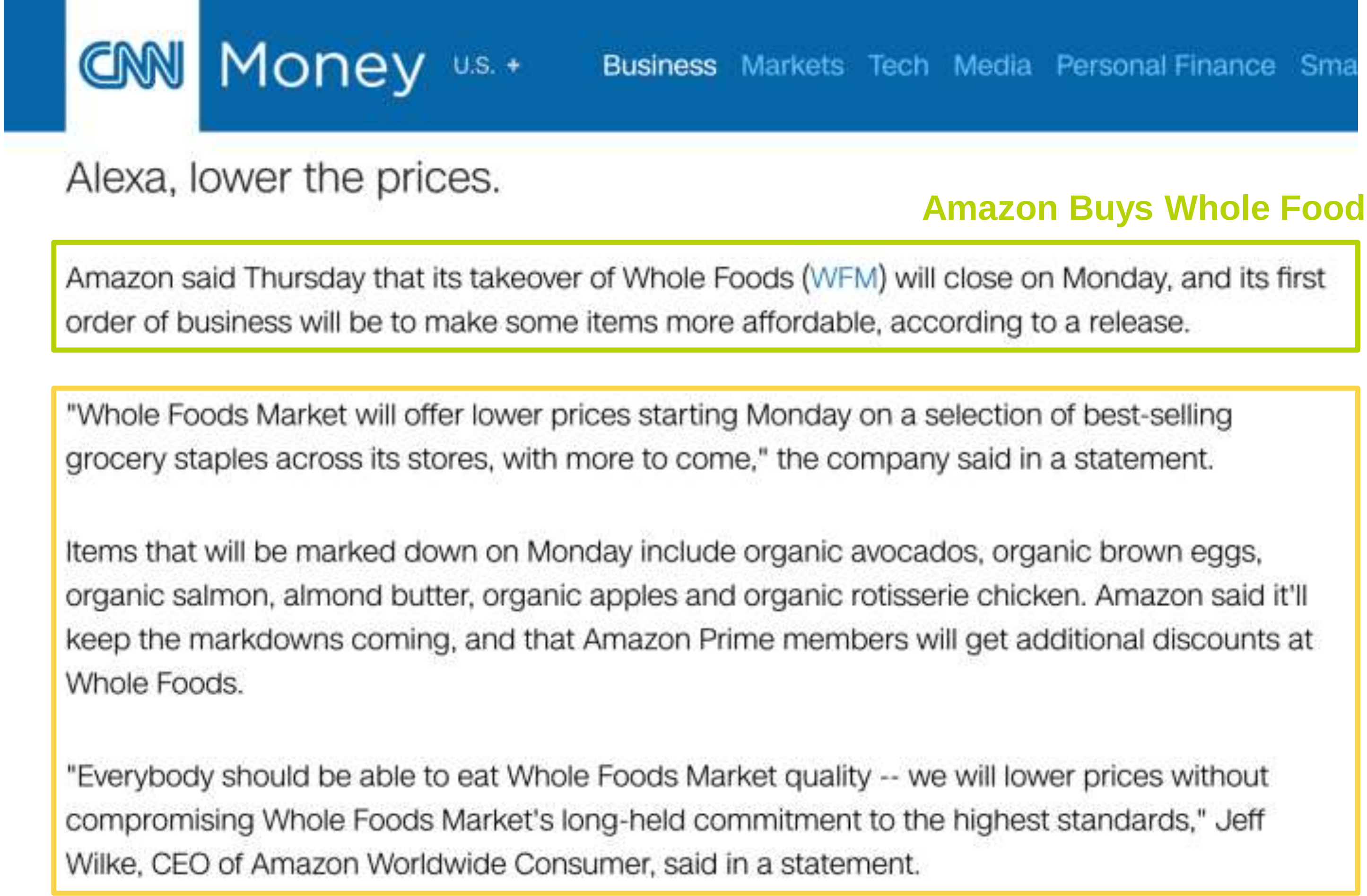
intel® Software

Latent Dirichlet Allocation and Topic Modeling

Topic Modeling

- Every document we read can be thought of as consisting of many topics all stacked upon one another.
- Today, we’re going to focus on how we can unpack these topics using NLP techniques.



Alexa, lower the prices.

Amazon Buys Whole Foods

Amazon said Thursday that its takeover of Whole Foods (WFM) will close on Monday, and its first order of business will be to make some items more affordable, according to a release.

"Whole Foods Market will offer lower prices starting Monday on a selection of best-selling grocery staples across its stores, with more to come," the company said in a statement.

