PREPROCESSING

Sampling

Feature extraction – TF-IDF

Data Normalization

DATA COLLECTION - SAMPLING

Data collection

- Suppose that you want to collect data from Twitter about the elections in USA
  - How do you go about it?
- Twitter Streaming/Search API:
  - Get a sample of all tweets that are posted on Twitter
  - Example of JSON object
- REST API:
  - Get information about specific users.
- There are several decisions that we need to make before we start collecting the data.
  - Time and Storage resources

Sampling

- Sampling is the main technique employed for data selection.
  - It is often used for both the preliminary investigation of the data and the final data analysis.

- Statisticians sample because obtaining the entire set of data of interest is too expensive or time consuming.
  - Example: What is the average height of a person in China?
    - We cannot measure the height of everybody

- Sampling is used in data mining because processing the entire set of data of interest is too expensive or time consuming.
  - Example: We have 1M documents. What fraction of pairs has at least 100 words in common?
    - Computing number of common words for all pairs requires 10^{12} comparisons
  - Example: What fraction of tweets in a year contain the word “China”?
    - 500M tweets per day, if 100 characters on average, 86.5TB to store all tweets

Sampling …

- The key principle for effective sampling is the following:
  - using a sample will work almost as well as using the entire data sets, if the sample is representative
  - A sample is representative if it has approximately the same property (of interest) as the original set of data
  - Otherwise we say that the sample introduces some bias
  - What happens if we take a sample from ShanghaiTech to compute the average height of a person in China?

Types of Sampling

- Simple Random Sampling
  - There is an equal probability of selecting any particular item
- Sampling without replacement
  - As each item is selected, it is removed from the population
- Sampling with replacement
  - Objects are not removed from the population as they are selected for the sample.
    - In sampling with replacement, the same object can be picked up more than once. This makes analytical computation of probabilities easier
    - E.g., we have 100 people, 51 are women $P(W) = 0.51$, 49 men $P(M) = 0.49$. If I pick two persons what is the probability $P(W,W)$ that both are women?
      - Sampling with replacement: $P(W,W) = 0.51^2$
      - Sampling without replacement: $P(W,W) = 51/100 * 50/99$

Types of Sampling

- Stratified sampling （分层抽样）
  - Split the data into several groups; then draw random samples from each group.
    - Ensures that all groups are represented.
  - Example 1. I want to understand the differences between legitimate and fraudulent credit card transactions. 0.1% of transactions are fraudulent. What happens if I select 1000 transactions at random?
    - I get 1 fraudulent transaction (in expectation). Not enough to draw any conclusions. Solution: sample 1000 legitimate and 1000 fraudulent transactions

Probability Reminder: If an event has probability p of happening and I do N trials, the expected number of times the event occurs is pN

Biased sampling

- Sometimes we want to bias our sample towards some subset of the data
  - Stratified sampling is one example
- Example: When sampling temporal data, we want to increase the probability of sampling recent data
  - Introduce recency bias
- Make the sampling probability to be a function of time, or the age of an item
  - Typical: Probability decreases exponentially with time
  - For item x_t after time t select with probability $p(x_t) \propto e^{-t}$

FEATURE EXTRACTION

TF-IDF word weighting

Data cleaning – Feature extraction

- Once we have the data, we most likely will not use it as is
- We need to do some cleaning
- We need to extract some features to represent our data

Data Quality

- Examples of data quality problems:
  - Noise and outliers
  - Missing values
  - Duplicate data

A mistake or a millionaire?

Missing values

Inconsistent duplicate entries

Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
1	Yes	Single	125K	**No**
2	No	Married	100K	**No**
3	No	Single	70K	**No**
4	Yes	Married	120K	**No**
5	No	Divorced	10000K	**Yes**
6	No	NULL	60K	**No**
7	Yes	Divorced	220K	**NULL**
8	No	Single	85K	**Yes**
9	No	Married	90K	**No**
9	No	Single	90K	**No**

Data preprocessing: feature extraction

- The data we obtain are not necessarily as a relational table
- Data may be in a very raw format
  - Examples: text, speech, mouse movements, etc
- We need to extract the features from the data

- Feature extraction:
  - Selecting the characteristics by which we want to represent our data
  - It requires some domain knowledge about the data
  - It depends on the application
- Deep learning: eliminates this step.

