0, banana=0.0,
speed=0.00069710700592540001, piec=0.0, born=0.0, oughta=0.0, hacker=0.0,
stream=0.0, smith=0.0, choir=0.0, variou=0.0, weed=0.0, boundari=0.0, panti=0.0,
accord=0.00034855350296270001, best=0.00034855350296270001, condom=0.0,
whatll=0.0, carlo=0.0, promis=0.00034855350296270001, barrett=0.0, pentagon=0.0,
sooner=0.0, nexu=0.0, bloodi=0.0, said=0.0, mississippi=0.0, legitim=0.0,

syndrom=0.0, repli=0.0, purpl=0.0, break=0.00069710700592540001, wash=0.0,
motel=0.0, wrap=0.0, fifteen=0.0, sell=0.0, bravo=0.00034855350296270001,
world=0.0, beaumont=0.0, switch=0.0, adrian=0.0, after=0.0, vanish=0.0,
window=0.0010456605088881, interfer=0.0, countri=0.0, wretch=0.0, sugar=0.0,
captur=0.0, wed=0.0, adrenalin=0.0, dorothi=0.0, jude=0.0, gorgeou=0.0,
corneliu=0.0, flip=0.0, beyond=0.0010456605088881, mix=0.0, bandit=0.0,
rabbit=0.0, rare=0.0, cecil=0.0, belong=0.0, collaps=0.0, resort=0.0, cuz=0.0,
trevor=0.0, lovebird=0.0, express=0.0, dutch=0.0, scare=0.00069710700592540001,
kept=0.0, stephani=0.0, signatur=0.0, warren=0.0, battl=0.0, insight=0.0,
next=0.00069710700592540001, lamp=0.0, trail=0.0, kent=0.0, forg=0.0, ran=0.0,
proce=0.0, shade=0.0, asylum=0.0, kubelik=0.0, vallen=0.0,
in=0.0069710700592541001, everyon=0.0020913210177761999, kidnei=0.0, hostil=0.0,
pie=0.0, written=0.0, bye=0.00069710700592540001, refer=0.0,
explos=0.00034855350296270001, tucker=0.0, swayzak=0.0, lord=0.0,
room=0.00034855350296270001, split=0.0, udou=0.0, whack=0.0, lo=0.0, drawer=0.0,
wild=0.0, hire=0.0, among=0.0, disgust=0.00034855350296270001, pocket=0.0,
margo=0.0, ignor=0.0, goodlook=0.0, claim=0.0, approach=0.0, fruit=0.0,
raid=0.0, phrase=0.0, luck=0.0, kevin=0.0, autograph=0.0, core=0.0,
download=0.0, evil=0.00034855350296270001, strand=0.0, woulda=0.0, branch=0.0,
ghost=0.0, pack=0.00069710700592540001, robber=0.0, dime=0.0, angri=0.0, ju=0.0,
uyouru=0.0, drill=0.0, preserv=0.0, serial=0.0, option=0.0, oldest=0.0,
pattern=0.0, herb=0.0, row=0.0, ball=0.0, equip=0.0, drift=0.0, mike=0.0,
tyler=0.0, choos=0.0, nag=0.0, bookstor=0.0, happi=0.0010456605088881,
haircut=0.0, candid=0.0, tim=0.0, friend=0.00069710700592540001, sneak=0.0,
those=0.00034855350296270001, wayn=0.0, straw=0.0, cynthia=0.0, ad=0.0,
two=0.00069710700592540001, eyebal=0.0, carv=0.0, overlook=0.0, contractor=0.0,
languag=0.0, fortyf=0.0, admit=0.0, slight=0.0, popular=0.0, compliment=0.0,
del=0.0, listen=0.00069710700592540001, mess=0.0, free=0.0, keen=0.0,
noth=0.0017427675148135001, acquir=0.0, earlier=0.0, bluff=0.0, twentyfour=0.0,
steak=0.0, california=0.0, dial=0.0, industri=0.0, hopefulli=0.0, probli=0.0,
certainli=0.0, magnet=0.0, mobil=0.0, elvi=0.0, parlor=0.0, utah=0.0,
we=0.0062739630533287004, phoni=0.0, sec=0.0, fellow=0.0, eagl=0.0, achiev=0.0,
denni=0.0, head=0.0017427675148135001, bore=0.0, dyou=0.0, junior=0.0, bite=0.0,
through=0.00034855350296270001, grenad=0.0, phase=0.0, fabul=0.0, diplomat=0.0,
wrong=0.00034855350296270001, itll=0.0, again=0.0, dock=0.0, bear=0.0,
minor=0.0, white=0.0, environ=0.0, patient=0.00034855350296270001, mug=0.0,
lighthous=0.0, upsid=0.0, nasal=0.0, navig=0.0, sad=0.0, research=0.0, case=0.0,
gu=0.0, brutal=0.0, aint=0.0010456605088881, sinc=0.00034855350296270001,
nichola=0.0, intrud=0.0, dare=0.0, were=0.0059254095503659997,
import=0.00069710700592540001, pi=0.0, babysit=0.0, race=0.0,
oh=0.0139421401185082, flag=0.0, shaw=0.0, teacher=0.0, pimp=0.0, diego=0.0,
victim=0.0, anymor=0.0, round=0.0, rough=0.0, occas=0.0, sulu=0.0, woke=0.0,
chose=0.0, convent=0.0, prioriti=0.0, treadston=0.0, drag=0.0, maud=0.0,
monster=0.0, ic=0.00034855350296270001, or=0.0010456605088881, gener=0.0,
penguin=0.0, earth=0.0, dc=0.0, cave=0.0, fleet=0.0, wrote=0.0, goddammit=0.0,
venic=0.0, wade=0.0, spoken=0.0, unfortun=0.0, campaign=0.0, fever=0.0,
marcu=0.0, almost=0.00034855350296270001, these=0.00069710700592540001,

sick=0.00034855350296270001, reach=0.00034855350296270001, screw=0.0, grace=0.0,
anyth=0.0, empir=0.0, rack=0.0, nell=0.0, divin=0.0, violet=0.0, sing=0.0,
friendship=0.0, loomi=0.0, basebal=0.0, stood=0.0, clark=0.0, bottl=0.0,
marylin=0.0, sin=0.0, juli=0.0, halfwai=0.0, spanish=0.0,
hannah=0.00034855350296270001, shower=0.0, murphi=0.0,
pal=0.00034855350296270001, why=0.0020913210177761999, leagu=0.0,
our=0.00034855350296270001, die=0.0, glad=0.00034855350296270001, depart=0.0,
mightv=0.0, deborah=0.0, spent=0.0, heather=0.0, grant=0.0, delici=0.0,
bloom=0.0, halloween=0.0, selfish=0.0, mack=0.0, pleas=0.00034855350296270001,
later=0.00034855350296270001, tomb=0.0, veri=0.0010456605088881, al=0.0,
filthi=0.0, bein=0.0, cream=0.00034855350296270001, invit=0.0, ay=0.0,
mountain=0.0, tax=0.0, liquid=0.0, suffici=0.0, start=0.0, particl=0.0,
taught=0.0, worthless=0.0, caitlin=0.0, paperwork=0.0, duffi=0.0, fade=0.0,
lothar=0.0, courag=0.0, puls=0.0, fighter=0.0, galleri=0.0, restaur=0.0,
sky=0.00034855350296270001, hollywood=0.0, accept=0.0, trip=0.0, grid=0.0,
vigo=0.0, cellar=0.0, urg=0.0, recal=0.0, task=0.0, nose=0.00034855350296270001,
circul=0.0, sperm=0.0, experi=0.0, rust=0.0, thirtyseven=0.0, lame=0.0,
editor=0.0, univers=0.00034855350296270001, children=0.0, slug=0.0, ration=0.0,
adam=0.0, mailbox=0.0, twentytwo=0.0, reveal=0.0, eat=0.00034855350296270001,
tobacco=0.0, schuyler=0.0, widow=0.0, forgiv=0.0, punch=0.0, transfer=0.0,
humor=0.0, thrill=0.0, jewish=0.0, elbow=0.0, teach=0.0, flush=0.0, pace=0.0,
copi=0.0, locat=0.0, senior=0.0, mantan=0.0, auto=0.0, chemistri=0.0, bolt=0.0,
steadi=0.0, kick=0.0, ego=0.0, underneath=0.0, approv=0.0, cloud=0.0, studi=0.0,
hunter=0.0, clue=0.0, roof=0.0, french=0.0, billion=0.0, streak=0.0,
chauncei=0.0, judi=0.0, cost=0.0, campbel=0.0, contrari=0.0, bought=0.0,
negro=0.0, seein=0.0, area=0.00034855350296270001, testifi=0.0, remain=0.0,
summer=0.00034855350296270001, dunbar=0.0, racket=0.0, no=0.018124782154060701,
decis=0.0, gardin=0.0, miami=0.0, dark=0.00034855350296270001,
bird=0.00069710700592540001, reverend=0.0, loyal=0.0, clair=0.0, master=0.0,
finger=0.0, stink=0.0, expedit=0.0, skip=0.0, pardon=0.00034855350296270001,
jone=0.0, fals=0.0, toe=0.0, gal=0.0, might=0.0, approxim=0.0, norvil=0.0,
sheriff=0.0013942140118508, behav=0.0, punk=0.0, uareu=0.0, inde=0.0,
maximum=0.0, vernon=0.0, hide=0.0, hampshir=0.0, put=0.0, shh=0.0, program=0.0,
plan=0.0, scratch=0.0, ban=0.0, adopt=0.0, ninth=0.0, gui=0.00034855350296270002,
physic=0.0, enhanc=0.0, dead=0.00034855350296270001, bounc=0.0, perspect=0.0,
column=0.0, refus=0.0, knife=0.00034855350296270001, dragon=0.0, caesar=0.0,
spacecraft=0.0, step=0.0, genet=0.0, bowler=0.0, driver=0.0, result=0.0,
bishop=0.0, crime=0.0, ram=0.0, howr=0.0, st=0.0, memori=0.0, dana=0.0,
upper=0.0, front=0.0, twice=0.0, susan=0.0, whoever=0.0,
know=0.0045311955385150997, actual=0.0013942140118508, war=0.0, egg=0.0,
johnson=0.0, island=0.0, pervert=0.0, muscl=0.0, darryl=0.0, umm=0.0,
fourth=0.0, fresh=0.0, recept=0.0, crack=0.0, vote=0.0, betcha=0.0, skywir=0.0,
donovan=0.0, societi=0.0, uncomfort=0.0, commit=0.0, mai=0.0, april=0.0,
purpos=0.0, stare=0.0, common=0.0, campu=0.0, mask=0.0, opera=0.0, kennedi=0.0,
bree=0.0, acm=0.0, dug=0.0, climb=0.00069710700592540001, chicago=0.0,
ey=0.00069710700592540001, pinta=0.0, lack=0.0, hung=0.0,
camp=0.00034855350296270001, clinic=0.0, spat=0.0, uthisu=0.0, sock=0.0,

