SECURITY, PRIVACY AND COMPLIANCE.

DISTRIBUTED COMPUTING, VIRTUALIZATION AND CLOUD COMPUTING

By

Porrselvi

DISTRIBUTED COMPUTING, VIRTUALIZATION AND CLOUD COMPUTING

Introduction:

Instead of using a centralized computer to solve computational problems, a parallel and distributed computing system uses multiple computers to solve large-scale problems over the Internet. In general, *distributed computing* is the opposite of *centralized computing*. The field of *parallel computing* overlaps with distributed computing to a great extent, and *cloud computing* overlaps with distributed, centralized, and parallel computing. In Centralized computing**,** all computer resources are centralized in one physical system. All resources (processors, memory, and storage) are fully shared and tightly coupled within one integrated OS. Many data centers and supercomputers are *centralized systems*, but they are used in parallel, distributed, and cloud computing applications. A *distributed system* consists of multiple autonomous computers, each having its own private memory, communicating through a computer network. Information exchange in a distributed system is accomplished through *message passing*. For example, distributed transaction processing is often practiced in the banking and finance industry. An *Internet cloud* of resources can be either a centralized or a distributed computing system. The cloud applies parallel or distributed computing, or both. Clouds can be built with physical or virtualized resources over large data centers that are centralized or distributed.

It’s time to explore hardware, software, and network technologies for distributed computing system design and applications. In particular, we will focus on viable approaches to building distributed operating systems for handling massive parallelism in a distributed environment.

MapReduce: Support MapReduce programming model including Hadoop on Linux, Dryad on Windows HPCS, and Twister on Windows and Linux. Include new associated languages such as Sawzall, Pregel, Pig Latin, and LINQ.

Monitoring: Many grid solutions such as Inca. Can be based on publish-subscribe.

Notification: Basic function of publish-subscribe systems.

Programming model: Cloud programming models are built with other platform features and are related to familiar web and grid models.

Queues: Queuing system possibly based on publish-subscribe.

Scalable synchronization: Apache Zookeeper or Google Chubby. Supports distributed locks and used by BigTable. Not clear if (effectively) used in Azure Table or Amazon SimpleDB.

SQL: Relational database.

Table: Support of table data structures modeled on Apache Hbase or Amazon SimpleDB/Azure Table. Part of NOSQL movement.

Web role: Used in Azure to describe important link to user and can be supported otherwise with a portal framework. This is the main purpose of GAE.

Worker role: Implicitly used in both Amazon and grids but was first introduced as a high-level construct by Azure.

Blobs and Drives*:*

The basic storage concept in clouds is blobs for Azure and S3 for Amazon.

SQL and Relational Databases*:*

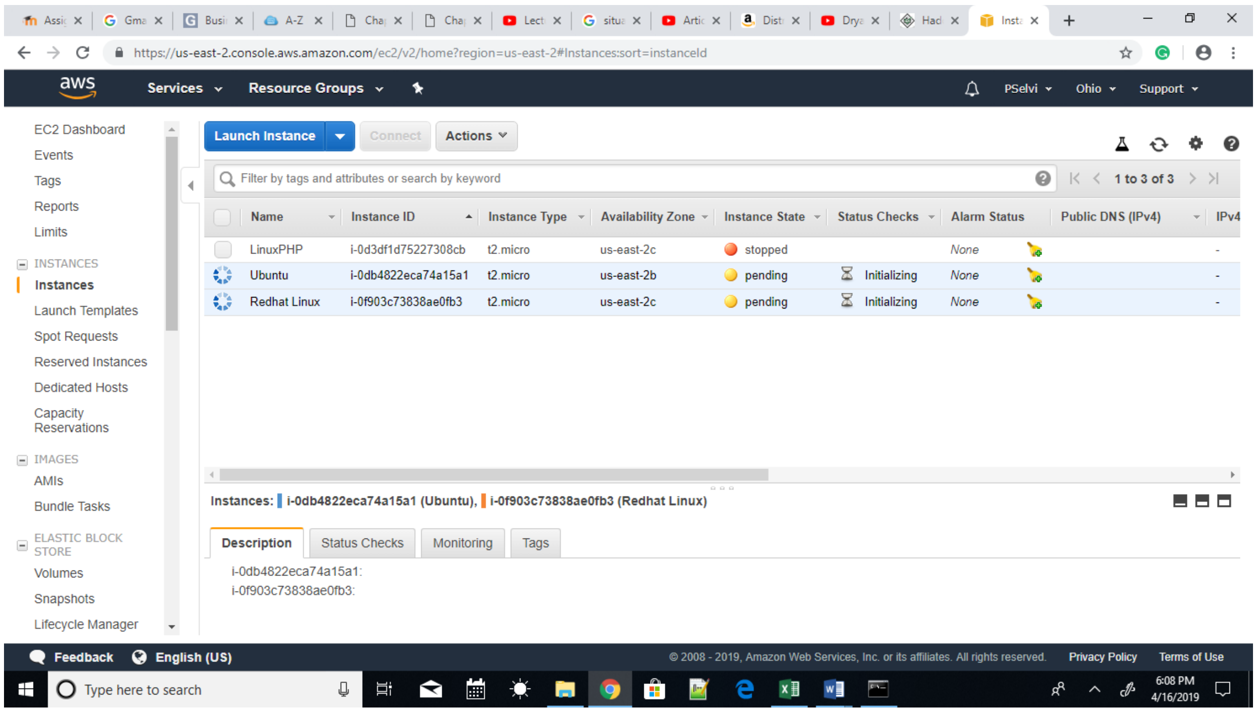
Both Amazon and Azure clouds offer relational databases and it is straightforward for academic systems to offer a similar capability unless there are issues of huge scale where, in fact, approaches based on tables and/or MapReduce might be more appropriate.

Table and NOSQL Non-relational Databases:

A substantial number of important developments have occurred regarding simplified database structures—termed “NOSQL” —typically emphasizing distribution and scalability. These are present in the three major clouds: BigTable in Google, SimpleDB in Amazon, and Azure Table for Azure.

Google’s Solution:

To dealing with huge amounts of scalable data, it is a hectic task to process such data through a single database bottleneck. Google solved this problem using an algorithm called MapReduce. This algorithm divides the task into small parts and assigns them to many computers, and collects the results from them which when integrated, form the result dataset.



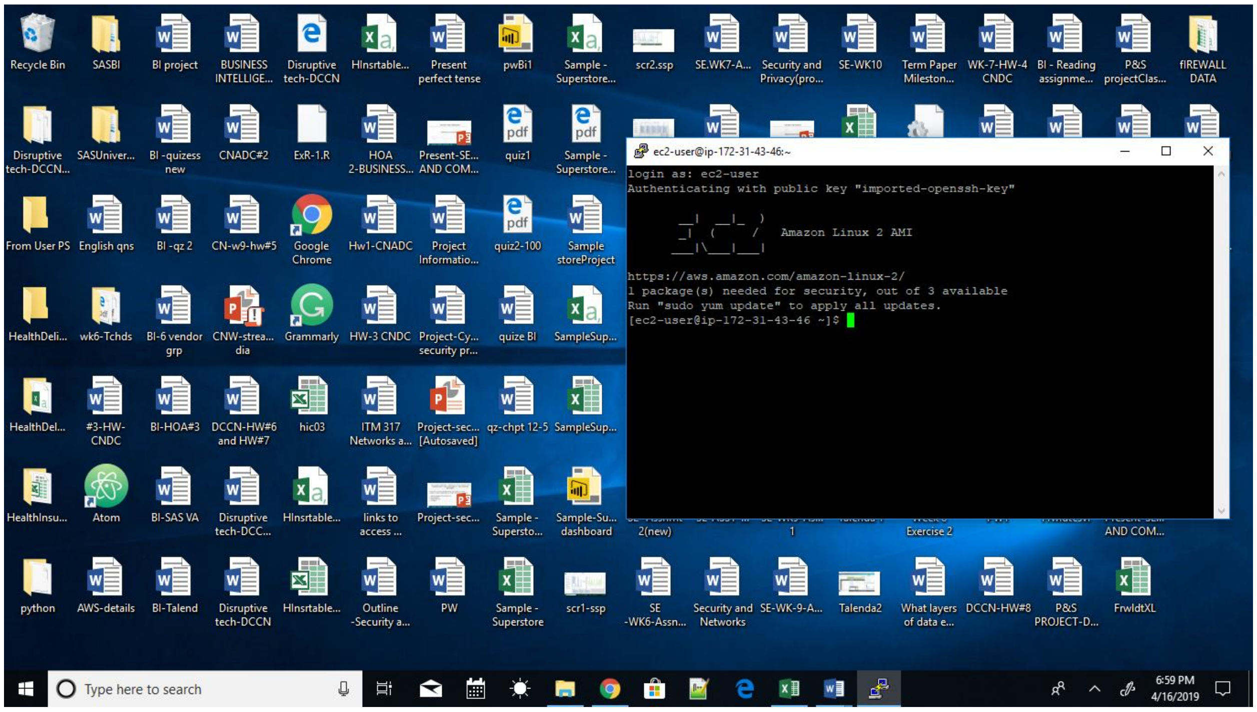
Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4
LinuxPHP	i-0d3df1d75227308cb	t2.micro	us-east-2c	stopped		None		-
Ubuntu	i-0db4822eca74a15a1	t2.micro	us-east-2b	pending	Initializing	None		-
Redhat Linux	i-0f903c73838ae0fb3	t2.micro	us-east-2c	pending	Initializing	None		-