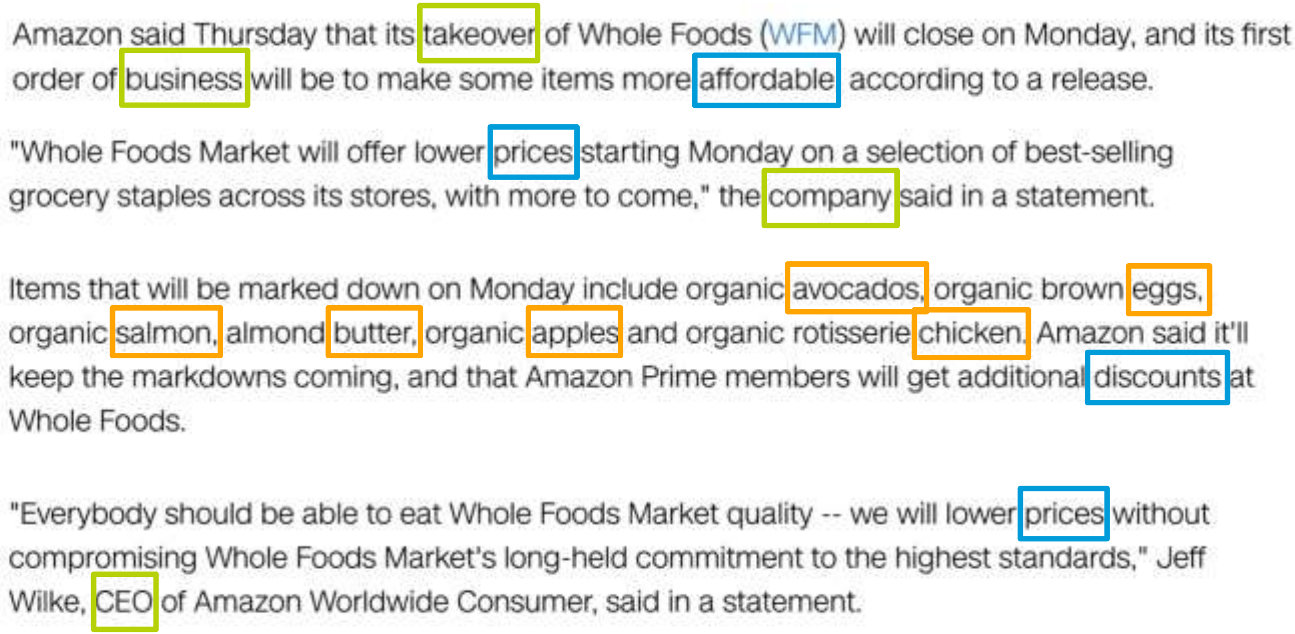
Items that will be marked down on Monday include organic avocados, organic brown eggs, organic salmon, almond butter, organic apples and organic rotisserie chicken. Amazon said it'll keep the markdowns coming, and that Amazon Prime members will get additional discounts at Whole Foods.

"Everybody should be able to eat Whole Foods Market quality -- we will lower prices without compromising Whole Foods Market's long-held commitment to the highest standards," Jeff Wilke, CEO of Amazon Worldwide Consumer, said in a statement.

Food types and price changes

Topic Modeling

- Goal: break text documents down into "topics" by word:



Amazon said Thursday that its takeover of Whole Foods (WFM) will close on Monday, and its first order of business will be to make some items more affordable according to a release.

"Whole Foods Market will offer lower prices starting Monday on a selection of best-selling grocery staples across its stores, with more to come," the company said in a statement.

Items that will be marked down on Monday include organic avocados, organic brown eggs, organic salmon, almond butter, organic apples and organic rotisserie chicken. Amazon said it'll keep the markdowns coming, and that Amazon Prime members will get additional discounts at Whole Foods.

"Everybody should be able to eat Whole Foods Market quality -- we will lower prices without compromising Whole Foods Market's long-held commitment to the highest standards," Jeff Wilke, CEO of Amazon Worldwide Consumer, said in a statement.

Topics:

Business

Prices

Food

Topic Modeling

	Takeover	Close	Prices	Discounts
Doc 1	1	1	1	1
Doc 2	1	1	0	0
Doc 3	0	0	1	1
Doc 4	1	1	1	0
Doc 5	1	0	0	0

Topic Modeling

	Takeover	Close	Prices	Discounts
Doc 1	1	1	1	1
Doc 2	1	1	0	0
Doc 3	0	0	1	1
Doc 4	1	1	1	0
Doc 5	1	0	0	0

In the first document, we see that all of our “words-of-interest” show up. So maybe they’re all just one big topic? We need to investigate more.

Topic Modeling

	Takeover	Close	Prices	Discounts
Doc 1	1	1	1	1
Doc 2	1	1	0	0
Doc 3	0	0	1	1
Doc 4	1	1	1	0
Doc 5	1	0	0	0

If we look at Doc 2, we see that ‘takeover’ and ‘close’ appear together again, but ‘prices’ and ‘discounts’ do not appear. Maybe we have two topics in Doc1, and only one of those topics also shows up in Doc 2.

Topic Modeling

	Takeover	Close	Prices	Discounts
Doc 1	1	1	1	1
Doc 2	1	1	0	0
Doc 3	0	0	1	1
Doc 4	1	1	1	0
Doc 5	1	0	0	0

The inverse happens in Doc 3. So it seems likely that we have 2 topics in document 1, since we see that these don't always have to appear together.

Topic Modeling

	Takeover	Close	Prices	Discounts
Doc 1	1	1	1	1
Doc 2	1	1	0	0
Doc 3	0	0	1	1
Doc 4	1	1	1	0
Doc 5	1	0	0	0