dawson=0.0, museum=0.0, vacuum=0.0, leap=0.0, romulan=0.0, soviet=0.0,
freewai=0.0, half=0.0, profit=0.0, iri=0.0, real=0.00034855350296270001,
client=0.0, an=0.0017427675148135001, dynamit=0.0, eugen=0.0, nuke=0.0,
robin=0.0, rub=0.0, gloria=0.0, becam=0.0, deceiv=0.0, distract=0.0,
matthew=0.0, ken=0.0, where=0.0048797490414778004, america=0.0, bela=0.0,
often=0.0, sox=0.0, dear=0.0, partner=0.0, devil=0.0, harm=0.0, shotgun=0.0,
adjust=0.0, scheme=0.0, let=0.0048797490414778004, trash=0.0, sherri=0.0,
script=0.0, join=0.0, clock=0.0, allison=0.0, confirm=0.0, clau=0.0, probat=0.0,
rhyme=0.0, whale=0.0, nsa=0.0, gruner=0.0, somedai=0.0, tripl=0.0, forward=0.0,
pool=0.0, flat=0.0, sole=0.0, american=0.0, pentang=0.0, grown=0.0,
frederick=0.0, perfectli=0.0, potenti=0.0, mccaffrei=0.0, form=0.0, nap=0.0,
slide=0.0, elev=0.0, thelma=0.0, fink=0.0, pumpkin=0.0, lilli=0.0, farewel=0.0,
swann=0.0, meal=0.0, fergu=0.0, mention=0.0, cant=0.0062739630533287004,
pain=0.00034855350296270001, tri=0.00034855350296270001,
ok=0.00034855350296270001, crisi=0.0, jose=0.0, would=0.0017427675148135001,
beach=0.0, size=0.0, angel=0.0, cancel=0.00034855350296270001, tuck=0.0,
cook=0.0, began=0.0, pregnanc=0.0, king=0.0, finish=0.0, automat=0.0,
overwhelm=0.0, fast=0.0, intens=0.0, translat=0.0, sucker=0.0, etern=0.0,
still=0.0013942140118508, beneath=0.0, funer=0.0, immedi=0.0, prai=0.0,
theatr=0.0, ceremoni=0.0, pregnant=0.0, capac=0.0, dya=0.0, killain=0.0,
maroon=0.0, sammi=0.0, gordon=0.0, bother=0.0, hah=0.0, cartman=0.0, crude=0.0,
bedroom=0.00034855350296270001, mustang=0.0, offer=0.0, path=0.0, doyl=0.0,
viktor=0.0, iim=0.0, buff=0.0, disk=0.0, birthdai=0.00069710700592540001,
kilo=0.0, lap=0.0, am=0.0010456605088881, poem=0.0, least=0.0, pick=0.0,
pad=0.0, slowli=0.0, garag=0.0, person=0.0, permit=0.0, further=0.0,
connect=0.0, surgeon=0.0, freezer=0.0, obviou=0.0,
dinosaur=0.00034855350296270001, audrei=0.0, furnitur=0.0, scientist=0.0,
roll=0.0, collect=0.0, chat=0.0, attend=0.0, document=0.0, chapter=0.0,
death=0.00034855350296270001, venkman=0.0, fingernail=0.0, sweet=0.0, ol=0.0,
stalk=0.0, salt=0.0, tobi=0.0, camera=0.0, entri=0.0, congratul=0.0, inspir=0.0,
jackson=0.0, simul=0.0, friedman=0.0, pale=0.0, kathryn=0.0,
somebodi=0.00034855350296270001, splendid=0.0, eighti=0.0, eras=0.0,
digniti=0.0, swim=0.0, ly=0.00069710700592540001, star=0.00069710700592540001,
ye=0.0024398745207389002, shy=0.0, pin=0.0, outpost=0.0, rais=0.0, yah=0.0,
leopard=0.0, exposur=0.0, puttin=0.00034855350296270001, martin=0.0, chip=0.0,
somehow=0.0, jaeger=0.0, silent=0.0, fact=0.0, ni=0.0, tub=0.0, bag=0.0,
initi=0.0, buri=0.0, council=0.0, andrew=0.0, brazil=0.0, neil=0.0, suit=0.0,
aliv=0.0, us=0.0038340885325897, gabriel=0.0, expertis=0.0,
secret=0.00034855350296270001, cherri=0.0, crippl=0.0, cross=0.0, silver=0.0,
thief=0.0, whip=0.0, insan=0.0, riplei=0.0, foreign=0.0, ted=0.0, glimps=0.0,
lebowski=0.0, me=0.0076681770651794998, peopl=0.00034855350296270001,
koessler=0.0, minist=0.0, kristen=0.0, faze=0.0, payment=0.0, dean=0.0,
regret=0.0, histor=0.0, slaughter=0.0, near=0.0, popcorn=0.0, japanes=0.0,
fuse=0.0, lucki=0.0, mous=0.0, wherer=0.0, powel=0.0, vanessa=0.0, inherit=0.0,
meat=0.0, sang=0.0, bert=0.0, scale=0.0, scam=0.0, servic=0.0, porno=0.0,
spring=0.0, storag=0.0, small=0.0, thing=0.0017427675148135001, tragedi=0.0,
makin=0.0, just=0.0076681770651794998, costum=0.0, averag=0.0, experienc=0.0,

```
click=0.0, lose=0.00034855350296270001, folk=0.0, danger=0.0, mayflow=0.0,
more=0.0010456605088881, view=0.0, jasper=0.0, hostag=0.0, swamp=0.0, cup=0.0,
hatch=0.0, echo=0.0, cindi=0.0, buddyboi=0.0, depend=0.0, rang=0.0, harold=0.0,
instrument=0.0, deeper=0.0, beam=0.0, fish=0.0, writer=0.0, stir=0.0,
penelop=0.0, react=0.0, knive=0.0, mafia=0.0, carter=0.0, whod=0.0, lincoln=0.0,
swell=0.0, boat=0.0, certain=0.0, jet=0.0, steal=0.0,
safer=0.00034855350296270001, larger=0.0, magazin=0.0, edgar=0.0,
pittsburgh=0.0, committe=0.0, truth=0.0, green=0.0, over=0.0024398745207389002,
prevent=0.0, impati=0.0, tv=0.0, ian=0.0, alarm=0.0, boom=0.0, tripp=0.0,
basement=0.0, torpedo=0.0, such=0.00034855350296270001, sir=0.0013942140118508,
whatr=0.0, jump=0.0, shari=0.0, kenni=0.0, oti=0.0, sweat=0.0, spine=0.0,
everybodi=0.00069710700592540001, salon=0.0, asian=0.0, miser=0.0, church=0.0,
trust=0.0, peac=0.0, surveil=0.0, understand=0.00069710700592540001, picnic=0.0,
skipper=0.0, bodi=0.0, itself=0.0, mental=0.0, come=0.0048797490414778004,
shortli=0.0, bizarr=0.0, evolv=0.0, tea=0.00034855350296270001, known=0.0,
unlik=0.0, arni=0.0, whistler=0.0, firm=0.0, occur=0.0, grandpa=0.0,
traitor=0.0, that=0.015336354130359, gosh=0.0, wynant=0.0, hopkin=0.0,
cough=0.0, butch=0.0, agn=0.0, wichita=0.0, tall=0.0, sent=0.0, manner=0.0,
artifici=0.0, dumper=0.0, palac=0.0, track=0.0, twin=0.0, darl=0.0, stark=0.0,
despit=0.0, alvi=0.0, pen=0.0, wore=0.0, ani=0.0010456605088881, marshal=0.0,
deaf=0.0, stretch=0.0, chopper=0.0, coloni=0.0, shaft=0.0, pat=0.0, hal=0.0,
allow=0.0, knee=0.0, ugli=0.0, combin=0.0, sheldrak=0.0, wall=0.0,
complaint=0.00034855350296270001, understood=0.0, varieti=0.0, roger=0.0,
page=0.0, alphabet=0.0, men=0.00034855350296270001, palm=0.0, terror=0.0,
li=0.0, skill=0.0, childhood=0.0, wilder=0.0, elliot=0.0, specialist=0.0,
travel=0.00034855350296270001, howev=0.0, exploit=0.0, quit=0.0010456605088881,
mph=0.0, traci=0.0, extrem=0.0, torranc=0.0, fiction=0.0, measur=0.0, bug=0.0,
lawrenc=0.0, pendergast=0.0, cb=0.0, art=0.0, convinc=0.0, float=0.0, hop=0.0,
armi=0.00034855350296270001, socal=0.0, sailor=0.0, circl=0.0, pursu=0.0,
notifi=0.0, creasi=0.0, sum=0.0, restrict=0.0, shield=0.00034855350296270001,
rocco=0.0, asleep=0.0, failur=0.0, memor=0.0, mon=0.0, cool=0.0010456605088881,
emma=0.0, shape=0.0, state=0.0, discount=0.0, youll=0.00069710700592540001,
repeat=0.00069710700592540001, neither=0.0, chart=0.0, soul=0.0, dose=0.0,
kai=0.0, wilson=0.0, valet=0.0, tore=0.0, sacr=0.0, rorschach=0.0, custom=0.0,
imit=0.0, rekal=0.0, railroad=0.0, monsieur=0.0, read=0.0, lighter=0.0,
utterli=0.0, loss=0.0, appropri=0.0, veget=0.0, oklahoma=0.0, oliv=0.0,
habit=0.0, slick=0.0, intuit=0.0, rod=0.0, dedic=0.0, propos=0.0, corridor=0.0,
nut=0.0, auction=0.0, flu=0.0, bottom=0.0, patron=0.0, uwhyu=0.0, richi=0.0,
link=0.0, signal=0.0, custodi=0.0, chanc=0.0, sayer=0.0, skin=0.0, mortal=0.0,
add=0.0, honeymoon=0.0, receipt=0.0, nun=0.0, usual=0.0, starv=0.0, eight=0.0,
morgu=0.0, comin=0.00034855350296270001, style=0.00069710700592540001,
cadet=0.0, uh=0.0027884280237016001, uhuh=0.0, earn=0.0, tini=0.0, volunt=0.0)
```

For example, the fastest way to find the frequency of "fun" in the movie *Toy Story* is to access the `'fun'` item from its row. Check the original table to see if this worked for you!