Text data

- Data will often not be in a nice relational table
- For example: Text data
  - We need to do additional effort to extract the useful information from the text data

- We will now see some basic text processing ideas.

A data preprocessing example

- Suppose we want to mine the restaurant comments/reviews of people on Yelp or Foursquare.

Mining Task

- Collect all reviews for the top-10 most reviewed restaurants in NY in Yelp


```
{"votes": {"funny": 0, "useful": 2, "cool": 1},
 "user_id": "Xqd0DzHaiyRqVH3WRG7hzg",
 "review_id": "15SdjuK7DmYqUAj6rjGowg",
 "stars": 5, "date": "2007-05-17",
 "text": "I heard so many good things about this place so I was pretty juiced to try
it.  I'm from Cali and I heard Shake Shack is comparable to IN-N-OUT and I gotta
say,  Shake Shake wins hands down.    Surprisingly, the line was short and we waited
about 10 MIN. to order.  I ordered a regular cheeseburger, fries and a black/white
shake.  So yummerz.   I love the location too!  It's in the middle of the city and the
view is breathtaking.   Definitely one of my favorite places to eat in NYC.",
 "type": "review",
 "business_id": "vcNAWiLM4dR7D2nwwJ7nCA"}
```


- Feature extraction: Find few terms that best describe the restaurants.

Example data

I heard so many good things about this place so I was pretty juiced to try it. I'm from Cali and I heard Shake Shack is comparable to IN-N-OUT and I gotta say, Shake Shake wins hands down.   Surprisingly, the line was short and we waited about 10 MIN. to order. I ordered a regular cheeseburger, fries and a black/white shake. So yummerz.   I love the location too! It's in the middle of the city and the view is breathtaking.   Definitely one of my favorite places to eat in NYC.

I'm from California and I must say, Shake Shack is better than IN-N-OUT, all day, err'day.

Would I pay $15+ for a burger here? No. But for the price point they are asking for, this is a definite bang for your buck (though for some, the opportunity cost of waiting in line might outweigh the cost savings) Thankfully, I came in before the lunch swarm descended and I ordered a shake shack (the special burger with the patty + fried cheese & portabella topping) and a coffee milk shake. The beef patty was very juicy and snugly packed within a soft potato roll. On the downside, I could do without the fried portabella-thingy, as the crispy taste conflicted with the juicy, tender burger. How does shake shack compare with in-and-out or 5-guys? I say a very close tie, and I think it comes down to personal affliations. On the shake side, true to its name, the shake was well churned and very thick and luscious. The coffee flavor added a tangy taste and complemented the vanilla shake well. Situated in an open space in NYC, the open air sitting allows you to munch on your burger while watching people zoom by around the city. It's an oddly calming experience, or perhaps it was the food

First cut

- Do simple processing to “normalize” the data (remove punctuation, make into lower case, clear white spaces, other?)
- Break into words, keep the most popular words

the 27514	the 16710	the 16010	the 14241
and 14508	and 9139	and 9504	and 8237
i 13088	a 8583	i 7966	a 8182
a 12152	i 8415	to 6524	i 7001
to 10672	to 7003	a 6370	to 6727
of 8702	in 5363	it 5169	of 4874
ramen 8518	it 4606	of 5159	you 4515
was 8274	of 4365	is 4519	it 4308
is 6835	is 4340	sauce 4020	is 4016
it 6802	burger 432	in 3951	was 3791
in 6402	was 4070	this 3519	pastrami 3748
for 6145	for 3441	was 3453	in 3508
but 5254	but 3284	for 3327	for 3424
that 4540	shack 3278	you 3220	sandwich 2928
you 4366	shake 3172	that 2769	that 2728
with 4181	that 3005	but 2590	but 2715
pork 4115	you 2985	food 2497	on 2247
my 3841	my 2514	on 2350	this 2099
this 3487	line 2389	my 2311	my 2064
wait 3184	this 2242	cart 2236	with 2040
not 3016	fries 2240	chicken 2220	not 1655
we 2984	on 2204	with 2195	your 1622
at 2980	are 2142	rice 2049	so 1610
on 2922	with 2095	so 1825	have 1585