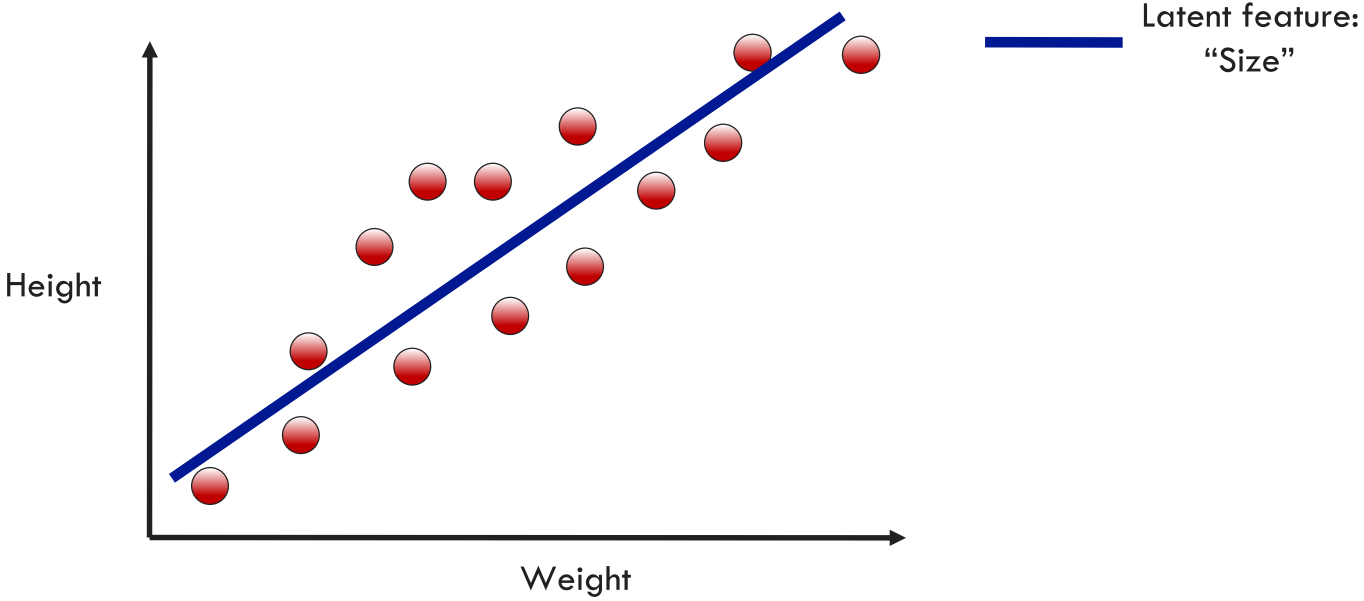
Doc 4 adds a bit of confusion, but we see that ‘takeover’ and ‘close’ appear together again. Even if there’s an extra appearance by ‘prices’ which doesn’t quite fit our original plan. So maybe ’prices’ is a cross-topic word.

Topic Modeling

	Takeover	Close	Prices	Discounts
Doc 1	1	1	1	1
Doc 2	1	1	0	0
Doc 3	0	0	1	1
Doc 4	1	1	1	0
Doc 5	1	0	0	0

Mathematically, we want to find “topics” that are collections of words that appear in similar documents. More generally, a collection of features in a dataset is called a “**latent feature**”. Let’s look at an example

Latent Features: Example



If we had data on people's heights and weights, we would likely observe a high correlation. That is because there is an "latent feature" of "Size" affecting both.

Latent Spaces for NLP

- In NLP, we use topic modeling techniques to find latent features that are collections of individual words.
- Under the hood, what we're doing when creating topics is shifting vectorizer axes in a clever way.
- This creates a new space, that's some combination of our previous columns, that's "hidden" or latent to us.
- We can use this new space to look at each document and understand more about it.
- Let's look at a simple example.

Latent Spaces

“I love my pet rabbit.”

“That dish yesterday was amazing.”

“She cooked the best rabbit dish ever.”

“I gave leftovers of that dish to my pet, mr. rabbit” “Rabbits make messy pets.”

“My rabbit growls when I pet her.” “He has five rabbits.”

“I had this weird dish with fried rabbit.” “That’s my pet rabbit’s favorite dish.”

…

“I love my pet rabbit.”

“That dish yesterday was amazing.”

“She cooked the best rabbit dish ever.”

“I gave leftovers of that dish to my pet, mr. rabbit” “Rabbits make messy pets.”

“My rabbit growls when I pet her.” “He has five rabbits.”

“I had this weird dish with fried rabbit.” “That’s my pet rabbit’s favorite dish.”

…

Remove stop words, only keep nouns, end up with 3 features: “rabbit”, “pet”, “dish”

"I love my pet rabbit."

"That dish yesterday was amazing."

"She cooked the best rabbit dish ever."