[162]:
```
row_for_title('toy story').item('fun')
```

[162]:
```
0.00034855350296270001
```

Question 1.0

Set `expected_row_sum` to the number that you **expect** will result from summing all proportions in each row, excluding the first five columns. Think about what any one row adds up to.

[163]:
```
# Set row_sum to a number that's the (approximate) sum of each row of word␣
↪proportions.
expected_row_sum = 1
```

[164]:
```
grader.check("q1_0")
```

[164]:
```
q1_0 results: All test cases passed!
```

This dataset was extracted from a dataset from Cornell University. After transforming the dataset (e.g., converting the words to lowercase, removing the inappropriate words, and converting the counts to frequencies), we created this new dataset containing the frequency of 5000 common words in each movie.

[165]:
```
print('Words with frequencies:', movies.drop(np.arange(5)).num_columns)
print('Movies with genres:', movies.num_rows)
```

```
Words with frequencies: 5000
Movies with genres: 333
```

2.1 1.1. Word Stemming

The columns other than "Title", "Year", "Rating", "Genre", and "# Words" in the `movies` table are all words that appear in some of the movies in our dataset. These words have been *stemmed*, or abbreviated heuristically, in an attempt to make different inflected forms of the same base word into the same string. For example, the column "manag" is the sum of proportions of the words "manage", "manager", "managed", and "managerial" (and perhaps others) in each movie. This is a common technique used in machine learning and natural language processing.

Stemming makes it a little tricky to search for the words you want to use, so we have provided another table called `vocab_table` that will let you see examples of unstemmed versions of each stemmed word. Run the code below to load it.

Note: You should use `vocab_table` for the rest of Section 1.1, not `vocab_mapping`.

[166]:
```
# Just run this cell.
vocab_mapping = Table.read_table('stem.csv')
stemmed = np.take(movies.labels, np.arange(3, len(movies.labels)))
vocab_table = Table().with_column('Stem', stemmed).join('Stem', vocab_mapping)
vocab_table.take(np.arange(1100, 1110))
```

[166]:
```
Stem | Word
bond | bonding
bone | bone
bone | boning
bone | bones
bonu | bonus
book | bookings
book | books
book | booking
book | booked
book | book
```

Question 1.1.1

Using `vocab_table`, find the stemmed version of the word "elements" and assign the value to `stemmed_message`.

[167]:
```
stemmed_message = vocab_table.where("Word", "elements").column("Stem").item(0)
stemmed_message
```

[167]:
```
'element'
```

[168]:
```
grader.check("q1_1_1")
```

[168]:
```
q1_1_1 results: All test cases passed!
```

Question 1.1.2

What stem in the dataset has the most words that are shortened to it? Assign `most_stem` to that stem.

[169]:
```
most_stem = vocab_table.group("Stem").sort("count", descending = True).
  ↪column("Stem").item(0)
most_stem
```

[169]:
```
'gener'
```

[170]:
```
grader.check("q1_1_2")
```

[170]:
```
q1_1_2 results: All test cases passed!
```

Question 1.1.3

What is the longest word in the dataset whose stem wasn't shortened? Assign that to `longest_uncut`. Break ties alphabetically from Z to A (so if your options are "cat" or "bat",

you should pick "cat"). Note that when sorting letters, the letter `a` is smaller than the letter `z`. You may also want to sort more than once.

Hint 1: `vocab_table` has 2 columns: one for stems and one for the unstemmed (normal) word. Find the longest word that wasn't cut at all (same length as stem).

Hint 2: There is a table function that allows you to compute a function on every element in a column. Check Python Reference if you aren't sure which one.

Hint 3: In our solution, we found it useful to first add columns with the length of the word and the length of the stem, and then to add a column with the difference between those lengths. What will the difference be if the word is not shortened?

[171]:
```
stem_len_array = vocab_table.apply(len, "Stem")
word_len_array = vocab_table.apply(len, "Word")
stem_word_len_diff_array = abs(stem_len_array - word_len_array)

vocab_len_diff_table = (vocab_table.with_columns("Stem Len", stem_len_array,
                                                 "Word Len", word_len_array,
                                                 "Difference",␣
 ↪stem_word_len_diff_array))

vocab_len_diff_table = vocab_len_diff_table.sort("Difference")
vocab_len_diff_table = vocab_len_diff_table.where("Difference", 0)
vocab_len_diff_table = vocab_len_diff_table.where("Stem Len", are.
 ↪equal_to(max(vocab_len_diff_table.column("Stem Len"))))
vocab_len_diff_table = vocab_len_diff_table.sort("Stem Len", descending = True)

longest_uncut = vocab_len_diff_table.column("Stem").item(0)
longest_uncut
```

[171]:
```
'extraordinari'
```

[172]:
```
grader.check("q1_1_3")
```

[172]:
```
q1_1_3 results: All test cases passed!
```

Question 1.1.4

How many stems have only one word that is shortened to them? For example, if the stem "book" only maps to the word "books" and if the stem "a" only maps to the word "a," both should be counted as stems that map only to a single word.

Assign `count_single_stems` to the count of stems that map to one word only.

[173]:
```
count_single_stems = vocab_table.group("Stem").where("count", 1).num_rows
count_single_stems
```

[173]:
```
1408
```

[174]:
```
grader.check("q1_1_4")
```

[174]:
```
q1_1_4 results: All test cases passed!
```

2.2 1.2. Exploratory Data Analysis: Linear Regression

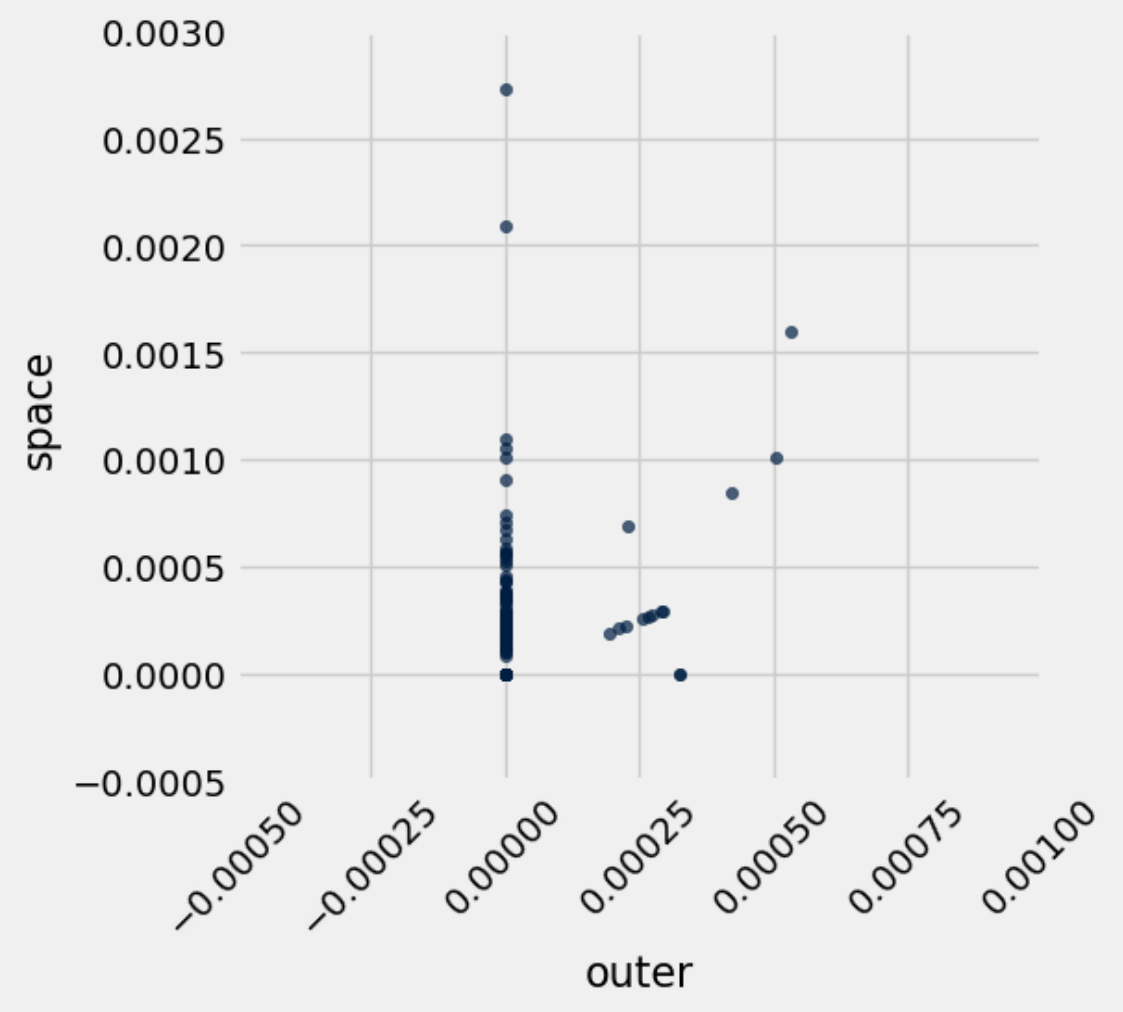
Let's explore our dataset before trying to build a classifier. To start, we'll use the associated proportions to investigate the relationship between different words.

The first association we'll investigate is the association between the proportion of words that are "outer" and the proportion of words that are "space".

As usual, we'll investigate our data visually before performing any numerical analysis.

Run the cell below to plot a scatter diagram of "space" proportions vs "outer" proportions and to create the `outer_space` table. Each point on the scatter plot represents one movie.

[175]:
```
# Just run this cell!
outer_space = movies.select("outer", "space")
outer_space.scatter("outer", "space")
plots.axis([-0.0005, 0.001, -0.0005, 0.003]);
plots.xticks(rotation=45);
```



Question 1.2.1

Looking at that chart it is difficult to see if there is an association. Calculate the correlation coefficient for the potential linear association between proportion of words that are "outer" and the proportion of words that are "space" for every movie in the dataset, and assign it to `outer_space_r`.

Hint: If you need a refresher on how to calculate the correlation coefficient check out Ch 15.1.

[176]:
```
# These two arrays should make your code cleaner!
outer = movies.column("outer")
space = movies.column("space")

# su = (value - mean) / SD
outer_su = (outer - np.mean(outer)) / np.std(outer)
space_su = (space - np.mean(space)) / np.std(space)
```

```
outer_space_r = np.mean(outer_su * space_su)
outer_space_r
```

[176]: 0.31942607876895912

[177]:
```
grader.check("q1_2_1")
```

[177]: q1_2_1 results: All test cases passed!