First cut

- Do simple processing to “normalize” the data (remove punctuation, make into lower case, clear white spaces, other?)
- Break into words, keep the most popular words

```
the 27514
and 14508
i 13088
a 12152
to 10672
of 8702
ramen 8518
was 8274
is 6835
it 6802
in 6402
for 6145
but 5254
that 4540
you 4366
with 4181
pork 4115
my 3841
this 3487
wait 3184
not 3016
we 2984
at 2980
on 2922
```

```
the 16710
and 9139
a 8583
i 8415
to 7003
in 5363
it 4606
of 4365
is 4340
burger 432
was 4070
for 3441
but 3284
shack 3278
shake 3172
that 3005
you 2985
my 2514
line 2389
this 2242
fries 2240
on 2204
are 2142
with 2095
```

```
the 16010
and 9504
i 7966
to 6524
a 6370
it 5169
of 5159
is 4519
sauce 4020
in 3951
this 3519
was 3453
for 3327
you 3220
that 2769
but 2590
food 2497
cart 2236
chicken 2220
with 2195
rice 2049
so 1825
```

```
the 14241
and 8237
a 8182
i 7001
to 6727
of 4874
you 4515
it 4308
is 4016
was 3791
pastrami 3748
in 3508
for 3424
sandwich 2928
that 2728
but 2715
on 2247
not 1655
your 1622
so 1610
have 1585
```

Most frequent words are stop words

Second cut

- Remove stop words
  - Stop-word lists can be found online.

```
a,about,above,after,again,against,all,am,an,and,any,are,aren't,as,at,be,becaus
e,been,before,being,below,between,both,but,by,can't,cannot,could,couldn't,did,
didn't,do,does,doesn't,doing,don't,down,during,each,few,for,from,further,had,h
adn't,has,hasn't,have,haven't,having,he,he'd,he'll,he's,her,here,here's,hers,h
erself,him,himself,his,how,how's,i,i'd,i'll,i'm,i've,if,in,into,is,isn't,it,it
's,its,itself,let's,me,more,most,mustn't,my,myself,no,nor,not,of,off,on,once,o
nly,or,other,ought,our,ours,ourselves,out,over,own,same,shan't,she,she'd,she'l
l,she's,should,shouldn't,so,some,such,than,that,that's,the,their,theirs,them,t
hemselves,then,there,there's,these,they,they'd,they'll,they're,they've,this,th
ose,through,to,too,under,until,up,very,was,wasn't,we,we'd,we'll,we're,we've,we
re,weren't,what,what's,when,when's,where,where's,which,while,who,who's,whom,wh
y,why's,with,won't,would,wouldn't,you,you'd,you'll,you're,you've,your,yours,yo
urself,yourselves,
```

Second cut

- Remove stop words
  - Stop-word lists can be found online.

ramen 8572	burger 4340	sauce 4023	pastrami 3782
pork 4152	shack 3291	food 2507	sandwich 2934
wait 3195	shake 3221	cart 2239	place 1480
good 2867	line 2397	chicken 2238	good 1341
place 2361	fries 2260	rice 2052	get 1251
noodles 2279	good 1920	hot 1835	katz's 1223
ippudo 2261	burgers 1643	white 1782	just 1214
buns 2251	wait 1508	line 1755	like 1207
broth 2041	just 1412	good 1629	meat 1168
like 1902	cheese 1307	lamb 1422	one 1071
just 1896	like 1204	halal 1343	deli 984
get 1641	food 1175	just 1338	best 965
time 1613	get 1162	get 1332	go 961
one 1460	place 1159	one 1222	ticket 955
really 1437	one 1118	like 1096	food 896
go 1366	long 1013	place 1052	sandwiches 813
food 1296	go 995	go 965	can 812
bowl 1272	time 951	can 878	beef 768
can 1256	park 887	night 832	order 720
great 1172	can 860	time 794	pickles 699
best 1167	best 849	long 792	time 662
		people 790	