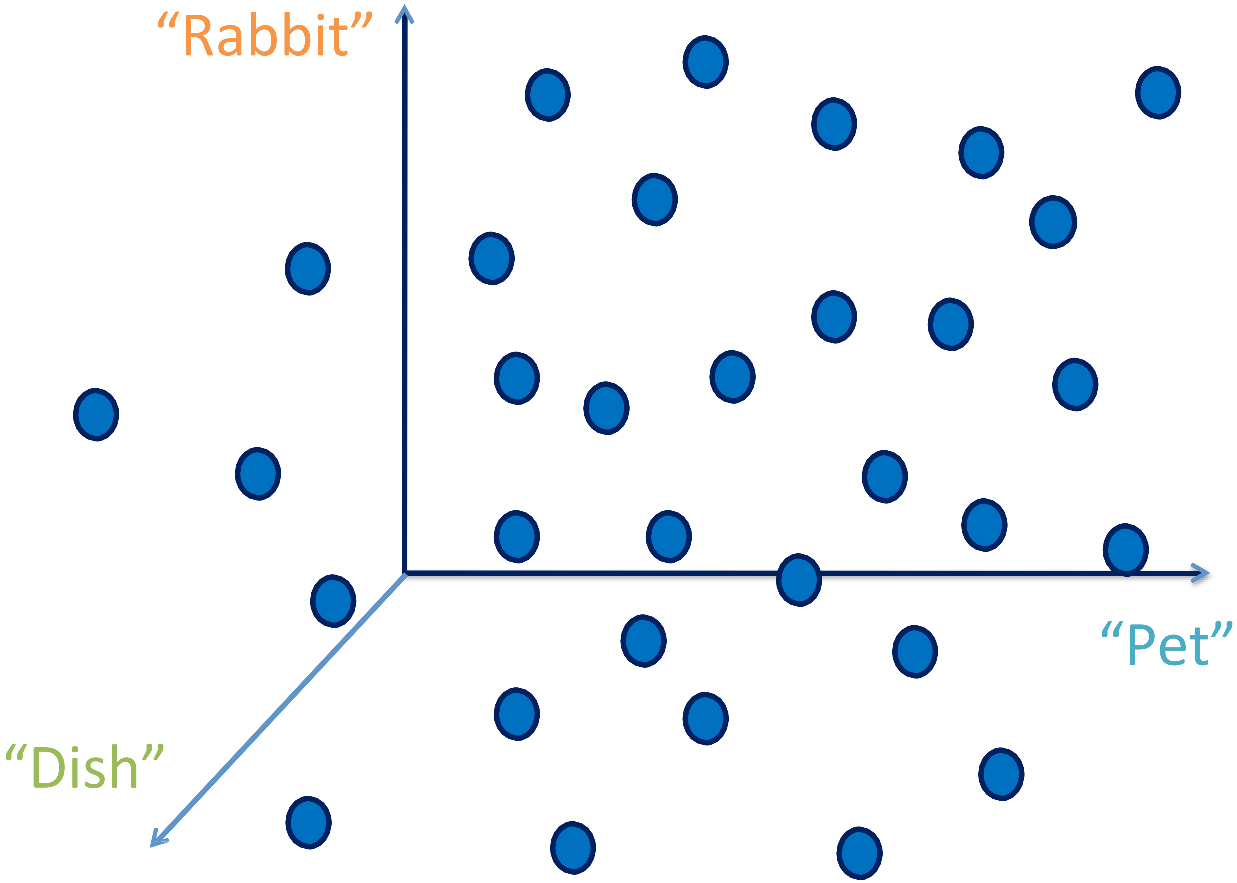
"I gave leftovers of that dish to my pet, mr. rabbit" "Rabbits make messy pets."

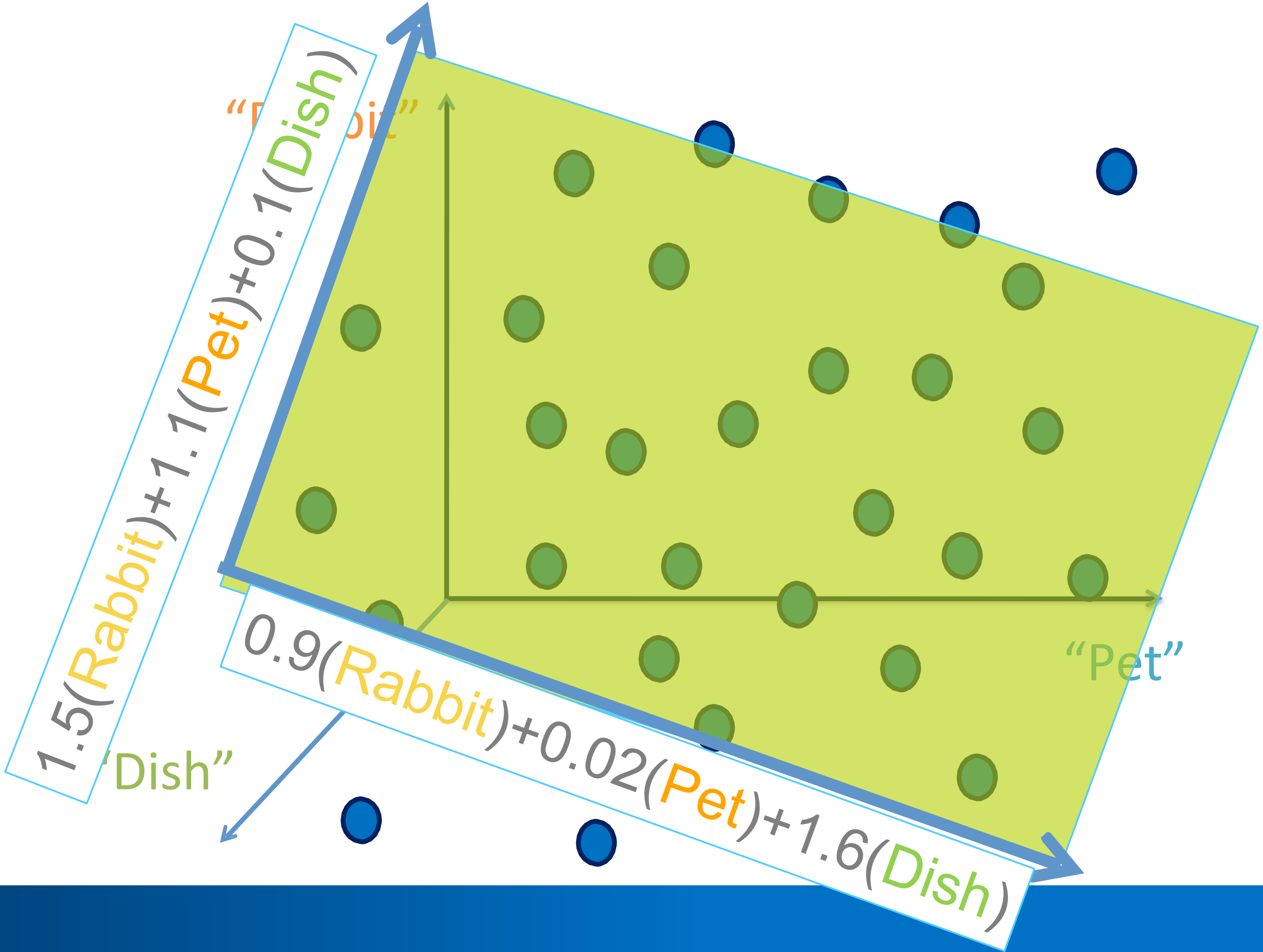
"My rabbit growls when I pet her." "He has five rabbits."