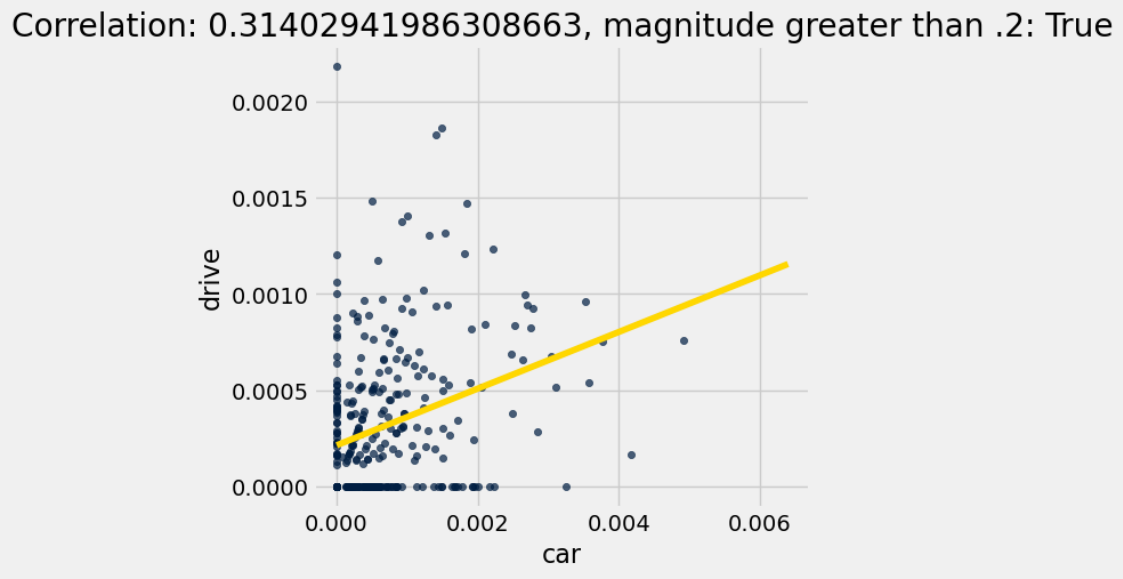
Question 1.2.2

Choose two *different* words in the dataset with a magnitude (absolute value) of correlation higher than 0.2 and plot a scatter plot with a line of best fit for them. Please do not pick "outer" and "space" or "san" and "francisco". The code to plot the scatter plot and line of best fit is given for you, you just need to calculate the correct values to `r`, `slope` and `intercept`.

Hint 1: It's easier to think of words with a positive correlation, i.e. words that are often mentioned together. Try to think of common phrases or idioms.

Hint 2: Refer to Section 15.2 of the textbook for the formulas. For additional past examples of regression, see Homework 9.

[178]:
```
word_x = "car"
word_y = "drive"

# These arrays should make your code cleaner!
arr_x = movies.column(word_x)
arr_y = movies.column(word_y)

# su = (value - mean) / SD
x_su = (arr_x - np.mean(arr_x)) / np.std(arr_x)
y_su = (arr_y - np.mean(arr_y)) / np.std(arr_y)

# r = mean[product of x_su and y_su]
r = np.mean(x_su * y_su)

# m = r * y_SD / x_SD
slope = r * (np.std(arr_y) / np.std(arr_x))
# b = y_avg - m * x_avg
intercept = np.mean(arr_y) - slope * np.mean(arr_x)

# DON'T CHANGE THESE LINES OF CODE
movies.scatter(word_x, word_y)
max_x = max(movies.column(word_x))
plots.title(f"Correlation: {r}, magnitude greater than .2: {abs(r) >= 0.2}")
plots.plot([0, max_x * 1.3], [intercept, intercept + slope * (max_x*1.3)],␣
 ↪color='gold');
```



Question 1.2.3

Imagine that you picked the words "san" and "francisco" as the two words that you would expect to be correlated because they compose the city name San Francisco. Assign `san_francisco` to either the number 1 or the number 2 according to which statement is true regarding the correlation between "san" and "francisco."

1. "san" can also precede other city names like San Diego and San Jose. This might lead to "san" appearing in movies without "francisco," and would reduce the correlation between "san" and "francisco."
2. "san" can also precede other city names like San Diego and San Jose. The fact that "san" could appear more often in front of different cities and without "francisco" would increase the correlation between "san" and "francisco."

[179]:
```
san_francisco = 1
```

[180]:
```
grader.check("q1_2_3")
```

[180]:
```
q1_2_3 results: All test cases passed!
```

2.3 1.3. Splitting the dataset

Now, we're going to use our `movies` dataset for two purposes.

1. First, we want to *train* movie genre classifiers.
2. Second, we want to *test* the performance of our classifiers.

Hence, we need two different datasets: *training* and *test*.

The purpose of a classifier is to classify unseen data that is similar to the training data. The test dataset will help us determine the accuracy of our predictions by comparing the actual genres of the movies with the genres that our classifier predicts. Therefore, we must ensure that there are no movies that appear in both sets. We do so by splitting the dataset randomly. The dataset has already been permuted randomly, so it's easy to split. We just take the first 85% of the dataset for training and the rest for test.

Run the code below (without changing it) to separate the datasets into two tables.

[181]:
```
# Here we have defined the proportion of our data
# that we want to designate for training as 17/20ths (85%)
# and, of our total dataset 3/20ths (15%) of the data is
# reserved for testing

training_proportion = 17/20

num_movies = movies.num_rows
num_train = int(num_movies * training_proportion)
num_test = num_movies - num_train

train_movies = movies.take(np.arange(num_train))
test_movies = movies.take(np.arange(num_train, num_movies))

print("Training: ",   train_movies.num_rows, ";",
      "Test: ",       test_movies.num_rows)
```

```
Training:  283 ; Test:  50
```

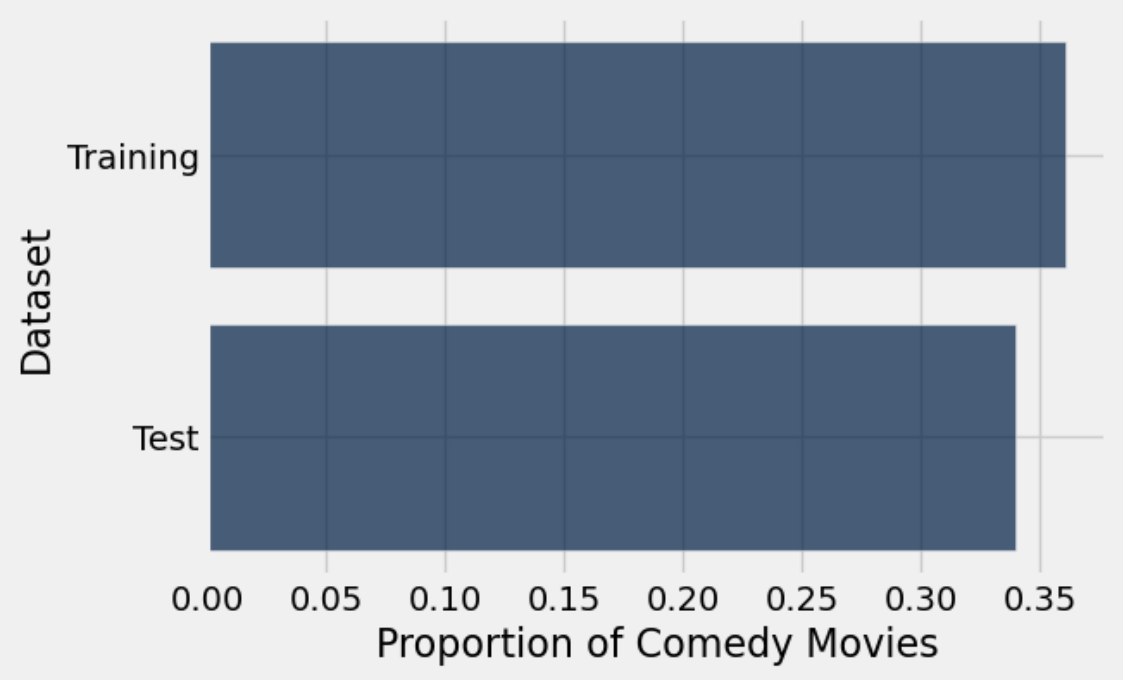
Question 1.3.1

Draw a horizontal bar chart with two bars that show the proportion of Comedy movies in each dataset (`train_movies` and `test_movies`). The two bars should be labeled "Training" and "Test". Complete the function `comedy_proportion` first; it should help you create the bar chart.

Hint: Refer to Section 7.1 of the textbook if you need a refresher on bar charts.

[182]:
```
def comedy_proportion(table):
    # Return the proportion of movies in a table that have the comedy genre.
    num_rows_in_table = table.num_rows
    only_comedy_table = table.where("Genre", "comedy")
    num_comedy_rows_in_table = only_comedy_table.num_rows

    return num_comedy_rows_in_table / num_rows_in_table

# The staff solution took multiple lines.  Start by creating a table.
# If you get stuck, think about what sort of table you need for barh to work
new_table = (Table().with_columns("Dataset", make_array("Training", "Test"),
```

```
                                    "Proportion of Comedy Movies",␣
 ↪make_array(comedy_proportion(train_movies), comedy_proportion(test_movies))))
new_table
new_table.barh("Dataset")
```



3 Part 2: K-Nearest Neighbors - A Guided Example

K-Nearest Neighbors (k-NN) is a classification algorithm. Given some numerical *attributes* (also called *features*) of an unseen example, it decides which category that example belongs to based on its similarity to previously seen examples. Predicting the category of an example is called *labeling*, and the predicted category is also called a *label*.

An attribute (feature) we have about each movie is *the proportion of times a particular word appears in the movie*, and the labels are two movie genres: comedy and thriller. The algorithm requires many previously seen examples for which both the attributes and labels are known: that's the `train_movies` table.

To build understanding, we're going to visualize the algorithm instead of just describing it.

3.1 2.1. Classifying a movie

In k-NN, we classify a movie by finding the `k` movies in the *training set* that are most similar according to the features we choose. We call those movies with similar features the *nearest neighbors*. The k-NN algorithm assigns the movie to the most common category among its `k` nearest neighbors.

Let's limit ourselves to just 2 features for now, so we can plot each movie. The features we will use

are the proportions of the words "water" and "feel" in the movie. Taking the movie *Monty Python and the Holy Grail* (in the test set), 0.000804074 of its words are "water" and 0.0010721 are "feel". This movie appears in the test set, so let's imagine that we don't yet know its genre.

