

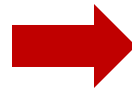
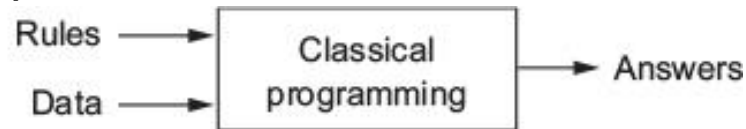
## 10. 데이터 분석 - 고급

# 새로운 프로그램 패러다임

기존의 프로그램 방식과는 달리 최근에는 Machine Learning을 이용하여 문제를 해결하는 방법이 많이 사용되고 있다.

## □ 기존방식

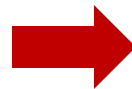
- 사전 준비한 Rules과 Data를 이용하여, 해답을 제공하는 프로그램을 수작업으로 개발함
- 프로그램 구조 : 정적인 구조(프로그램 개발후에 Rule이 변경되면, 코드 수정을 통하여 해당 내용을 반영함)



Logic이 이미 결정됨  
(정적인 프로그램)

## □ 새로운 방식

- 사전 준비한 Data와 학습용 해답을 이용하여, 문제를 해결하기 위한 Rules을 자동으로 만들어 냄
- 프로그램 구조 : 동적인 구조(Data만 있으면 기계가 스스로 학습하여 Rules을 만들어 내고 프로그램에서



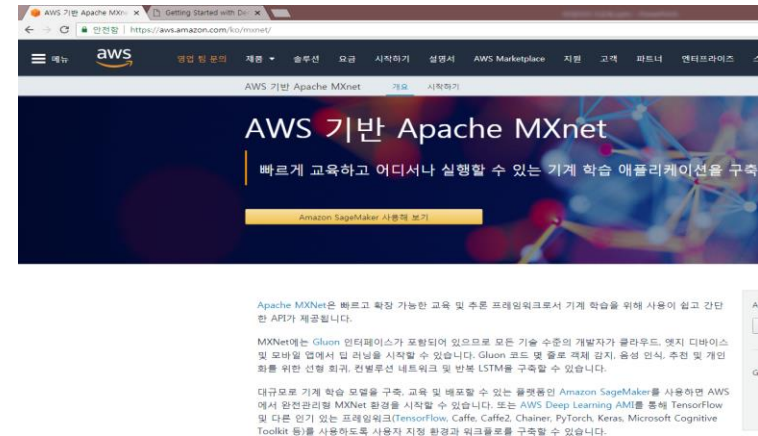
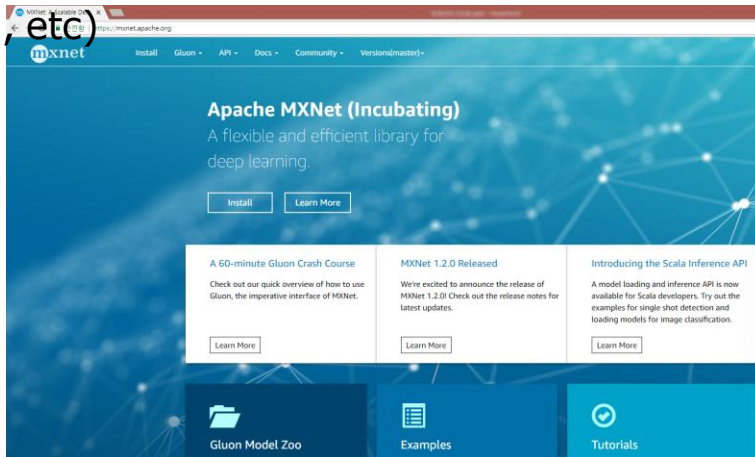
Logic이 결정되어 있지 않음  
(동적인 프로그램)

mxnet이란 CMU에서 개발한 이후 Apache DMLC에서 운영하고 있는 오픈소스 형태의 딥러닝 프레임워크이다.

