Week 8: Working with Text and Databases

8.1 Engagement: Working With Databases

Relational Data Model

By the end of this video, you should be able to:

- Describe the structural components of a relational data model
- Demonstrate which components make up a data model's 'schema'
- Explain the purpose of primary and foreign keys
- Describe a "Join" operation

In this lesson, we will review a Jupyter Notebook to interact with a relational database well known to the data science community. Before we do that, let's spend a few minutes reviewing what we mean when we say relational data model, and how to interact with relational data.

By the end of this video, you should be able to describe the structural components of a relational data model, demonstrate which components make up a data model's schema, explain the purpose of primary and foreign keys, and describe a join operation.

A Collection of Tables

ID	FName	LName	Department	Title	Salary
202	John	Gonzales	IT	DB Specialist	104750
203	Mary	Roberts	Research	Director	175400
204	Janaki	Rao	HR	Financial Analyst	63850
205	Alex	Knight	IT	Security Specialist	123500
206	Pamela	Ziegler	IT	Programmer	85600
207	Harry	Dawson	HR	Director	115450

No Duplicates

ID	FName	LName	Department	Title	Salary
202	John	Gonzales	IT	DB Specialist	104750
203	Mary	Roberts	Research	Director	175400
204	Janaki	Rao	HR	Financial Analyst	63850
205	Alex	Knight	IT	Security Specialist	123500
206	Pamela	Ziegler	IT	Programmer	85600
207	Harry	Dawson	HR	Director	115450
~~207~~	~~Harry~~	~~Dawson~~	~~HR~~	~~Director~~	~~115450~~

The primary data structurefor a relational model is a table, like the one shown here for a toy application. You might have also noticed that table structure is very similar to a Python DataFrame. DataFrames can be used to load relational tables which are also called relations. This table actually represents a set of tuples. This is a relational tuple represented as a row in the table. We were informally calling this a record before but now we call it a tuple. Thus, this relation is a set of six tuples. There are six rows and six tuples.
Remember the definition of sets, it's a collection of distinct elements of the same type. That means I cannot add the red record at the end of this table as a tuple into this relation because if I do, it will be introducing a duplicate. In practice, many systems will allow duplicate tuples in their relation but mechanisms are provided to prevent duplicate entries if the user so chooses.

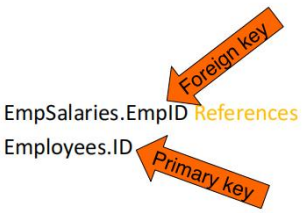
Dissimilar Tuples Disallowed

ID	Fname	Lname	Department	Title	Salary
202	John	Gonzales	IT	DB Specialist	104750
203	Mary	Roberts	Research	Director	175400
204	Janaki	Rao	HR	Financial Analyst	63850
205	Alex	Knight	IT	Security Specialist	123500
206	Pamela	Ziegler	IT	Programmer	85600
207	Harry	Dawson	HR	Director	115450
~~Jane~~	~~Doe~~	~~208~~	~~Res. Associate~~	~~65800~~	~~Research~~

Foreign Keys

EmpSalaries

EmpID	Date	Salary
202	1/1/2016	104750
203	2/15/1016	175400
204	6/1/2015	63850
205	9/15/2015	123500
206	10/1/2015	85600
207	4/15/2015	115450
202	9/15/2014	101250
204	3/1/2015	48000
207	9/15/2013	106900
205	10/1/2014	113400