Second cut

- Remove stop words
  - Stop-word lists can be found online.

ramen 8572	burger 4340	sauce 4023	pastrami 3782
pork 4152	shack 3291	food 2507	sandwich 2934
wait 3195	shake 3221	cart 2239	place 1480
good 2867	line 2397	chicken 2238	good 1341
place 2361	fries 2260	rice 2052	**get** 1251
noodles 2279	good 1920	hot 1835	katz's 1223
ippudo 2261	burgers 1643	white 1782	just 1214
buns 2251	wait 1508	line 1755	**like** 1207
broth 2041	just 1412	good 1629	meat 1168
like 1902	cheese 1307	lamb 1422	one 1071
just 1896	**like** 1204	halal 1343	deli 984
get 1641	food 1175	just 1338	best 965
time 1613	**get** 1162	**get** 1332	go 961
one 1460	place 1159	one 1222	ticket 955
really 1437	one 1118	**like** 1096	food 896
go 1366	long 1013	place 1052	
food 1296			
bowl 1272			
can 1256	park 887	night 832	order 720
great 1172	can 860	time 794	pickles 699
best 1167	best 849	long 792	time 662
		people 790	

Commonly used words in reviews, not so interesting

TF-IDF

- The words that are best for describing a document are the ones that are important for the document, but also unique to the document.

- $TF(w, d)$: term frequency of word w in document d
  - Number of times that the word appears in the document
  - Natural measure of importance of the word for the document

- $IDF(w)$: inverse document frequency
  - Natural measure of the uniqueness of the word w

- $TF\text{-}IDF(w, d) \; = \; TF(w, d) \times IDF(w)$

IDF

- Important words are the ones that are unique to the document (differentiating) compared to the rest of the collection
  - All reviews use the word "like". This is not interesting
  - We want the words that characterize the specific restaurant

- Document Frequency $DF(w)$: fraction of documents that contain word w.

$$DF(w) = \frac{D(w)}{D}$$

$D(w)$: num of docs that contain word w
D: total number of documents

- Inverse Document Frequency $IDF(w)$:

$$IDF(w) = \log\left(\frac{1}{DF(w)}\right)$$

- Maximum when unique to one document : $IDF(w) = \log(D)$
- Minimum when the word is common to all documents: $IDF(w) = \log\left(\frac{1}{1}\right) = 0$

Third cut

- Ordered by TF-IDF

ramen 3057.4176194	fries 806.08537330	lamb 985.655290756243	pastrami 1931.94250908298 6
akamaru 2353.24196	custard 729.607519	halal 686.038812717726	katz's 1120.62356508209 4
noodles 1579.68242	shakes 628.4738038	53rd 375.685771863491	rye 1004.28925735888 2
broth 1414.7133955	shroom 515.7790608	gyro 305.809092298788	corned 906.113544700399 2
miso 1252.60629058	burger 457.2646379	pita 304.984759446376	pickles 640.487221580035 4
hirata 709.1962086	crinkle 398.347221	cart 235.902194557873	reuben 515.779060830666 1
hakata 591.7643688	burgers 366.624854	platter 139.459903080044	matzo 430.583412389887 1
shiromaru 587.1591	madison 350.939350	chicken/lamb 135.852520	sally 428.110484707471 2
noodle 581.8446147	shackburger 292.42	carts 120.274374158359	harry 226.323810772916 4
tonkotsu 529.59457	'shroom 287.823136	hilton 84.2987473324223	mustard 216.079238853014 6
ippudo 504.5275695	portobello 239.806	lamb/chicken 82.8930633	cutter 209.535243462458 1
buns 502.296134008	custards 211.83782	yogurt 70.0078652365545	carnegie 198.655512713779 3
ippudo's 453.60926	concrete 195.16992	52nd 67.5963923222322	katz 194.387844446609 7
modern 394.8391629	bun 186.9621782983	6th 60.7930175345658 9	knish 184.206807439524 1
egg 367.3680056967	milkshakes 174.996	4am 55.4517744447956 5	sandwiches 181.415707218 8
shoyu 352.29551922	concretes 165.7861	yellow 54.4470265206673	brisket 131.945865389878 4
chashu 347.6903490	portabello 163.483	tzatziki 52.959457138863	fries 131.613054313392 7
karaka 336.1774235	shack's 159.334353	lettuce 51.323016802268	salami 127.621117258549 3
kakuni 276.3102111	patty 152.22603588	sammy's 50.656872045869	knishes 124.339595021678 1
ramens 262.4947006	ss 149.66803104461	sw 50.5668577816893 3	delicatessen 117.488967607 2
bun 236.5122638036	patties 148.068287	platters 49.906597000316	deli's 117.431839742696 1
wasabi 232.3667512	cam 105.9496067806	falafel 49.479699521204	carver 115.129254649702 1
dama 221.048168927	milkshake 103.9720	sober 49.2211422635451	brown's 109.441778045519 2
brulee 201.1797390	lamps 99.011158998	moma 48.1589121730374	matzoh 108.22149937072 1