First, we need to make our notion of similarity more precise. **We will say that the *distance* between two movies is the straight-line distance between them when we plot their features on a scatter diagram.**

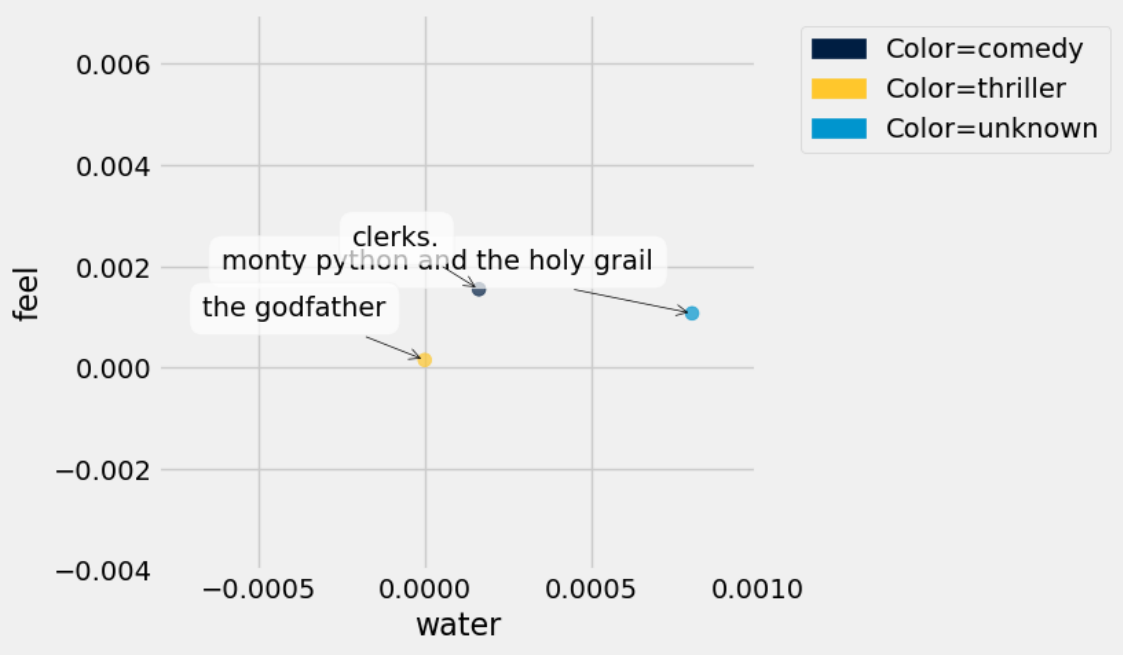
This distance is called the Euclidean ("yoo-KLID-ee-un") distance, whose formula is $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$.

For example, in the movie *Clerks.* (in the training set), 0.00016293 of all the words in the movie are "water" and 0.00154786 are "feel". Its distance from *Monty Python and the Holy Grail* on this 2-word feature set is $\sqrt{(0.000804074 - 0.000162933)^2 + (0.0010721 - 0.00154786)^2} \approx 0.000798379$. (If we included more or different features, the distance could be different.)

A third movie, *The Godfather* (in the training set), has 0 "water" and 0.00015122 "feel".

The function below creates a plot to display the "water" and "feel" features of a test movie and some training movies. As you can see in the result, *Monty Python and the Holy Grail* is more similar to *Clerks.* than to the *The Godfather* based on these features, which makes sense as both movies are comedy movies, while *The Godfather* is a thriller.

[183]:
```
# Just run this cell.
def plot_with_two_features(test_movie, training_movies, x_feature, y_feature):
    """Plot a test movie and training movies using two features."""
    test_row = row_for_title(test_movie)
    distances = Table().with_columns(
            x_feature, [test_row.item(x_feature)],
            y_feature, [test_row.item(y_feature)],
            'Color',   ['unknown'],
            'Title',   [test_movie]
        )
    for movie in training_movies:
        row = row_for_title(movie)
        distances.append([row.item(x_feature), row.item(y_feature), row.
 ↪item('Genre'), movie])
    distances.scatter(x_feature, y_feature, group='Color', labels='Title', s=50)

training = ["clerks.", "the godfather"]
plot_with_two_features("monty python and the holy grail", training, "water",␣
 ↪"feel")
plots.axis([-0.0008, 0.001, -0.004, 0.007]);
```



Question 2.1.1

Compute the Euclidean distance (defined in the section above) between the two movies, *Monty Python and the Holy Grail* and *The Godfather*, using the `water` and `feel` features only. Assign it the name `one_distance`.

Hint 1: If you have a row, you can use `item` to get a value from a column by its name. For example, if `r` is a row, then `r.item("Genre")` is the value in column `"Genre"` in row `r`.

Hint 2: Refer to the beginning of Part 1 if you don't remember what `row_for_title` does.

Hint 3: In the formula for Euclidean distance, think carefully about what `x` and `y` represent. Refer to the example in the text above if you are unsure.

[184]:
```
python = row_for_title("monty python and the holy grail")
godfather = row_for_title("the godfather")

# x1 = proportion of "water" in MPTHG
x1 = python.item("water")
# y1 = proportion of "feel" in MPTGH
y1 = python.item("feel")

x2 = godfather.item("water")
y2 = godfather.item("feel")

diff_of_x = x1 - x2
squared_diff_of_x = np.square(diff_of_x)
```

```
diff_of_y = y1 - y2
squared_diff_of_y = np.square(diff_of_y)

one_distance = np.sqrt(squared_diff_of_x + squared_diff_of_y)
one_distance
```

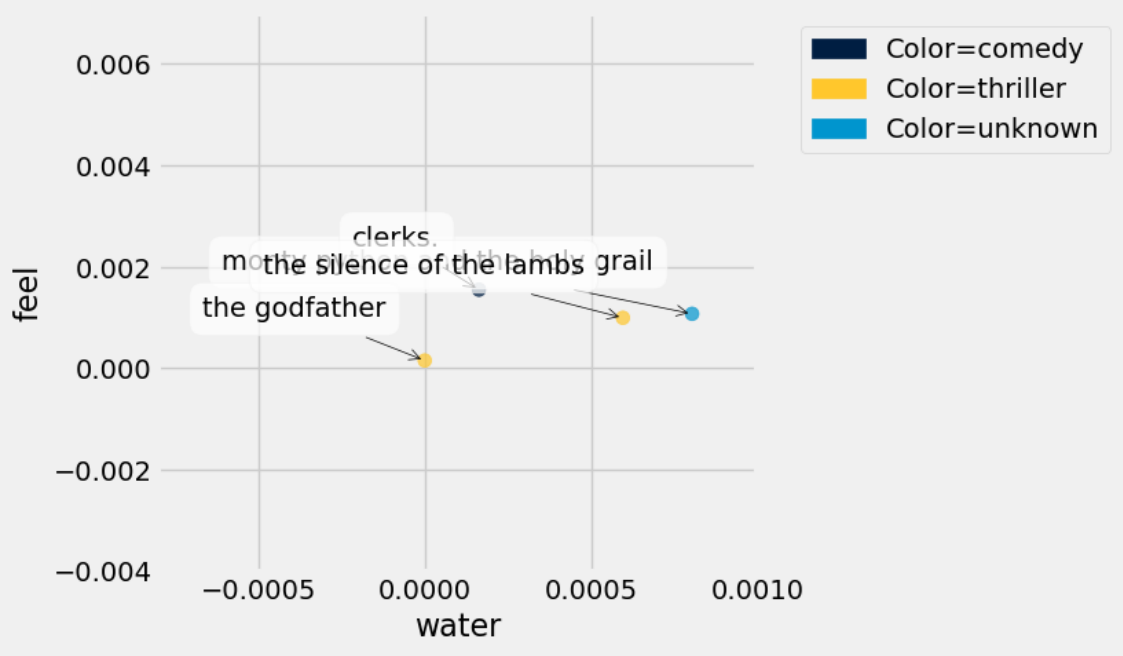
[184]: 0.0012225209151294461

[185]:
```
grader.check("q2_1_1")
```

[185]: q2_1_1 results: All test cases passed!

Below, we've added a third training movie, *The Silence of the Lambs*. Before, the point closest to *Monty Python and the Holy Grail* was *Clerks.*, a comedy movie. However, now the closest point is *The Silence of the Lambs*, a thriller movie.

[186]:
```
training = ["clerks.", "the godfather", "the silence of the lambs"]
plot_with_two_features("monty python and the holy grail", training, "water",␣
↪"feel")
plots.axis([-0.0008, 0.001, -0.004, 0.007]);
```



Question 2.1.2

Complete the function `distance_two_features` that computes the Euclidean distance between any

two movies, using two features. The last two lines call your function to show that *Monty Python and the Holy Grail* is closer to *The Silence of the Lambs* than it is to *Clerks*.

[187]:
```
def distance_two_features(title0, title1, x_feature, y_feature):
    """Compute the distance between two movies with titles title0 and title1.

    Only the features named x_feature and y_feature are used when computing the␣
 ↪distance.
    """
    row0 = row_for_title(title0)
    row1 = row_for_title(title1)

    # for title0
    x1 = row0.item(x_feature)
    y1 = row0.item(y_feature)

    # for title1
    x2 = row1.item(x_feature)
    y2 = row1.item(y_feature)

    # squared difference of x1 - x2
    squared_difference_of_x = np.square(x1 - x2)

    # squared difference of y1 - y2
    squared_difference_of_y = np.square(y1 - y2)

    # sqrt(a + b)
    return np.sqrt(squared_difference_of_x + squared_difference_of_y)

for movie in make_array("clerks.", "the silence of the lambs"):
    movie_distance = distance_two_features(movie, "monty python and the holy␣
 ↪grail", "water", "feel")
    print(movie, 'distance:\t', movie_distance)
```

```
clerks. distance:        0.000798381068723
the silence of the lambs distance:       0.000222563148556
```

[188]:
```
grader.check("q2_1_2")
```

[188]:
```
q2_1_2 results: All test cases passed!
```

Question 2.1.3

Define the function `distance_from_python` so that it works as described in its documentation/docstring(the string right below where the function is defined).

Note: Your solution should not use arithmetic operations directly. Instead, it should make use of

previously defined functions!