Instances: i-0db4822eca74a15a1 (Ubuntu), i-0f903c73838ae0fb3 (Redhat Linux)

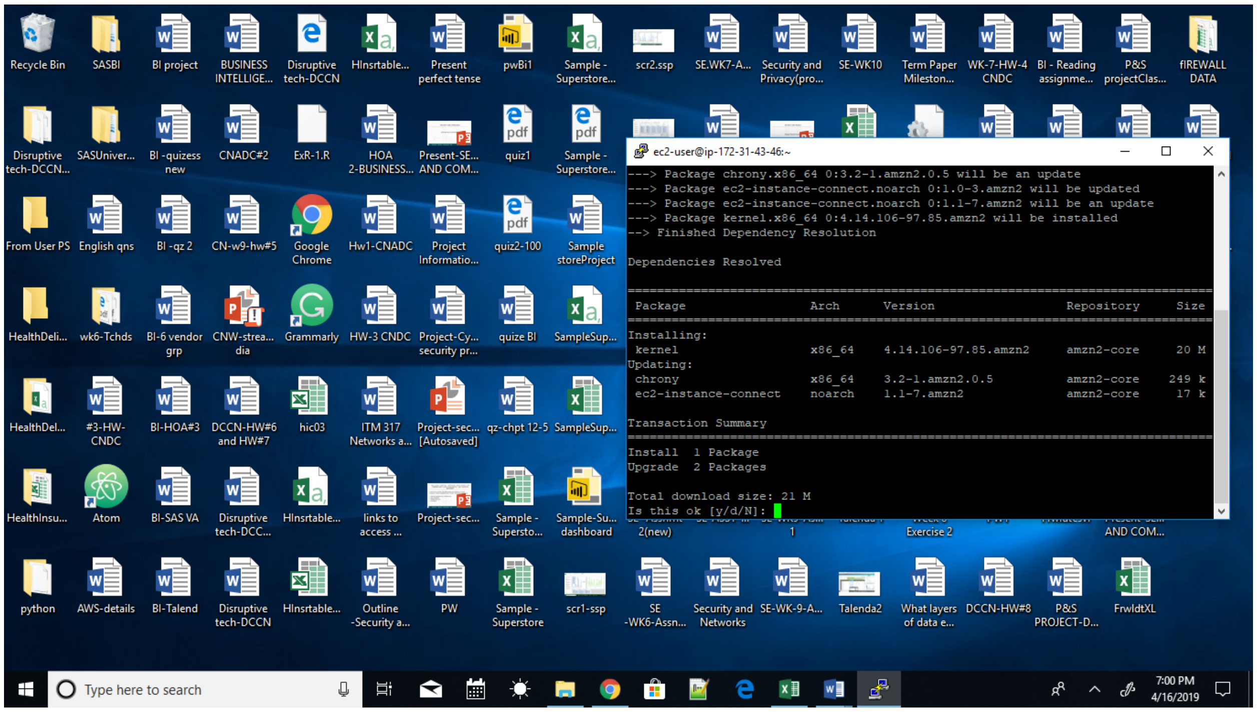
Description | Status Checks | Monitoring | Tags

i-0db4822eca74a15a1:
i-0f903c73838ae0fb3:

Authenticate as a ec2 user.

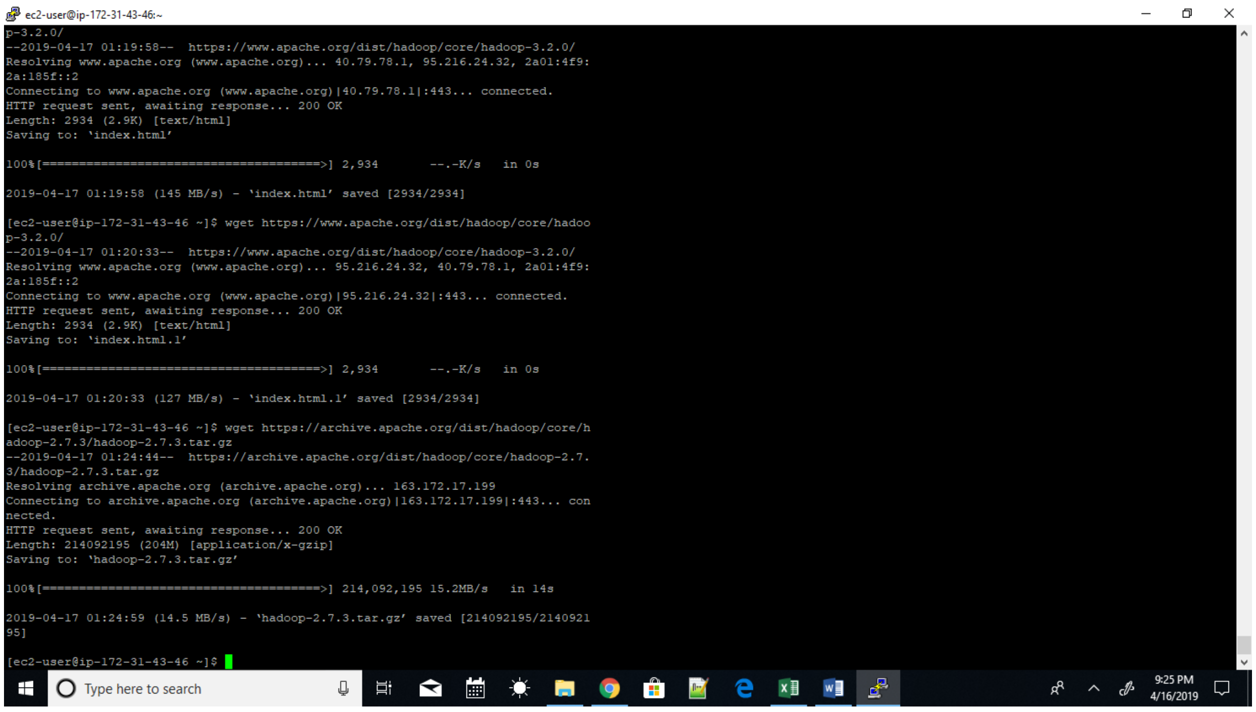


```
login as: ec2-user
Authenticating with public key "imported-openssh-key"

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
1 package(s) needed for security, out of 3 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-43-46 ~]$
```



```
---> Package chrony.x86_64 0:3.2-1.amzn2.0.5 will be an update
---> Package ec2-instance-connect.noarch 0:1.0-3.amzn2 will be updated
---> Package ec2-instance-connect.noarch 0:1.1-7.amzn2 will be an update
---> Package kernel.x86_64 0:4.14.106-97.85.amzn2 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package                 Arch      Version                   Repository    Size
================================================================================
Installing:
 kernel                  x86_64    4.14.106-97.85.amzn2      amzn2-core    20 M
Updating:
 chrony                  x86_64    3.2-1.amzn2.0.5           amzn2-core   249 k
 ec2-instance-connect    noarch    1.1-7.amzn2               amzn2-core    17 k

Transaction Summary
================================================================================
Install  1 Package
Upgrade  2 Packages

Total download size: 21 M
Is this ok [y/d/N]:
```