Here is another tuple I cannot add.The red tuple on the bottom has all the right pieces of information but it is all in the wrong order. So, this tuple is dissimilar with the other six tuples in the relation. Okay, how does the system know that this tuple is different? This brings our attention to the first row, that is the header of this table painted in black. This row is part of the schema of the table. These are our columns in a Pandas DataFrame. The schema in a relational table can also specify keys. The first column says that ID is a primary key. This means it's unique for each employee. And for every employee knowing the primary key for an employee, we also uniquely know the other five attributes for that employee like first name, last name, department, title and salary. You should now see that a table with a primary key logically implies that the table cannot have a duplicate record because if you do, it will violate the uniqueness constraint associated with the primary key.
I will call our first table employeeswith the ID as the primary key. Let us now introduce a new table containing the salary history of employees. The employees are identified by the column employee ID, EmpID but these are not new values that this table happens to have. They are the same IDs that are present in the ID column of the employees table presented earlier. This is reflected in the statement made on the right. The term references means the values in this column can exist only if the same values appear in employees, the table being referenced. This table is also called the **parent table**. In the terminology of the relational model, the EmpID column of the EmpSalaries table is called a foreign key that refers to the primary key of the employees table. Note that EmpID is not a primary key in the EmpSalaries table because it has multiple tuples with the same EmpIDs reflecting the salary of the employee at different times.

总结:

ID	FName	LName
202	John	Gonzales
203	Mary	Roberts
204	Janaki	Rao
205	Alex	Knight
206	Pamela	Ziegler
207	Harry	Dawson

EmpID	Date	Salary
202	1/1/2016	104750
203	2/15/1016	175400
204	6/1/2015	63850
205	9/15/2015	123500
206	10/1/2015	85600
207	4/15/2015	115450
202	9/15/2014	101250
204	3/1/2015	48000
207	9/15/2013	106900
205	10/1/2014	113400

Joining Relations

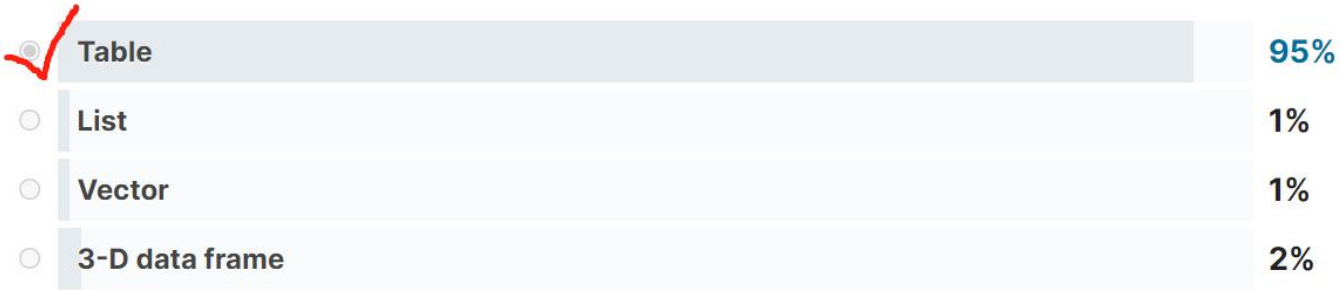
ID	FName	LName	Date	Salary
202	John	Gonzales	1/1/2016	104750
202	John	Gonzales	9/15/2014	101250
203	Mary	Roberts	2/15/1016	175400
204	Janaki	Rao	6/1/2015	63850
204	Janaki	Rao	3/1/2015	48000
205	Alex	Knight	9/15/2015	123500
205	Alex	Knight	10/1/2014	113400
206	Pamela	Ziegler	10/1/2015	85600
207	Harry	Dawson	4/15/2015	115450
207	Harry	Dawson	9/15/2013	106900

You may remember join as an operation we discussedas a part of Pandas. Here's an example of a relational join performed on the first three columns of employee and the EmpSalaries table where employees.ID and the EmpSalaries that EmpID columns are matched for equality. The output table shows all columns involved. The common column is represented only once. This form of join is called a natural join. It is important to understand that join is one of the most expensive, that is time and space consuming, operations. As data becomes larger and the tables contains hundreds of millions of tuples, the join operation can easily become a bottleneck in a larger analytical application. So, for data science that involves big data, when we need joins, it's very important to choose a suitable data management platform that makes its operation efficient.

As a summary,the relational model provides a way to describe unique relationships between entities and data records, including the relationships between record identifiers. Pandas DataFrames implements some features from the relational data model, making it easier to work with relational databases.

What is the primary data structure for a relational data model?