Third cut

- TF-IDF takes care of stop words as well
- We do not need to remove the stopwords since they will get $IDF(w) = 0$
- Important: IDF is collection-dependent!
  - For some other corpus the words *get, like, eat*, may be important

- What would you do for Chinese reviews?

- What would you do for Chinese reviews?
  - 中文没有天然空格作为分隔
  - 分词+（去除停用词）
  - 分词工具: Jieba in Python

Decisions, decisions…

- When mining real data you often need to make some decisions
  - What data should we collect? How much? For how long?
  - Should we throw out some data that does not seem to be useful?

An actual review

```
AAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAA AAAAAAAAAAAAAAAAAAAAAAAAAA AAA
```

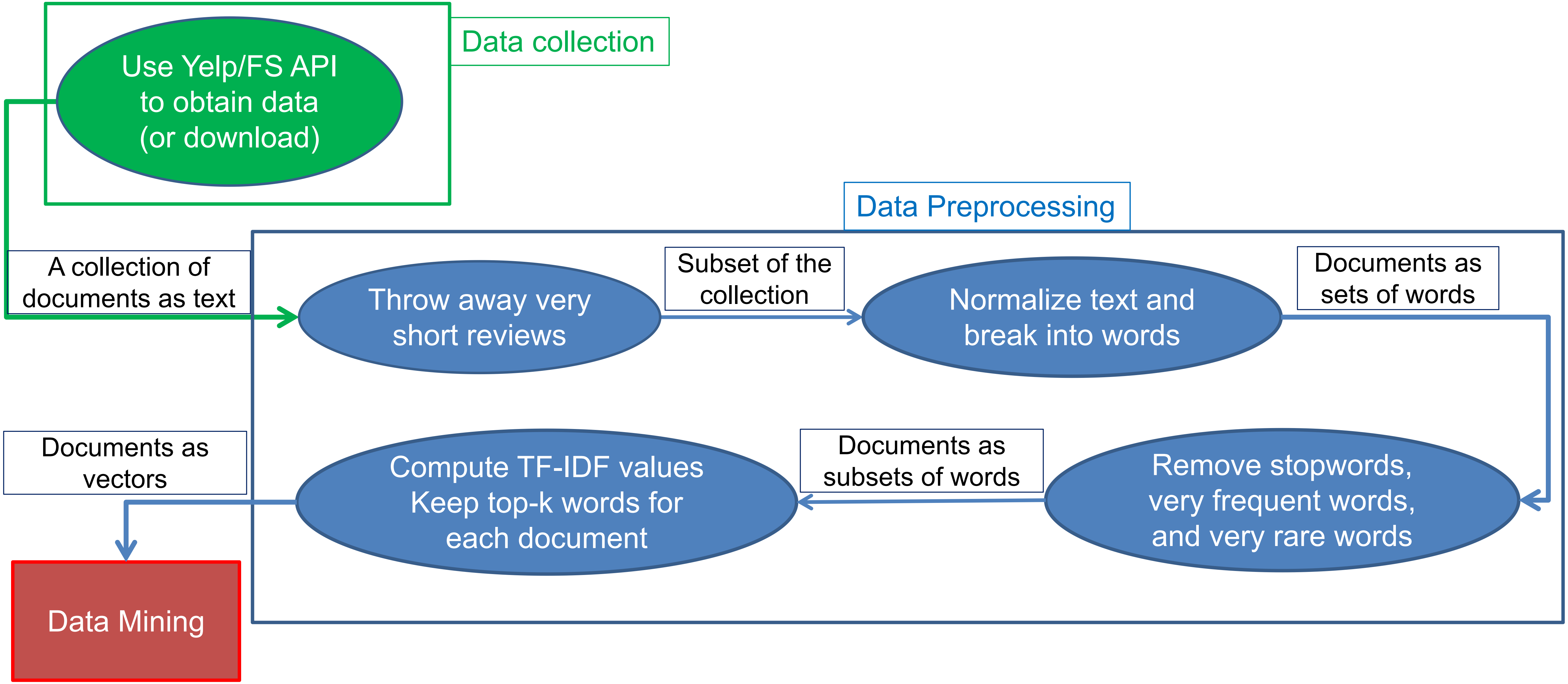
    - Too frequent data (stop words), too infrequent (errors?), erroneous data, missing data, outliers
  - How should we weight the different pieces of data?