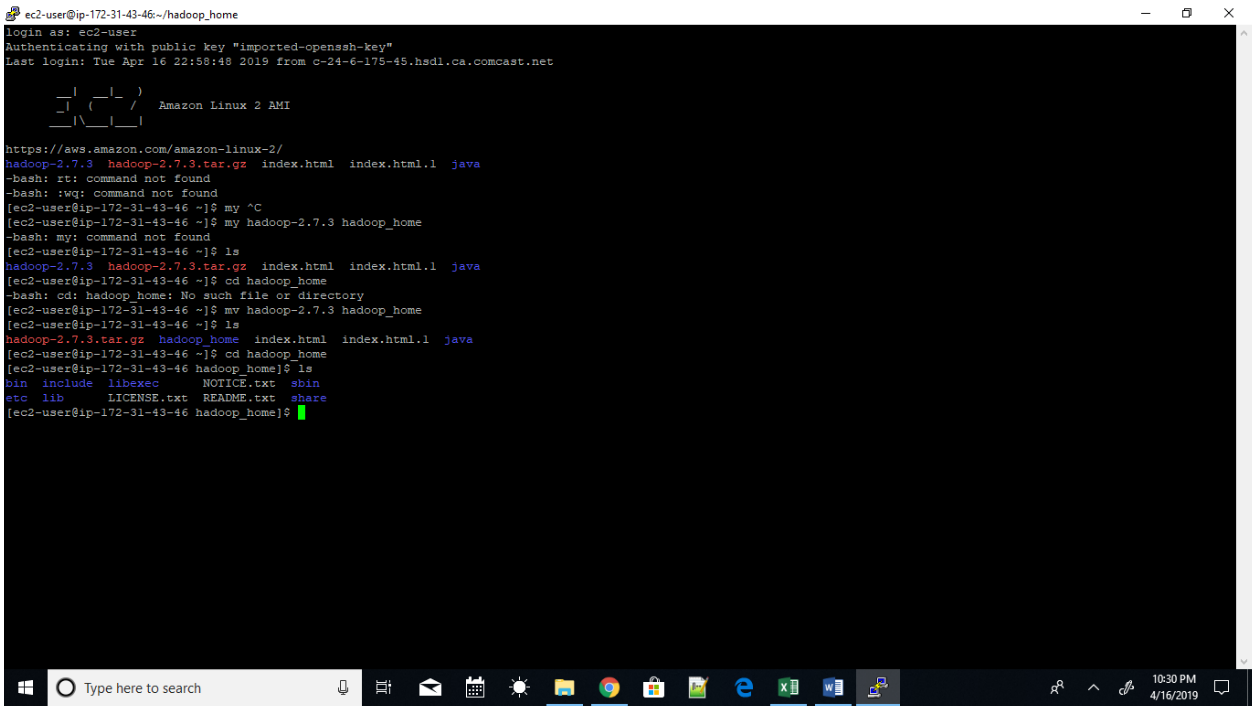
Here I installed Java and Hadoop in aws linux environment.



ec2-user@ip-172-31-43-46:~

```
p-3.2.0/
--2019-04-17 01:19:58--  https://www.apache.org/dist/hadoop/core/hadoop-3.2.0/
Resolving www.apache.org (www.apache.org)... 40.79.78.1, 95.216.24.32, 2a01:4f9:
2a:185f::2
Connecting to www.apache.org (www.apache.org)|40.79.78.1|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2934 (2.9K) [text/html]
Saving to: ‘index.html’

100%[=====================================>] 2,934       --.-K/s   in 0s

2019-04-17 01:19:58 (145 MB/s) - ‘index.html’ saved [2934/2934]

[ec2-user@ip-172-31-43-46 ~]$ wget https://www.apache.org/dist/hadoop/core/hadoo
p-3.2.0/
--2019-04-17 01:20:33--  https://www.apache.org/dist/hadoop/core/hadoop-3.2.0/
Resolving www.apache.org (www.apache.org)... 95.216.24.32, 40.79.78.1, 2a01:4f9:
2a:185f::2
Connecting to www.apache.org (www.apache.org)|95.216.24.32|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2934 (2.9K) [text/html]
Saving to: ‘index.html.1’

100%[=====================================>] 2,934       --.-K/s   in 0s

2019-04-17 01:20:33 (127 MB/s) - ‘index.html.1’ saved [2934/2934]

[ec2-user@ip-172-31-43-46 ~]$ wget https://archive.apache.org/dist/hadoop/core/h
adoop-2.7.3/hadoop-2.7.3.tar.gz
--2019-04-17 01:24:44--  https://archive.apache.org/dist/hadoop/core/hadoop-2.7.
3/hadoop-2.7.3.tar.gz
Resolving archive.apache.org (archive.apache.org)... 163.172.17.199
Connecting to archive.apache.org (archive.apache.org)|163.172.17.199|:443... con
nected.
HTTP request sent, awaiting response... 200 OK
Length: 214092195 (204M) [application/x-gzip]
Saving to: ‘hadoop-2.7.3.tar.gz’

100%[=====================================>] 214,092,195 15.2MB/s   in 14s

2019-04-17 01:24:59 (14.5 MB/s) - ‘hadoop-2.7.3.tar.gz’ saved [214092195/2140921
95]

[ec2-user@ip-172-31-43-46 ~]$
```

Here I installed Hadoop.