RESULTS



总结:

Structured Query Language

By the end of this video, you should be able to:

- Describe what data retrieval means
- Explain the purpose of SQL
- Create simple SELECT queries

What is Data Retrieval?

- Data retrieval
  - The way in which the desired data is specified and retrieved from a data store
- Our focus
  - How to specify a data request
  - The internal mechanism of data retrieval

Now that we simply went over the relational model without going into the details, we will talk about how we can retrieve data from a relational database. By the end of this video, you should be able to describe what data retrieval means, explain the purpose of SQL, and create simple SELECT queries.

Data retrieval refers to the way in whichdata desired by a user is specified and retrieved from a data source. Note that in this course, we are using the term data retrieval in two ways. I assume that your data is stored in a data store that follows a specific data model, like, for example, the relational data model. By data retrieval, we will refer to the way you specify how to get the desired data out of the relational data store, and the internal processing that occurs within the data management system to compute or evaluate that specified retrieval request.

Structured Query Language

- The standard for structured data
- Example Database Schema

Bars(name, addr, license)
Beers(name, manf)
Sells(bar, beer, price)

name	addr	license
Great American Bar	363 Main St., SD, CA 92390	41-437844098
Beer Paradise	6450 Mango Drive, SD, CA 92130	41-973428319
Have a Good Time	8236 Adams Avenue, SD, CA 92116	32-032263401

SELECT-FROM-WHERE

- Which beers are made by Heineken?

```
SELECT name
FROM Beers
WHERE manf = 'Heineken'
```

Output attribute(s)

Table(s) to use

The condition(s) to satisfy

Strings like 'Heineken' are case-sensitive and are put in quotes

name
Heineken Lager Beer
Amstel Lager
Amstel Light
...

Select $_{manf='Heineken'}$ (Beers)

⬇

Project(name)

Now let's see the query specificationhappening using a specific language called Structured Query Language, or SQL. SQL is the ubiquitous query language when the data is structured, but it has been extended in many ways to accommodate other types of data. For this lesson, we will stick to the structured aspect of the language. You should know that SQL is used in classical database management systems like Oracle as well as modern distributive big data systems such as Spark in the form of Spark SQL. Let's work, now, with an illustrative example. Our schema for this business has three relations, or tables. The first table lists these bars, their name, address, and the license number of the bar. Notice that the attribute called name is underlined because it is the primary key of the bars relationship. Recall that primary key refers to a set of attributes, in this case just name, that make a record unique. Note that the relation bars with the attribute names within parentheses is the same as the table shown on the right. We will use both representations as we go forward. The second table, called beers, lists the names and manufacturers of beer. Not every bar sells the same brands of beer. Even when they do, they may have different prices for the same product, because difference in establishment cost. So the sells table records which bar sells which beer at what price.

The most basic structure of a SQL queryis a select from where clause, in this example where we are looking for beer names that are made by Heineken. So we need to specify our attribute, which is our output, in this case the name of the beer, the logical table which should be used to answer the query, in this case beers, and the condition that all the desired data items should satisfy, namely the value of the attribute called manf is equal to Heineken. There are a few things to notice here. So our query is select name from beers, where manf equals Heineken. First, the literal Heineken is put within quotes because it's a string literal. Remember that, in this case, the string is supposed to match exactly, including the case. Secondly, if you go back to the data operations we discussed in pandas, you will recognize that this form of query can also be represented as a selection operation on the relation beers with the condition on the manf attribute, followed by a projection operation that outputs the name attribute from the result of the selection operation. So the selection operation finds all tuples in beers for which the manufacturer is Heineken, and from those tuples it projects only the name column. The result of this query is a table with one single attribute called name. This would be the column inside pandas or a series object.