- Most decisions are application dependent. Some information may be lost but we can usually live with it (most of the times)

- We should make our decisions clear since they affect our findings.

- Dealing with real data is hard…

The preprocessing pipeline for our text mining task



Word and document representations

- Using TF-IDF values has a very long history in text mining
  - Assigns a numerical value to each word, and a vector to a document
- Recent trend: Use word embeddings
  - Map every word into a multidimensional vector
- Use the notion of context: the words that surround a word in a phrase
  - Similar words appear in similar contexts
  - Similar words should be mapped to close-by vectors
- Example: words “movie” and “film”

The actor for the movie / film Joker is candidate for an Oscar

- Both words are likely to appear with similar words
  - director, actor, actress, scenario, script, Oscar, cinemas etc

word2vec

- Two approaches

CBOW: Learn an embedding for words so that given the context you can predict the missing word

Skip-Gram: Learn an embedding for words such that given a word you can predict the context

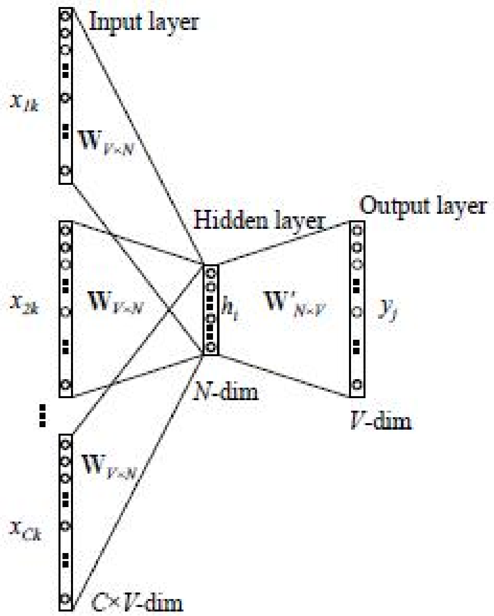


Figure 2: Continuous bag-of-word model

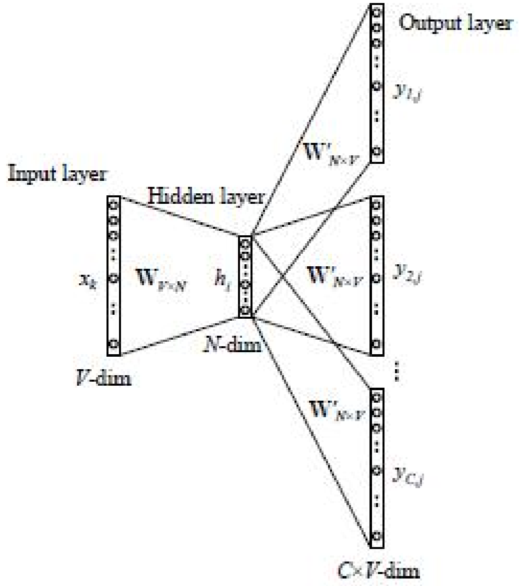


Figure 3: The skip-gram model.