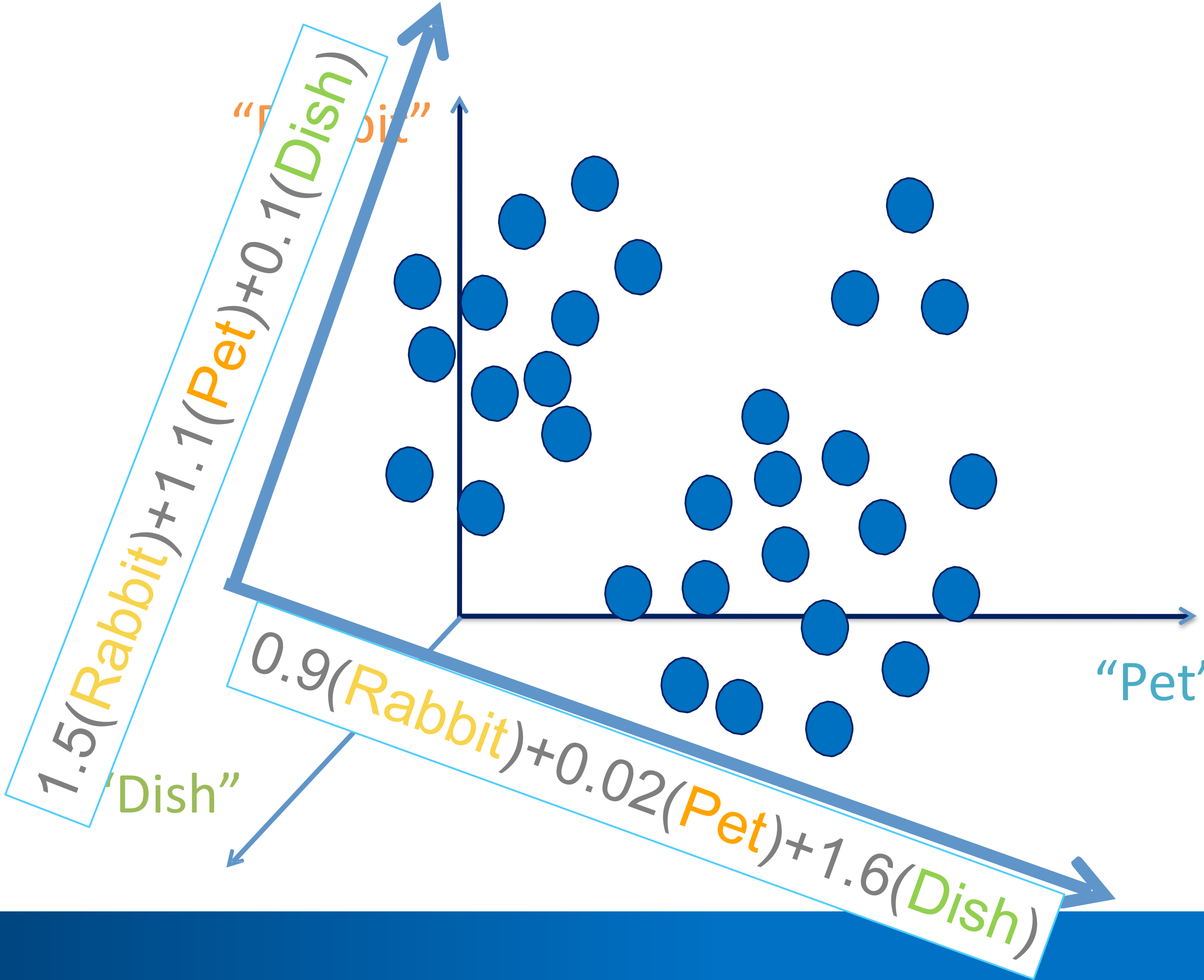
"I had this weird dish with fried rabbit." "That's my pet rabbit's favorite dish."