[189]:
```
def distance_from_python(title):
    """Return the distance between the given movie and "monty python and the␣
 ↪holy grail",
    based on the features "water" and "feel".

    This function takes a single argument:
      title: A string, the name of a movie.
    """

    return distance_two_features("monty python and the holy grail", title,␣
 ↪"water", "feel")

# Calculate the distance between "Clerks." and "Monty Python and the Holy Grail"
distance_from_python('clerks.')
```

[189]:
```
0.00079838106872277164
```

[190]:
```
grader.check("q2_1_3")
```

[190]:
```
q2_1_3 results: All test cases passed!
```

Question 2.1.4

Using the features `water` and `feel`, what are the names and genres of the 5 movies in the **training set** closest to *Monty Python and the Holy Grail*? To answer this question, make a **table** named `close_movies` containing those 5 movies with columns `"Title"`, `"Genre"`, `"water"`, and `"feel"`, as well as a column called `"distance from python"` that contains the distance from *Monty Python and the Holy Grail*. The table should be **sorted in ascending order by `"distance from python"`**.

Note: Why are smaller distances from *Monty Python and the Holy Grail* more helpful in helping us classify the movie?

Hint: Your final table should only have 5 rows. How can you get the first five rows of a table?

[191]:
```
# The staff solution took multiple lines.
train_movies_distance_from_python_array = train_movies.
 ↪apply(distance_from_python, "Title")
train_movies_distance_from_python_table = train_movies.with_column("distance␣
 ↪from python", train_movies_distance_from_python_array)
train_movies_distance_from_python_table =␣
 ↪train_movies_distance_from_python_table.sort("distance from python",␣
 ↪descending = False)
```

```
train_movies_distance_from_python_table =␣
 ↪train_movies_distance_from_python_table.select("Title", "Genre", "water",␣
 ↪"feel", "distance from python")

close_movies = train_movies_distance_from_python_table.take(np.arange(5))
close_movies
```

[191]:
```
Title                    | Genre    | water       | feel        | distance from
python
alien                    | thriller | 0.00070922  | 0.00124113  | 0.000193831
tomorrow never dies      | thriller | 0.000888889 | 0.000888889 | 0.00020189
the silence of the lambs | thriller | 0.000595948 | 0.000993246 | 0.000222563
innerspace               | comedy   | 0.000522193 | 0.00104439  | 0.00028324
some like it hot         | comedy   | 0.000528541 | 0.000951374 | 0.00030082
```

[192]:
```
grader.check("q2_1_4")
```

[192]:
```
q2_1_4 results: All test cases passed!
```

Question 2.1.5

Next, we'll classify *Monty Python and the Holy Grail* based on the genres of the closest movies.

To do so, define the function `most_common` so that it works as described in its documentation below.

[193]:
```
def most_common(label, table):
    """This function takes two arguments:
      label: The label of a column, a string.
      table: A table.

    It returns the most common value in the label column of the table.
    In case of a tie, it returns any one of the most common values.
    """
    array_to_search = table.column(label)

    return table.group(label).sort("count", descending=True).column(0).item(0)

# Calling most_common on your table of 5 nearest neighbors classifies
# "monty python and the holy grail" as a thriller movie, 3 votes to 2.
most_common('Genre', close_movies)
```

[193]:
```
'thriller'
```

[194]:
```
grader.check("q2_1_5")
```

[194]:
```
q2_1_5 results: All test cases passed!
```

Congratulations are in order — you've classified your first movie! However, we can see that the classifier doesn't work too well since it categorized *Monty Python and the Holy Grail* as a thriller movie (unless you count the thrilling holy hand grenade scene). In next part, we will try to do better!

4 Checkpoint (due Friday 11/22 by 5:00 pm PT)

Toby is so proud of you for reaching the checkpoint!

Run the following cells and submit to the Gradescope assignment titled **Project 3 Checkpoint**. **Before that, remove any `tbl.show()` calls** (calls to `.show` with no argument) and any unnecessary print statements. Forgetting to do this may cause issues when running your notebook on Gradescope.

NOTE: This checkpoint does not represent the halfway point of the project. You are *highly* encouraged to continue on to the next section early.

To get full credit for this checkpoint, you must pass all the public autograder tests above this cell. The cell below will rerun all of the autograder tests for Part 1 and Part 2 so that you can double check your work.

[195]:
```
checkpoint_tests = ["q1_0", "q1_1_1", "q1_1_2", "q1_1_3", "q1_1_4",
                    "q1_2_1", "q1_2_3",
                    "q2_1_1", "q2_1_2", "q2_1_3", "q2_1_4", "q2_1_5"]

for test in checkpoint_tests:
    display(grader.check(test))
```

```
q1_0 results: All test cases passed!

q1_1_1 results: All test cases passed!

q1_1_2 results: All test cases passed!

q1_1_3 results: All test cases passed!

q1_1_4 results: All test cases passed!

q1_2_1 results: All test cases passed!

q1_2_3 results: All test cases passed!

q2_1_1 results: All test cases passed!

q2_1_2 results: All test cases passed!

q2_1_3 results: All test cases passed!

q2_1_4 results: All test cases passed!

q2_1_5 results: All test cases passed!
```

4.1 Submission

Make sure you have run all cells in your notebook in order before running the cell below, so that all images/graphs appear in the output. You can do this by going to `Cell > Run All`. The cell below will generate a zip file for you to submit. **Please save before exporting!**

Reminders: - Please remember to **add your partner to your Gradescope submission**. If you resubmit, make sure to re-add your partner, as Gradescope does not save any partner information. - Make sure to wait until the autograder finishes running to ensure that your submission was processed properly and that you submitted to the correct assignment.

[196]:
```
# Save your notebook first, then run this cell to export your submission.
grader.export(pdf=False, force_save=True)
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

---

5 Part 3: Features

Now, we're going to extend our classifier to consider more than two features at a time to see if we can get a better classification of our movies.

Euclidean distance still makes sense with more than two features. For `n` different features, we compute the difference between corresponding feature values for two movies, square each of the `n` differences, sum up the resulting numbers, and take the square root of the sum.

Question 3.0

Write a function called `distance` to compute the Euclidean distance between two **arrays** of **numerical** features (e.g. arrays of the proportions of times that different words appear). The function should be able to calculate the Euclidean distance between two arrays of arbitrary (but equal) length.

Next, use the function you just defined to compute the distance **between the first and second movie** in the **training set** *using all of the features*. (Remember that the first five columns of your tables are not features.)

Hint 1: To convert rows to arrays, use `np.array`. For example, if `t` was a table, `np.array(t.row(0))` converts row 0 of `t` into an array.

Hint 2: Make sure to drop the first five columns of the table before you compute `distance_first_to_second`, as these columns do not contain any features (the proportions of words).

[197]:
```
# SCRATCH WORK
train_movies.show(1)
train_movies.drop(np.arange(5)).show(1)
np.array(train_movies.drop(np.arange(5)).row(0))
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>
```

[197]:
```
array([ 0.0061667 ,  0.00059678,  0.00159141, …,  0.        ,
        0.        ,  0.        ])
```

[198]:
```
def distance(features_array1, features_array2):
    """The Euclidean distance between two arrays of feature values."""
    return np.sum((features_array1 - features_array2) ** 2) ** 0.5

train_movies_drop_first_5_columns = train_movies.drop(np.arange(5))

array_values_movie_1  = np.array(train_movies_drop_first_5_columns.row(0))
array_values_movie_2 = np.array(train_movies_drop_first_5_columns.row(1))