Mikolov, Tomas, et al. "Efficient estimation of word representations in vector space."

DATA NORMALIZATION

Normalization of numeric data

- In many cases it is important to normalize the data rather than use the raw values
- The kind of normalization that we use depends on what we want to achieve

Column normalization

- In this data, different attributes take very different range of values. For distance/similarity the small values will disappear
- We need to make them comparable

Temperature	Humidity	Pressure
30	0.8	90
32	0.5	80
24	0.3	95

Column Normalization

- Divide (the values of a column) by the maximum value for each attribute
  - maximum is 1

Temperature	Humidity	Pressure
0.9375	1	0.9473
1	0.625	0.8421
0.75	0.375	1

new value = old value / max value in the column

Temperature	Humidity	Pressure
30	0.8	90
32	0.5	80
24	0.3	95

Column Normalization

- Subtract the minimum value and divide by the difference of the maximum value and minimum value for each attribute
  - Brings everything in the [0,1] range, maximum is one, minimum is zero

Temperature	Humidity	Pressure
0.75	1	0.33
1	0.6	0
0	0	1

new value = (old value – min column value) / (max col. value –min col. value)

Temperature	Humidity	Pressure
30	0.8	90
32	0.5	80
24	0.3	95

Row Normalization

- Are these documents similar?

	Word 1	Word 2	Word 3
Doc 1	28	50	22
Doc 2	12	25	13

Row Normalization

- Are these documents similar?
- Divide by the sum of values for each document (row in the matrix)
  - Transform a vector into a distribution*

	Word 1	Word 2	Word 3
Doc 1	0.28	0.5	0.22
Doc 2	0.24	0.5	0.26

new value = old value / Σ old values in the row

*For example, the value of cell (Doc1, Word2) is the probability that a randomly chosen word of Doc1 is Word2

	Word 1	Word 2	Word 3
Doc 1	28	50	22
Doc 2	12	25	13

Row Normalization

- Do these two users rate movies in a similar way?

	Movie 1	Movie 2	Movie 3
User 1	1	2	3
User 2	2	3	4

Row Normalization

- Do these two users rate movies in a similar way?
- Subtract the mean value for each user (row) – centering of data
  - Captures the deviation from the average behavior

	Movie 1	Movie 2	Movie 3
User 1	-1	0	+1
User 2	-1	0	+1

new value = (old value – mean row value) [/ (max row value –min row value)]

	Movie 1	Movie 2	Movie 3
User 1	1	2	3
User 2	2	3	4

Row Normalization

- Z-score:

$$z_i = \frac{x_i - \text{mean}(x)}{\text{std}(x)}$$

$$\text{mean}(x) = \frac{1}{N}\sum_{j=1}^{N} x_j$$

$$\text{std}(x) = \sqrt{\frac{\sum_{j=1}^{N}\left(x_j - \text{mean}(x)\right)^2}{N}}$$

Average "distance" from the mean

- Measures the number of standard deviations away from the mean

	Movie 1	Movie 2	Movie 3
User 1	1.01	-0.87	-0.22
User 2	-1.01	0.55	0.93

	Movie 1	Movie 2	Movie 3	Mean	STD
User 1	5	2	3	3.33	1.53
User 2	1	3	4	2.66	1.53

- Individual HW: 分析诗人的诗句，计算TF-IDF，绘制直方图、词云图，了解诗人的创作风格。
- You will be given **two weeks** for a mini report and code submission. Data and templates will be provided.
- Details to be released soon.