总结:

More Example Queries

- Find expensive beer
  - SELECT DISTINCT beer, price
  - FROM Sells
  - WHERE price > 15
- Which businesses have a Temporary License (starts with 32) in San Diego?
  - SELECT name
  - FROM Bars
  - WHERE addr LIKE '%SD%' **AND** license LIKE '32%' LIMIT 5

name	addr	license
Great American Bar	363 Main St., SD, CA 92390	41-437844098
Beer Paradise	6450 Mango Drive, SD, CA 92130	41-973428319
Have a Good Time	8236 Adams Avenue, SD, CA 92116	32-032263401

Summary

- SQL is the standard querying language for structured relational data
- Resembles pandas data frames operations
- Allow for selection of data and more

We illustrate some more features of SQL using two more example queries. The first looks for expensive beer and it's price. Select distinct beer and price, from sells, where price is larger than 15. Let's say, because they're beers, to be expensive if it costs more than 15 dollars a bottle, from the schema we know that price information is available from the table called Sells. So the FROM clause should use Sells in this case. The WHERE clause is intuitive and specifies the price to be greater than 15. Now, notice that the sells relationship also has a column called bar. Now, if two different bars sells the same beer at the same price, we will get both entries in the result. That's not what we want. Regardless of the multiplicity of bars that have the same price for the same beer, we want the result just once. This is achieved through the SELECT DISTINCT statement in the first row, which ensures that the result relationship has no duplicates.

The second example shows the casewhere more than one condition must be satisfied by the result. In this query, the business must be in San Diego, and at the same time it must be a temporary license holder, which means the license number should start with 32. As we see here, these conditions are put together by the AND operator. Thus, this query will pick the third record in the table, because the first two records satisfy the first condition and not the second condition. One can also place a limit on the number of results to return. If our database is large and we need only five results, for example, a sample to display, we can say LIMIT 5. The exact syntax of this LIMIT clause may vary between database management system vendors or even in Python.

As a summary, SQL is the standard querying languagefor structured relational data, and it resembles pandas dataframes, although it can provide more operations. SQL allows for joining and other operations in addition to the simple data selection query as we focused on in this video, but we will not go into the details for those operations, and we'll show you a notebook next, how to use some of these selection queries in the context of Python Jupyter Notebooks.

Live Code, Structured Query Language

详见 jupyter notebook: Working with Databases.ipynb

总结:

8.2 Engagement: Natural Language Processing with NLTK

nltk

By the end of this video, you should be able to:

- Describe what natural language processing (NLP) means
- List a few examples of NLP in everyday life
- Explain what NLTK is

Intro to Natural Language Processing (NLP)

- Algorithms to analyze, understand and derive meaning from human language
- Hard computational problem because human language is **ambiguous**, needs **context** and ability to **link concepts**
- **Applications:** summarize text, generate keywords, identify sentiment of text

In this lesson, we will focus onNatural Language Processing using a popular Python package called NLTK. By the end of this video, you should be able to describe what Natural Language Processing, or NLP, means, list a very examples of NLP in everyday life, and explain what NLTK is.

Natural Language Processing, or shortly NLP, is a data science term used to refer to the interaction of computers, and natural language humans use. This is not an easy task because human language is ambiguous. As humans, we are good at understanding the context of something said to us, and link that to our understanding of the concepts around it. Doing this algorithmically, however, is not trivial. This is what the field of NLP tries to improve and develop algorithms and data techniques that make it possible in an effective and fast fashion. Chances are, you have used an NLP application, if you used an online summary of news or books, looked for keywords generated for most popular twitter topics, or used a virtual assistant on your phone.

Real Life examples of NLP

- Speech recognition engines like Siri, Google Now or Alexa
- Automatic translation like Google Translate or Facebook automatic translation of statuses
- Chat bots that can answer question via Facebook Messenger, for example provided by Techcrunch, Disney or Whole Foods

nltk

Natural Language Toolkit in Python

- Work with human language data
- Includes over 50 datasets
- Complete library of easy to use algorithms for processing text
- Available for free under open source license

http://nltk.org

Let's list some of these applications.NLP techniques are applied in speech recognition engines, like Siri, Google Now, or Alexa. These engines are designed to learn what and how a human talks over time, and constantly improve their accuracy. Similarly, automatic translators, like Google Translate or Facebook automatic translation of statuses use an LP, using some recent very effective neural network based techniques that take not only words and phrases into account, but also the context, by looking at the word surrounding the text they are translating. Chat bots that can answer questions via Facebook Messenger are another example of NLP. They use NLP engines to process the questions, often simply categorize them, and match them to existing answers to your question.