## □ 개요

- 개발 : CMU(<http://www.cs.cmu.edu/~muli/file/mxnet-learning-sys.pdf>)
- 운영 : DMLC(Distributed Machine Learning Community : CMU, NYU, NVIDIA, MS, Baidu,

Amazon, etc)



아파치 mxnet사이트(<https://mxnet.apache.org>) 아마존 AWS사이트(<https://aws.amazon.com/ko/mxnet/>)

## □ 장점

- 여러 언어 지원 : Python, C++, R, Scala, Julia, Matlab, JavaScript
- CPU와 GPU연산을 지원
- GPU, mobile에서 동작 가능
- 혼합 패러다임 지원(symbolic/imperative)
- 최적화된 C++엔진으로 좋은 성능을 발휘할 수 있음


# 딥러닝 플랫폼

딥러닝 개발을 위한 플랫폼에 대한 비교 분석 결과는 하기와 같다.

F/W	주체	플랫폼	모바일	언어	인터페이스	OpenMP	CUDA	OpenCL	멀티GPU	분산
<b>Caffe</b>	BAIR	Linux, Mac	-	C++	Python, MATAB	Y	Y	-	Y	
<b>Chainer</b>	Preferred Networks	Linux	-	Python	Python	-	Y	-	Y	Y
<b>CNTK</b>	Microsoft	Linux, Windows	-	C++	Python, C++	Y	Y	-	Y	Y
<b>DL4J</b>	SkyMind	Cross-platform (JVM)	Android	Java	Java, Scala, Python	Y	Y	-	Y	Y (Spark)
<b>Keras</b>	François Chollet	Linux, Mac, Windows	-	Python	Python	Y(Theano) N(TF)	Y	-	Y	
<b>MXNet</b>	DMLC	Linux, Mac, Windows, Javascript	Android, iOS	C++	C++, Python, Julia, MATLAB, JavaScript, Go, R, Scala, Perl	Y	Y	-	Y	Y
<b>TensorFlow</b>	Google	Linux, Mac, Windows	Android, iOS	C++, Python	Python, C/C++, Java, Go	N	Y	-	Y	Y
<b>Theano</b>	Université de Montréal	Linux, Mac, Windows	-	Python	Python	Y	Y	-	Y	
<b>Torch</b>	Ronan, Clément, Koray, Soumith	Linux, Mac, Windows	Android, iOS	C, Lua	Lua	Y	Y	Y	Y	Not officiall y

# 딥러닝 플랫폼

활용성 측면에서 딥러닝 개발을 위한 플랫폼을 비교한 결과는 하기와 같다.

	Languages	Tutorials and training materials	CNN modeling capability	RNN modeling capability	Architecture: easy-to-use and modular front end	Speed	Multiple GPU support	Keras compatible
Theano	Python, C++	++	++	++	+	++	+	+
Tensor-Flow	Python	+++	+++	++	+++	++	++	+
Torch	Lua, Python (new)	+	+++	++	++	+++	++	
Caffe	C++	+	++		+	+	+	
MXNet	R, Python, Julia, Scala	++	++	+	++	++	+++	
Neon	Python	+	++	+	+	++	+	
CNTK	C++	+	+	+++	+	++	+	

R에서 mxnet CPU버전을 사용하기 위해서는 하기와 같은 방법으로 관련 패키지를

설치해야 한다

## □ mxnet 설치 명령

```
cran <- getOption("repos")
cran["dmlc"] <- "https://apache-mxnet.s3-accelerate.dualstack.amazonaws.com/R/CRAN/"
options(repos = cran)
```

## □ mxnet 설치 진행화면

```
also installing the dependencies 'XML', 'Rook', 'downloader', 'igraph', 'influenceR', 'rgexf', 'DiagrammeR', 'visNetwork'

Warning in install.packages :
  unable to access index for repository https://apache-mxnet.s3-accelerate.dualstack.amazonaws.com/R/CRAN/bin/windows/contrib/3.5/:
  cannot open URL 'https://apache-mxnet.s3-accelerate.dualstack.amazonaws.com/R/CRAN/bin/windows/contrib/3.5/PACKAGES'
Package which is only available in source form, and may need compilation of C/C++/Fortran:
  'mxnet'
These will not be installed
trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.5/XML_3.98-1.11.zip'
Content type 'application/zip' length 4601014 bytes (4.4 MB)
downloaded 4.4 MB

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.5/Rook_1.1-1.zip'
Content type 'application/zip' length 436638 bytes (426 KB)
downloaded 426 KB

trying URL 'https://cran.rstudio.com/bin/windows/contrib/3.5/downloader_0.4.zip'
Content type 'application/zip' length 24715 bytes (24 KB)
downloaded 24 KB
```

## □ R버전이 최신버전이어서 mxnet 패키지 설치시 오류가 발생할 경우 구글링을 하여 문제를

해결한다.

```
install.packages("https://s3.ca-central-1.amazonaws.com/jeremiedb/share/mxnet/CPU/mxnet.zip",
  repos = NULL)
```

# mxnet 설치 및 확인 - 행렬연산

mxnet를 설치가 제대로 되었는지 예제 프로그램을 실행하여 확인한다.