distance_first_to_second = distance(array_values_movie_1, array_values_movie_2)
distance_first_to_second
```

[198]:
```
0.033354468908813169
```

[199]:
```
grader.check("q3_0")
```

[199]:
```
q3_0 results: All test cases passed!
```

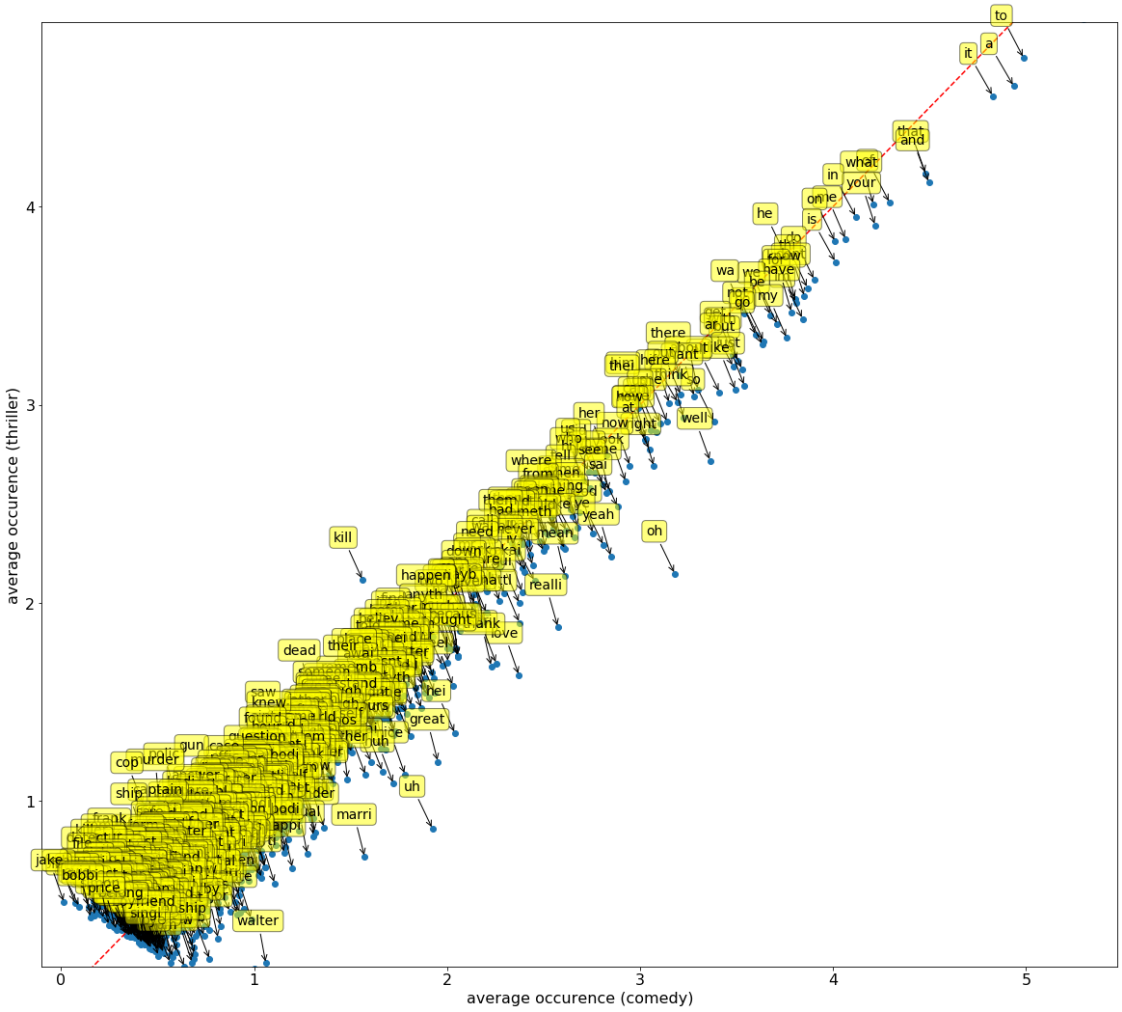
5.1 3.1. Creating your own feature set

Unfortunately, using all of the features has some downsides. One clear downside is the lack of *computational efficiency* – computing Euclidean distances just takes a long time when we have lots of features. You might have noticed that in the last question!

So we're going to select just 20. We'd like to choose features that are very *discriminative*. That is, features which lead us to correctly classify as much of the test set as possible. This process of choosing features that will make a classifier work well is sometimes called *feature selection*, or, more broadly, *feature engineering*.

In this question, we will help you get started on selecting more effective features for distinguishing comedy from thriller movies. The plot below (generated for you) shows the average number of times each word occurs in a comedy movie on the horizontal axis and the average number of times it occurs in an thriller movie on the vertical axis.

Note: The line graphed is the line of best fit, NOT the line y=x.



Run the cell below for an interactive plot! Hover over the invididual points to see what word each point is! For reference, the μ character you may see in some of the entries means to multiply by 10^{-6} or `0.000001`

[200]:
```
from IPython.display import HTML
HTML("https://raw.githubusercontent.com/jonathanferrari/static/main/word_plot.
  ↪html")
```

[200]:
```
<IPython.core.display.HTML object>
```

Questions 3.1.1 through 3.1.4 will ask you to interpret the plot above. If no words are present in the specified area of the graph, try to think what characteristics would those words exhibit if points were located there. For each question, select one of the following choices and assign its number to the provided name. 1. The word is common in both comedy and thriller movies 2. The word is uncommon in comedy movies and common in thriller movies 3. The word is common in comedy movies and uncommon in thriller movies 4. The word is uncommon in both comedy and thriller

movies 5. It is not possible to say from the plot

Question 3.1.1

What properties do words in the bottom left corner of the plot have? Your answer should be a single integer from 1 to 5, corresponding to the correct statement from the choices above.

[201]:
```
bottom_left = 4
```

[202]:
```
grader.check("q3_1_1")
```

[202]:
```
q3_1_1 results: All test cases passed!
```

Question 3.1.2

What properties do words in the bottom right corner have?

[203]:
```
bottom_right = 3
```

[204]:
```
grader.check("q3_1_2")
```

[204]:
```
q3_1_2 results: All test cases passed!
```

Question 3.1.3

What properties do words in the top right corner have?

[205]:
```
top_right = 1
```

[206]:
```
grader.check("q3_1_3")
```

[206]:
```
q3_1_3 results: All test cases passed!
```

Question 3.1.4

What properties do words in the top left corner have?

[207]:
```
top_left = 2
```

[208]:
```
grader.check("q3_1_4")
```

[208]:
```
q3_1_4 results: All test cases passed!
```

Question 3.1.5

If we see a movie with a lot of words that are common for comedy movies but uncommon for thriller movies, what would be a reasonable guess about the genre of the movie? Assign `movie_genre` to the integer corresponding to your answer: 1. It is a thriller movie. 2. It is a comedy movie.

[209]:
```
movie_genre_guess = 2
```

[210]:
```
grader.check("q3_1_5")
```

[210]:
```
q3_1_5 results: All test cases passed!
```

Question 3.1.6

Using the plot above, make an array of at least 10 common words that you think might let you **distinguish** between comedy and thriller movies. Make sure to choose words that are **frequent enough** that every movie contains at least one of them. Don't just choose the most frequent words though–you can do much better.

[211]:
```
# Set my_features to an array of at least 10 features (strings that are column␣
↪labels)

my_features = make_array("oh", "uh", "dude", "love", "big", "window", "beat",␣
↪"kill", "hang", "gun")

# Select the 10 features of interest from both the train and test sets
train_my_features = train_movies.select(my_features)
test_my_features = test_movies.select(my_features)
```

[212]:
```
grader.check("q3_1_6")
```

[212]:
```
q3_1_6 results: All test cases passed!
```

This test makes sure that you have chosen words such that at least one appears in each movie. If you can't find words that satisfy this test just through intuition, try writing code to print out the titles of movies that do not contain any words from your list, then look at the words they do contain.

Question 3.1.7

In two sentences or less, describe how you selected your features.

We viewed the interactive graph and we chose 5 words from above and below the line of best fit, which were far from the line of best fit and towards the center of the graph.

Next, let's classify the first movie from our test set using these features. You can examine the movie by running the cells below. Do you think it will be classified correctly?

[213]:
```
print("Movie:")
test_movies.take(0).select('Title', 'Genre').show()
print("Features:")
test_my_features.take(0).show()
```

```
Movie:
```

```
<IPython.core.display.HTML object>
```

```
Features:
```

```
<IPython.core.display.HTML object>
```

As before, we want to look for the movies in the training set that are most like our test movie. We will calculate the Euclidean distances from the test movie (using `my_features`) to all movies in the training set. You could do this with a `for` loop, but to make it computationally faster, we have provided a function, `fast_distances`, to do this for you. Read its documentation to make sure you understand what it does. (You don't need to understand the code in its body unless you want to.)

[214]:
```
# Just run this cell to define fast_distances.

def fast_distances(test_row, train_table):
    """Return an array of the distances between test_row and each row in␣
 ↪train_table.

    Takes 2 arguments:
      test_row: A row of a table containing features of one
        test movie (e.g., test_my_features.row(0)).
      train_table: A table of features (for example, the whole
        table train_my_features)."""
    assert train_table.num_columns < 50, "Make sure you're not using all the␣
 ↪features of the movies table."
    assert type(test_row) != datascience.tables.Table, "Make sure you are␣
 ↪passing in a row object to fast_distances."
    assert len(test_row) == len(train_table.row(0)), "Make sure the length of␣
 ↪test row is the same as the length of a row in train_table."
    counts_matrix = np.asmatrix(train_table.columns).transpose()
    diff = np.tile(np.array(list(test_row)), [counts_matrix.shape[0], 1]) -␣
 ↪counts_matrix
    np.random.seed(0) # For tie breaking purposes
    distances = np.squeeze(np.asarray(np.sqrt(np.square(diff).sum(1))))
    eps = np.random.uniform(size=distances.shape)*1e-10 #Noise for tie break
    distances = distances + eps
    return distances
```

Question 3.1.8

Use the `fast_distances` function provided above to compute the distance from the first movie in your test set to all the movies in your training set, **using your set of features**. Make a new table called `genre_and_distances` with one row for each movie in the training set and two columns: * The `"Genre"` of the training movie * The `"Distance"` from the first movie in the test set

Ensure that `genre_and_distances` is **sorted in ascending order by distance to the first test movie**.

Hint: Think about how you can use the variables you defined in 3.1.6.

[215]:
```
# The staff solution took multiple lines of code.
train_table_custom_features = train_movies.select("oh", "uh", "dude", "love",␣
 ↪"big", "window", "beat", "kill", "hang", "gun")
test_table_custom_features = test_movies.select("oh", "uh", "dude", "love",␣
 ↪"big", "window", "beat", "kill", "hang", "gun")

distances = fast_distances(test_table_custom_features.row(0),␣
 ↪train_table_custom_features)

genre_and_distances = Table().with_columns("Genre", train_movies.
 ↪column("Genre"), "Distance", distances)
genre_and_distances = genre_and_distances.sort("Distance", descending = False)

genre_and_distances
```

[215]:
```
Genre    | Distance
comedy   | 0.00103501
thriller | 0.00108052
thriller | 0.00110477
comedy   | 0.00114754
comedy   | 0.00114936
thriller | 0.00121164
thriller | 0.00124358
comedy   | 0.00127499
thriller | 0.00131187
comedy   | 0.00132315
… (273 rows omitted)
```

[216]:
```
grader.check("q3_1_8")
```

[216]:
```
q3_1_8 results: All test cases passed!
```

Question 3.1.9

Now compute the 7-nearest neighbors classification of the first movie in the test set. That is, decide on its genre by finding the most common genre among its 7 nearest neighbors in the training set, according to the distances you've calculated. Then check whether your classifier chose the right

genre. (Depending on the features you chose, your classifier might not get this movie right, and that's okay.)

Hint 1: You should use the `most_common` function that was defined earlier.

Hint 2: You should use a comparison operator.

[217]:
```
# Set my_assigned_genre to the most common genre among these.
my_assigned_genre = most_common("Genre", test_movies.take(np.arange(7))) #␣
 ↪label, table

# Set my_assigned_genre_was_correct to True if my_assigned_genre
# matches the actual genre of the first movie in the test set, False otherwise.
my_assigned_genre_was_correct = my_assigned_genre == test_movies.
 ↪column("Genre").item(0) # == first movie in test set

print("The assigned genre, {}, was{}correct.".format(my_assigned_genre, " " if␣
 ↪my_assigned_genre_was_correct else " not "))
```

```
The assigned genre, thriller, was not correct.
```

[218]:
```
grader.check("q3_1_9")
```

[218]:
```
q3_1_9 results: All test cases passed!
```

5.2 3.2. A classifier function

Now we can write a single function that encapsulates the whole process of classification.

Question 3.2.1

Write a function called `classify`. It should take the following four arguments: * A row of features for a movie to classify (e.g., `test_my_features.row(0)`). * A table with a column for each feature (e.g., `train_my_features`). * An array of classes (e.g. the labels "comedy" or "thriller") that has as many items as the previous table has rows, and in the same order. *Hint:* What are the labels of each row in the training set? * `k`, the number of neighbors to use in classification.

It should return the class (the string `'comedy'` or the string `'thriller'`) a `k`-nearest neighbor classifier picks for the given row of features.

[219]:
```
def classify(test_row, train_features, train_labels, k):
    """Return the most common class among k nearest neigbors to test_row."""
    distances = fast_distances(test_row, train_features)

    genre_and_distances = Table().with_columns("Genre", train_movies.
 ↪column("Genre"), "Distance", distances)
    genre_and_distances = genre_and_distances.sort("Distance", descending =␣
 ↪False)
```