In this lesson, we will go through a notebookthat uses NLTK for Natural Language Processing. NLTK is the most popular Python package for NLP. It is an open source library that provides modules for importing, cleaning, pre-processing text data, in human language, and then apply computational linguistics algorithms, or machine learning algorithms, like sentiment analysis, to these datasets. It also provides over 50 datasets to start working with, including the movie database we will be using in our example notebook.

Let's get started with our notebook.

总结:

nltk corpora

另详见 jupyter notebook: Natural Language Processing of Movie Reviews using nltk.ipynb

By the end of this video, you should be able to:

- Describe what corpus means
- List some datasets in the nltk corpora
- Recite the basic features of the movie reviews corpus in nltk

nltk corpora

corpus (plural corpora) is a collection of text in digital form, assembled for text processing

nltk provides a **download interface** to pre-processed text datasets.

Hope you were able to explore the datasetsnltk offers interactively. Now, we will review of the corpora nltk provides. By the end of this video, you should be able to describe what corpus means, list some datasets in the nltk corpora, and recite the basic features of the movie reviews corpus in nltk.
NL techniques depends on large amounts of text or other linguistics data. These digital collections are called corpora all together. Another word you will hear is a corpus, which is the singular form of corpora. As you have seen in our brief live session, nltk provides means to download some these large datasets.

```
nltk.download()
```

```
NLTK Downloader
---------------------------------------------------------------------------
    d) Download   l) List    u) Update   c) Config   h) Help   q) Quit
---------------------------------------------------------------------------
Downloader> l

Packages:
  [ ] abc..................... Australian Broadcasting Commission 2006
  [ ] alpino.................. Alpino Dutch Treebank
  [ ] averaged_perceptron_tagger Averaged Perceptron Tagger
  [ ] averaged_perceptron_tagger_ru Averaged Perceptron Tagger (Russian)
  [ ] basque_grammars..... Grammars for Basque
  [ ] biocreative_ppi..... BioCreAtIvE (Critical Assessment of Information
                           Extraction Systems in Biology)
  [ ] bllip_wsj_no_aux.... BLLIP Parser: WSJ Model
  [ ] book_grammars....... Grammars from NLTK Book
  [ ] brown............... Brown Corpus
  [ ] brown_tei........... Brown Corpus (TEI XML Version)
  [ ] cess_cat............ CESS-CAT Treebank
  [ ] cess_esp............ CESS-ESP Treebank
  [ ] chat80.............. Chat-80 Data Files
  [ ] city_database....... City Database
  [ ] cmudict............. The Carnegie Mellon Pronouncing Dictionary (0.6)
  [ ] comparative_sentences Comparative Sentence Dataset
  [ ] comtrans............ ComTrans Corpus Sample
  [ ] conll2000........... CONLL 2000 Chunking Corpus
  [ ] conll2002........... CONLL 2002 Named Entity Recognition Corpus

Hit Enter to continue:
```

nltk movie reviews corpus

nltk.download("movie_reviews")

```
~ altintas$ ls nltk_data/corpora/movie_reviews

   README neg pos
```

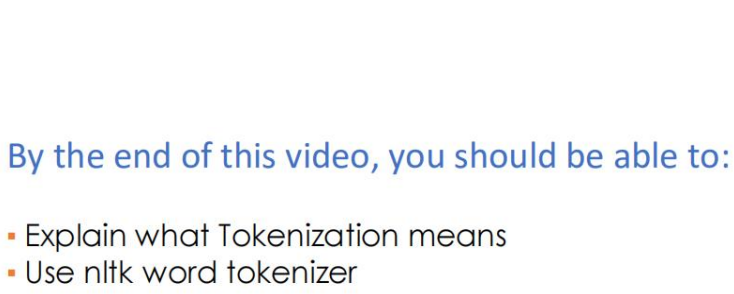
2000 files:

- 1000 positive reviews in the pos/ folder
- 1000 negative reviews in the neg/ folder
- average 800 words per review

总结:

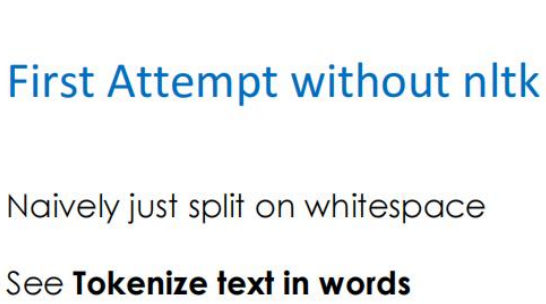
tokenize

另详见 jupyter notebook: Natural Language Processing of Movie Reviews using nltk.ipynb



Now let's talk about Tokenizing the words and text.By the end of this video, you should be able to explain what Tokenization means, and use an nltk word Tokenizer.

The first step in NLP is generally to split the text into words. This process might appear simple, but it is very tedious to handle all corner cases. What are corner cases? They include inconsistent use of punctuation, or contractions, or shortened versions of words. They can also be hyphenated words that include characters like the New York-based example here.



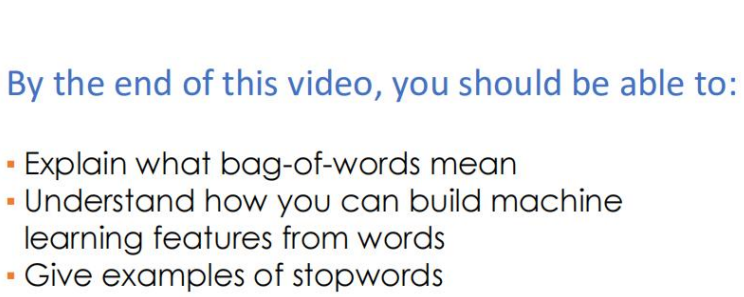
How do we Tokenize such cases?Nltk offers libraries to remedy these challenges. When we switch to a notebook in a few seconds, we will first use a simple white space based Tokenizer. Then, we will learn how to do it better and easier using nltk.

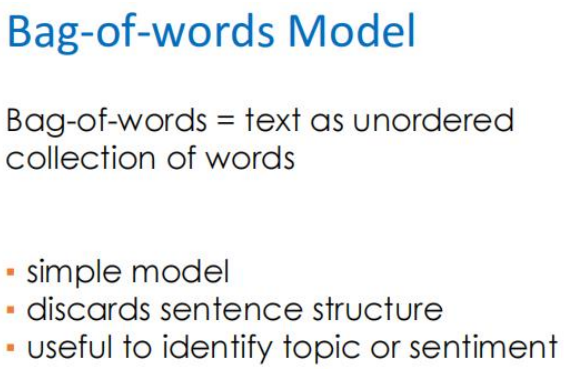
Okay, let's now switch back to our notebook.

总结:

Build a Bag-of-Words Model

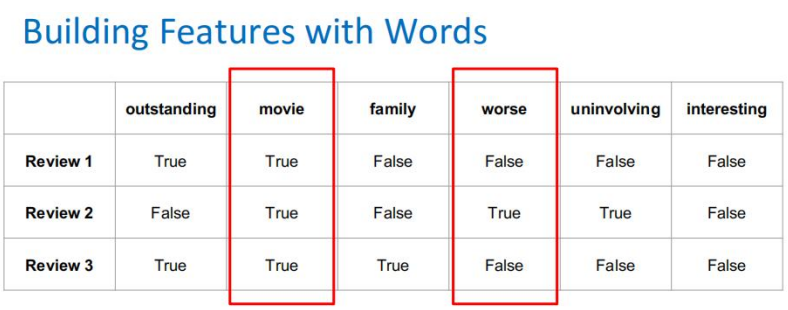
另详见 jupyter notebook: Natural Language Processing of Movie Reviews using nltk.ipynb



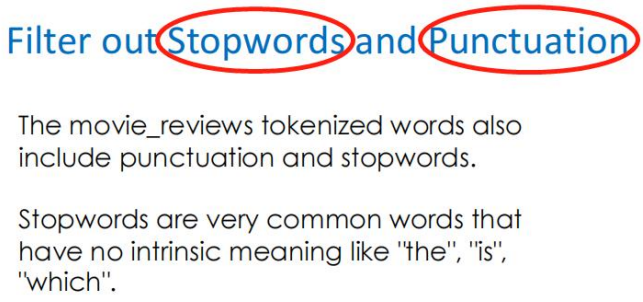


In this video, we will review what bag-of-words means. By the end of this video, you should be able to explain what bag-of-words mean, understand how you can build machine learning features from words, and give examples of stopwords.