## □ mxnet 설치가 제대로 되었는지 확인

```
> library(mxnet)
# 2행 3열의 1값을 갖는 행렬을 생성한다.
> a <- mx.nd.ones(c(2,3), ctx = mx.cpu())
# 각 행렬의 값에 2를 곱한 후 1을 더하는 연산을
실행한다.
> b <- a * 2 + 1
> b
      [,1] [,2] [,3]
[1,]    3    3    3
[2,]    3    3    3
> a
      [,1] [,2] [,3]
[1,]    1    1    1
[2,]    1    1    1
```

# Classification

mxnet를 이용하여 classification 분석을 수행하기 위한 준비작업은 하기와 같다.

## □ 필요한 package 설치

```
> install.packages('mlbench')
```

## □ 데이터 불러오기

```
> require(mlbench)
```

필요한 패키지를 로딩중입니다: mlbench

```
> require(mxnet)
```

```
> data(Sonar, package='mlbench')
```

```
> Sonar
```

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10
1	0.0200	0.0371	0.0428	0.0207	0.0954	0.0986	0.1539	0.1601	0.3109	0.2111
2	0.0453	0.0523	0.0843	0.0689	0.1183	0.2583	0.2156	0.3481	0.3337	0.2872
3	0.0262	0.0582	0.1099	0.1083	0.0974	0.2280	0.2431	0.3771	0.5598	0.6194

.....

```
> nrow(Sonar)
```

```
[1] 208
```

```
> ncol(Sonar)
```

```
[1] 61
```

```
> #입력변수 60개, 출력변수1개(Class)
```

Sonar 데이터셋은 다양한 각도와 다양한 조건하에서 금속 실린더로부터 수중 음파 탐지기 신호를 수신하여 얻은 111 가지 패턴에 대한 데이터셋이다.

각 데이터는 0.0에서 1.0의 범위를 갖는 60개의 변수값을 갖는다.

각 신호패턴에 대해 물체가 바위(Rock)인 경우 R로 분류하고, 광석(Metal)인 경우 M으로 분류한 데이터이다.



## Classification

mxnet를 이용하여 classification 분석을 수행하기 위한 준비작업은 하기와 같다.

## □ 데이터 전처리(종속변수에 대한 양자화)

[illegible]

mxnet를 이용하여 classification 분석을 수행하기 위한 준비작업은 하기와 같다.

## □ 학습용, 훈련용 데이터 준비

```
> train.ind=c(1:50,100:150)
> train.ind
[1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
[19] 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36
[37] 37 38 39 40 41 42 43 44 45 46 47 48 49 50 100 101 102 103
[55] 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121
[73] 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139
[91] 140 141 142 143 144 145 146 147 148 149 150
> # 훈련용으로 101개의 데이터를 활용하고, 나머지는 테스트용으로 데이터를 활용함
> train.x=data.matrix(Sonar[train.ind,1:60])
> train.y=Sonar[train.ind,61]
> test.x=data.matrix(Sonar[-train.ind, 1:60])
> test.y=Sonar[-train.ind,61]
```

# Classification

mxnet를 이용하여 classification 분석 모델링은 하기와 같이 수행한다.