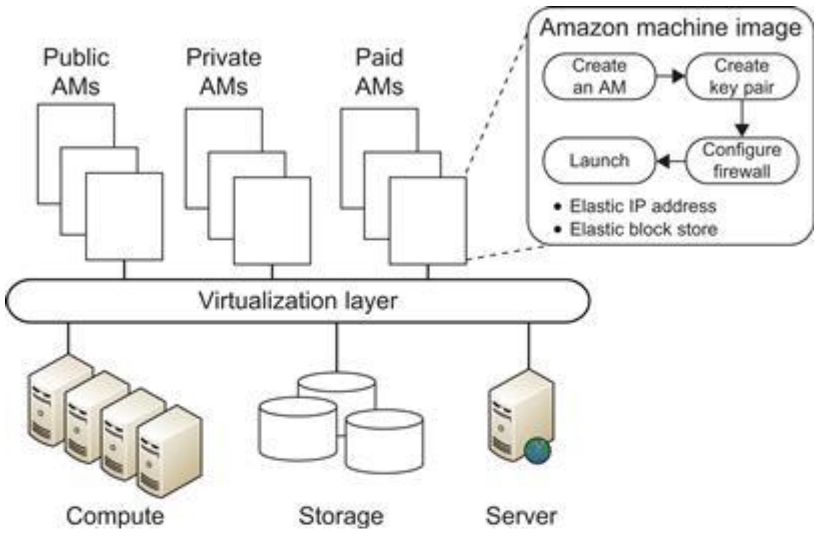


ec2-user@ip-172-31-43-46:~/hadoop_home

```
login as: ec2-user
Authenticating with public key "imported-openssh-key"
Last login: Tue Apr 16 22:58:48 2019 from c-24-6-175-45.hsdl.ca.comcast.net

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
hadoop-2.7.3  hadoop-2.7.3.tar.gz  index.html  index.html.1  java
-bash: rt: command not found
-bash: :wq: command not found
[ec2-user@ip-172-31-43-46 ~]$ my ^C
[ec2-user@ip-172-31-43-46 ~]$ my hadoop-2.7.3 hadoop_home
-bash: my: command not found
[ec2-user@ip-172-31-43-46 ~]$ ls
hadoop-2.7.3  hadoop-2.7.3.tar.gz  index.html  index.html.1  java
[ec2-user@ip-172-31-43-46 ~]$ cd hadoop_home
-bash: cd: hadoop_home: No such file or directory
[ec2-user@ip-172-31-43-46 ~]$ mv hadoop-2.7.3 hadoop_home
[ec2-user@ip-172-31-43-46 ~]$ ls
hadoop-2.7.3.tar.gz  hadoop_home  index.html  index.html.1  java
[ec2-user@ip-172-31-43-46 ~]$ cd hadoop_home
[ec2-user@ip-172-31-43-46 hadoop_home]$ ls
bin  include  libexec      NOTICE.txt  sbin
etc  lib      LICENSE.txt  README.txt  share
[ec2-user@ip-172-31-43-46 hadoop_home]$
```

PROGRAMMING ON AMAZON EC2:

Amazon was the first company to introduce VMs in application hosting. Customers can rent VMs instead of physical machines to run their own applications. By using VMs, customers can load any software of their choice. The elastic feature of such a service is that a customer can create, launch, and terminate server instances as needed, paying by the hour for active servers. Amazon provides several types of preinstalled VMs. Instances are often called *Amazon Machine Images (AMIs)* which are preconfigured with operating systems based on Linux or Windows, and additional software.



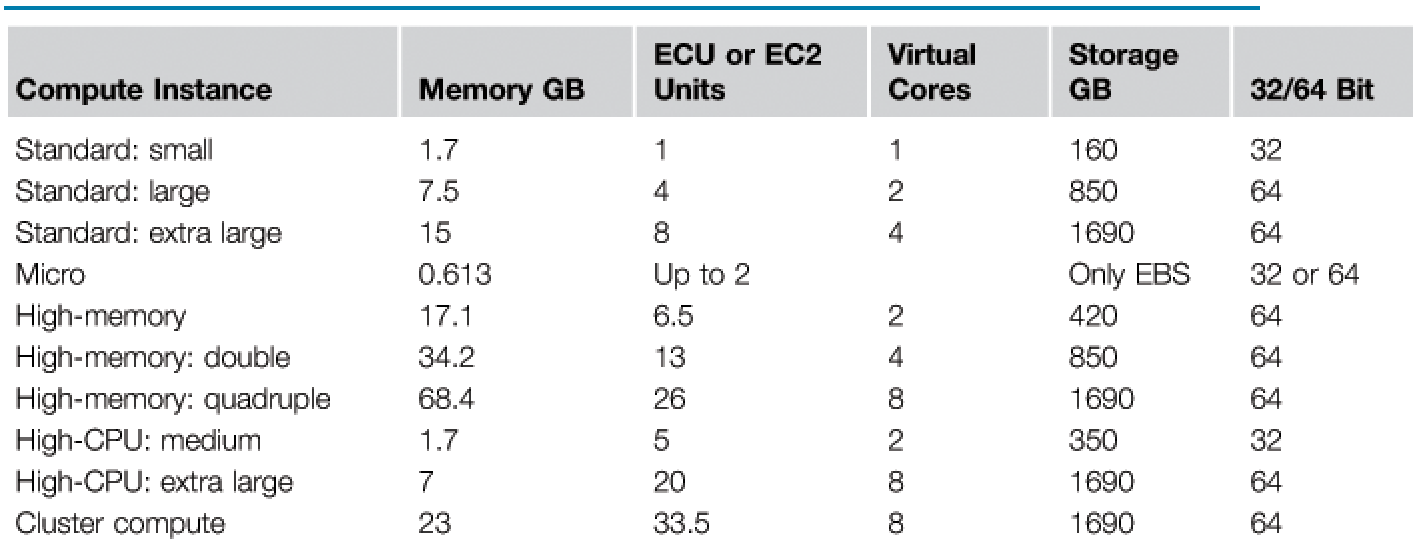
Use of EC2 Services in the AWS Platform:

The IaaS instances available in October 2010 in five broad classes:

1. **Standard instances** are well suited for most applications.
2. **Micro instances** provide a small number of consistent CPU resources and allow you to burst CPU capacity when additional cycles are available. They are well suited for lower throughput applications and web sites that consume significant compute cycles periodically.

3. **High-memory instances** offer large memory sizes for high-throughput applications, including database and memory caching applications.
4. **High-CPU instances** have proportionally more CPU resources than memory (RAM) and are well suited for compute-intensive applications.
5. **Cluster compute instances** provide proportionally high CPU resources with increased network performance and are well suited for high-performance computing (HPC) applications and other demanding network-bound applications. They use 10 Gigabit Ethernet interconnections.

Instance Types Available on Amazon EC2 (October 6, 2010)



Compute Instance	Memory GB	ECU or EC2 Units	Virtual Cores	Storage GB	32/64 Bit
Standard: small	1.7	1	1	160	32
Standard: large	7.5	4	2	850	64
Standard: extra large	15	8	4	1690	64
Micro	0.613	Up to 2		Only EBS	32 or 64
High-memory	17.1	6.5	2	420	64
High-memory: double	34.2	13	4	850	64
High-memory: quadruple	68.4	26	8	1690	64
High-CPU: medium	1.7	5	2	350	32
High-CPU: extra large	7	20	8	1690	64
Cluster compute	23	33.5	8	1690	64

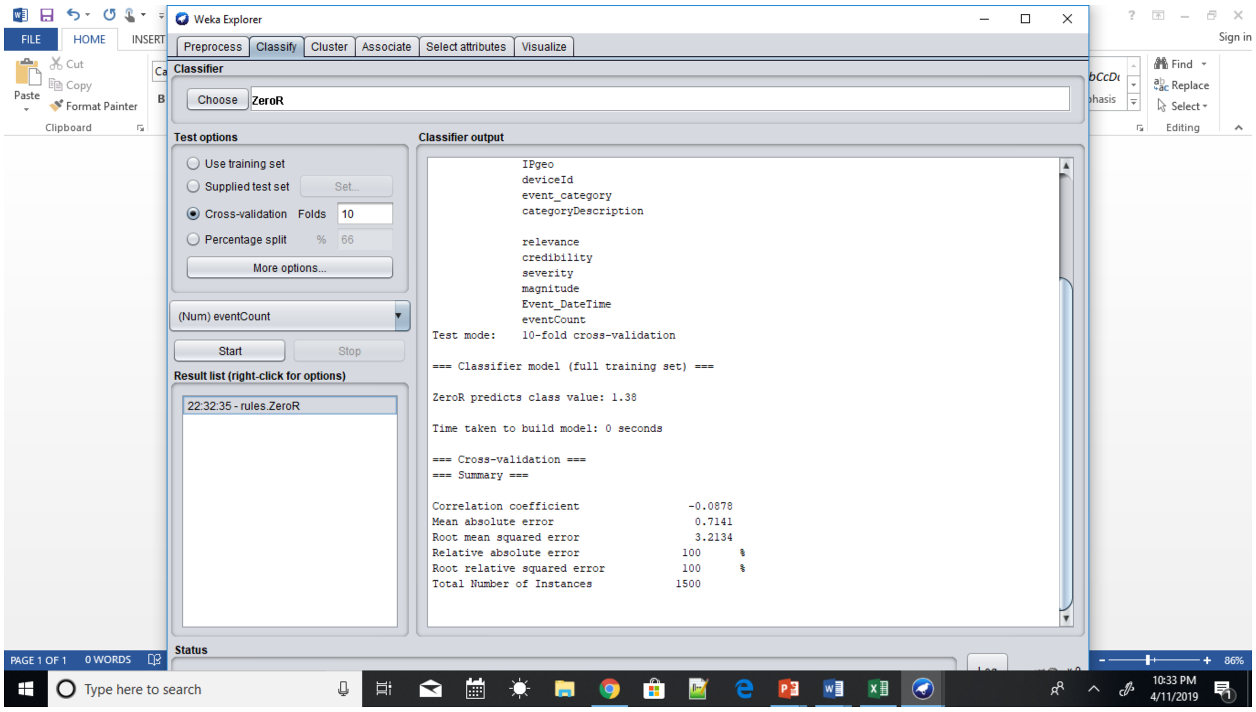
Data in machine learning:

Data is the fuel that drives the machine learning engine. Data, when fed to machine learning systems, helps in detecting patterns and mining data.