…

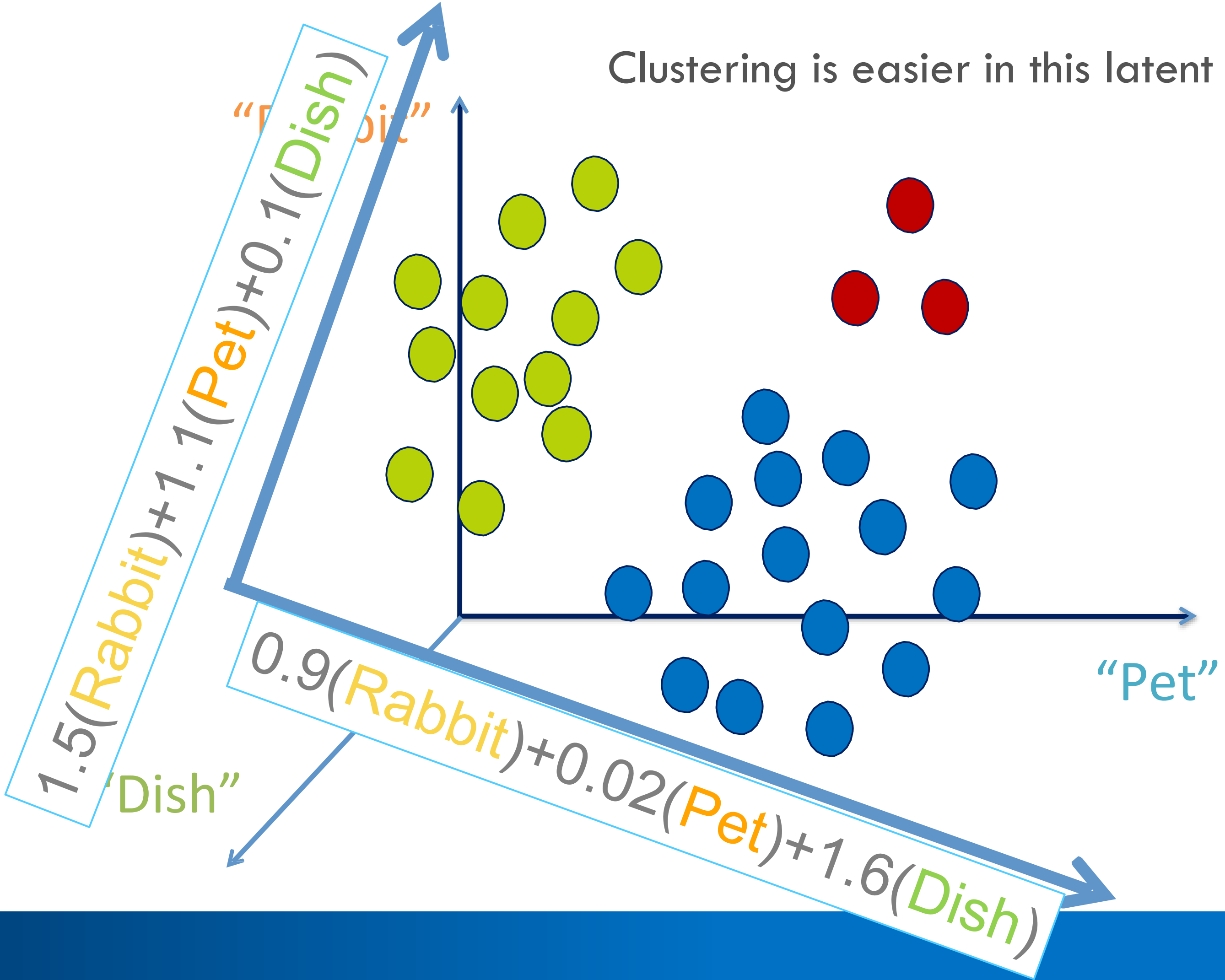
Remove stop words, only keep nouns, end up with 3 features: "rabbit", "pet", "dish"


“Rabbit”
“Pet”
“Dish”


1.5(Rabbit)+1.1(Pet)+0.1(Dish)
"Rabbit"
"Pet"
"Dish"
0.9(Rabbit)+0.02(Pet)+1.6(Dish)


“Rabbit”
“Pet”
“Dish”
1.5(Rabbit)+1.1(Pet)+0.1(Dish)
0.9(Rabbit)+0.02(Pet)+1.6(Dish)

Clustering is easier in this latent space



1.5(Rabbit)+1.1(Pet)+0.1(Dish)
"Rabbit"
"Pet"
"Dish"
0.9(Rabbit)+0.02(Pet)+1.6(Dish)

What are our clusters?

“I love my pet rabbit.” “Rabbits make messy pets.”

“My rabbit growls when I pet her.” “He has five rabbits.”

“That dish yesterday was amazing.”
“She cooked the best rabbit dish ever.”
“I had this weird dish with fried rabbit.”

“I gave leftovers of that dish to my pet, Mr. Rabbit” “That’s my pet rabbit’s favorite dish.”

Axis 1: 1.5(Rabbit) +1.1 (Pet) + 0.1(Dish)

Axis 2: 0.9(Rabbit) + 0.02(Pet) + 1.6(Dish)

Axis 1: High
Axis 2: Low

"I love my pet rabbit." "Rabbits make messy pets."

"My rabbit growls when I pet her." "He has five rabbits."

Axis 1: Low
Axis 2: High

"That dish yesterday was amazing."
"She cooked the best rabbit dish ever."
"I had this weird dish with fried rabbit."

Axis 1: High
Axis 2: High

"I gave leftovers of that dish to my pet, Mr. Rabbit" "That's my pet rabbit's favorite dish."

TOPIC 1: 1.5(Rabbit) +1.1 (Pet) + 0.1(Dish) ← Pet Rabbits

TOPIC 2: 0.9(Rabbit) + 0.02(Pet) + 1.6(Dish) ← Eating Rabbit

TOPIC 1: High
TOPIC 2: Low

"I love my pet rabbit." "Rabbits make messy pets."