## □ 신경망 모델링

```
> mx.set.seed(0)
> model<-mx.mlp(train.x,train.y, #훈련용 데이터 설정
+ hidden_node=100, # 은닉노드 100개
+ out_node=2, #출력노드 2개
+ out_activation='softmax', #활성화 함수는 소프트맥스 설정
+ num.round=20, #훈련 iteration은 20회 실시
+ array.batch.size=15, #매 iteration별로 배치사이즈는 15개
+ learning.rate=0.07, #학습률은 0.07
+ momentum=0.9, #최적화 방법으로 모멘텀사용하고
설정값은 0.9
+ eval.metric=mx.metric.accuracy) #평가척도는 정확도
```

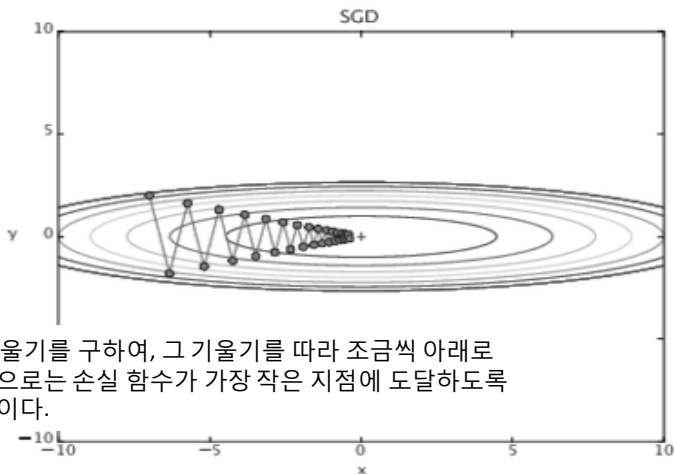
## □ 훈련 과정 출력

```
Start training with 1 devices
[1] Train-accuracy=0.485714299338205
[2] Train-accuracy=0.466666677168437
[3] Train-accuracy=0.523809535162789
[4] Train-accuracy=0.580952397414616
[5] Train-accuracy=0.704761922359467
[6] Train-accuracy=0.714285731315613
[7] Train-accuracy=0.733333349227905
[8] Train-accuracy=0.752380967140198
[9] Train-accuracy=0.809523820877075
[10] Train-accuracy=0.819047629833221
[11] Train-accuracy=0.809523820877075
[12] Train-accuracy=0.790476202964783
[13] Train-accuracy=0.77142858505249
[14] Train-accuracy=0.819047629833221
[15] Train-accuracy=0.838095247745514
[16] Train-accuracy=0.876190483570099
[17] Train-accuracy=0.857142865657806
[18] Train-accuracy=0.838095247745514
[19] Train-accuracy=0.828571438789368
[20] Train-accuracy=0.84761905670166
```

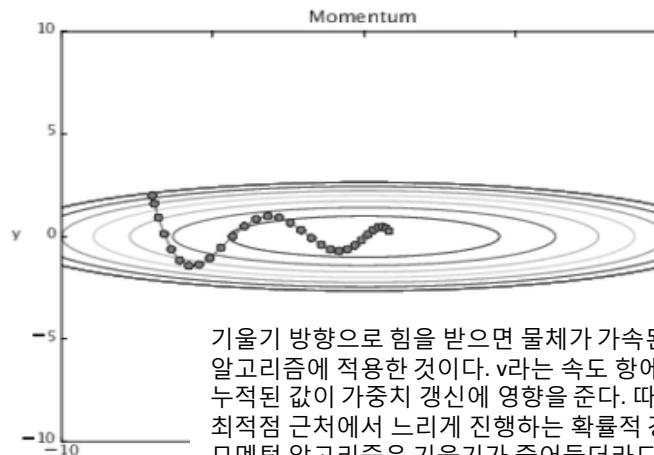
# 가중치 최적화 방법

신경망에서 학습은 매 iteration마다 가중치를 조정하여 최적값에 도출하도록 하는 작업을 수행하는 것을 말하며

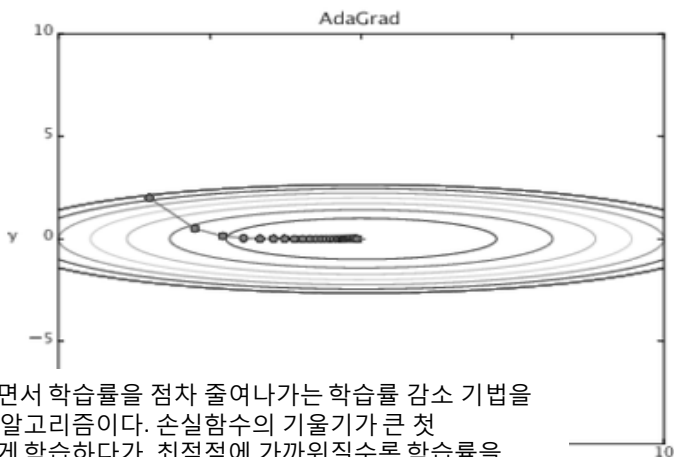
여기서 최적화 방법에는 하기와 같이 다양한 방법이 있다.