Bayesian networks:

Bayesian network (**BN**) are probabilistic models that are primarily used for prediction and decision making. These are belief networks that use the principles of probability theory along with

statistics.



Classifier

Choose ZeroR

Test options

Use training set

Supplied test set

Cross-validation Folds 10

Percentage split % 66

(Num) eventCount

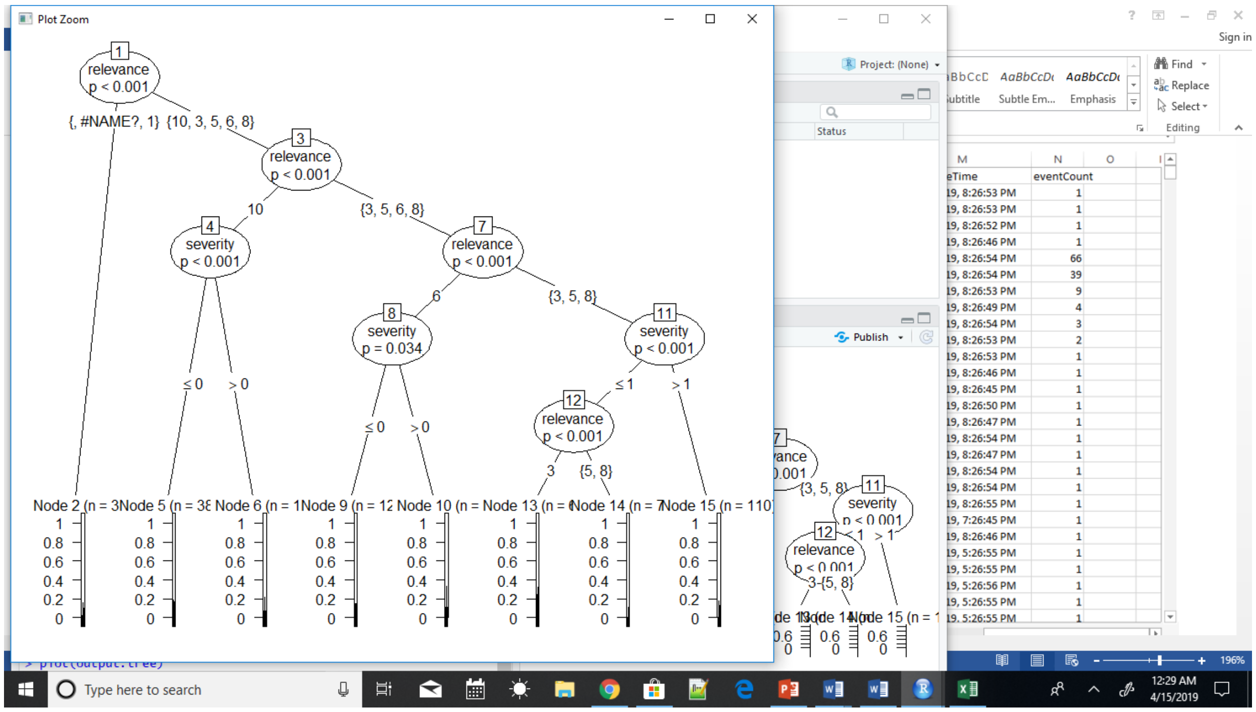
Result list (right-click for options)

22:32:35 - rules.ZeroR

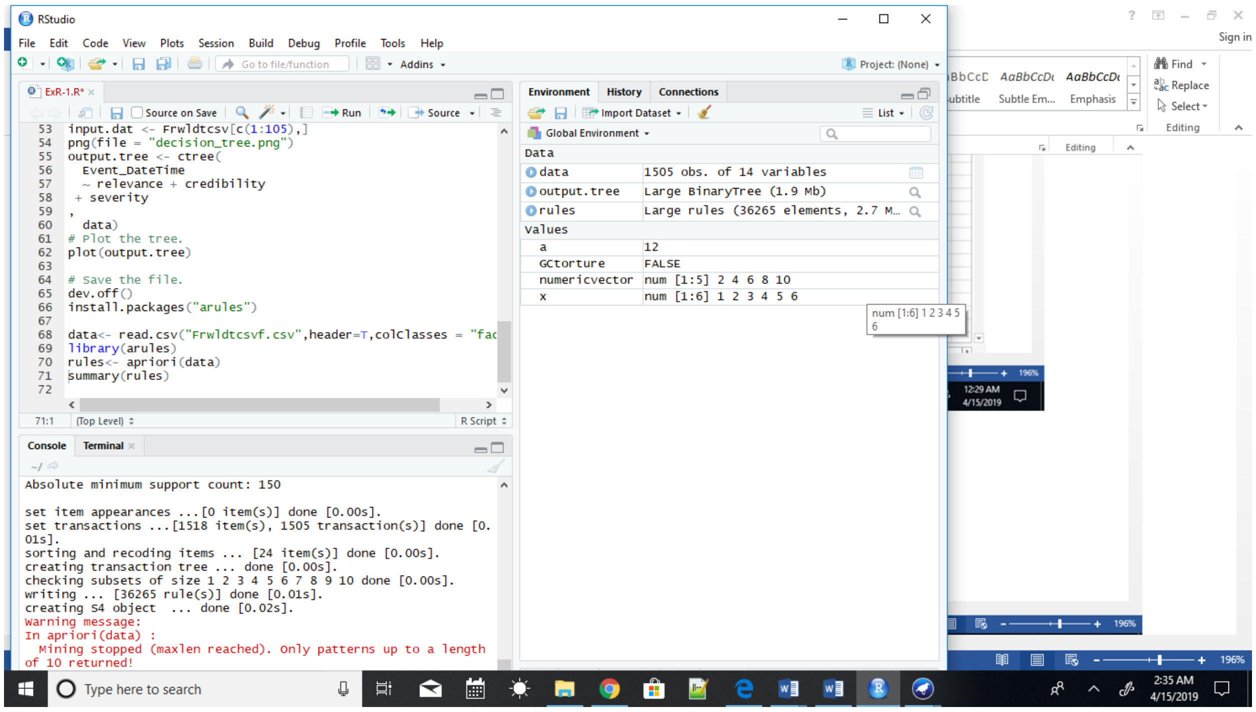
Classifier output

```
              IPgeo
              deviceId
              event_category
              categoryDescription

              relevance
              credibility
              severity
              magnitude
              Event_DateTime
              eventCount
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

ZeroR predicts class value: 1.38

Time taken to build model: 0 seconds

=== Cross-validation ===
=== Summary ===

Correlation coefficient                 -0.0878
Mean absolute error                      0.7141
Root mean squared error                  3.2134
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances             1500
```

DECISION TREE:



ASSOCIATION RULE:

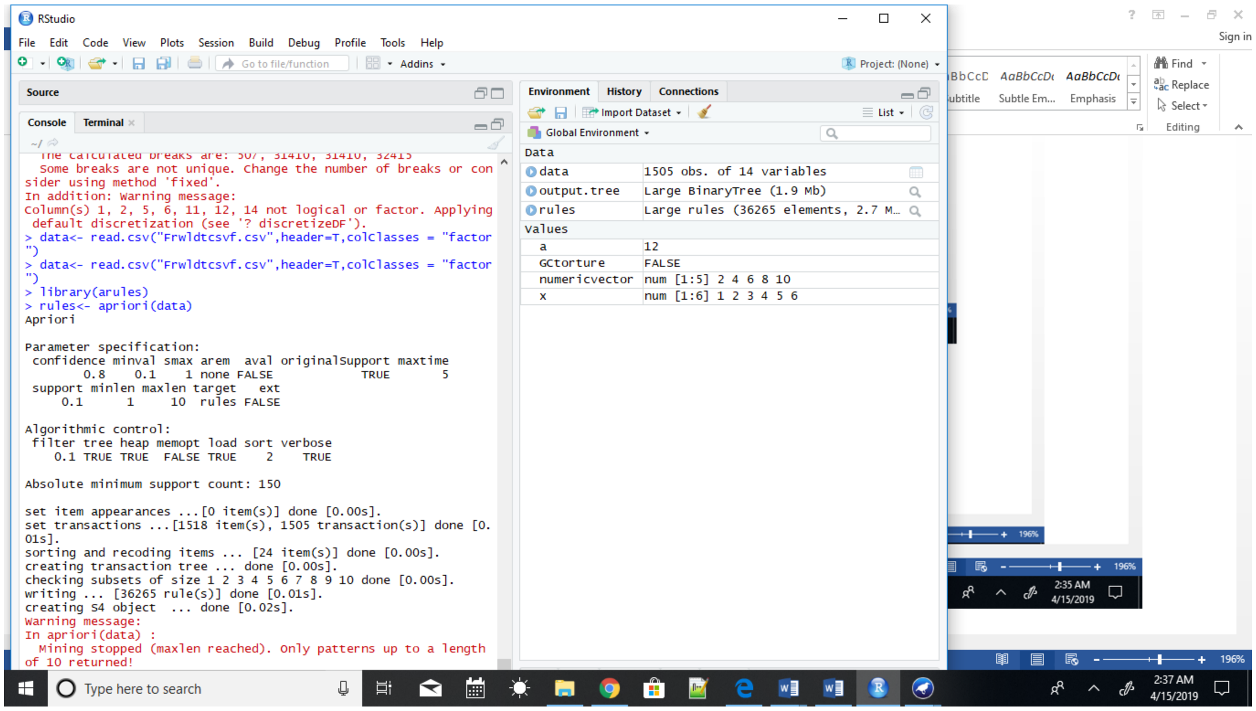


```r
input.dat <- Frwldtcsv[c(1:105),]
png(file = "decision_tree.png")
output.tree <- ctree(
  Event_DateTime
  ~ relevance + credibility
 + severity
,
  data)
# Plot the tree.
plot(output.tree)

# Save the file.
dev.off()
install.packages("arules")

data<- read.csv("Frwldtcsvf.csv",header=T,colClasses = "fac
library(arules)
rules<- apriori(data)
summary(rules)
```

```
Data
data            1505 obs. of 14 variables
output.tree     Large BinaryTree (1.9 Mb)
rules           Large rules (36265 elements, 2.7 M...
Values
a               12
GCtorture       FALSE
numericvector   num [1:5] 2 4 6 8 10
x               num [1:6] 1 2 3 4 5 6
```

```
Absolute minimum support count: 150

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[1518 item(s), 1505 transaction(s)] done [0.
01s].
sorting and recoding items ... [24 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 8 9 10 done [0.00s].
writing ... [36265 rule(s)] done [0.01s].
creating S4 object  ... done [0.02s].
Warning message:
In apriori(data) :
  Mining stopped (maxlen reached). Only patterns up to a length
of 10 returned!
```

36365 elements, so that many rules are generated when running this rules.



```
  The calculated breaks are: 507, 31410, 31410, 32415
  Some breaks are not unique. Change the number of breaks or con
sider using method 'fixed'.
In addition: Warning message:
Column(s) 1, 2, 5, 6, 11, 12, 14 not logical or factor. Applying
 default discretization (see '? discretizeDF').
> data<- read.csv("Frwldtcsvf.csv",header=T,colClasses = "factor
")
> data<- read.csv("Frwldtcsvf.csv",header=T,colClasses = "factor
")
> library(arules)
> rules<- apriori(data)
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime
        0.8    0.1    1 none FALSE            TRUE       5
 support minlen maxlen target   ext
     0.1      1     10  rules FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 150

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[1518 item(s), 1505 transaction(s)] done [0.
01s].
sorting and recoding items ... [24 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 8 9 10 done [0.00s].
writing ... [36265 rule(s)] done [0.01s].
creating S4 object  ... done [0.02s].
Warning message:
In apriori(data) :
  Mining stopped (maxlen reached). Only patterns up to a length
of 10 returned!
```

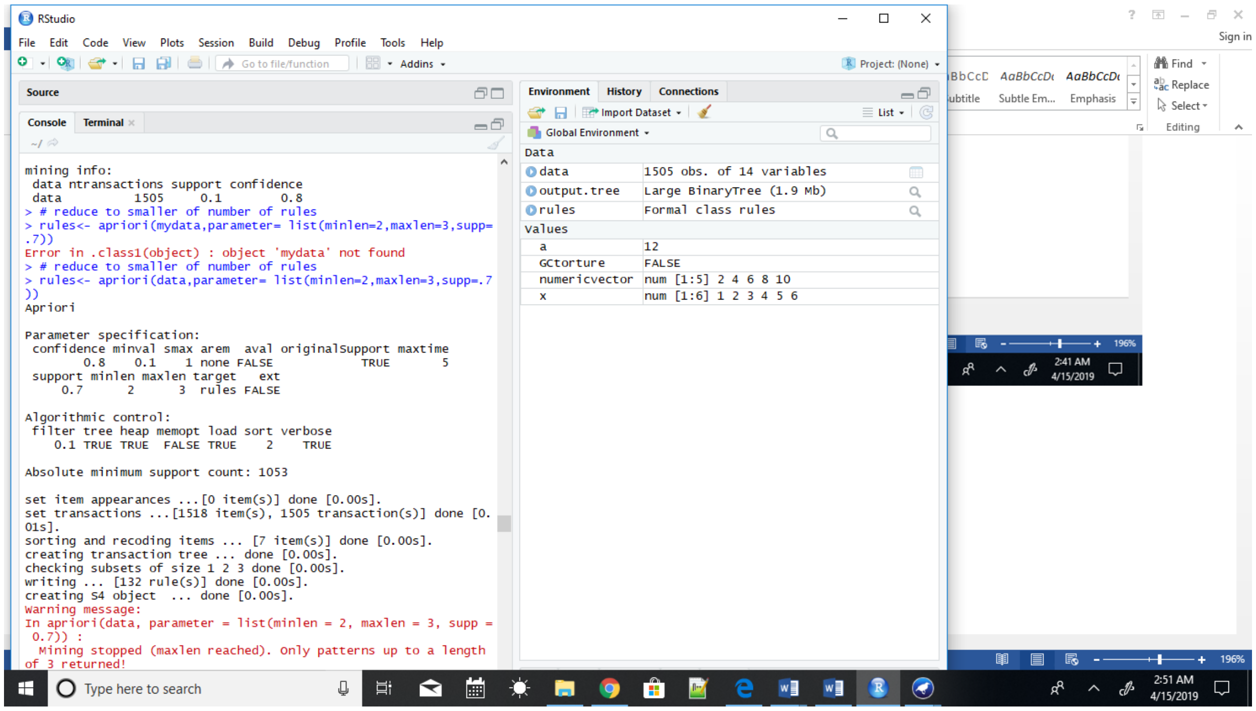
```
Data
data            1505 obs. of 14 variables
output.tree     Large BinaryTree (1.9 Mb)
rules           Large rules (36265 elements, 2.7 M…
Values
a               12
GCtorture       FALSE
numericvector   num [1:5] 2 4 6 8 10
x               num [1:6] 1 2 3 4 5 6
```

Summary.