```
    assigned_genre = most_common("Genre", genre_and_distances.take(np.
 ↪arange(k)))

    return assigned_genre
```

[220]:
```
grader.check("q3_2_1")
```

[220]:
```
q3_2_1 results: All test cases passed!
```

Question 3.2.2

Assign `godzilla_genre` to the genre predicted by your classifier for the movie "godzilla" in the test set, using **15 neighbors** and using your 10 features.

Hint: The `row_for_title` function will not work here.

[221]:
```
# The staff solution first defined a row called godzilla_features.
godzilla_features = (test_movies.where("Title", "godzilla")
                                .select("oh", "uh", "dude", "love", "big",␣
 ↪"window", "beat", "kill", "hang", "gun")
                                .row(0))
godzilla_genre = classify(godzilla_features, train_my_features, train_movies.
 ↪column("Genre"), 15)
godzilla_genre
```

[221]:
```
'thriller'
```

[222]:
```
grader.check("q3_2_2")
```

[222]:
```
q3_2_2 results: All test cases passed!
```

Finally, when we evaluate our classifier, it will be useful to have a classification function that is specialized to use a fixed training set and a fixed value of `k`.

Question 3.2.3

Create a classification function that takes as its argument a row containing your 10 features and classifies that row using the 15-nearest neighbors algorithm with `train_my_features` as its training set.

[223]:
```
def classify_feature_row(row):
    return classify(row, train_my_features, train_movies.column("Genre"), 15)

# When you're done, this should produce 'thriller' or 'comedy'.
classify_feature_row(test_my_features.row(0))
```

[223]:
```
'thriller'
```

[224]:
```
grader.check("q3_2_3")
```

[224]:
```
q3_2_3 results: All test cases passed!
```

5.3 3.3. Evaluating your classifier

Now that it's easy to use the classifier, let's see how accurate it is on the whole test set.

Question 3.3.1

Use `classify_feature_row` and `apply` to classify every movie in the test set. Assign these guesses as an array to `test_guesses`. Then, compute the proportion of correct classifications.

Hint 1: If you do not specify any columns in `tbl.apply(...)`, your function will be applied to every row object in `tbl`.

Hint 2: Which dataset do you want to apply this function to?

[225]:
```
test_guesses = test_my_features.apply(classify_feature_row)
proportion_correct = np.mean( test_guesses == test_movies.column("Genre") )
proportion_correct
```

[225]:
```
0.73999999999999999
```

[226]:
```
grader.check("q3_3_1")
```

[226]:
```
q3_3_1 results: All test cases passed!
```

Question 3.3.2

An important part of evaluating your classifiers is figuring out where they make mistakes. Assign the name `test_movie_correctness` to a table with three columns, `'Title'`, `'Genre'`, and `'Was correct'`.

- The `'Genre'` column should contain the original genres, not the ones you predicted.
- The `'Was correct'` column should contain `True` or `False` depending on whether or not the movie was classified correctly.

[280]:
```
# Feel free to use multiple lines of code
# but make sure to assign test_movie_correctness to the proper table!
was_correct_array = test_my_features.apply(classify_feature_row) == test_movies.
 ↪column("Genre")

test_movie_correctness = (Table().with_columns("Title", test_movies.
 ↪column("Title"),
                                                "Genre", test_movies.
 ↪column("Genre"),
```

```
                                                    "Was correct",␣
↪was_correct_array))

test_movie_correctness.sort('Was correct', descending = True).show(5)
```

```
<IPython.core.display.HTML object>
```

[228]:
```
grader.check("q3_3_2")
```

[228]:
```
q3_3_2 results: All test cases passed!
```

Question 3.3.3

Do you see a pattern in the types of movies your classifier misclassifies? In two sentences or less, describe any patterns you see in the results or any other interesting findings from the table above. If you need some help, try looking up the movies that your classifier got wrong on Wikipedia.

The classifier misclassified 13 movies, 6 being thrillers and 7 comedies. This looks like an almost even split with a bias towards misclassifying comedy movies.

At this point, you've gone through one cycle of classifier design. Let's summarize the steps: 1. From available data, select test and training sets. 2. Choose an algorithm you're going to use for k-NN classification. 3. Identify some features. 4. Define a classifier function using your features and the training set. 5. Evaluate its performance (the proportion of correct classifications) on the test set.

6 Part 4: Explorations

Now that you know how to evaluate a classifier, it's time to build another one.

Your friends are big fans of comedy and thriller movies. They have created their own dataset of movies that they want to watch, but they need your help in determining the genre of each movie in their dataset (comedy or thriller). You have never seen any of the movies in your friends' dataset, so none of your friends' movies are present in your training or test set from earlier. In other words, this new dataset of movies can function as another test set that we are going to make predictions on based on our original training data.

Run the following cell to load your friends' movie data.

> **Note:** The `friend_movies` table has 5005 columns, so we only show the first 105 to stop this cell from crashing your notebook.

[230]:
```
friend_movies = Table.read_table('friend_movies.csv')
friends_subset = friend_movies.select(np.arange(106))
friends_subset.show(5)
```

```
<IPython.core.display.HTML object>
```

Question 4.1

Your friend's computer is not as powerful as yours, so they tell you that the classifier you create for them can only have up to 5 words as features. Develop a new classifier with the constraint of **using no more than 5 features.** Assign `new_features` to an array of your features.

Your new function should have the same arguments as `classify_feature_row` and return a classification. Name it `another_classifier`. Then, output your accuracy using code from earlier. We can look at the value to compare the new classifier to your old one.

Some ways you can change your classifier are by using different features or trying different values of `k`. (Of course, you still have to use `train_movies` as your training set!)

Make sure you don't reassign any previously used variables here, such as `proportion_correct` from the previous question.

Note: There's no one right way to do this! Just make sure that you can explain your reasoning behind the new choices.

[273]:
```
# "oh", "uh", "dude", "love", "big", "window", "beat", "kill", "hang", "gun"
v1 = make_array("oh", "dude", "big", "beat", "hang")
v2 = make_array("uh", "love", "window", "kill", "gun")

new_features = v1 # array

train_new = train_movies.select(new_features)
test_new = friend_movies.select(new_features)

def another_classifier(row):
    return classify(row, train_my_features.select(new_features), train_movies.
 ↪column("Genre"), 15)
```

[275]:
```
grader.check("q4_1")
```

[275]:
```
q4_1 results: All test cases passed!
```

Question 4.2

Do you see a pattern in the mistakes your new classifier makes? How good an accuracy were you able to get with your limited classifier? Did you notice an improvement from your first classifier to the second one? Describe in two sentences or less.

Hint: You may not be able to see a pattern.

The new classifier shows a continuing pattern where comedies are more likely to be misclassified than thrillers, with five comedy misclassifications compared to three thriller misclassifications. The second classifier showed improvement over the first, achieving a higher accuracy rate of 0.78 compared to the first classifier's 0.74 an increase of 0.04.

Question 4.3

Given the constraint of five words, how did you select those five? Describe in two sentences or less.

From our initial set of 10 words, which contained an equal mix of common "comedy" and "thriller" terms, we selected every other word to create our five-word subset.

Archie wants to say: ***Congratulations! You have finished the final Data 8 project!***

Using your statistics and machine learning skills you have: - Investigated a movie script dataset - Identified word associations - Built a machine learning model to classify movies by their scripts

7 Part 5: Other Classification Methods (OPTIONAL)

Note: Everything below is **OPTIONAL**. Please only work on this part after you have finished and submitted the project. If you create new cells below, do NOT reassign variables defined in previous parts of the project.

Now that you've finished your k-NN classifier, you might be wondering what else you could do to improve your accuracy on the test set. Classification is one of many machine learning tasks, and there are plenty of other classification algorithms! If you feel so inclined, we encourage you to try any methods you feel might help improve your classifier.

We've compiled a list of blog posts with some more information about classification and machine learning. Create as many cells as you'd like below–you can use them to import new modules or implement new algorithms.

Blog posts:

- Classification algorithms/methods
- Train/test split and cross-validation
- More information about k-nearest neighbors
- Overfitting

In future data science classes, such as Data Science 100, you'll learn about some of the algorithms in the blog posts above, including logistic regression. You'll also learn more about overfitting, cross-validation, and approaches to different kinds of machine learning problems.

One module that you can consider using is Scikit-learn. There's a lot to think about, so we encourage you to find more information on your own!

Lastly, please make sure the runtime of any cells you make aren't too long, as it may time out the autograder.

Please make sure you have added your partner on Gradescope. If you have done it, set `add_partner` to True.

Note: The variable should be set to True if you don't have a partner.

[281]:
```
add_partner = True
```

[282]:
```
grader.check("q5_1")
```

[282]:
```
q5_1 results: All test cases passed!
```

You're finished! Time to submit.

Reminders: - Please remember to **add your partner to your Gradescope submission**. If you resubmit, make sure to re-add your partner, as Gradescope does not save any partner information. - Please note that filling out the Official Project 2 Partner form is NOT the same as adding your partner to your Gradescope submission. - **Remove any `tbl.show()` calls** (calls to `.show` with no argument) and any unneccessary print statements. Forgetting to do this may cause issues when running your notebook on Gradescope. - Make sure to wait until the autograder finishes running to ensure that your submission was processed properly and that you submitted to the correct assignment.

7.1 Written Work Submission

Below, you will see two cells. Running the first cell will automatically generate a PDF of all questions that need to be manually graded, and running the second cell will automatically generate a zip with your autograded answers. You are responsible for submitting both the coding portion (the zip) and the written portion (the PDF) to their respective Gradescope portals. **Please save before exporting!**

> **Important: You must correctly assign the pages of your PDF after you submit to the correct gradescope assignment. If your pages are not correctly assigned and/or not in the correct PDF format by the deadline, we reserve the right to award no points for your written work.**

If there are issues with automatically generating the PDF in the first cell, you can try downloading the notebook as a PDF by clicking on `File -> Save and Export Notebook As... -> Webpdf`. If that doesn't work either, you can manually take screenshots of your answers to the manually graded questions and submit one single PDF of your screenshots. Either way, **you are responsible for ensuring your submision follows our requirements, we will NOT be granting regrade requests for submissions that don't follow instructions.**

You must submit the PDF generated via one of these methods, we will not accept screenshots or Word documents.

[283]:
```
from otter.export import export_notebook
from os import path
from IPython.display import display, HTML
name = 'project3'
export_notebook(f"{name}.ipynb", filtering=True, pagebreaks=True)
if(path.exists(f'{name}.pdf')):
    display(HTML(f"Download your PDF <a href='{name}.pdf' download>here</a>."))
else:
    print("\n Pdf generation failed, please try the other methods described␣
 ↪above")
```

```
<IPython.core.display.HTML object>
```

7.2 Submission

Make sure you have run all cells in your notebook in order before running the cell below, so that all images/graphs appear in the output. The cell below will generate a zip file for you to submit.

Please save before exporting!

[ ]:
```
# Save your notebook first, then run this cell to export your submission.
grader.export(pdf=False, run_tests=True)
```

```
Running your submission against local test cases…
```