Bag-of-words model is a very simple representation of a body of text as a loose set of words. It flattens any text into an unordered collection of words. Although it disregards the sentence structure associated to the words, this simple technique is pretty useful to identify a topic or sentiment in text, like if a product review has a negative or positive sentiment, or what a body of text talks about.



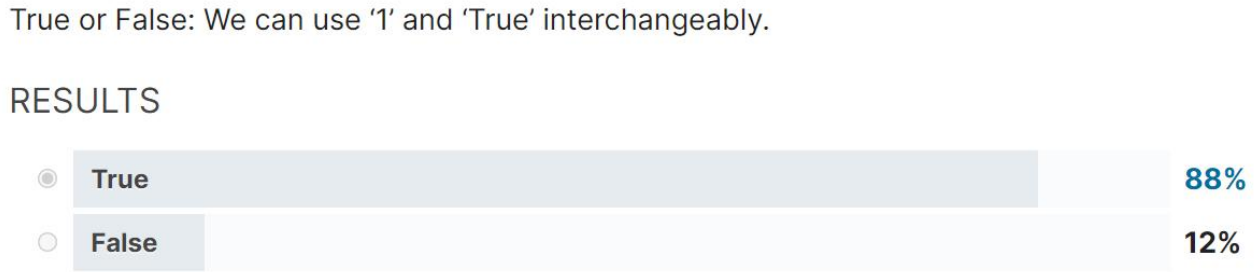
	outstanding	movie	family	worse	uninvolving	interesting
Review 1	True	True	False	False	False	False
Review 2	False	True	False	True	True	False
Review 3	True	True	True	False	False	False



We can use the words in a feature matrixwhere each word is a column, and each text body or review in our movie example, is a row that has boolean data values. A cell in the review row gets assigned true if the word appears in the review, and false if it doesn't. Just by looking at these three rows in this matrix, with a limited set of words, we can identify that the topic of these reviews are movies, and probably review 1 and review 3 are positive, and review 2 is negative.

Before we move on to our notebook, I would like to mentionthat it is often practice to filter our stopwords and maybe even the punctuations from the bag-of-words before further analysis. Stopwords are words like the, that, and is, which occur a lot but don't have a big significance in identifying the context of the text being processed.

Now, let's switch back to our notebook to execute our first bag-of-words model.



总结:

Plotting Frequency of Words

另详见 jupyter notebook: Natural Language Processing of Movie Reviews using nltk.ipynb

By the end of this video, you should be able to:

- Count how many times an item appears in a list
- Plot word frequency in logarithmic axes
- Plot word counts histograms

Number of Words in Movie Reviews Corpus

- ~1.6 million words
- just 710 thousand after filtering punctuation and stopwords

Now let's look at what we can do further.To analyze word frequencies. By the end of this video, you should be able to: count how many times an item appears in a list, plot word frequency in logarithmic axes, and plot word counts histograms.

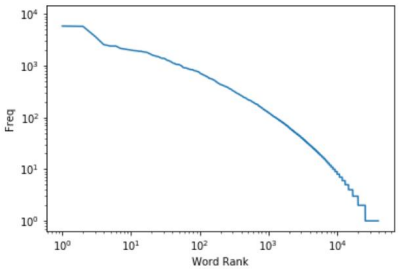
When we switch to our notebook, we will see that a quick check of the amount of words in our movie review database shows 1.6 million words, which can then be reduced to 710 thousand wire removal of useless words as we called it. This is still a lot of words.