7 rules for 1 item 791 rules with 10 items.

Quality measures:

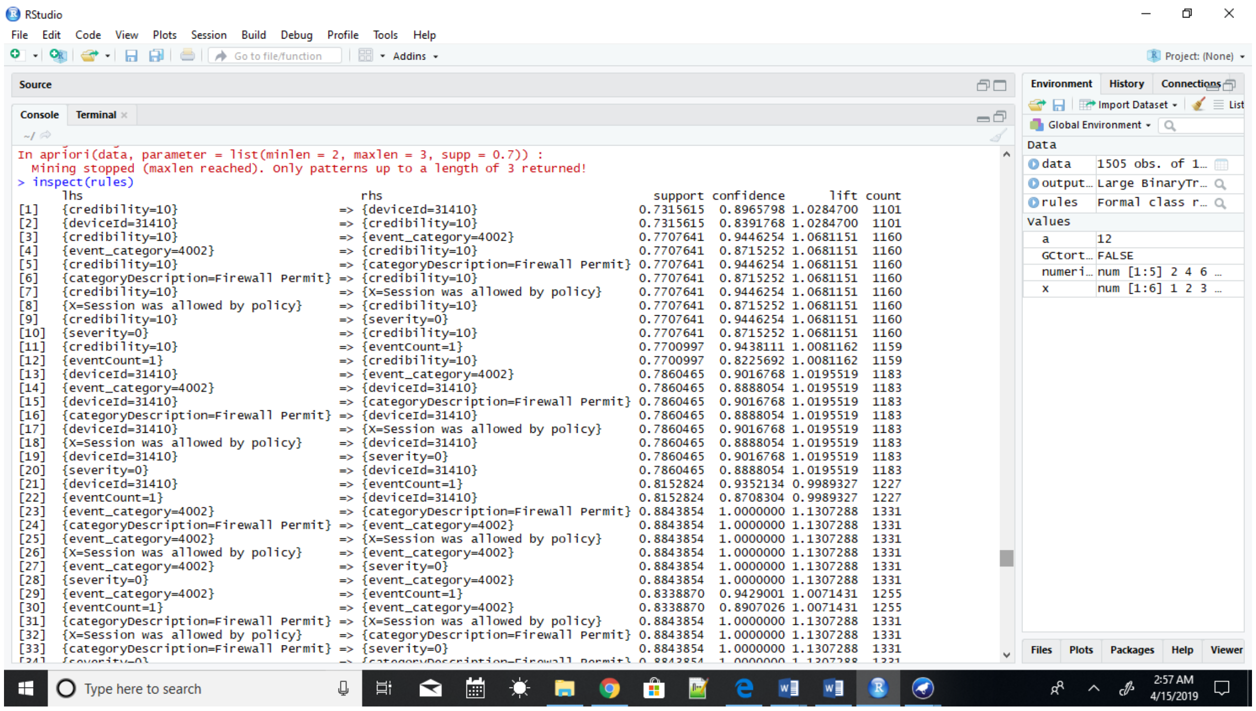
Support: Min. : 10% and Max. : 0.9362, Confidence: Max. :1.0000



```
mining info:
 data ntransactions support confidence
 data          1505     0.1        0.8
> # reduce to smaller of number of rules
> rules<- apriori(mydata,parameter= list(minlen=2,maxlen=3,supp=
.7))
Error in .class1(object) : object 'mydata' not found
> # reduce to smaller of number of rules
> rules<- apriori(data,parameter= list(minlen=2,maxlen=3,supp=.7
))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime
        0.8    0.1    1 none FALSE            TRUE       5
 support minlen maxlen target   ext
     0.7      2      3  rules FALSE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 1053

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[1518 item(s), 1505 transaction(s)] done [0.
01s].
sorting and recoding items ... [7 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done [0.00s].
writing ... [132 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
Warning message:
In apriori(data, parameter = list(minlen = 2, maxlen = 3, supp =
 0.7)) :
  Mining stopped (maxlen reached). Only patterns up to a length
of 3 returned!
```

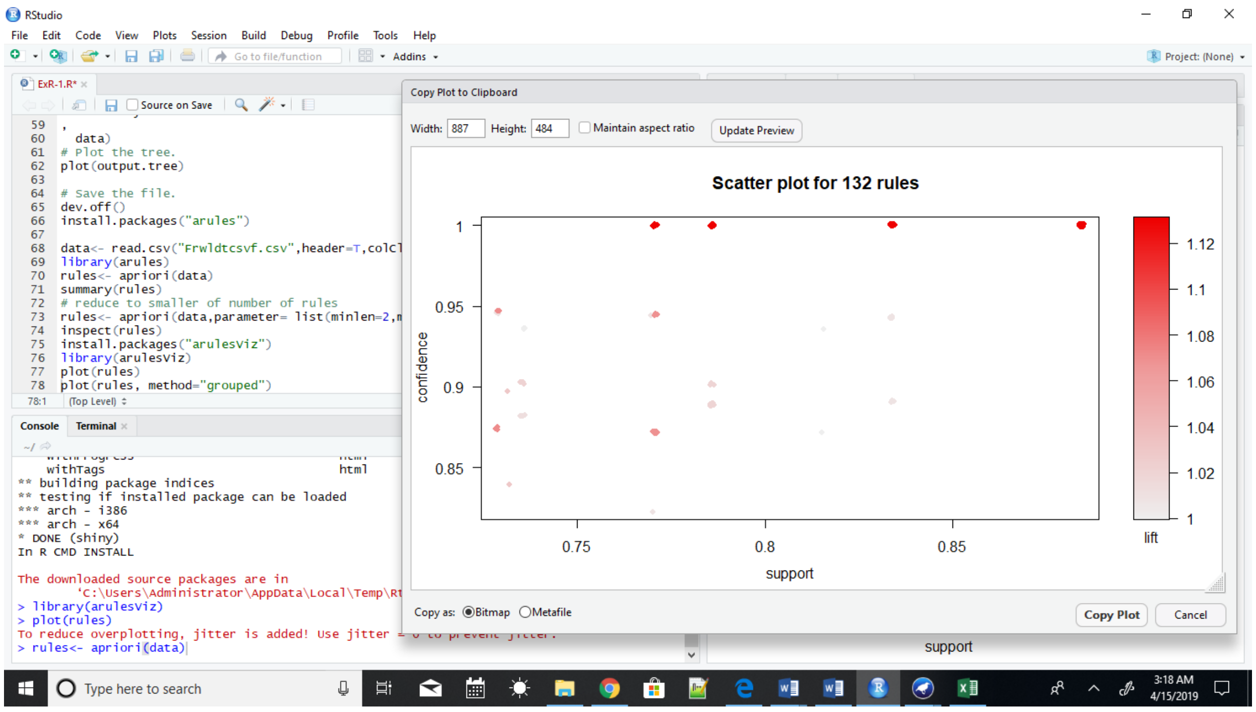
```
Data
 data            1505 obs. of 14 variables
 output.tree     Large BinaryTree (1.9 Mb)
 rules           Formal class rules
Values
 a               12
 GCtorture       FALSE
 numericvector   num [1:5] 2 4 6 8 10
 x               num [1:6] 1 2 3 4 5 6
```

Support: 0.7, minlen=2, maxlen=3, and confidence= 0.8.



```
In apriori(data, parameter = list(minlen = 2, maxlen = 3, supp = 0.7)) :
  Mining stopped (maxlen reached). Only patterns up to a length of 3 returned!
> inspect(rules)
     lhs                                  rhs                                support   confidence lift      count
[1]  {credibility=10}                  => {deviceId=31410}                   0.7315615 0.8965798  1.0284700 1101
[2]  {deviceId=31410}                  => {credibility=10}                   0.7315615 0.8391768  1.0284700 1101
[3]  {credibility=10}                  => {event_category=4002}              0.7707641 0.9446254  1.0681151 1160
[4]  {event_category=4002}             => {credibility=10}                   0.7707641 0.8715252  1.0681151 1160
[5]  {credibility=10}                  => {categoryDescription=Firewall Permit} 0.7707641 0.9446254 1.0681151 1160
[6]  {categoryDescription=Firewall Permit} => {credibility=10}               0.7707641 0.8715252  1.0681151 1160
[7]  {credibility=10}                  => {X=Session was allowed by policy}  0.7707641 0.9446254  1.0681151 1160
[8]  {X=Session was allowed by policy} => {credibility=10}                   0.7707641 0.8715252  1.0681151 1160
[9]  {credibility=10}                  => {severity=0}                       0.7707641 0.9446254  1.0681151 1160
[10] {severity=0}                      => {credibility=10}                   0.7707641 0.8715252  1.0681151 1160
[11] {credibility=10}                  => {eventCount=1}                     0.7700997 0.9438111  1.0081162 1159
[12] {eventCount=1}                    => {credibility=10}                   0.7700997 0.8225692  1.0081162 1159
[13] {deviceId=31410}                  => {event_category=4002}              0.7860465 0.9016768  1.0195519 1183
[14] {event_category=4002}             => {deviceId=31410}                   0.7860465 0.8888054  1.0195519 1183
[15] {deviceId=31410}                  => {categoryDescription=Firewall Permit} 0.7860465 0.9016768 1.0195519 1183
[16] {categoryDescription=Firewall Permit} => {deviceId=31410}               0.7860465 0.8888054  1.0195519 1183
[17] {deviceId=31410}                  => {X=Session was allowed by policy}  0.7860465 0.9016768  1.0195519 1183
[18] {X=Session was allowed by policy} => {deviceId=31410}                   0.7860465 0.8888054  1.0195519 1183
[19] {deviceId=31410}                  => {severity=0}                       0.7860465 0.9016768  1.0195519 1183
[20] {severity=0}                      => {deviceId=31410}                   0.7860465 0.8888054  1.0195519 1183
[21] {deviceId=31410}                  => {eventCount=1}                     0.8152824 0.9352134  0.9989327 1227
[22] {eventCount=1}                    => {deviceId=31410}                   0.8152824 0.8708304  0.9989327 1227
[23] {event_category=4002}             => {categoryDescription=Firewall Permit} 0.8843854 1.0000000 1.1307288 1331
[24] {categoryDescription=Firewall Permit} => {event_category=4002}          0.8843854 1.0000000  1.1307288 1331
[25] {event_category=4002}             => {X=Session was allowed by policy}  0.8843854 1.0000000  1.1307288 1331
[26] {X=Session was allowed by policy} => {event_category=4002}              0.8843854 1.0000000  1.1307288 1331
[27] {event_category=4002}             => {severity=0}                       0.8843854 1.0000000  1.1307288 1331
[28] {severity=0}                      => {event_category=4002}              0.8843854 1.0000000  1.1307288 1331
[29] {event_category=4002}             => {eventCount=1}                     0.8338870 0.9429001  1.0071431 1255
[30] {eventCount=1}                    => {event_category=4002}              0.8338870 0.8907026  1.0071431 1255
[31] {categoryDescription=Firewall Permit} => {X=Session was allowed by policy} 0.8843854 1.0000000 1.1307288 1331
[32] {X=Session was allowed by policy} => {categoryDescription=Firewall Permit} 0.8843854 1.0000000 1.1307288 1331
[33] {categoryDescription=Firewall Permit} => {severity=0}                   0.8843854 1.0000000  1.1307288 1331
```

Credibility=10 lhs and deviceId = 31410 rhs when the confidence level = 0.8965798. The confidence level is changing when credibility is in rhs, and deviceId is lhs.



```
,
  data)
# Plot the tree.
plot(output.tree)

# Save the file.
dev.off()
install.packages("arules")

data<- read.csv("Frwldtcsvf.csv",header=T,colcl
library(arules)
rules<- apriori(data)
summary(rules)
# reduce to smaller of number of rules
rules<- apriori(data,parameter= list(minlen=2,m
inspect(rules)
install.packages("arulesViz")
library(arulesViz)
plot(rules)
plot(rules, method="grouped")
```

```
    withTags                                        html
** building package indices
** testing if installed package can be loaded
*** arch - i386
*** arch - x64
* DONE (shiny)
In R CMD INSTALL

The downloaded source packages are in
        'C:\Users\Administrator\AppData\Local\Temp\Rt
> library(arulesViz)
> plot(rules)
To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.
> rules<- apriori(data)
```

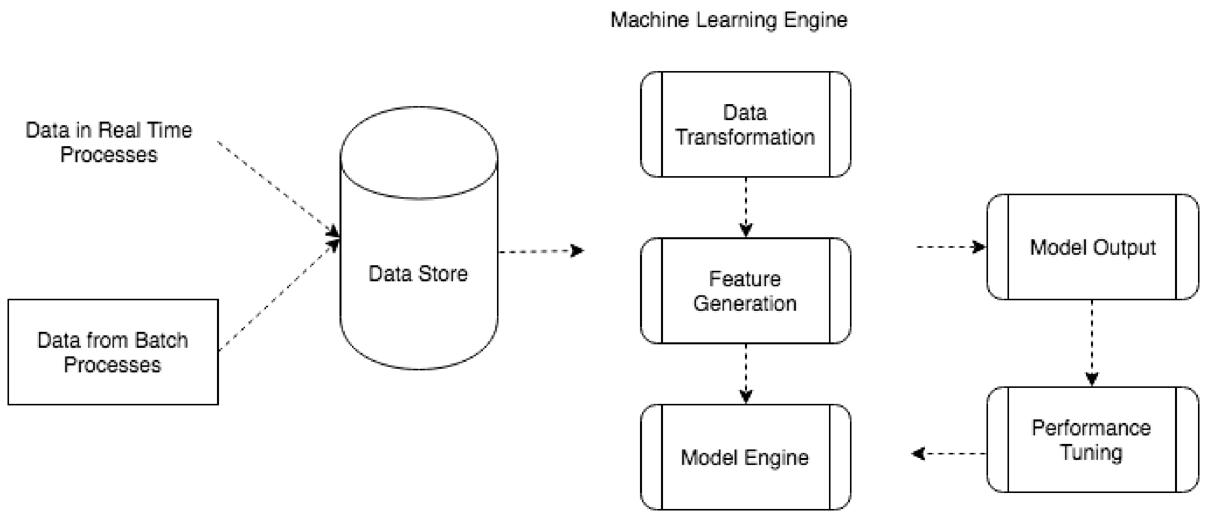
Dark color is indicating the lift level s high that is the confidence level and support level is high. The light color is indicating the lift value is low.

Decision trees:

Decision tree learning is a predictive machine learning technique that uses decision trees. Decision trees make use of decision analysis and predict the value of the target. Decision trees are simple implementations of classification problems and popular in operations research. Decisions are made by the output value predicted by the conditional variable.

The machine learning architecture:

A typical machine learning system comprises a pipeline of processes that happens in a sequence for any type of machine learning system, irrespective of the industry. The following diagram shows a typical machine learning system and the sub-processes involved:



CONCLUSION:

In conclusion, to solve computational problems, a parallel and distributed computing system uses multiple computers to solve large-scale problems over the Internet. In particular, we focused on viable approaches to building distributed operating systems for handling massive parallelism in a distributed environment. Moreover, I described about the cost to build and maintain data center servers and other maintenance cost as well. To increase the performance speed of the programs, the Amdahl's law teaches us that we should make the sequential bottleneck as small as possible. Consequently, explained in detail about a distributed computing system. Furthermore, I provided an example for amazon EC2 service also. After that, how the decision tree algorithm primarily used for prediction and decision making. Finally, in this paper I explained about the Association Rule to explain the quality measure of confidence level and support for firewall log data.