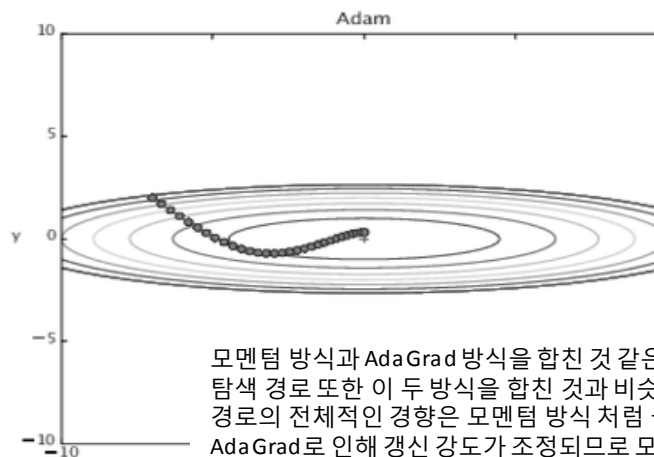
손실 함수의 기울기를 구하여, 그 기울기를 따라 조금씩 아래로 내려가 최종적으로는 손실 함수가 가장 작은 지점에 도달하도록 하는 알고리즘이다.



기울기 방향으로 힘을 받으면 물체가 가속된다는 물리 법칙을 알고리즘에 적용한 것이다.  $v$ 라는 속도 항에 기울기 값이 누적되고, 그 누적된 값이 가중치 갱신에 영향을 준다. 따라서 기울기가 줄어드는 최적점 근처에서 느리게 진행하는 확률적 경사 하강법(SGD)와는 다르게, 모멘텀 알고리즘은 기울기가 줄어들더라도 누적된 기울기 값으로 인해 빠르게 최적점으로 수렴하게 된다.



학습을 진행하면서 학습률을 점차 줄여나가는 학습률 감소 기법을 적용한 최적화 알고리즘이다. 손실함수의 기울기가 큰 첫 부분에서는 크게 학습하다가, 최적점에 가까워질수록 학습률을 줄여 조금씩 작게 학습하는 방식이다



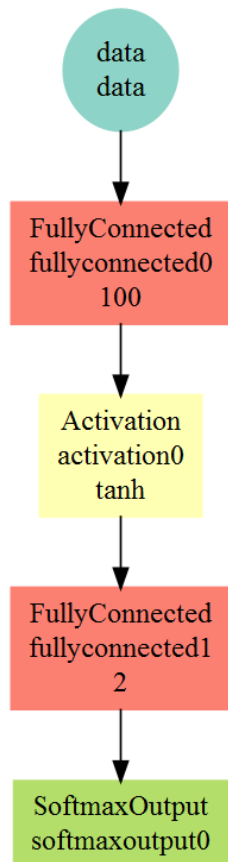
모멘텀 방식과 AdaGrad 방식을 합친 것 같은 알고리즘이므로, 최적점 탐색 경로 또한 이 두 방식을 합친 것과 비슷한 양상으로 나타난다. 탐색 경로의 전체적인 경향은 모멘텀 방식처럼 공이 굴러가는 듯하고, AdaGrad로 인해 갱신 강도가 조정되므로 모멘텀 방식보다 좌우 흔들림이 덜 한 것을 볼 수 있다.

# Classification

mxnet를 이용하여 classification 분석 결과는 하기와 같다.

## □ 신경망 모델링 구조 그래프 출력

```
> graph.viz(model$symbol) #Rstudio viewer창에서  
조회가능
```



## □ 모델링 결과 테스트

```
> preds=predict(model, test.x)  
> pred.label<-max.col(t(preds))-1  
#confusion matrix를 이용하여 모델링 적합도  
검정  
> table(pred.label, test.y)
```

		test.y	
pred.label	0	1	
	0	22	13
	1	38	34