Using Counter

- Part of the collections package in the Python Standard Library
- Counts how many time an item is repeated

```
counter = Counter(filtered_words)
counter["movie"]
5771
```

Plotting Word Frequency



Histogram of Word Counts

- Use hist from matplotlib to create a histogram
- Choose bin number and optionally log axes

How can you find out the frequency of each wordthat is how many times each word appears in this corpus? We will use the counter object from the collection package in Python for this purpose. Counter works very similar to the unique commanding units, we discussed before. We can provide it with a list of words. And it returns an object wire, which we can find out the reputation of each word. You would already guess that "movie" is a very frequent word in a movie database. And, indeed, it occurs 5,771 times in this script corpus.

Once we have frequency, in a counter,for each word, we will see, how we can plot the distribution of words, using matplotlib. This graph, just copy it, from our notebook, where we sorted the word counts and plotted their values on logarithmic axes. To check the shape of the distribution. This visualization is particularly useful, if you're comparing two or more data sets. If flatter distribution indicates a large vocabulary, while a peak distribution indicates a restricted vocabulary. Often due to a focused topic, or a specialized language.

We will also create histograms of the wordsfor visualization of the frequencies.

Let's now switch to our notebook to see what we just reviewed in action.

What is the benefit of a log graph over a graph that has not modified the scales?

RESULTS

There is no outright advantage to converting a graph into a log scale.	1%
Log scales are significantly better in representing every type of data compared to unmodified scale graphs.	15%
✓ Log scales allow a large range of values that are compressed into one point of the graph to be shown.	84%

总结：

Sentiment Analysis

另详见 jupyter notebook: Natural Language Processing of Movie Reviews using nltk.ipynb

By the end of this video, you should be able to:

- Explain what is Sentiment Analysis
- Train a Sentiment Analysis classifier with nltk
- Check accuracy on training and test data

What is Sentiment Analysis

- Identify attitude or emotion encoded in a text
- Can be implemented as a Machine Learning Classifier
- **Example:** prediction on the appearance of words in a review

We will now start using the bag-of-words modelwe created in a classifier for Sentiment Analysis of movie reviews. By the end of this video, you should be able to explain what Sentiment Analysis is, train a Sentiment Analysis classifier with nltk, and check accuracy on training and test data of this model.

So, what is Sentiment Analysis? The term refers to the activity of identifying attitude or emotion encoded in a body of text, like a product review or work of literature. Classification using machine learning is a technique used for sentiment models. In our notebook, we will build a predictor of sentiment using the movie review corpus.

Build features/label pairs

The function implemented previously creates a set of features.

Create a pair of feature and positive/negative label for each review.

Naive Bayes Classifier

Naive Bayes Classifier is a simple classifier based on Conditional Probabilities.

In the training phase, it detects the probability that each feature (word) appears in a category (positive or negative).

Once trained, it collects the "votes" for all words in the new review and finds the most probable label.

As you would remember, classification is asupervised activity and requires labels from a ground truth data. This is where we will take advantage of bag-of-words and a curated negative and positive reviews we downloaded. We will use our previously implemented bag-of-words function to create a positive or negative label for each review bag-of-words.

We will use Naive Bayes Classifier for this task.Although we haven't reviewed Naive Bayes before, it is a very simple classifier with a probabilistic approach to classification. What this means is that the relationships between the input features and the class labels is expressed as probabilities. So, given the input features, for example, the probability for each class is estimated. The class with the highest probability then determines the label for the sample. Let's now switch to our notebook one last time to end with the Sentiment Classification Task using Naive Bayes from nltk this time, not scikit-learn.

总结:

8.3 Engagement: Twitter - Working with Text, Incomplete

另详见 jupyter notebook: Using Tweepy for Tweet Analysis.ipynb

整个视频就不看了，只打印了上述的 jupyter notebook。

Advanced String Formatting in Python

```
print("{:20} | {:>6}".format(k,v))
```

"{:20}" format(s) pad the string to 20 spaces

"{:^20}" format(s) centered

"{:>20}" format(s) right-aligned

More at https://pyformat.info

总结: