Lecture 2: DS 402: Big Data Concepts

HDFS, Inside Map-Reduce, Introduction to Machine Learning

Outline

- Hadoop Distributed File System
- Recap SQL, Data Structure and Programming
  - HashMap
- Inside Map-Reduce -Twitter Word Count
- Introduction to Machine Learning
  - Naive Bayes

Lab/Assigment
Spam-Ham Problem

The Google File System

Distributed File System

*** Fault-tolerance at scale*

- application bugs
- operating system bugs
- human errors
- failures of disks, memory
- connectors
- networking
- power supplies

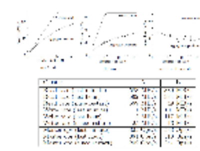
Therefore, constant monitoring, error detection, fault tolerance, and automatic recovery must be integral to the system.

The Design of GFS/HDFS

No arbitrary file modifications!

Block / Chunk Size

Read and Writes Rates

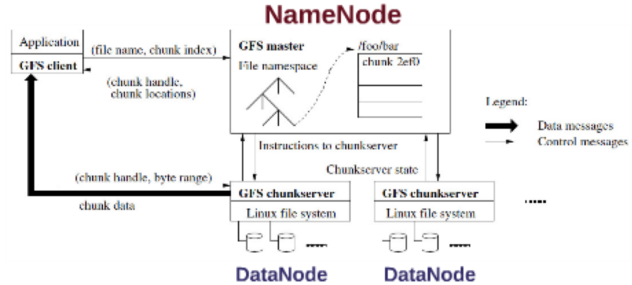


The Word-Count Example

Data Flow

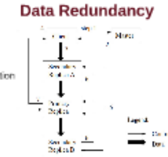
*** Single Master*

- However, we must minimize its involvement in reads and writes so that it does not become a bottleneck.
- Clients never read and write file data through the master.
- Instead, a client asks the master which chunk-servers it should contact. It caches



Data Redundancy

Mechanism
- Re-replication
- Re-balancing
- Garbage Collection



Recap-SQL, DS and Basic Programming

Recap Data Structures and Programming

- HashMaps
- Java
- Sorting
- Searching

Introduction to Machine Learning

Why Machine Learning?

Why Machine Learning? (continue..)

Spam or Ham

Web Page Categorization

- News
- Sports
- Wiki
- Blog
- Product
- ...

** Impossible to write a concise rule-set!!

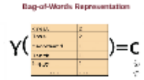
What is Machine Learning?

Naive Bayes Algorithm

Naive Bayes

- Supervised
- "Naive" = Simple
- simple representation of document
  - Bag of words

Bag-of-Words Representation



NB-Mathematical Foundation

NB-Working Example

Assignment

Training

Testing

Lecture 2: DS 402: Big Data Concepts

HDFS, Inside Map-Reduce, Introduction to Machine Learning

Outline

- Hadoop Distributed File System
- Recap SQL, Data Structure and Programming
  - HashMap
- Inside Map-Reduce -Twitter Word Count
- Introduction to Machine Learning
  - Naive Bayes

Lab/Assigment
Spam-Ham Problem

The Google File System

Sanjay Ghemawat , Howard Gobioff , Shun-Tak Leung, The google file system, Proceedings of the nineteenth ACM symposium on Operating systems principles, October 19-22, 2003, Bolton Landing, NY, USA

Distributed File System

*** Fault-tolerance at scale*

- application bugs
- operating system bugs
- human errors
- failures of disks, memory
- connectors
- networking
- power supplies

Therefore, constant monitoring, error detection, fault tolerance, and automatic recovery must be integral to the system.

The Design of GFS/HDFS

Assumptions

In designing a file system for our needs, we have been guided by assumptions that offer both **challenges** and **opportunities**.

- The system is built from many **inexpensive commodity components** that often fail. It must constantly monitor itself and detect, tolerate, and recover promptly from component failures on a routine basis.
- The system stores a modest number of large files. We expect a few million files, each typically 100 MB or **larger** in size. Multi-GB files are the common case and should be managed efficiently. **Small files** must be supported, but we need not optimize for them.
- The workloads primarily consist of two kinds of reads: large **streaming** reads and small random reads.
- The workloads also have many large, **sequential writes that append data to files**.

No arbitrary file modifications!

Files are mutated by ***appending new data rather than overwriting existing data***. Random writes within a file are practically non-existent. Once written, the files are only read, and often only ***sequentially***.

Block / Chunk Size

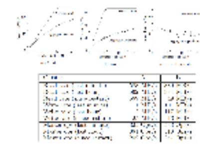
A disk has a block size, which is the minimum amount of data that it can read or write.

512 bytes

64 MB

minimize the cost of seeks.
Thus the time to transfer a large file made of multiple blocks operates at the disk transfer rate.

Read and Writes Rates

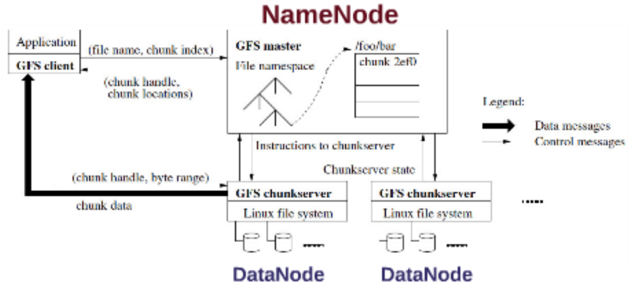


The Word-Count Example

Data Flow

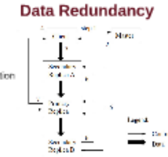
*** Single Master*

- However, we must minimize its involvement in reads and writes so that it does not become a bottleneck.
- Clients never read and write file data through the master.
- Instead, a client asks the master which chunk-servers it should contact. It caches



Data Redundancy

Mechanism
- Re-replication
- Re-balancing
- Garbage Collection



Recap-SQL, DS and Basic Programming

SQL: SELECT SUM(Sales) from Table GROUP BY City

** HBase and Hive

Recap Data Structures and Programming

- HashMaps
- Java
- Sorting
- Searching

Introduction to Machine Learning

Why Machine Learning?

Big Data - The Problem
- Robust Technology
  - Hadoop
- Intelligent Analytic
  - Unstructured data
  - Machine Learning

Why Machine Learning? (continue..)

Spam or Ham

Web Page Categorization
- News
- Sports
- Wiki
- Blog
- Product
- ...

** impossible to write a concise rule-set!!

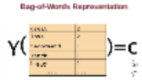
What is Machine Learning?

Naive Bayes Algorithm

Naive Bayes

- Supervised
- "Naive" = Simple
- simple representation of document
  - Bag of words

Bag-of-Words Representation



NB-Mathematical Foundation

NB-Working Example

Assignment

Training

Take Set 1
- Learn prior polarities: spam and ham
- Stop Words
- Part-of-Speech tagging
- Stemming

Testing

- Given an unknown mail: judge automatically whether it is a ham or spam
- Run on Set 2

Measuring System's Accuracy

Outline

- Hadoop Distributed File System
- Recap SQL, Data Structure and Programming
    - HashMap
- Inside Map-Reduce -Twitter Word Count
- Introduction to Machine Learning
    - Naive Bayes

Lab/Assigment

Spam-Ham Problem

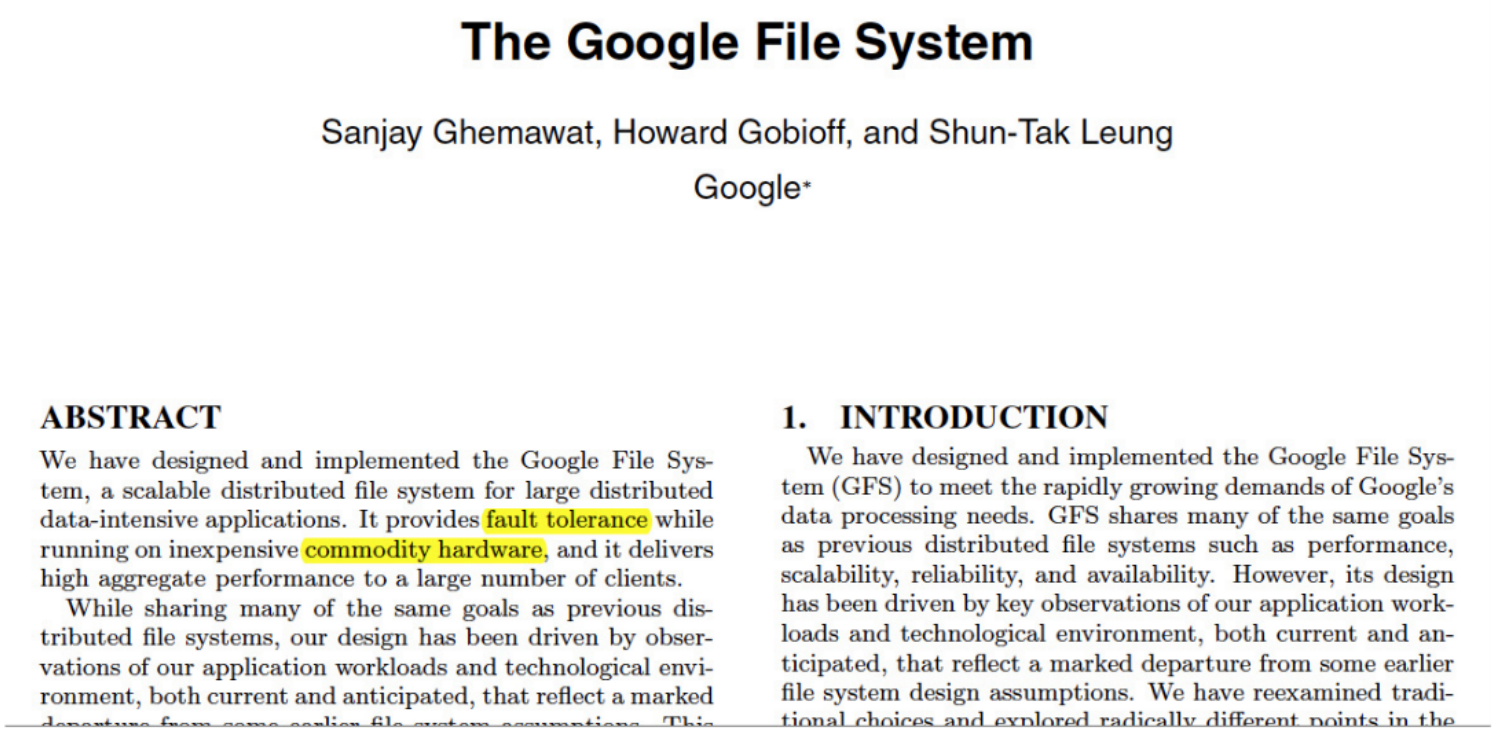
The Google File System

Sanjay Ghemawat , Howard Gobioff , Shun-Tak Leung, The Google file system, Proceedings of the nineteenth ACM symposium on Operating systems principles, October 19-22, 2003, Bolton Landing, NY, USA

Sanjay Ghemawat

Howard Gobioff

Shun-Tak Leung



The Google File System

Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung

Google*

ABSTRACT

We have designed and implemented the Google File System, a scalable distributed file system for large distributed data-intensive applications. It provides fault tolerance while running on inexpensive commodity hardware, and it delivers high aggregate performance to a large number of clients.

While sharing many of the same goals as previous distributed file systems, our design has been driven by observations of our application workloads and technological environment, both current and anticipated, that reflect a marked departure from some earlier file system assumptions. This

1. INTRODUCTION

We have designed and implemented the Google File System (GFS) to meet the rapidly growing demands of Google's data processing needs. GFS shares many of the same goals as previous distributed file systems such as performance, scalability, reliability, and availability. However, its design has been driven by key observations of our application workloads and technological environment, both current and anticipated, that reflect a marked departure from some earlier file system assumptions. We have reexamined traditional choices and explored radically different points in the

Distributed File System

*** Fault-tolerance at scale*

- application bugs
- operating system bugs
- human errors
- failures of disks, memory
- connectors
- networking
- power supplies

Therefore, constant monitoring, error detection, fault tolerance, and automatic recovery must be integral to the system.

The Design of GFS/HDFS

Assumptions

In designing a file system for our needs, we have been guided by assumptions that offer both **challenges** and **opportunities**.

- The system is built from many **inexpensive commodity components** that often fail. It must constantly monitor itself and detect, tolerate, and recover promptly from component failures on a routine basis.
- The system stores a modest number of large files. We expect a few million files, each typically 100 MB or **larger** in size. Multi-GB files are the common case and should be managed efficiently. **Small files** must be supported, but we need not optimize for them.
- The workloads primarily consist of two kinds of reads: large **streaming** reads and small random reads.
- The workloads also have many large, **sequential writes that append data to files**.

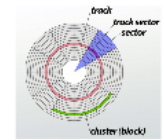
No arbitrary file modifications!

Files are mutated by ***appending new data rather than overwriting existing data***. Random writes within a file are practically non-existent. Once written, the files are only read, and often only ***sequentially***.

Block / Chunk Size

A disk has a block size, which is the minimum amount of data that it can read or write.

512 bytes



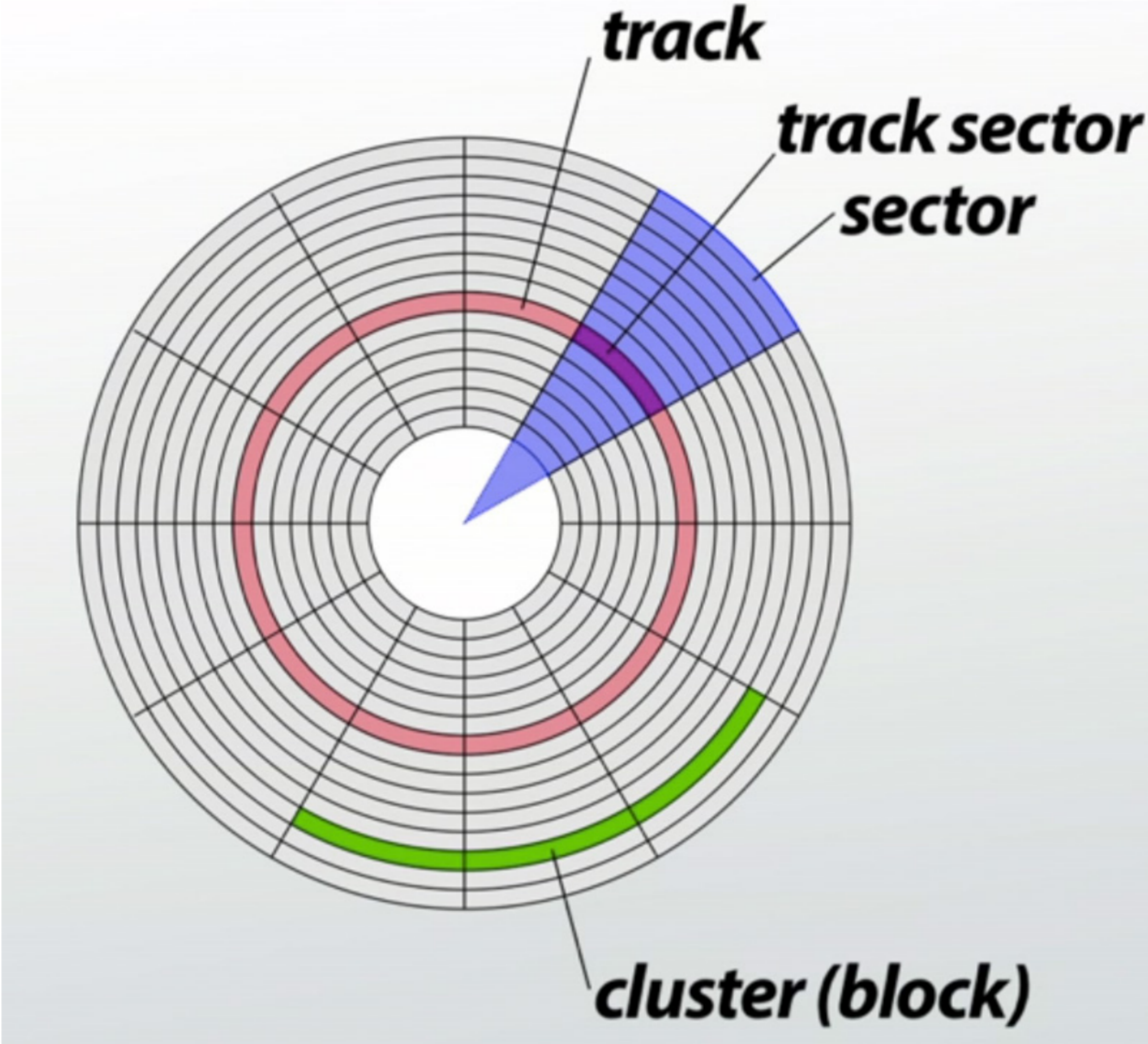
64 MB

minimize the cost of seeks.
Thus the time to transfer a large file made of multiple blocks operates at the disk transfer rate.

A quick calculation shows that if the seek time is around 10 ms, and the transfer rate is 100 MB/s, then to make the seek time 1% of the transfer time, we need to make the block size around 100 MB.

The default is actually 64 MB, although many HDFS installations use 128 MB blocks. This figure will continue to be revised upward as transfer speeds grow with new generations of disk drives.

** Map tasks in MapReduce normally operate on one block at a time, so if you have too few tasks (fewer than nodes in the cluster), your jobs will run slower than they could otherwise.

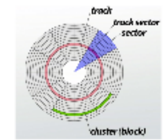


track
track sector
sector
cluster (block)

Block / Chunk Size

A disk has a block size, which is the minimum amount of data that it can read or write.

512 bytes



64 MB

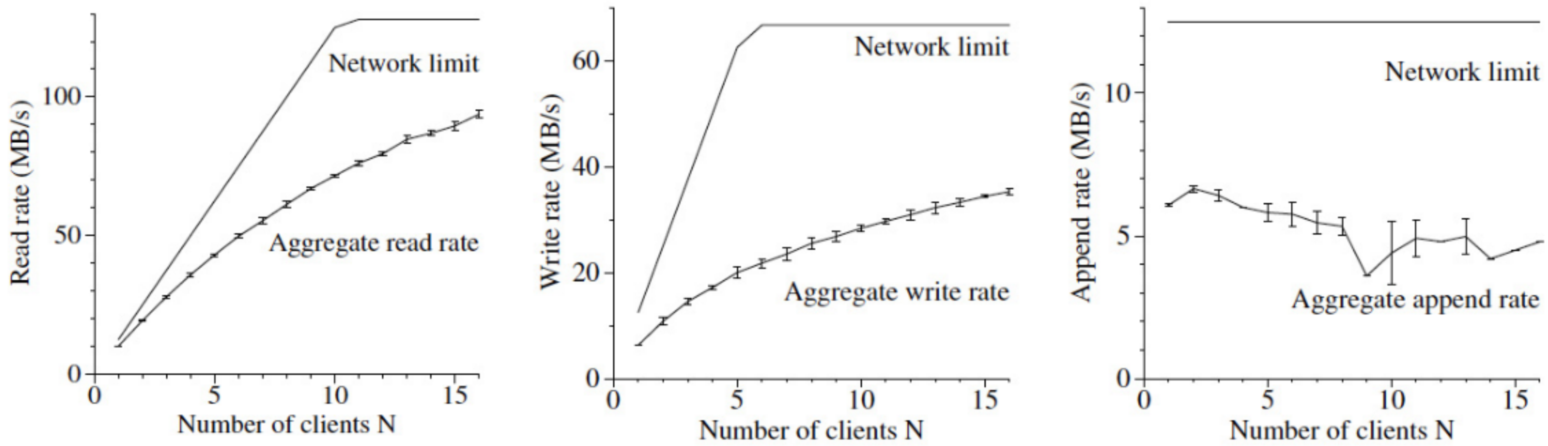
minimize the cost of seeks.
Thus the time to transfer a large file made of multiple blocks operates at the disk transfer rate.

A quick calculation shows that if the seek time is around 10 ms, and the transfer rate is 100 MB/s, then to make the seek time 1% of the transfer time, we need to make the block size around 100 MB.

The default is actually 64 MB, although many HDFS installations use 128 MB blocks. This figure will continue to be revised upward as transfer speeds grow with new generations of disk drives.

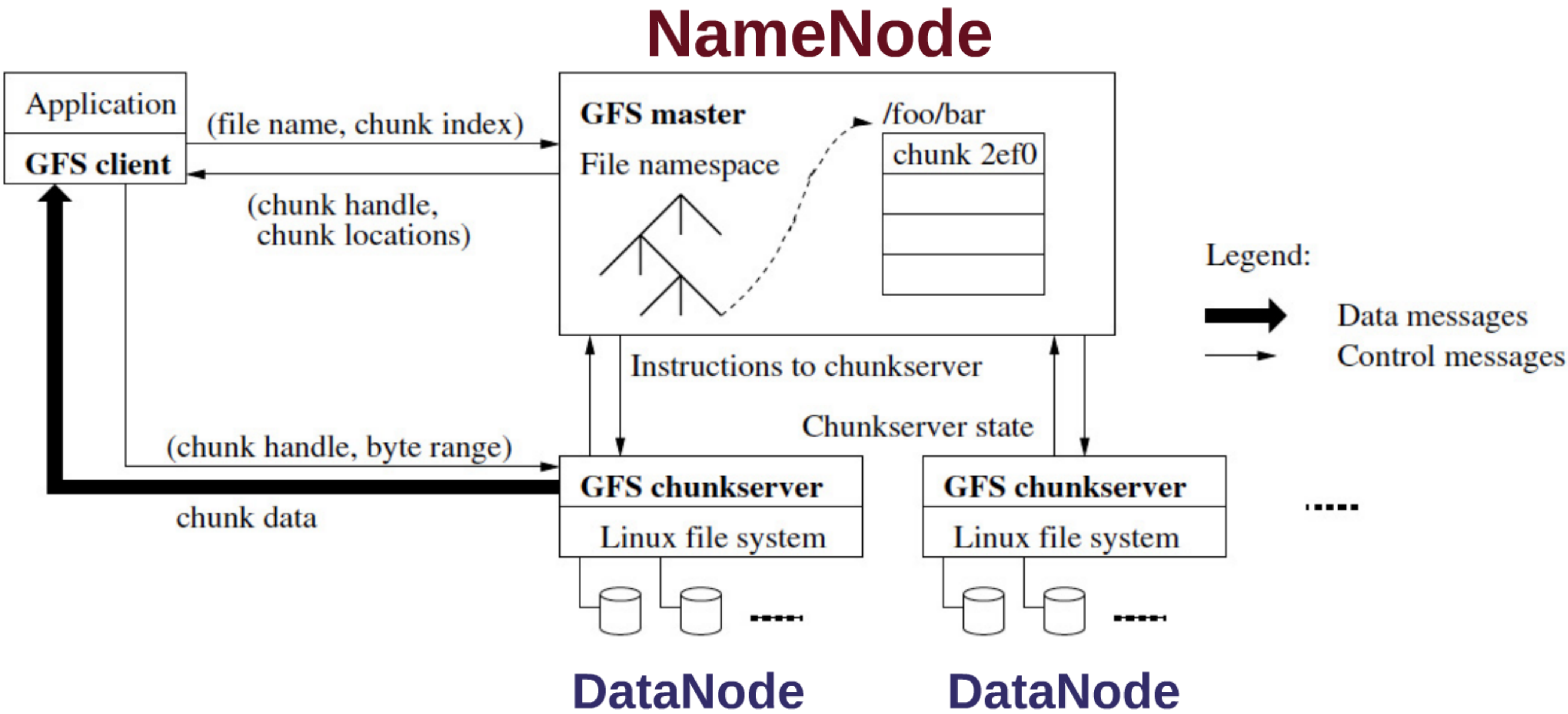
** Map tasks in MapReduce normally operate on one block at a time, so if you have too few tasks (fewer than nodes in the cluster), your jobs will run slower than they could otherwise.

Read and Writes Rates



(a) Reads (b) Writes (c) Record appends

Cluster	A	B
Read rate (last minute)	583 MB/s	380 MB/s
Read rate (last hour)	562 MB/s	384 MB/s
Read rate (since restart)	589 MB/s	49 MB/s
Write rate (last minute)	1 MB/s	101 MB/s
Write rate (last hour)	2 MB/s	117 MB/s
Write rate (since restart)	25 MB/s	13 MB/s
Master ops (last minute)	325 Ops/s	533 Ops/s
Master ops (last hour)	381 Ops/s	518 Ops/s
Master ops (since restart)	202 Ops/s	347 Ops/s



NameNode
Application
GFS client
(file name, chunk index)
(chunk handle,
chunk locations)
GFS master
File namespace
/foo/bar
chunk 2ef0
Legend:
Data messages
Control messages
Instructions to chunkserver
Chunkserver state
(chunk handle, byte range)
chunk data
GFS chunkserver
Linux file system
GFS chunkserver
Linux file system
DataNode
DataNode

Lab/Assigment
Spam-Ham Problem

Howard Gobioff

Shun-Tak Leung

Therefore, constant monitoring, error detection, fault tolerance, and automatic recovery must be integral to the system.

- The workloads primarily consist of two kinds of reads: large **streaming** reads and small random reads.
- The workloads also have many large, **sequential writes** that **append data to files**.

No arbitrary file modifications!

Files are mutated by ***appending new data rather than overwriting existing data***. Random writes within a file are practically non-existent. Once written, the files are only read, and often only ***sequentially***.

Block / Chunk Size

A disk has a block size, which is the minimum amount of data that it can read or write.

512 bytes

64 MB

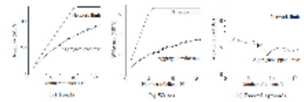
minimize the cost of seeks.
Thus the time to transfer a large file made of multiple blocks operates at the disk transfer rate.

A quick calculation shows that if the seek time is around 10 ms, and the transfer rate is 100 MB/s, then to make the seek time 1% of the transfer time, we need to make the block size around 100 MB.

The default is actually 64 MB, although many HDFS installations use 128 MB blocks. This figure will continue to be revised upward as transfer speeds grow with new generations of disk drives.

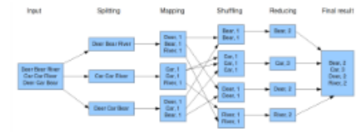
** Map tasks in MapReduce normally operate on one block at a time, so if you have too few tasks (fewer than nodes in the cluster), your jobs will run slower than they could otherwise.

Read and Writes Rates



Cluster	A	B
Read rate (last minute)	[illegible] MB/s	380 MB/s
Read rate (last hour)	562 MB/s	384 MB/s
Read rate (since restart)	[illegible] MB/s	49 MB/s
Write rate (last minute)	1 MB/s	101 MB/s
Write rate (last hour)	2 MB/s	117 MB/s
Write rate (since restart)	25 MB/s	13 MB/s
Master ops (last minute)	[illegible] Ops/s	[illegible] Ops/s
Master ops (last hour)	381 Ops/s	518 Ops/s
Master ops (since restart)	202 Ops/s	347 Ops/s

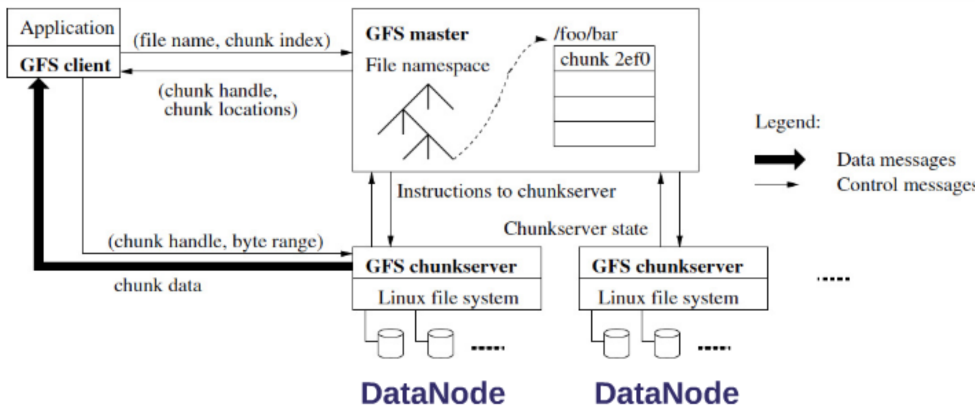
The Word-Count Example



Data Flow

***** Single Master***

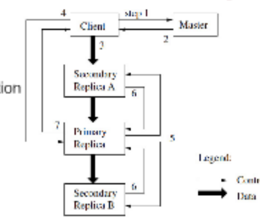
- However, we must minimize its involvement in reads and writes so that it does not become a bottleneck.
- Clients never read and write file data through the master.
- Instead, a client asks the master which chunk-servers it should contact. It caches



Data Redundancy

Mechanism

- Re-replication
- Re-balancing
- Garbage Collection



Hadoop Filesystems

HDFS is just one implementation.

HDFS Command-Line Interface

Recap-SQL, DS and Basic Programming

City	Sales
Delhi	[illegible]
Kolkata	₹40000
Gurgaon	[illegible]
Bangalore	₹90000
Mumbai	₹70000
Delhi	[illegible]
Bangalore	[illegible]
...	...
...	...

City	Sales
Bangalore	[illegible]
Delhi	[illegible]
Gurgaon	[illegible]
Kolkata	₹40000
Mumbai	₹70000
...	...
...	...

SQL: SELECT SUM(Sales) from Table GROUP BY City

**** HBase and Hive**

Recap Data Structures and Programming

- HashMaps
- Java
- Sorting
- Searching

Introduction to Machine Learning

Why Machine Learning?

Big Data - The Problem

- Robust Technology
  - Hadoop
- Intelligent Analytic
  - Unstructured data
    - Machine Learning

Why Machine Learning? (continue..)

Spam or Ham

Web Page Categorization

- News
- Sports
- Wiki
- Blog
- Product
- ...

** Impossible to write a concise rule-set!!

What is Machine Learning?

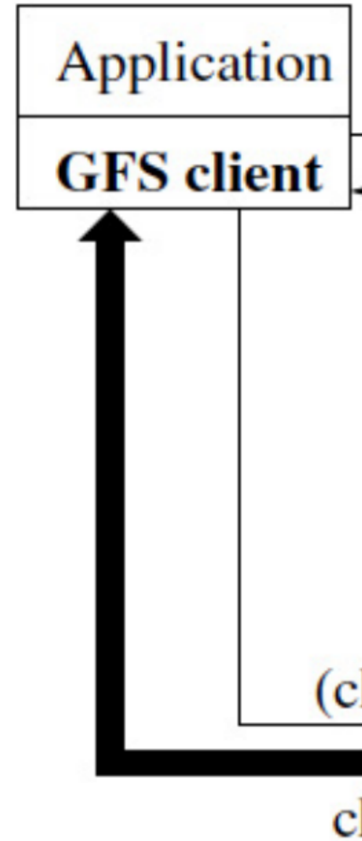
ML Types

Naive Bayes Algorithm

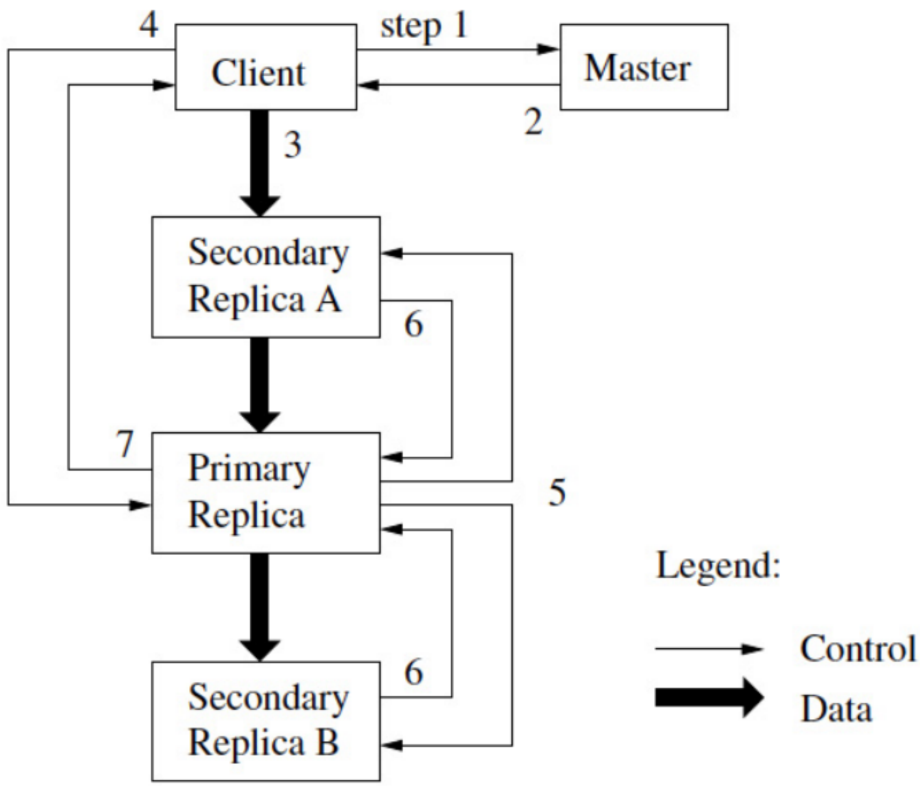
Data Flow

***** Single Master***

- However, we must minimize its involvement in reads and writes so that it does not become a bottleneck.
- Clients never read and write file data through the master.
- Instead, a client asks the master which chunk-servers it should contact. It caches



Data Redundancy



Mechanism

- Re-replication
- Re-balancing
- Garbage Collection

Hadoop Filesystems

HDFS is just one implementation.

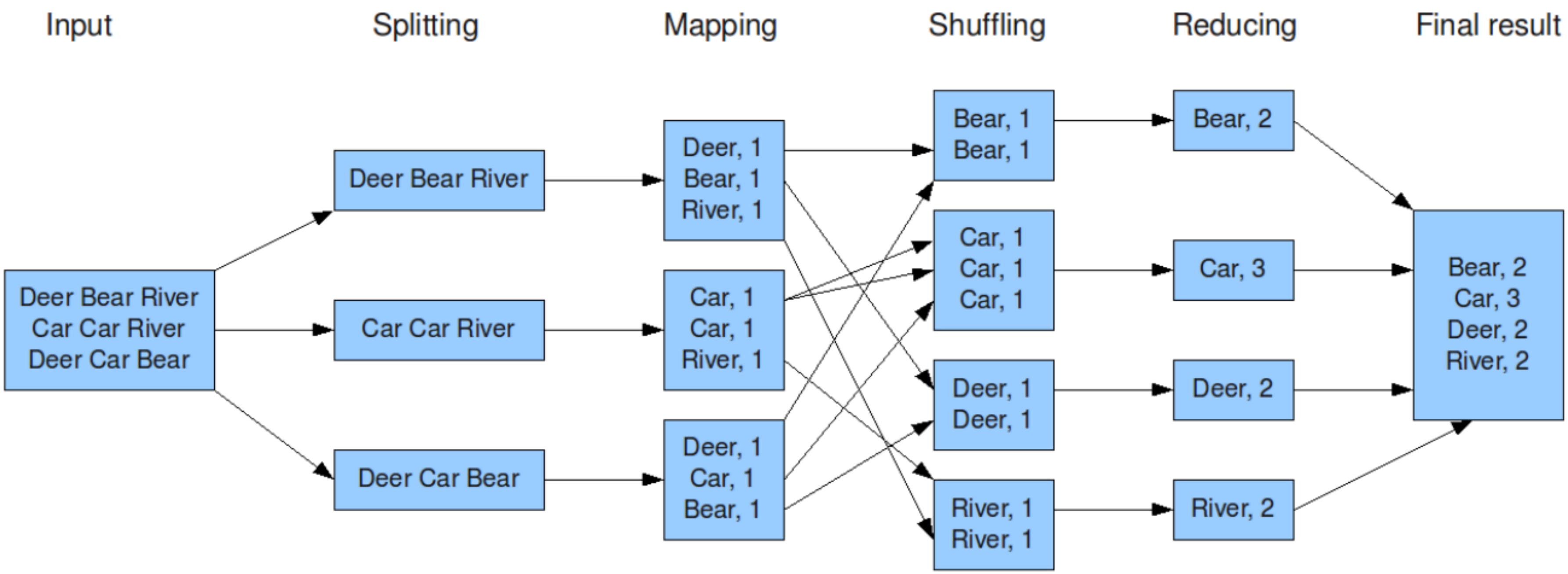
Filesystem	URI scheme	Java implementation (all under org.apache.hadoop)	Description
Local	*file*	fs.LocalFileSystem	A filesystem for a locally connected disk with client-side checksums. Use RawLocalFileSystem for a local filesystem with no checksums. See "LocalFileSystem" on page 76.
HDFS	*hdfs*	hdfs.DistributedFileSystem	Hadoop's distributed filesystem. HDFS is designed to work efficiently in conjunction with MapReduce.
HFTP	*hftp*	hdfs.HftpFileSystem	A filesystem providing read-only access to HDFS over HTTP. (Despite its name, HFTP has no connection with FTP.) Often used with *distcp* (see "Parallel Copying with distcp" on page 70) to copy data between HDFS clusters running different versions.
HSFTP	*hsftp*	hdfs.HsftpFileSystem	A filesystem providing read-only access to HDFS over HTTPS. (Again, this has no connection with FTP.)
HAR	*har*	fs.HarFileSystem	A filesystem layered on another filesystem for archiving files. Hadoop Archives are typically used for archiving files in HDFS to reduce the namenode's memory usage. See "Hadoop Archives" on page 71.
KFS (CloudStore)	*kfs*	fs.kfs.KosmosFileSystem	CloudStore (formerly Kosmos filesystem) is a distributed filesystem like HDFS or Google's GFS, written in C++. Find more information about it at *http://kosmosfs.sourceforge.net/*.
FTP	*ftp*	fs.ftp.FTPFileSystem	A filesystem backed by an FTP server.
S3 (native)	*s3n*	fs.s3native.NativeS3FileSystem	A filesystem backed by Amazon S3. See *http://wiki.apache.org/hadoop/AmazonS3*.
S3 (block-based)	*s3*	fs.s3.S3FileSystem	A filesystem backed by Amazon S3, which stores files in blocks (much like HDFS) to overcome S3's 5 GB file size limit.

HDFS Command-Line Interface

hadoop fs -**copyFromLocal** input/abc.txt hdfs://localhost/user/amitava/xyz.txt

hadoop fs **-copyToLocal** /user/amitava/xyz.txt abc.copy.txt

The Word-Count Example



Recap-SQL, DS and Basic Programming

City	Sales
Delhi	₹50000
Kolkata	₹40000
Gurgaon	₹30000
Bangalore	₹80000
Mumbai	₹70000
Delhi	₹55000
Bangalore	₹88000
...	...
...	...

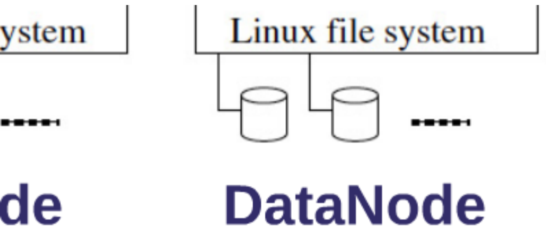
City	Sales
Bangalore	₹168000
Delhi	₹105000
Gurgaon	₹30000
Kolkata	₹40000
Mumbai	₹70000
...	...
...	...

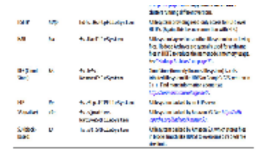
SQL: SELECT SUM(Sales) from Table GROUP BY City

**** HBase and Hive**

Recap Data Structures and Programming

- HashMaps
- Java
- Sorting
- Searching





HDFS Command-Line Interface

hadoop fs **-copyFromLocal** input/abc.txt hdfs://localhost/user/amitava/xyz.txt

hadoop fs **-copyToLocal** /user/amitava/xyz.txt abc.copy.txt

Introduction to Machine Learning

Why Machine Learning?

Big Data - The Problem

- Robust Technology
  - Hadoop
- Intelligent Analytic
  - Unstructured data
    - Machine Learning

Why Machine Learning? (continue..)

Spam or Ham

Web Page Categorization

- News
- Sports
- Wiki
- Blog
- Product
- ...

** Impossible to write a concise rule-set!!

What is Machine Learning?

Definition 1

Arthur Samuel (1959). Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.

Definition 2

Tom Mitchell (1998) Well-posed Learning Problem: A computer program is said to learn from experience *E* with respect to some task *T* and some performance measure *P*, if its performance on *T*, as measured by *P*, improves with experience *E*.

ML Types

- Supervised learning
- Unsupervised learning
- Online Learning
- Others:
  Reinforcement learning

ve Bayes Algorithm

NB-Working Example

$$P(c) = \frac{N_c}{N}$$

$$P(w \mid c) = \frac{count(w,c)+1}{count(c)+|V|}$$

Priors:
$P(c) = \frac{3}{4}$
$P(j) = \frac{1}{4}$

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

nditional Probabilities:
hinese|c) = (5+1) / (8+6) = 6/14 = 3/7
okyo|c) = (0+1) / (8+6) = 1/14
apan|c) = (0+1) / (8+6) = 1/14
hinese|j) = (1+1) / (3+6) = 2/9

Choosing a class:
P(c|d5) ∝ 3/4 * (3/7)³ * 1/14 * 1/14
≈ 0.0003

Assignment

http://www.aueb.gr/users/ion/data/enron-spam/

- readme.txt
- Enron-Spam in pre-processed form:
  - Enron1
  - Enron2
  - Enron3
  - Enron4
  - Enron5
  - Enron6
- Enron-Spam in raw form:
  - ham messages:
    - beck-s
    - farmer-d
    - kaminski-v
    - kitchen-l
    - lokay-m
    - williams-w3
  - spam messages:

Spam

Ham

Training

Take Set 1

- Learn prior polarities: ***spam and ham***
- Stop Words:
  - http://www.ranks.nl/stopwords 
- Part-of-Speech tagging
  - http://nlp.stanford.edu/software/tagger.shtml 
- Stemming
  - http://ir.dcs.gla.ac.uk/resources/linguistic_utils/porter.java

Testing

- Given an unknown mail: judge automatically whether it is a ham or spam
- Run on Set 2

Measuring System's Accuracy

Accuracy = correctly identified/total instances

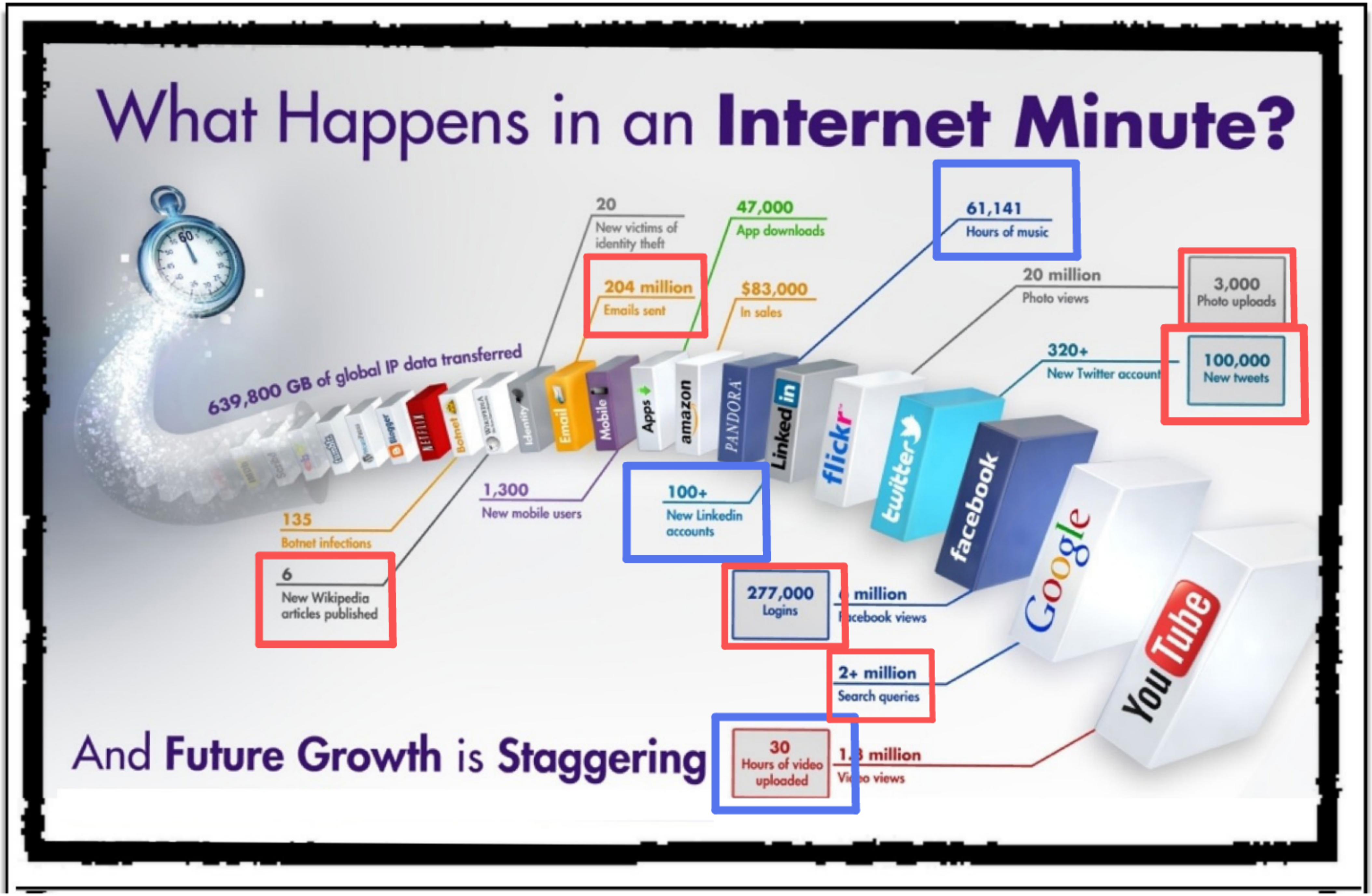
Precision (P): % of selected items that are correct

Why Machine Learning?

Big Data - The Problem

- Robust Technology
  - Hadoop
- Intelligent Analytic
  - Unstructured data
    - Machine Learning

Learning


What Happens in an Internet Minute?
639,800 GB of global IP data transferred
20
New victims of identity theft
204 million
Emails sent
47,000
App downloads
$83,000
In sales
61,141
Hours of music
20 million
Photo views
3,000
Photo uploads
320+
New Twitter accounts
100,000
New tweets
135
Botnet infections
6
New Wikipedia articles published
1,300
New mobile users
100+
New Linkedin accounts
277,000
Logins
million
2+ million
Search queries
30
Hours of video uploaded
million
And Future Growth is Staggering

Why Machine Learning? (continue..)

Spam or Ham

Web Page Categorization

- News
- Sports
- Wiki
- Blog
- Product
- ...

** Impossible to write a concise rule-set!!

What is Machine Learning?

Definition 1

Arthur Samuel (1959). Machine Learning: Field of study that gives computers the ability to learn without being explicitly programmed.

Definition 2

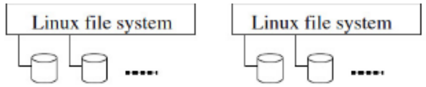
Tom Mitchell (1998) Well-posed Learning Problem: A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.

ML Types

- Supervised learning
- Unsupervised learning
- Online Learning
- Others:
  Reinforcement learning

master which chunk-servers it should contact. It caches

chunk data



HDFS Command-Line Interface

Recap-SQL, DS and Basic Programming

City	Sales
Delhi	[illegible]
Kolkata	[illegible]
[illegible]	[illegible]
Bangalore	[illegible]
Mumbai	[illegible]
Delhi	[illegible]
Bangalore	[illegible]
...	...
...	...

City	Sales
Bangalore	₹160000
Delhi	[illegible]
Gurgaon	[illegible]
Kolkata	[illegible]
Mumbai	₹70000
...	...
...	...

SQL: SELECT SUM(Sales) from Table GROUP BY City

** HBase and Hive

Recap Data Structures and Programming

- HashMaps
- Java
- Sorting
- Searching

Introduction to Machine Learning

Why Machine Learning?

Big Data - The Problem
- Robust Technology
  - Hadoop
- Intelligent Analytic
  - Unstructured data
    - Machine Learning

Why Machine Learning? (continue..)

Spam or Ham

Web Page Categorization
- News
- Sports
- Wiki
- Blog
- Product
- ...

** Impossible to write a concise rule-set!!

What is Machine Learning?

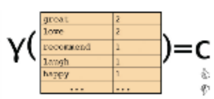
ML Types

Naive Bayes Algorithm

Naive Bayes

- Supervised
- "Naïve" = Simple
- simple representation of document
  - Bag of words

Bag-of-Words Representation



NB-Mathematical Foundation

For a document d and a class c

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d) = \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c)P(c)}{P(d)} = \underset{c \in C}{\operatorname{argmax}} P(d \mid c)P(c) = \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c)P(c)$$

Prior Polarity

Conditional Independence

NB-Working Example

Assignment

http://www.aueb.gr/users/ion/data/enron-spam/

Spam

Ham

Training

Take Set 1
- Learn prior polarities: ***spam and ham***
- Stop Words:
  - http://www.ranks.nl/stopwords
- Part-of-Speech tagging
  - http://nlp.stanford.edu/software/tagger.shtml
- Stemming
  - http://ir.dcs.gla.ac.uk/resources/linguistic_utils/porter.java

Testing

- Given an unknown mail: judge automatically whether it is a ham or spam
- Run on Set 2

Measuring System's Accuracy

Accuracy = correctly identified/total instances

Precision (P): % of selected items that are correct
Recall (R): % correct items that are selected
F-Measure (F): 2PR / (P+R)

Naive Bayes

- Supervised
- “Naïve” = Simple
- simple representation of document
    - Bag of words

Bag-of-Words Representation

$\gamma($

great	2
love	2
recommend	1
laugh	1
happy	1
...	...

$) = c$

NB-Mathematical Foundation

For a document *d* and a class *c*

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

$$= \underset{c \in C}{\operatorname{argmax}} \frac{P(d \mid c)P(c)}{P(d)}$$

$$= \underset{c \in C}{\operatorname{argmax}} P(d \mid c)P(c)$$

Prior Polarity

$$= \underset{c \in C}{\operatorname{argmax}} P(x_1, x_2, \ldots, x_n \mid c)P(c)$$

Conditional Independence: Assume the feature probabilities P(**x**i|**c**j) are independent given the class *c*.

$$P(x_1, \ldots, x_n \mid c) = P(x_1 \mid c) \bullet P(x_2 \mid c) \bullet P(x_3 \mid c) \bullet \ldots \bullet P(x_n \mid c)$$

NB-Working Example

$$P(c) = \frac{N_c}{N}$$

$$P(w \mid c) = \frac{count(w,c)+1}{count(c)+|V|}$$

Priors:

$P(c) = \frac{3}{4}$

$P(j) = \frac{1}{4}$

	Doc	Words	Class
Training	1	Chinese Beijing Chinese	c
	2	Chinese Chinese Shanghai	c
	3	Chinese Macao	c
	4	Tokyo Japan Chinese	j
Test	5	Chinese Chinese Chinese Tokyo Japan	?

Conditional Probabilities:

P(Chinese|*c*) = (5+1) / (8+6) = 6/14 = 3/7

P(Tokyo|*c*) = (0+1) / (8+6) = 1/14

P(Japan|*c*) = (0+1) / (8+6) = 1/14

P(Chinese|*j*) = (1+1) / (3+6) = 2/9

P(Tokyo|*j*) = (1+1) / (3+6) = 2/9

P(Japan|*j*) = (1+1) / (3+6) = 2/9

Choosing a class:

$P(c|d5) \propto 3/4 * (3/7)^3 * 1/14 * 1/14 \approx 0.0003$

$P(j|d5) \propto 1/4 * (2/9)^3 * 2/9 * 2/9 \approx 0.0001$

Assignment

http://www.aueb.gr/users/ion/data/enron-spam/

- readme.txt
- Enron-Spam in pre-processed form:
  - Enron1
  - Enron2
  - Enron3
  - Enron4
  - Enron5
  - Enron6
- Enron-Spam in raw form:
  - ham messages:
    - beck-s
    - farmer-d
    - kaminski-v
    - kitchen-l
    - lokay-m
    - williams-w3
  - spam messages:
    - BG
    - GP
    - SH

Spam

Ham

Training

Take Set 1

- Learn prior polarities: ***spam and ham***
- Stop Words:
  - http://www.ranks.nl/stopwords

- Part-of-Speech tagging
  - http://nlp.stanford.edu/software/tagger.shtml

- Stemming
  - http://ir.dcs.gla.ac.uk/resources/linguistic_utils/porter.java

Stop Words - Unmeaningful Words

a	been	doesn't	he	i'm	myself	own
about	before	doing	he'd	i've	no	same
above	being	don't	he'll	if	nor	shan't
after	below	down	he's	in	not	she
again	between	during	her	into	of	she'd
against	both	each	here	is	off	she'll
all	but	few	here's	isn't	on	she's
am	by	for	hers	it	once	should
an	can't	from	herself	it's	only	shouldn't
and	cannot	further	him	its	or	so
any	could	had	himself	itself	other	some
are	couldn't	hadn't	his	let's	ought	such
aren't	did	has	how	me	our	than
as	didn't	hasn't	how's	more	ours	that
at	do	have	i	most	ourselves	that's
be	does	haven't	i'd	mustn't	out	the
because	doesn't	having	i'll	my	over	their

POS Tagging

I hope this'll show the server working.

I/PRP hope/VBP this/DT 'll/MD show/VB the/DT server/NN working/VBG ./.

Content Words: Noun, Verb, Adjective, and Adverbs

Tag	Description
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential *there*
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun

Tag	Description
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	*to*
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb

Surface Forms

car, cars, car's, cars'

Stems

Car

** Stems are not root!!

goes -> goe
happily -> happili

Testing

- Given an unknown mail: judge automatically whether it is a ham or spam
- Run on Set 2

Measuring System's Accuracy

Accuracy = correctly identified/total instances

Precision (P): % of selected items that are correct
Recall (R): % correct items that are selected
F-Measure (F): 2PR / (P+R)

Today in the Lab: Twitter Word Counts

- Simple Word Count
- Stop Words:
  - http://www.ranks.nl/stopwords
- Part-of-Speech tagging
  - http://www.ark.cs.cmu.edu/TweetNLP/
- Stemming
  - http://ir.dcs.gla.ac.uk/resources/linguistic_utils/porter.java