"My rabbit growls when I pet her." "He has five rabbits."

TOPIC 1: Low
TOPIC 2: High

"That dish yesterday was amazing."
"She cooked the best rabbit dish ever."
"I had this weird dish with fried rabbit."

TOPIC 1: High
TOPIC 2: High

"I gave leftovers of that dish to my pet, Mr. Rabbit" "That's my pet rabbit's favorite dish."

Topic Modeling

- Documents are not required to be one topic or another. They will be an overlap of many topics.
- For each document, we will have a ‘percentage breakdown’ of how much of each topic is involved.
- The topics will not be defined by the machine, we must interpret the word groupings.

	Pet Rabbits	Eating Rabbit
“I love my pet rabbit.”	87%	13%
“That’s my rabbit’s favorite dish.”	42%	58%
“She cooks the best rabbit dish.”	14%	84%

What is a topic?

- When writing about a topic, certain words are more likely to come up. For example, if I used the words - hoop, backboard, ball, sneakers, and coach – you'd be able to assume I'm talking about basketball.
- A topic is just a conglomeration of words/ideas that tend to appear together.
- Mathematically, a topic is a probability distribution over all possible words.

What is a topic?

Word	Probability in “Pet Rabbit”	Probability in “Eating Rabbit”
pet	2.3E-7	1.2E-10
rabbit	7.9E-7	3.4E-8
dish	6.8E-11	4.5E-7
car	3.1E-12	1.8E-12
plate	8.3E-14	1.4E-9
affair	3.0E-13	3.1E-13
love	5.4E-8	3.9E-10
the	3.1E-5	3.2E-5

What is a topic?

Word	Probability in “Pet Rabbit”	Probability in “Eating Rabbit”
pet	2.3E-7	1.2E-10
rabbit	7.9E-7	3.4E-8
dish	6.8E-11	4.5E-7
car	3.1E-12	1.8E-12
plate	8.3E-14	1.4E-8
affair	3.0E-13	3.1E-13
love	5.4E-8	3.9E-10
the	3.1E-5	3.2E-5

Topic Modeling Summary

- Choose some algorithm to figure out how words are related
- Create a latent space that makes use of those in-document correlations to transition from a “word space” to a latent “topic space”
- Each axis in the latent space represents a topic
- Construct a probabilistic understanding of how much each topic contributes to each document.
- We, as the humans, must understand the meaning of the topics. The machine doesn’t understand the meanings, just the correlations.

Let’s look at some example topics from the Fetch20 Dataset

```
Topic 0:
jesus matthew said people den col prophecy int away war men messiah den
den radius prophet row isaiah psalm row col sea

Topic 1:
year game good team think don just games like players better runs hit
won league time baseball season win pitching

Topic 2:
graphics image edu data mail software ftp pub available send images
package computer information use files thanks program processing code
```

It’s still on us to interpret these topics.

Let’s look at some example topics from the Fetch20 Dataset

Atheism

jesus matthew said people den col prophecy int away war men messiah den den radius prophet row isaiah psalm row col sea

Baseball

year game good team think don just games like players better runs hit won league time baseball season win pitching

Graphics

graphics image edu data mail software ftp pub available send images package computer information use files thanks program processing code

Latent Dirichlet Allocation (LDA)

LDA is one of the algorithms we can choose to convert between word and topic spaces. It works by simulating the process of writing.

- Choose a topic to write about
- Choose words that are about that topic
- Add those words to the document

Start by deciding which topics will make up the document and in what percentage:

Technology: 50%   Music: 25%   Movies: 25%

Start by deciding which topics will make up the document and in what percentage:

Technology: 50%   Music: 25%   Movies: 25%

Now we roll a dice to decide which topic we start with: Technology

Start by deciding which topics will make up the document and in what percentage:

Technology: 50%   Music: 25%   Movies: 25%

Computer

Now we roll a dice to decide which topic we start with: Technology

Now we roll a new dice within the technology topic to see which word and we get: Computer

Now we repeat the process for the next word.

Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%

Computer screen

Now we roll a dice to decide which topic we start with: Technology

Now we roll a new dice within the technology topic to see which word and we get: screen

Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%

Computer screen the

Now we roll a dice to decide which topic we start with: Technology

Now we roll a new dice within the technology topic to see which word and we get: the

Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%

Computer screen the guitar

Now we roll a dice to decide which topic we start with:
Music

Now we roll a new dice within the technology topic to see which word and we get: guitar

Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%

Computer screen the guitar theatre

Now we roll a dice to decide which topic we start with:

Movies

Now we roll a new dice within the technology topic to see which word and we get: theatre

Start by deciding which topics will make up the document and in what percentage:

Technology: 50% Music: 25% Movies: 25%

Computer screen the guitar theatre headphones. Actress sings best mouse performance.

We can continue this process over and over until we’ve generated essentially an entire document. This document will be made up of the topics we selected, and the words that are associated with that topic, all based on the probabilities.

Latent Dirichlet Allocation in Practice

We typically do this in reverse on the modeling side, relying on Bayesian methodology.

- Assume some topic distribution in the corpus
- Assume some word distribution for each topic
- Look at the corpus and try to find what topic and word distributions would be most likely to generate that data.
- For this to converge, we need to tell it how many topics to look for.

Why Dirichlet?

- Any probability distribution could be used for assigning the words to topics, it's just a Bayesian process where we must select a prior. So why Dirichlet?
- The sparse Dirichlet distribution assumes that each topic will only be made from a small subset of the total available space.
- In our case, the vast majority of words are not related to technology (or any other topic), so assuming that only a small subset of the total word probabilities are large makes sense.
- Dirichlet does a better job than other choices for the Bayesian prior.

LDA in Python

- There are several libraries for LDA: two popular ones are Scikit-learn and Gensim
- Scikit-learn is nice because it quickly integrates with the other, more familiar Scikit-learn/NLTK tools and has a consistent API.
- GenSim is another Python text processing package we'll use for Word2Vec and a few other tools. It also has LDA as a built in package.
- We can either prepare our corpus with GenSim tools or use the `corpus = matutils.Sparse2Corpus(counts)` tool to convert from CountVectorizer to a format GenSim can use.
- For the exercises and following slides, you'll see the Scikit-Learn API; you can see a conceptual Gensim example in the appendix.

Code: LDA (Scikit-learn)

Input:

```
from sklearn.datasets import fetch_20newsgroups


ng_train = fetch_20newsgroups(subset=‘train’,
                              remove=(‘headers’, ‘footers’, ‘quotes’))


print("Data has {0:d} documents".format(len(ng_train.data)))
print(ng_train.data[0][:100]) # print requires Python 3
```

Output:

```
Data has 11314 documents
'I was wondering if anyone out there could enlighten me on this car I saw\nthe
other day. It was a 2-d’
```

Code: LDA (Scikit-learn)

Input:

```
from sklearn.feature_extraction.text import CountVectorizer


count_vectorizer = CountVectorizer(ngram_range=(1, 2),
                                   stop_words='english',
                                   token_pattern=“\\b[a-z][a-z]+\\b",
                                   lowercase=True,
                                   max_features=1000)


X = count_vectorizer.fit_transform(ng_train.data)

# “X” is now our transformed data
```

Code: LDA (Scikit-learn)

Input:

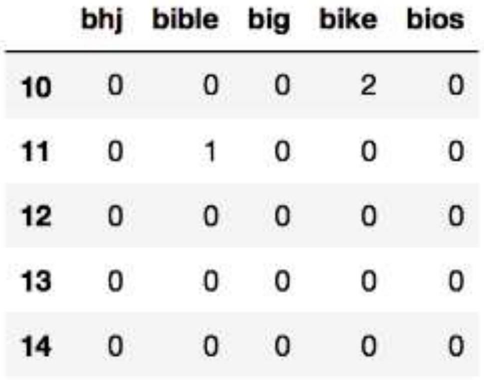
```
import pandas as pd


print(count_vectorizer.get_feature_names()[92:97]) # print 5 random columns
df = pd.DataFrame(X.toarray(), columns=count_vectorizer.get_feature_names())
# create data frame
print(df.iloc[10:15, 92:97]) # values of these features on documents 10-15
```

Output:

```
“['bhj', 'bible', 'big', 'bike', ‘bios']"
```

df:



	bhj	bible	big	bike	bios
10	0	0	0	2	0
11	0	1	0	0	0
12	0	0	0	0	0
13	0	0	0	0	0
14	0	0	0	0	0

Code: LDA (Scikit-learn)

Input:

```
from sklearn.decomposition import LatentDirichletAllocation


lda = LatentDirichletAllocation(n_topics=4, random_state=42,
                                     learning_method='online')


data = lda.fit_transform(X)

print(data[0])
```

Output:

```
[ 0.00246896, 0.00251041, 0.99253159, 0.00248904]
```

Code: LDA (Scikit-learn)

Input:

```
from sklearn.decomposition import LatentDirichletAllocation


lda = LatentDirichletAllocation(n_topics=4, random_state=42,
                                     learning_method='online')


data = lda.fit_transform(X)

print(data[0])
```

Output:

```
[ 0.00246896, 0.00251041, 0.99253159, 0.00248904]
```

This document is 99% topic 3!

Code: LDA (Scikit-learn)

Input:

```
for word in lda.components_[2].argsort()[:10:-1]:
    print(word)
```

Output:

```
year game good team think don just games like players
```

The top 10 words for topic 2!

intel®

Software

Appendix

Code: LDA (gensim)

Input:

```
from gensim import corpora, models, similarities, matutils

lda = LdaModel(corpus, num_topics=5) # train model

print(lda[doc_bow]) # get topic probability distribution for a document

lda.update(corpus2) # update the LDA model with additional documents

print(lda[doc_bow])
```

Output:

```
[(2: 0.95, 4: 0.21)]
[(0: 0.75, 1: 0.15, 5: 0.11)]
```

Code: LDA (gensim)

Input:

```
lda.print_topics()
```

Output:

```
[(0, u'0.002*"image" + 0.002*"don" + 0.002*"jpeg" + 0.002*"good" +
0.001*"think" + 0.001*"people" + 0.001*"file" + 0.001*"year" + 0.001*"just" +
0.001*"like"'),

(1, u'0.002*"graphics" + 0.002*"edu" + 0.002*"god" + 0.001*"just" +
0.001*"like" + 0.001*"does" + 0.001*"people" + 0.001*"know" + 0.001*"data" +
0.001*"jesus"'),

(2, u'0.002*"think" + 0.002*"don" + 0.002*"just" + 0.002*"like" + 0.002*"does"
+ 0.002*"god" + 0.001*"know" + 0.001*"people" + 0.001*"time" + 0.001*"good"')
… ]
```