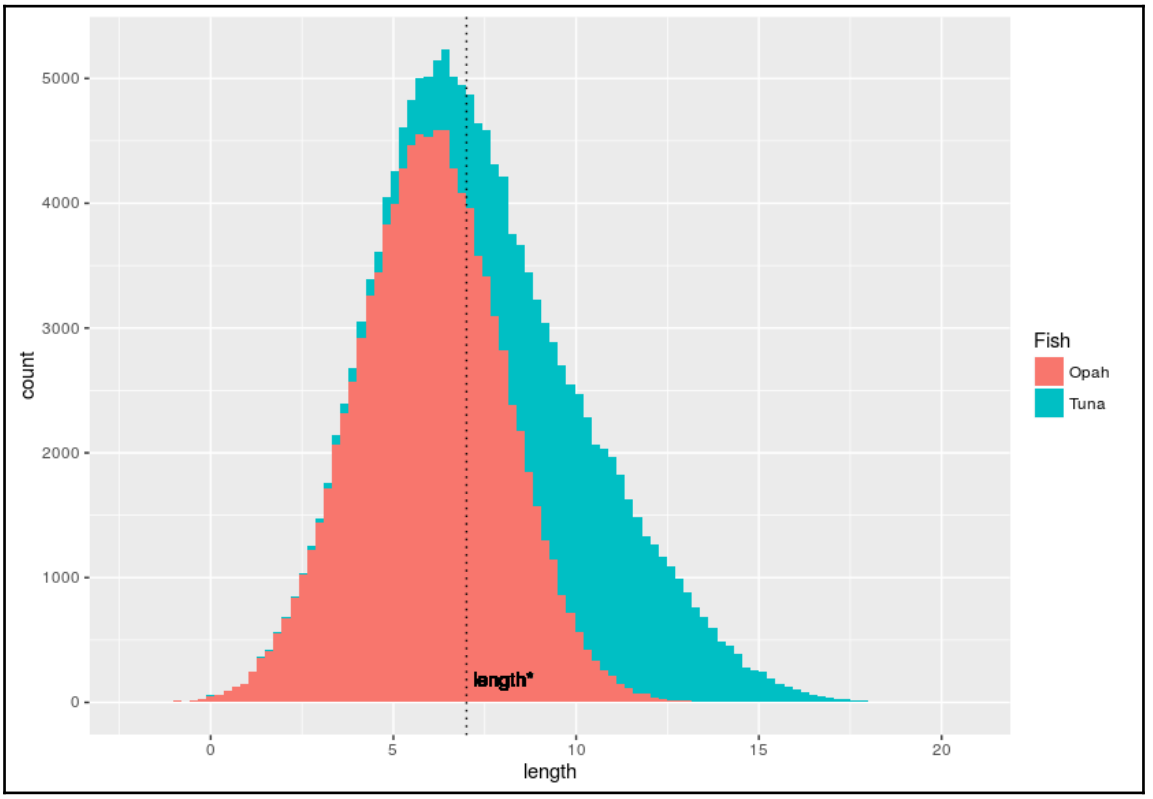
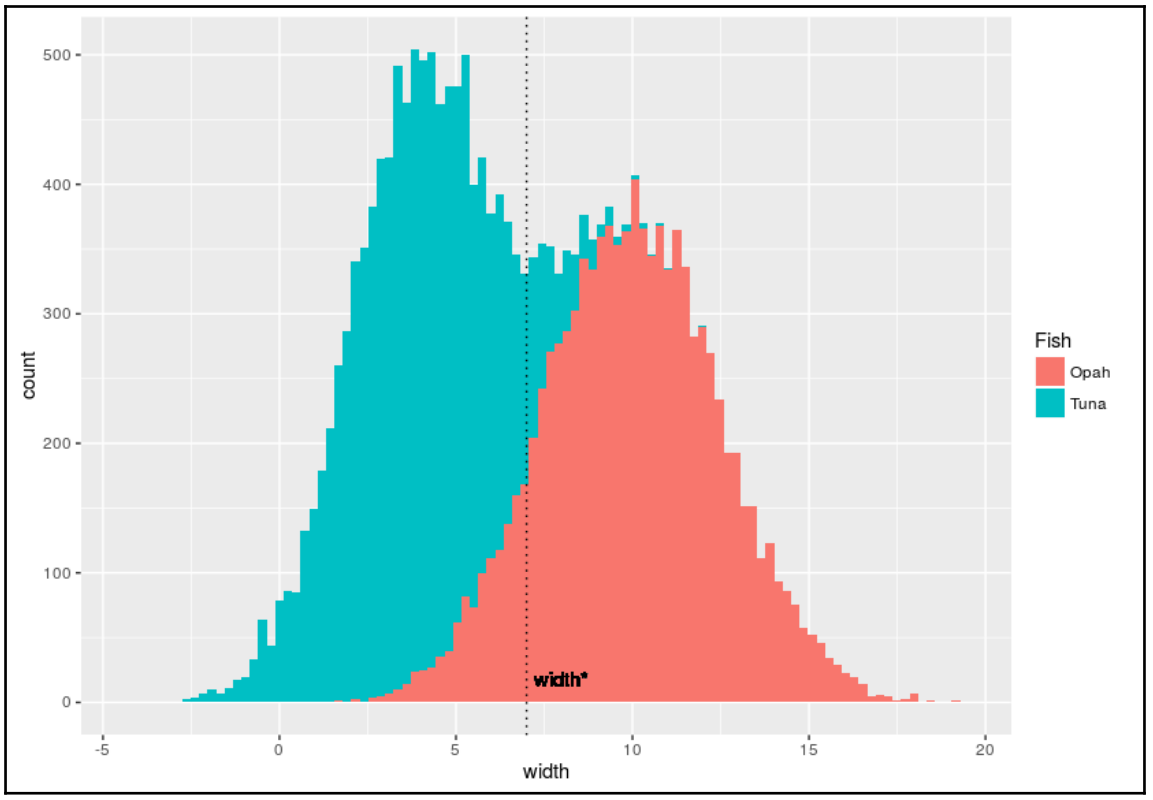
Table of Contents

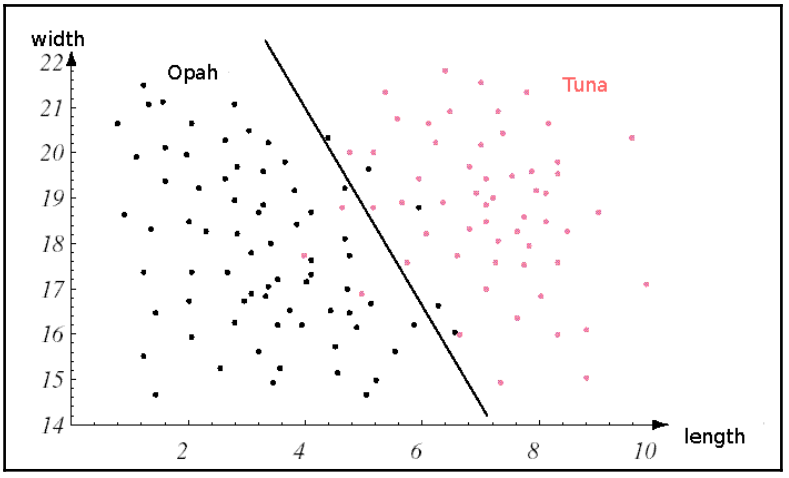
1

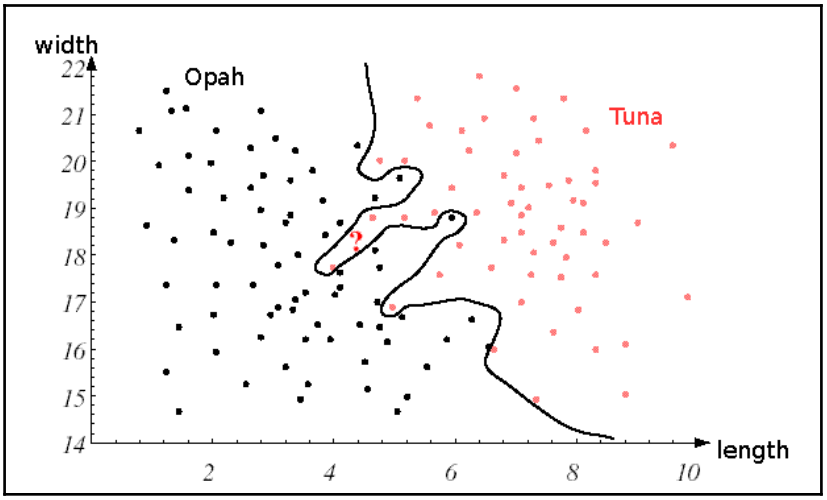
Index 76

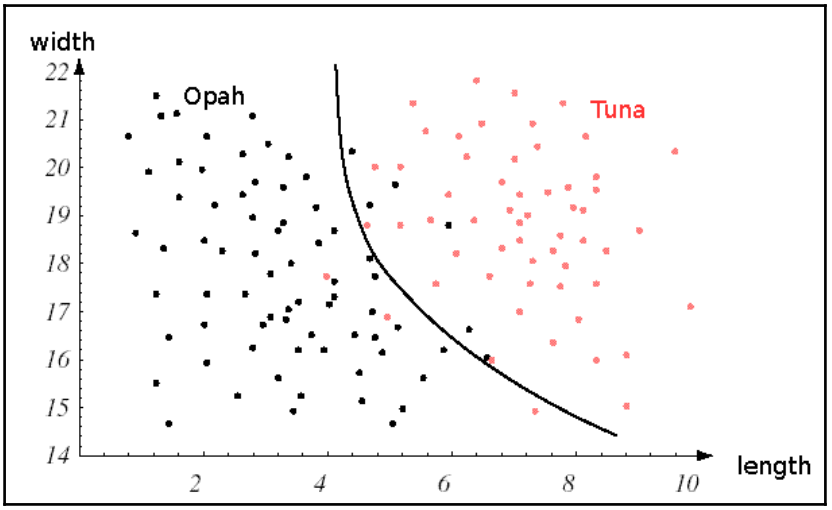
Chapter 1: Data Science - A Birds' Eye View

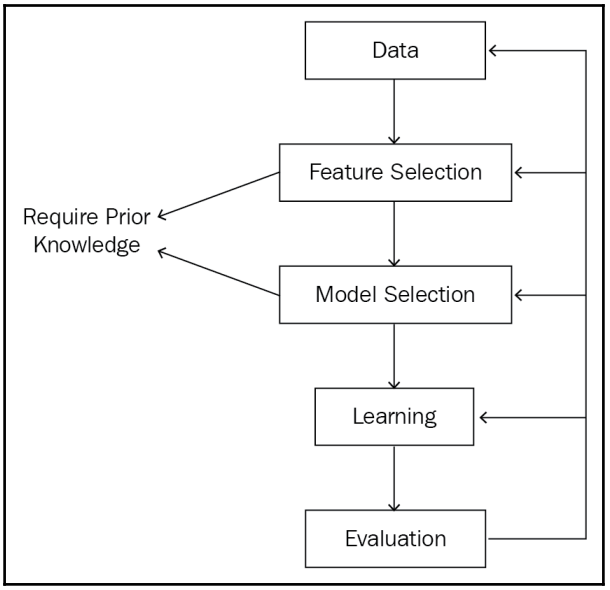


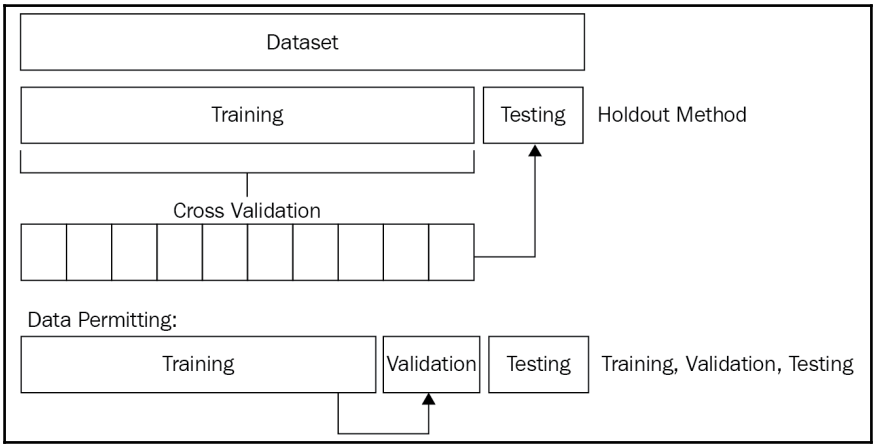

500
400
300
200
100
0
count
-5
0
5
10
15
20
width
width*
Fish
Opah
Tuna

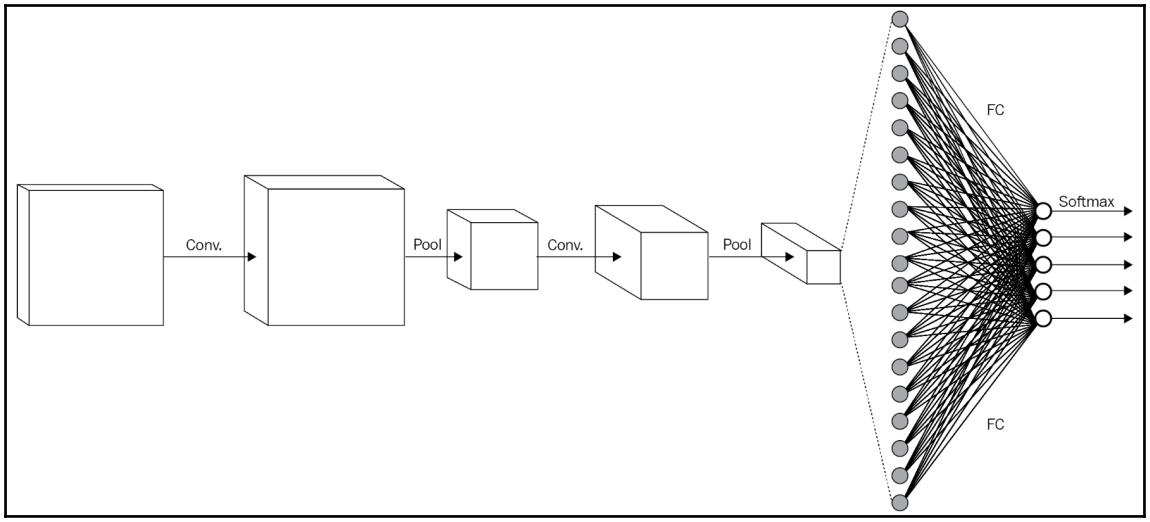

width
22
21
20
19
18
17
16
15
14
Opah
Tuna
2
4
6
8
10
length

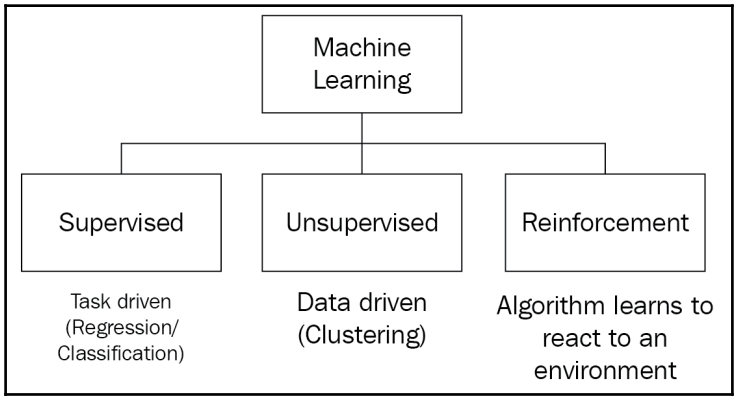

width
22
21
20
19
18
17
16
15
14
Opah
Tuna
?
2
4
6
8
10
length

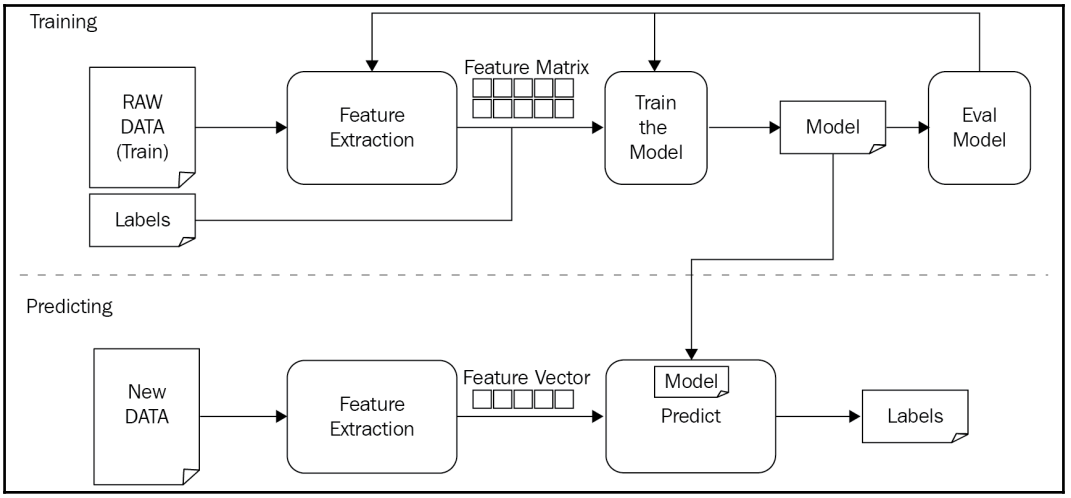

width
22
21
20
19
18
17
16
15
14
Opah
Tuna
2
4
6
8
10
length

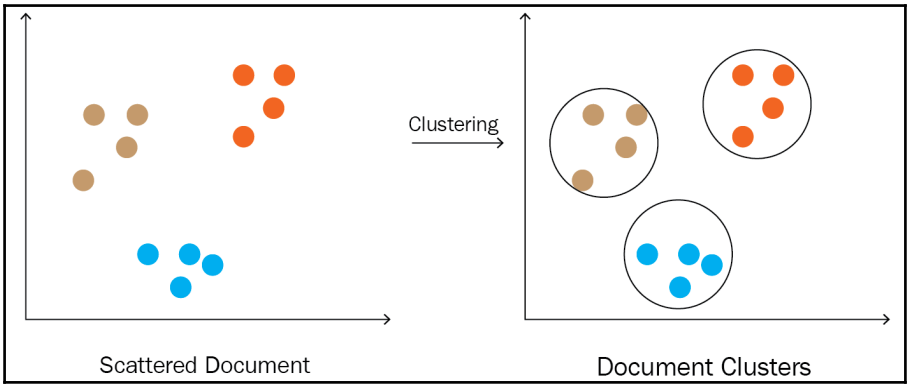

Data
Feature Selection
Require Prior Knowledge
Model Selection
Learning
Evaluation


Dataset
Training
Testing
Holdout Method
Cross Validation
Data Permitting:
Training
Validation
Testing
Training, Validation, Testing

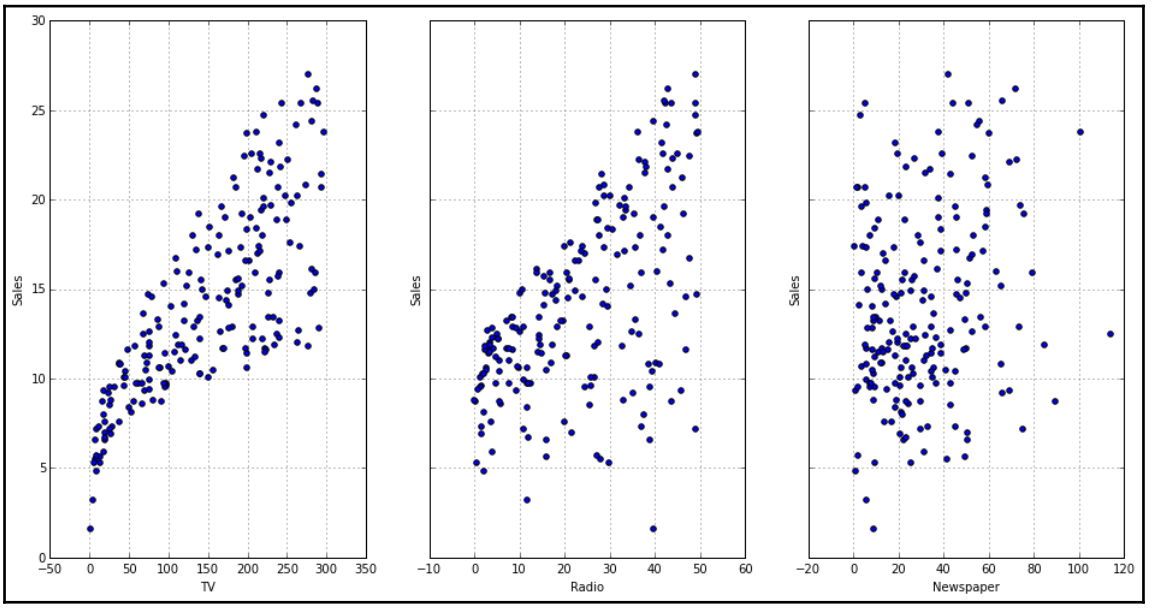

Conv.
Pool
Conv.
Pool
FC
Softmax
FC

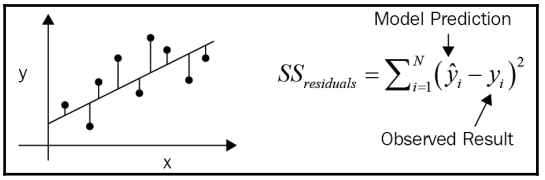

Machine Learning
Supervised
Unsupervised
Reinforcement
Task driven (Regression/ Classification)
Data driven (Clustering)
Algorithm learns to react to an environment

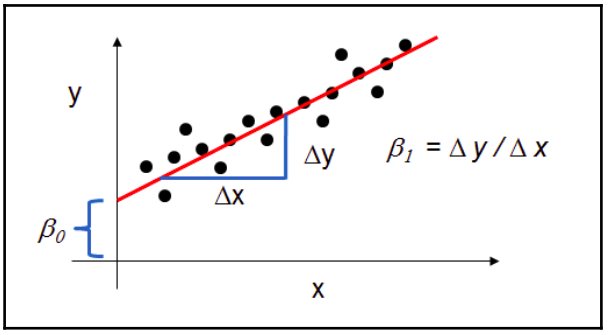

Training
RAW DATA (Train)
Labels
Feature Extraction
Feature Matrix
Train the Model
Model
Eval Model
Predicting
New DATA
Feature Extraction
Feature Vector
Model
Predict
Labels

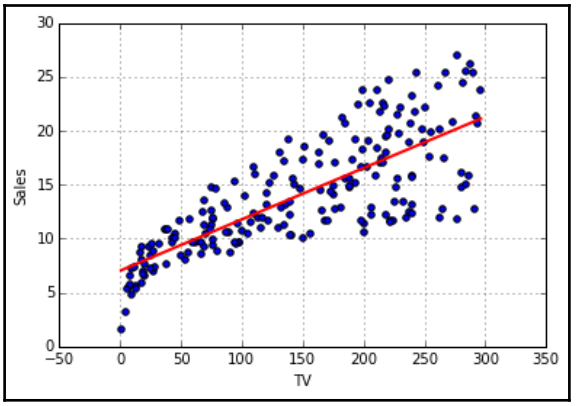

Clustering
Scattered Document
Document Clusters

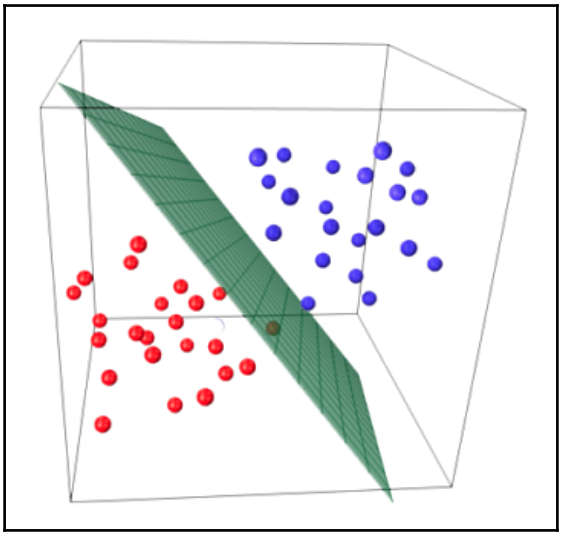
Chapter 2: Data Modeling in Action - The Titanic Example

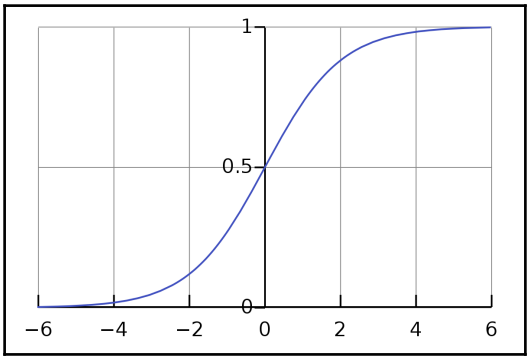




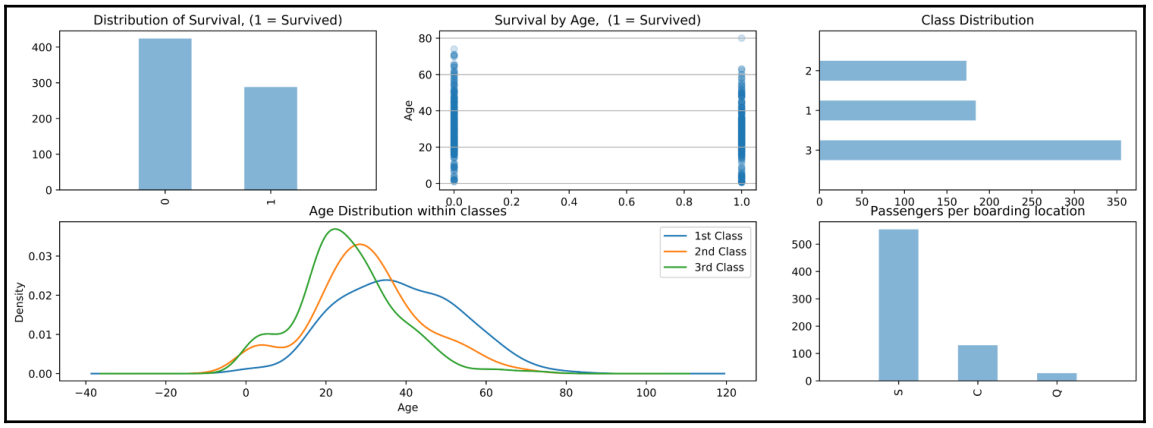


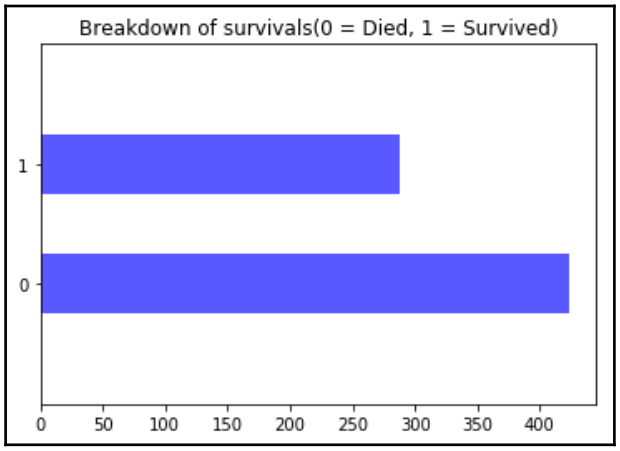

Sales
TV
30
25
20
15
10
5
0
−50
0
50
100
150
200
250
300
350

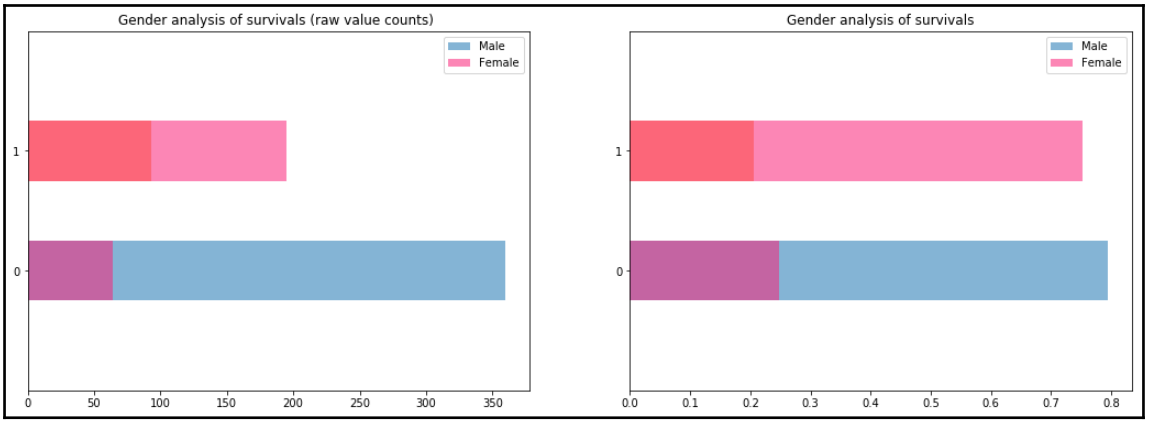



1
0.5
0
−6
−4
−2
0
2
4
6

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
500	501	0	3	Calic, Mr. Petar	male	17.0	0	0	315086	8.6625	NaN	S
501	502	0	3	Canavan, Miss. Mary	female	21.0	0	0	364846	7.7500	NaN	Q
502	503	0	3	O'Sullivan, Miss. Bridget Mary	female	NaN	0	0	330909	7.6292	NaN	Q
503	504	0	3	Laitinen, Miss. Kristina Sofia	female	37.0	0	0	4135	9.5875	NaN	S
504	505	1	1	Maioni, Miss. Roberta	female	16.0	0	0	110152	86.5000	B79	S
505	506	0	1	Penasco y Castellana, Mr. Victor de Satode	male	18.0	1	0	PC 17758	108.9000	C65	C
506	507	1	2	Quick, Mrs. Frederick Charles (Jane Richards)	female	33.0	0	2	26360	26.0000	NaN	S
507	508	1	1	Bradley, Mr. George ("George Arthur Brayton")	male	NaN	0	0	111427	26.5500	NaN	S
508	509	0	3	Olsen, Mr. Henry Margido	male	28.0	0	0	C 4001	22.5250	NaN	S
509	510	1	3	Lang, Mr. Fang	male	26.0	0	0	1601	56.4958	NaN	S



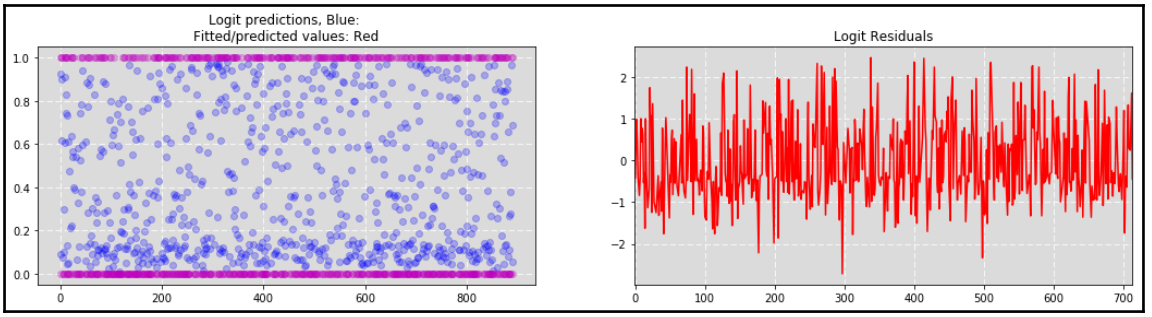


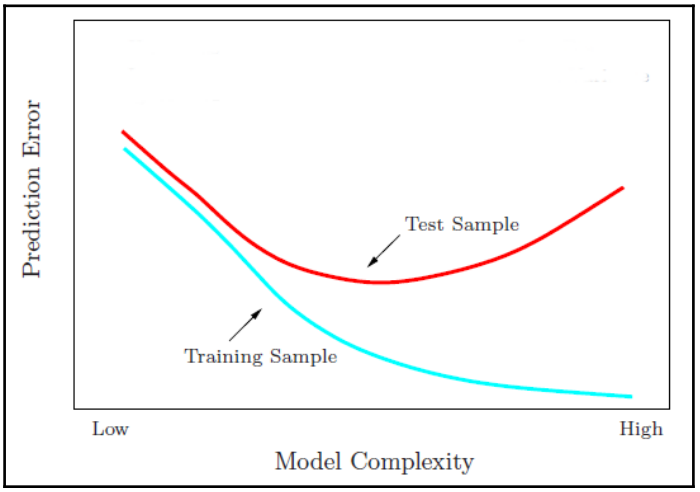


Logit Regression Results

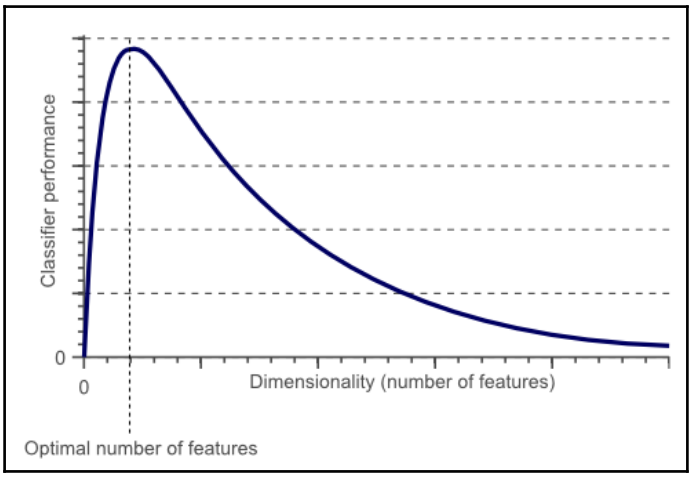
Dep. Variable:	Survived	**No. Observations:**	712
Model:	Logit	**Df Residuals:**	704
Method:	MLE	**Df Model:**	7
Date:	Sun, 20 Dec 2015	**Pseudo R-squ.:**	0.3414
Time:	11:27:33	**Log-Likelihood:**	-316.40
converged:	True	**LL-Null:**	-480.45
		LLR p-value:	5.992e-67

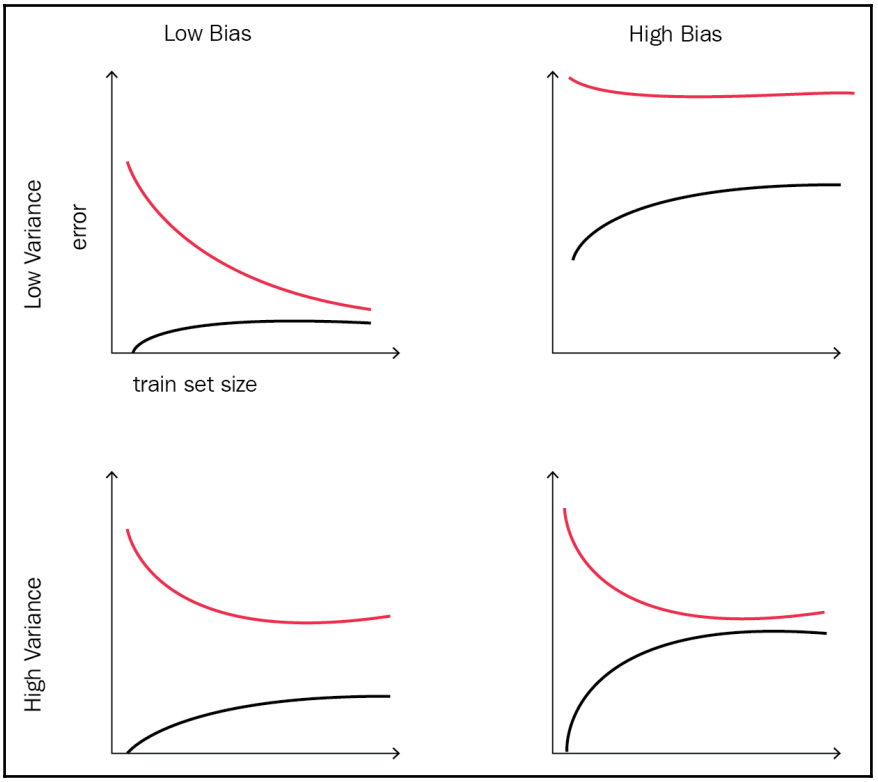
	coef	std err	z	P>|z|	[95.0% Conf. Int.]
Intercept	4.5423	0.474	9.583	0.000	3.613 5.471
C(Pclass)[T.2]	-1.2673	0.299	-4.245	0.000	-1.852 -0.682
C(Pclass)[T.3]	-2.4966	0.296	-8.422	0.000	-3.078 -1.916
C(Sex)[T.male]	-2.6239	0.218	-12.060	0.000	-3.050 -2.197
C(Embarked)[T.Q]	-0.8351	0.597	-1.398	0.162	-2.006 0.335
C(Embarked)[T.S]	-0.4254	0.271	-1.572	0.116	-0.956 0.105
Age	-0.0436	0.008	-5.264	0.000	-0.060 -0.027
SibSp	-0.3697	0.123	-3.004	0.003	-0.611 -0.129


Logit predictions, Blue:
Fitted/predicted values: Red
1.0
0.8
0.6
0.4
0.2
0.0
0
200
400
600
800
Logit Residuals
2
1
0
−1
−2
0
100
200
300
400
500
600
700


Prediction Error
Test Sample
Training Sample
Low
High
Model Complexity

Chapter 3: Feature Engineering and Model Complexity – The Titanic Example Revisited

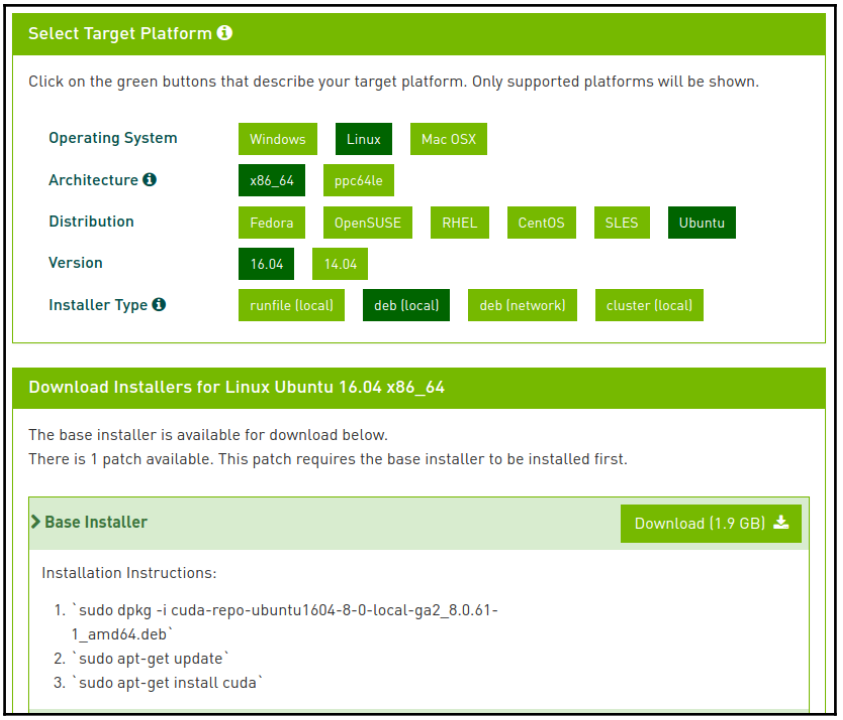



Low Bias
High Bias
Low Variance
error
train set size
High Variance

Chapter 4: Get Up and Running with TensorFlow

```
ahmed@ahmed-Inspiron-7559:~$ lspci | grep -i nvidia
02:00.0 3D controller: NVIDIA Corporation GM107M [GeForce GTX 960M] (rev a2)
ahmed@ahmed-Inspiron-7559:~$
```

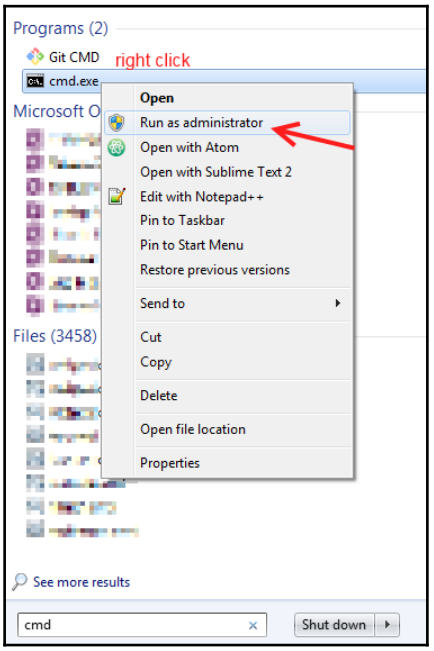
```
ahmed@ahmed-Inspiron-7559:~$ cat /proc/driver/nvidia/version
NVRM version: NVIDIA UNIX x86_64 Kernel Module  375.26  Thu Dec  8 18:36:43 PST
2016
GCC version:  gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.4)
ahmed@ahmed-Inspiron-7559:~$
```

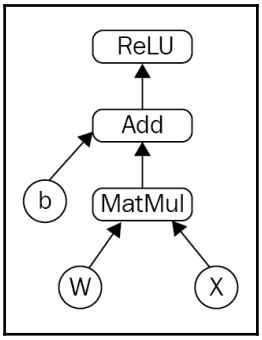


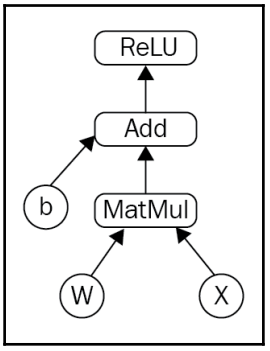
```
ahmed@ahmed-Inspiron-7559:~$ nvcc -V
nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2016 NVIDIA Corporation
Built on Tue_Jan_10_13:22:03_CST_2017
Cuda compilation tools, release 8.0, V8.0.61
ahmed@ahmed-Inspiron-7559:~$
```

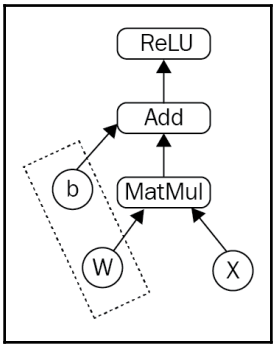
```
ahmed@ahmed-Inspiron-7559:~$ python3
Python 3.6.0 |Anaconda 4.3.0 (64-bit)| (default, Dec 23 2016, 12:22:00)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow as tf
I tensorflow/stream_executor/dso_loader.cc:128] successfully opened CUDA library libcublas.so locally
I tensorflow/stream_executor/dso_loader.cc:128] successfully opened CUDA library libcudnn.so locally
I tensorflow/stream_executor/dso_loader.cc:128] successfully opened CUDA library libcufft.so locally
I tensorflow/stream_executor/dso_loader.cc:128] successfully opened CUDA library libcuda.so.1 locally
I tensorflow/stream_executor/dso_loader.cc:128] successfully opened CUDA library libcurand.so locally
>>> a = tf.constant(5)
>>> b = tf.constant(6)
>>> sess = tf.Session()
I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:937] successful NUMA node read from SysFS had
negative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
I tensorflow/core/common_runtime/gpu/gpu_device.cc:885] Found device 0 with properties:
name: GeForce GTX 960M
major: 5 minor: 0 memoryClockRate (GHz) 1.176
pciBusID 0000:02:00.0
Total memory: 3.95GiB
Free memory: 3.43GiB
I tensorflow/core/common_runtime/gpu/gpu_device.cc:906] DMA: 0
I tensorflow/core/common_runtime/gpu/gpu_device.cc:916] 0:   Y
I tensorflow/core/common_runtime/gpu/gpu_device.cc:975] Creating TensorFlow device (/gpu:0) -> (devic
e: 0, name: GeForce GTX 960M, pci bus id: 0000:02:00.0)
>>> sess.run(a+b)
11
>>> sess.close()
>>>
```

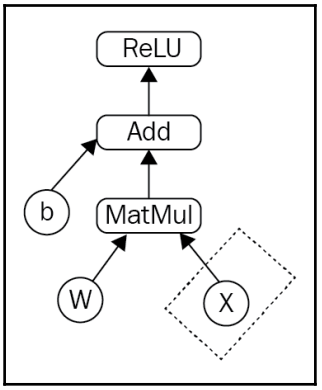
```
ahmed@ahmed-Inspiron-7559:~$ python3
Python 3.6.0 |Anaconda 4.3.0 (64-bit)| (default, Dec 23 2016, 12:22:00)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow as tf
>>> a = tf.constant(5)
>>> b = tf.constant(6)
>>> sess = tf.Session()
2017-09-11 15:39:50.792072: W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow librar
y wasn't compiled to use SSE4.1 instructions, but these are available on your machine and could speed
 up CPU computations.
2017-09-11 15:39:50.792125: W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow librar
y wasn't compiled to use SSE4.2 instructions, but these are available on your machine and could speed
 up CPU computations.
2017-09-11 15:39:50.792143: W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow librar
y wasn't compiled to use AVX instructions, but these are available on your machine and could speed up
 CPU computations.
2017-09-11 15:39:50.792158: W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow librar
y wasn't compiled to use AVX2 instructions, but these are available on your machine and could speed u
p CPU computations.
2017-09-11 15:39:50.792177: W tensorflow/core/platform/cpu_feature_guard.cc:45] The TensorFlow librar
y wasn't compiled to use FMA instructions, but these are available on your machine and could speed up
 CPU computations.
>>> sess.run(a+b)
11
>>> sess.close()
>>>
```

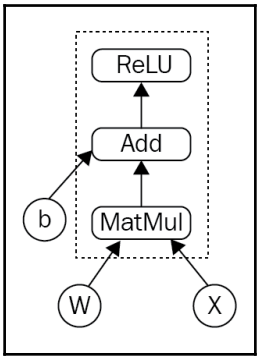



ReLU
Add
b
MatMul
W
X


ReLU
Add
b
MatMul
W
X

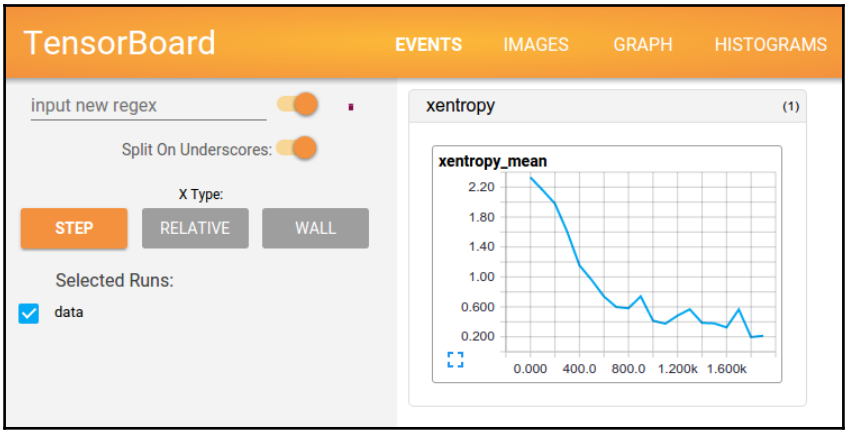

ReLU
Add
b
MatMul
W
X

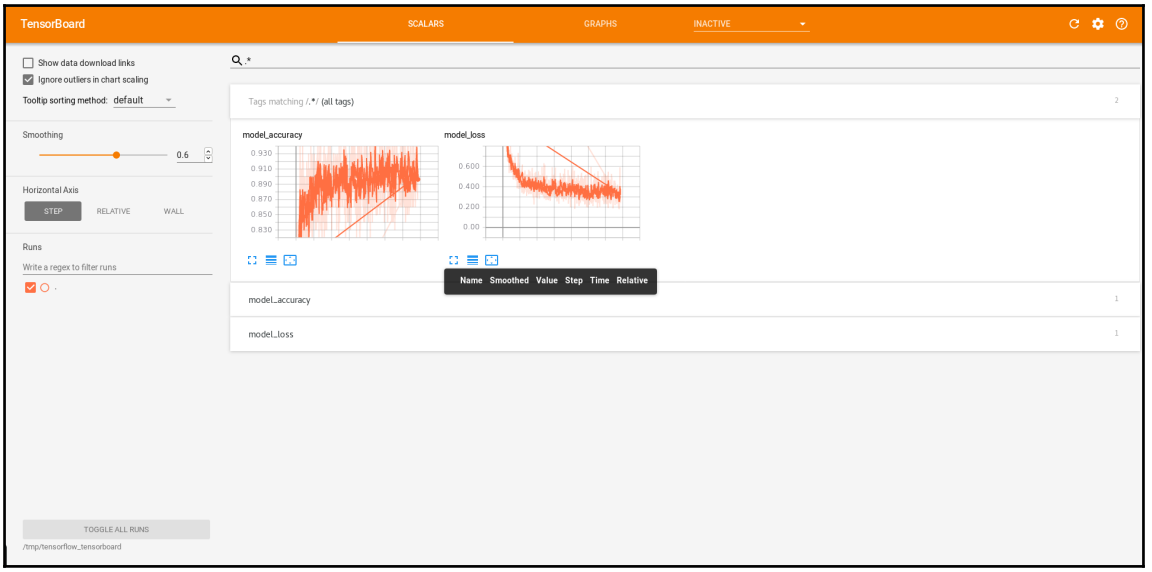

ReLU
Add
b
MatMul
W
X

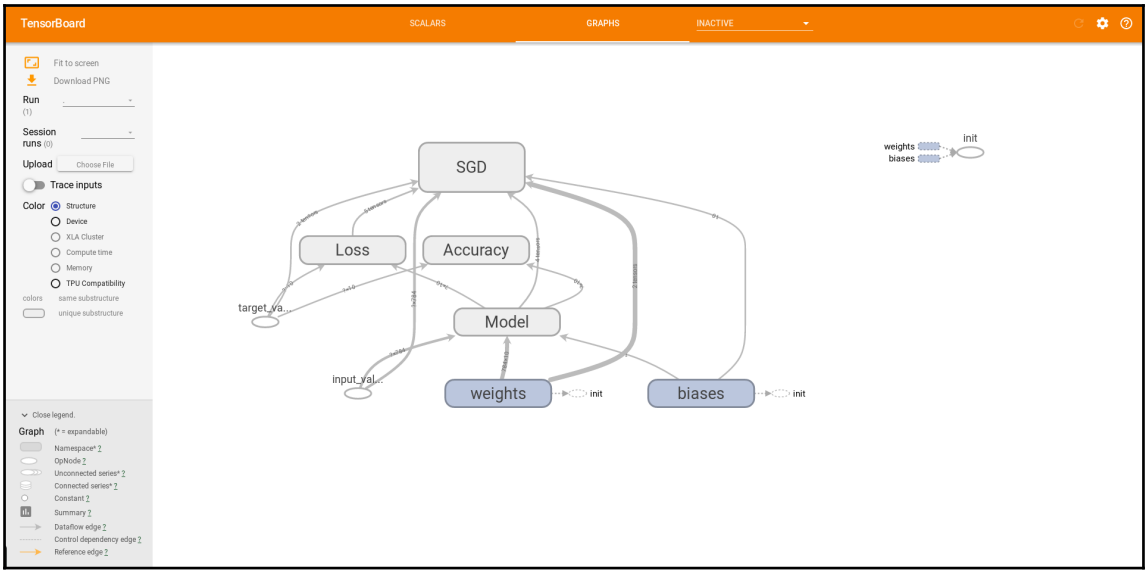

ReLU
Add
b
MatMul
W
X

```
ahmed@ahmed-Inspiron-7559:~$ python3
Python 3.6.0 |Anaconda 4.3.0 (64-bit)| (default, Dec 23 2016, 12:22:00)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> # import TensorFlow package
... import tensorflow as tf
>>> # build a TensorFlow variable b taking in initial zeros of size 100
... # ( a vector of 100 values)
... b  = tf.Variable(tf.zeros((100,)))
>>> # TensorFlow variable uniformly distributed values between -1 and 1
... # of shape 784 by 100
... W = tf.Variable(tf.random_uniform((784, 100),-1,1))
>>> # TensorFlow placeholder for our input data that doesn't take in
... # any initial values, it just takes a data type 32 bit floats as
... # well as its shape
... x = tf.placeholder(tf.float32, (100, 784))
>>> # express h as Tensorflow ReLU of the TensorFlow matrix
... #Multiplication of x and W and we add b
... h = tf.nn.relu(tf.matmul(x,W) + b )
>>> print(W)
<tf.Variable 'Variable_1:0' shape=(784, 100) dtype=float32_ref>
>>> print(b)
<tf.Variable 'Variable:0' shape=(100,) dtype=float32_ref>
>>>
```

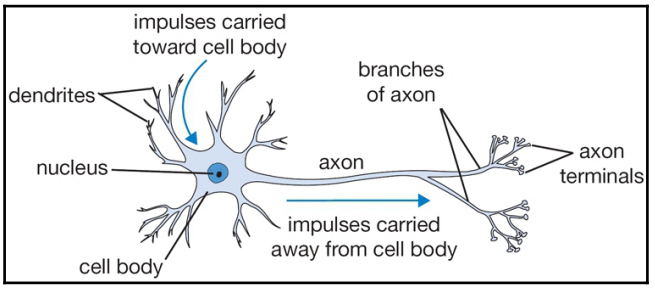
```
>>> # importing the numpy package for generating random variables for
... # our placeholder x
... import numpy as np
>>> # build a TensorFlow session object which takes a default execution
... # environment which will be most likely a CPU
... sess = tf.Session()
>>> # calling the run function of the sess object to initialize all the
... # variables.
... sess.run(tf.global_variables_initializer())
>>> # calling the run function on the node that we are interested in,
... # the h, and we feed in our second argument which is a dictionary
... # for our placeholder x with the values that we are interested in.
... sess.run(h, {x: np.random.random((100,784))})
array([[ 4.95583916,  0.13156724,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       [ 0.        ,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ],
       ...,
       [ 2.81067681,  8.28696823,  0.        , ...,  0.        ,
         0.55022001,  0.        ],
       [ 0.        ,  6.67730427,  0.        , ...,  0.        ,
         4.28411198,  3.10559845],
       [ 1.92718267,  0.        ,  0.        , ...,  0.        ,
         0.        ,  0.        ]], dtype=float32)
>>>
```

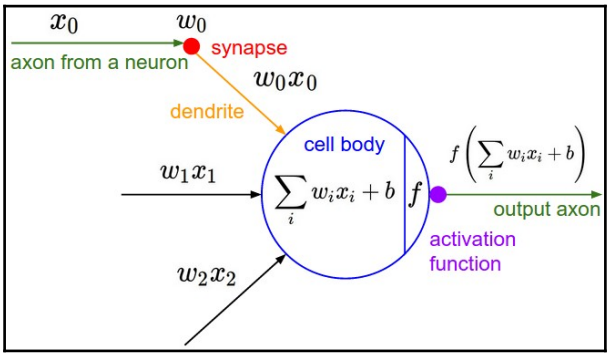

TensorBoard
EVENTS
IMAGES
GRAPH
HISTOGRAMS
input new regex
Split On Underscores:
X Type:
STEP
RELATIVE
WALL
Selected Runs:
data
xentropy
(1)
xentropy_mean
2.20
1.80
1.40
1.00
0.600
0.200
0.000
400.0
800.0
1.200k
1.600k

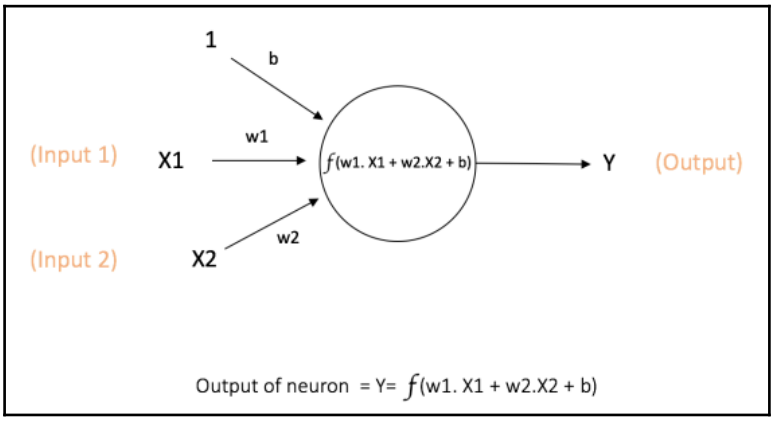

TensorBoard
SCALARS
GRAPHS
INACTIVE
Show data download links
Ignore outliers in chart scaling
Tooltip sorting method: default
Smoothing
0.6
Horizontal Axis
STEP
RELATIVE
WALL
Runs
Write a regex to filter runs
TOGGLE ALL RUNS
/tmp/tensorflow_tensorboard
.*
Tags matching /.*/ (all tags)
model_accuracy
model_loss
Name Smoothed Value Step Time Relative
model_accuracy
model_loss

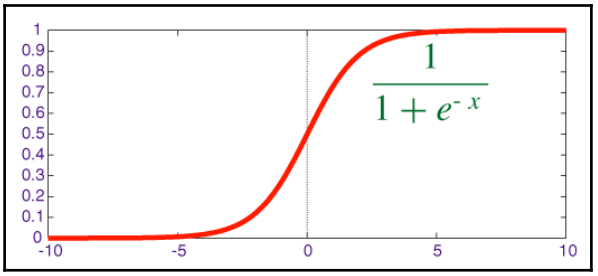

TensorBoard
SCALARS
GRAPHS
INACTIVE
Fit to screen
Download PNG
Run (1)
Session runs (0)
Upload
Choose File
Trace inputs
Color
Structure
Device
XLA Cluster
Compute time
Memory
TPU Compatibility
colors
same substructure
unique substructure
Close legend.
Graph (* = expandable)
Namespace* ?
OpNode ?
Unconnected series* ?
Connected series* ?
Constant ?
Summary ?
Dataflow edge ?
Control dependency edge ?
Reference edge ?
SGD
Loss
Accuracy
Model
target_va...
input_val...
weights
biases
init
init
weights
biases
init

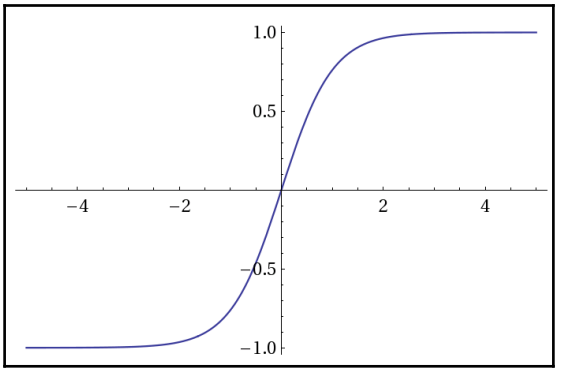
Chapter 5: TensorFlow in Action - Some Basic Examples

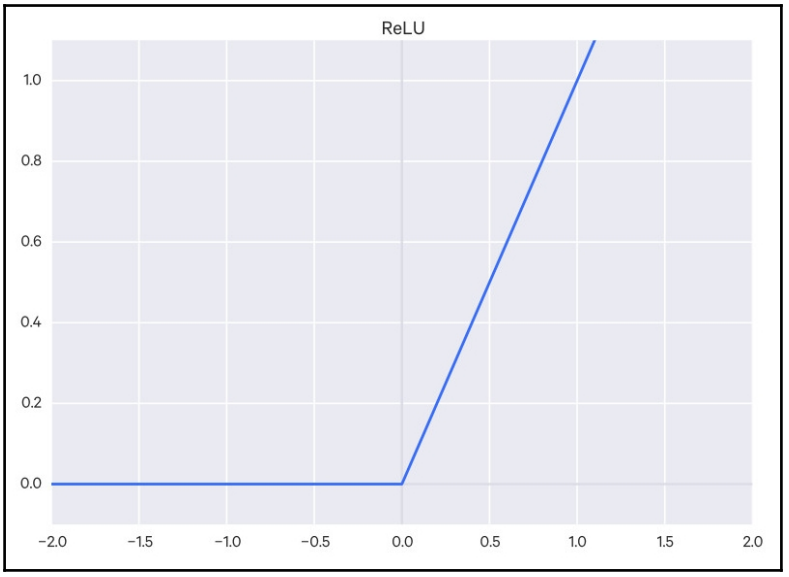


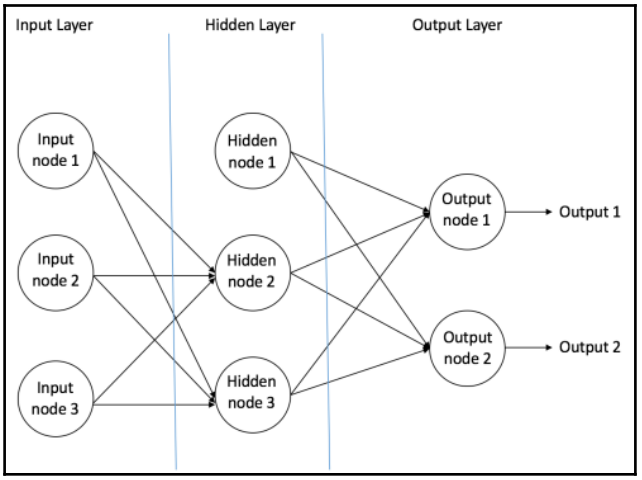


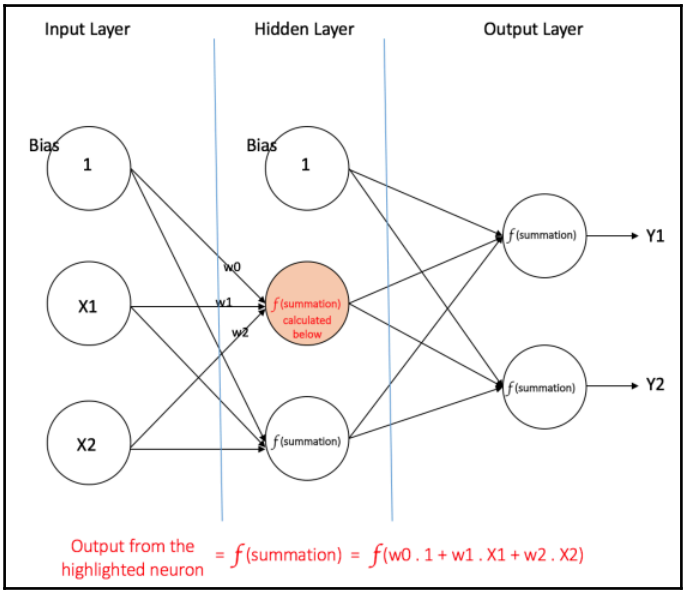


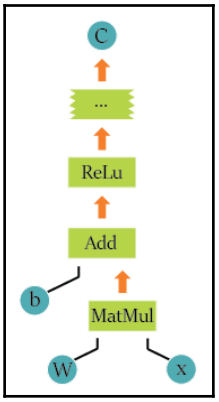

1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-10
-5
0
5
10
$\frac{1}{1+e^{-x}}$

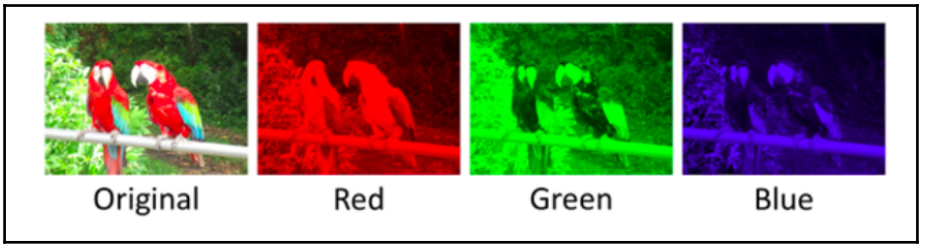

1.0
0.5
−4
−2
2
4
−0.5
−1.0

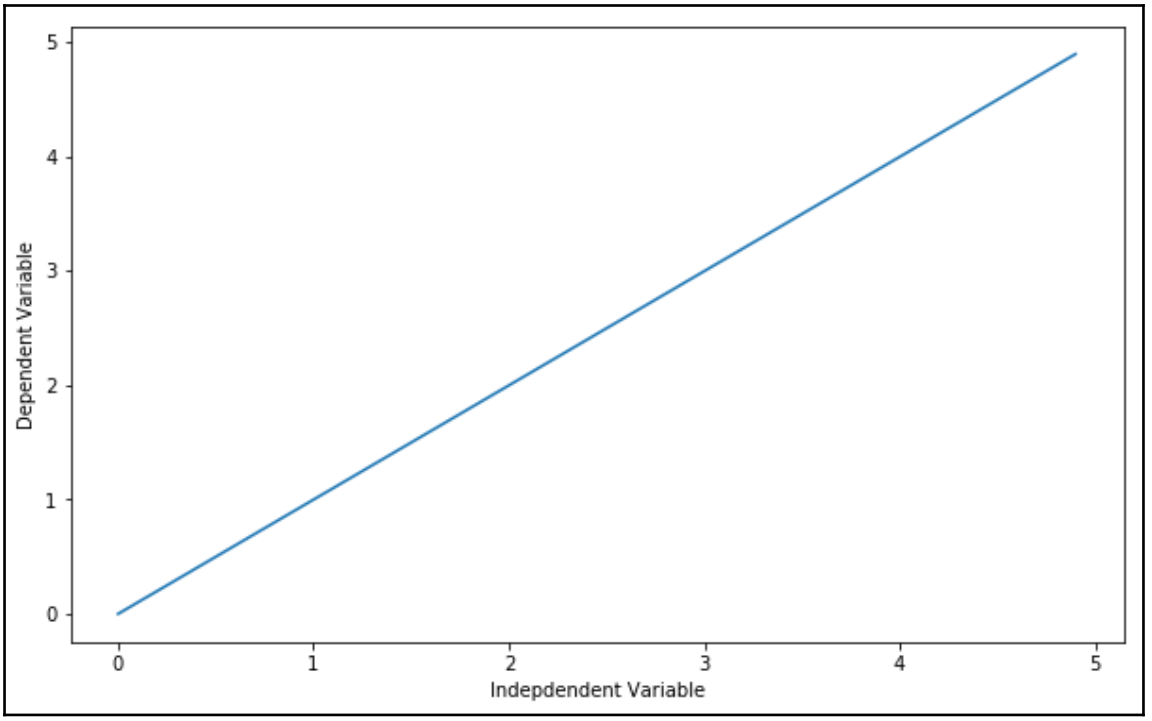

ReLU
1.0
0.8
0.6
0.4
0.2
0.0
−2.0
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0

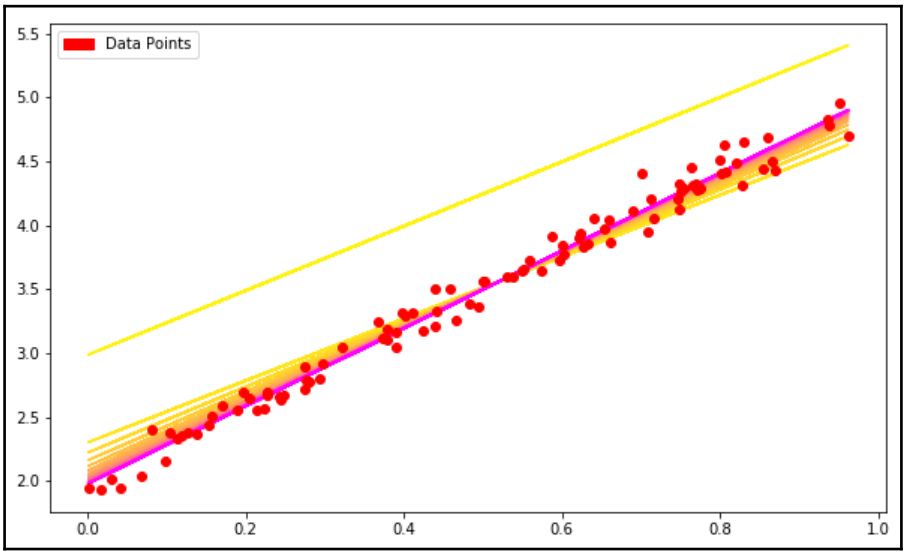

Input Layer
Hidden Layer
Output Layer
Input node 1
Input node 2
Input node 3
Hidden node 1
Hidden node 2
Hidden node 3
Output node 1
Output node 2
Output 1
Output 2

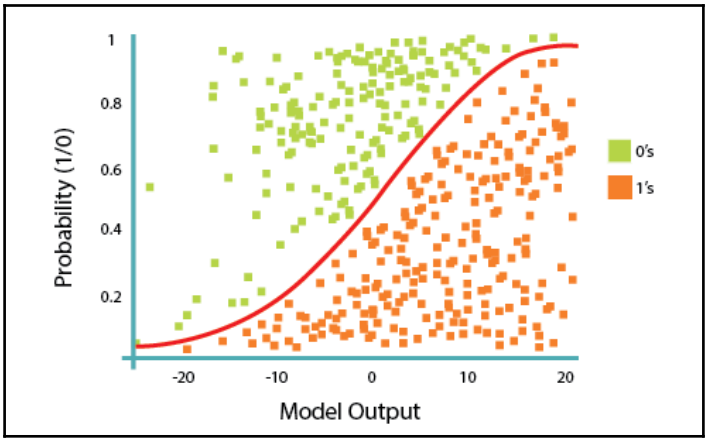

Input Layer
Hidden Layer
Output Layer
Bias
1
X1
X2
Bias
1
w0
w1
w2
f (summation) calculated below
f (summation)
f (summation)
f (summation)
Y1
Y2
Output from the highlighted neuron
= f (summation) = f (w0 . 1 + w1 . X1 + w2 . X2)


C
...
ReLu
Add
b
MatMul
W
x

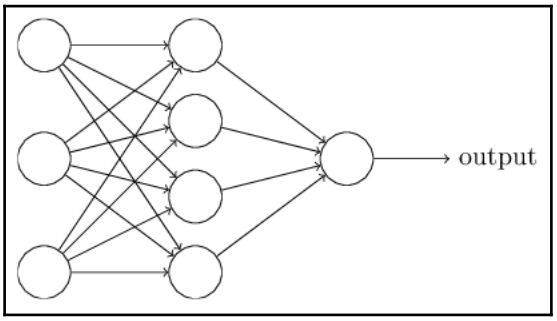

Original
Red
Green
Blue

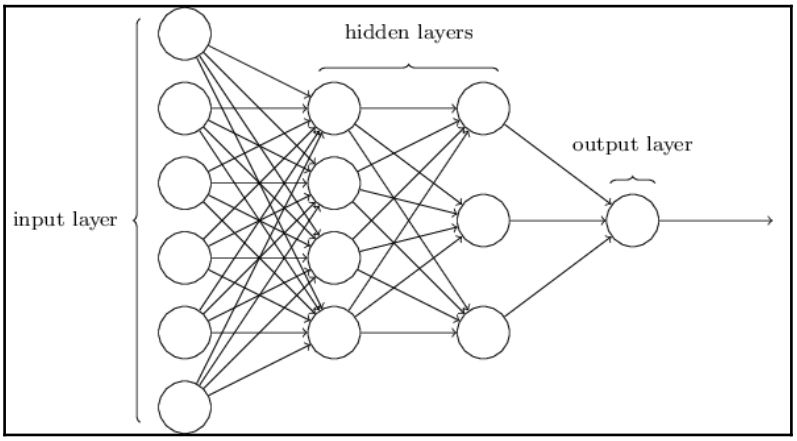

Dependent Variable
Indepdendent Variable
0
1
2
3
4
5

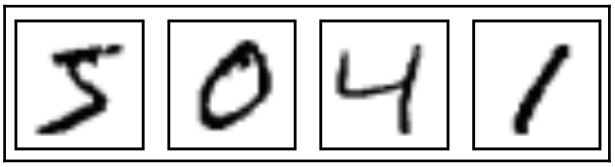

Data Points
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0

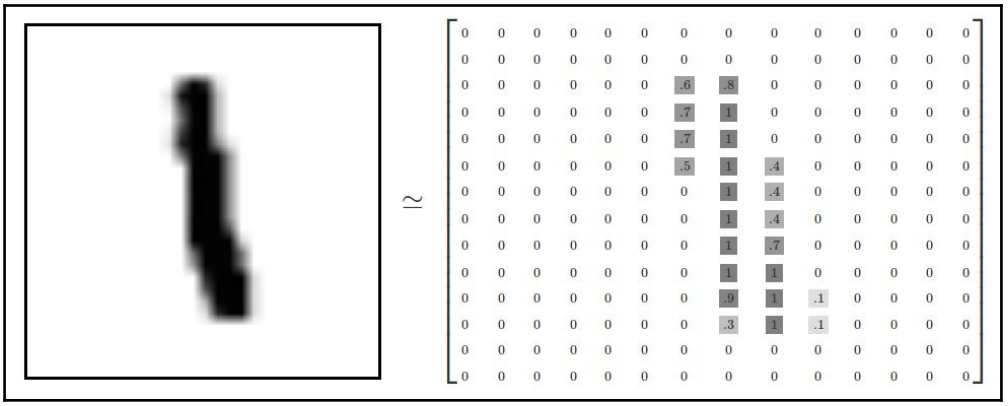

Probability (1/0)
1
0.8
0.6
0.4
0.2
-20
-10
0
10
20
Model Output
0's
1's

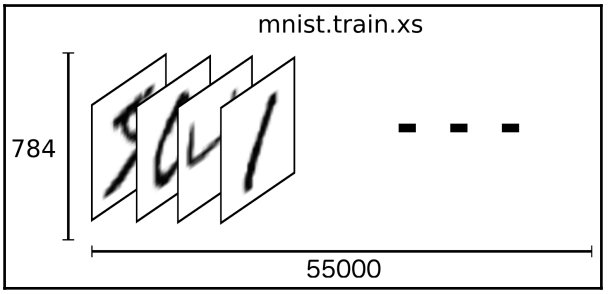
Chapter 6: Deep Feed-forward Neural Networks - Implementing Digit Classification

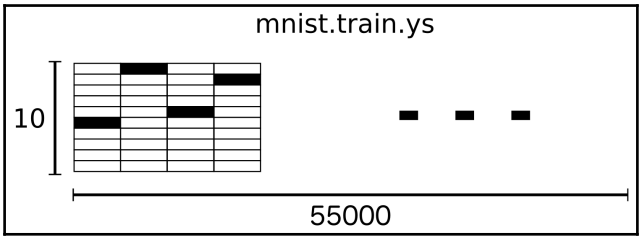


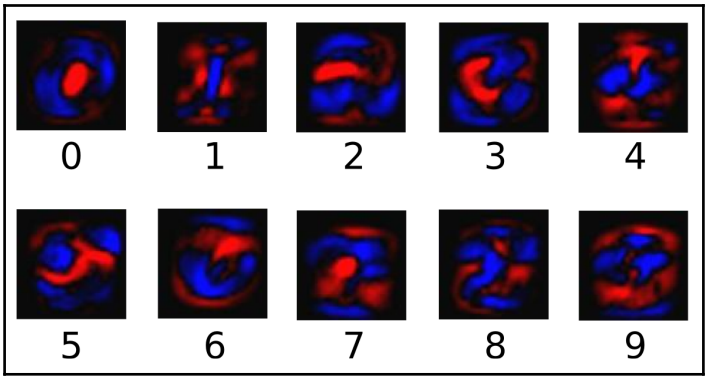


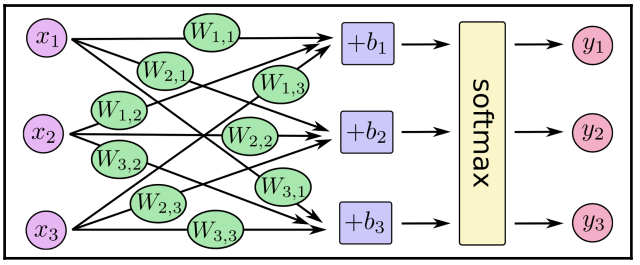



≃
0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 .6 .8 0 0 0 0 0 0
0 0 0 0 0 0 .7 1 0 0 0 0 0 0
0 0 0 0 0 0 .7 1 0 0 0 0 0 0
0 0 0 0 0 0 .5 1 .4 0 0 0 0 0
0 0 0 0 0 0 0 1 .4 0 0 0 0 0
0 0 0 0 0 0 0 1 .4 0 0 0 0 0
0 0 0 0 0 0 0 1 .7 0 0 0 0 0
0 0 0 0 0 0 0 1 1 0 0 0 0 0
0 0 0 0 0 0 0 .9 1 .1 0 0 0 0
0 0 0 0 0 0 0 .3 1 .1 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0


mnist.train.xs
784
55000


mnist.train.ys
10
55000

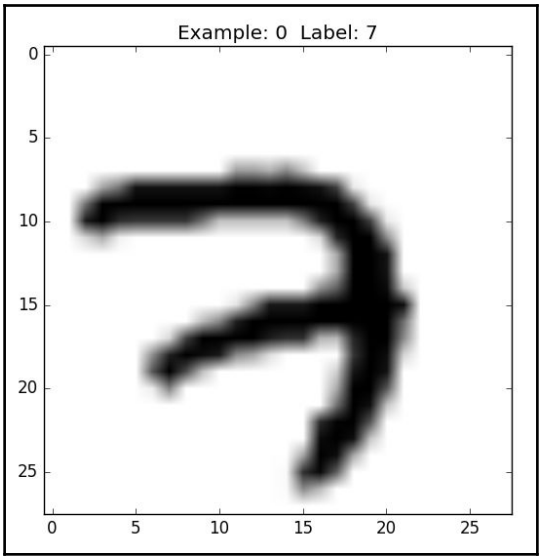


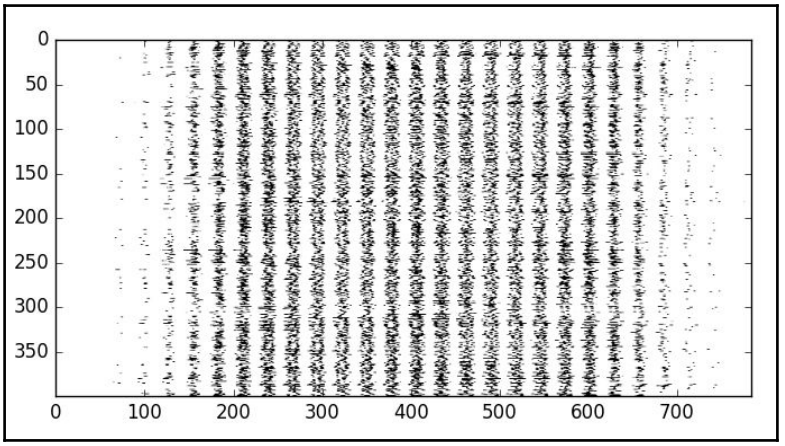


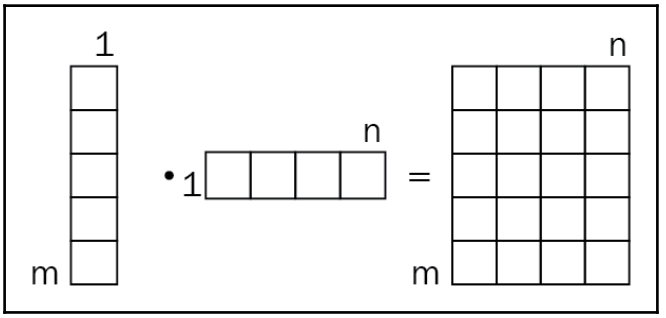
$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax}\left(\begin{bmatrix} W_{1,1}x_1 + W_{1,2}x_2 + W_{1,3}x_3 + b_1 \\ W_{2,1}x_1 + W_{2,2}x_2 + W_{2,3}x_3 + b_2 \\ W_{3,1}x_1 + W_{3,2}x_2 + W_{3,3}x_3 + b_3 \end{bmatrix}\right)$$

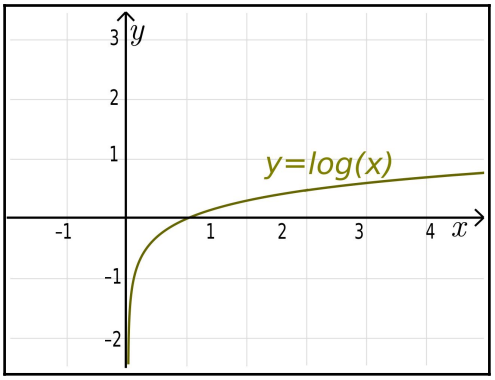
$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax}\left(\begin{bmatrix} W_{1,1} & W_{1,2} & W_{1,3} \\ W_{2,1} & W_{2,2} & W_{2,3} \\ W_{3,1} & W_{3,2} & W_{3,3} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}\right)$$

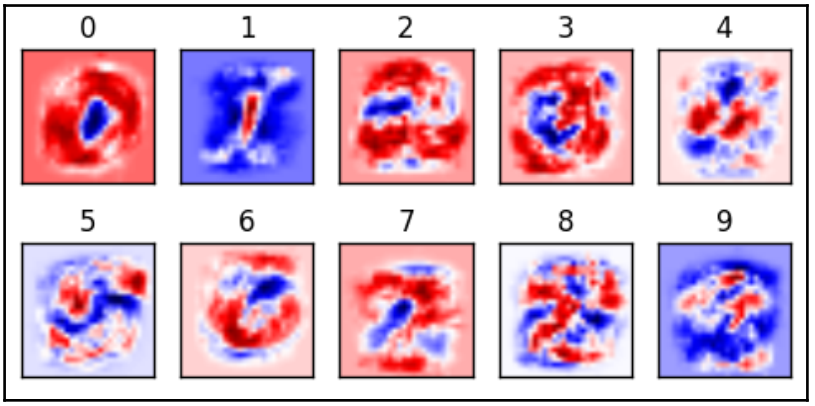
Label	0	1	2	3	4	5	6	7	8	9
Array	[0 ,	0,	0,	0,	0,	0,	0,	1,	0,	0]

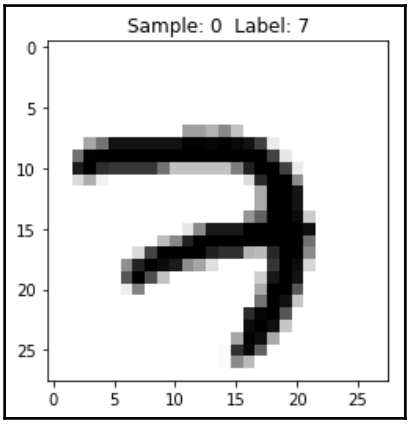

Example: 0 Label: 7
0
5
10
15
20
25
0
5
10
15
20
25

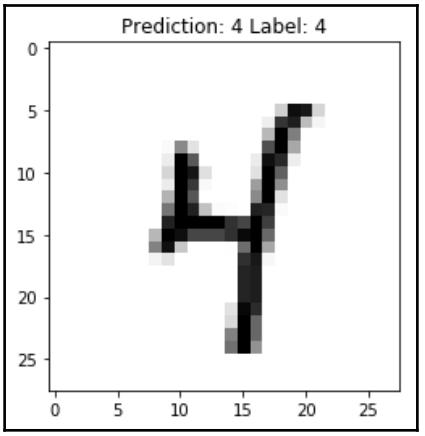

0
50
100
150
200
250
300
350
0
100
200
300
400
500
600
700


1
m
n
1
=
n
m

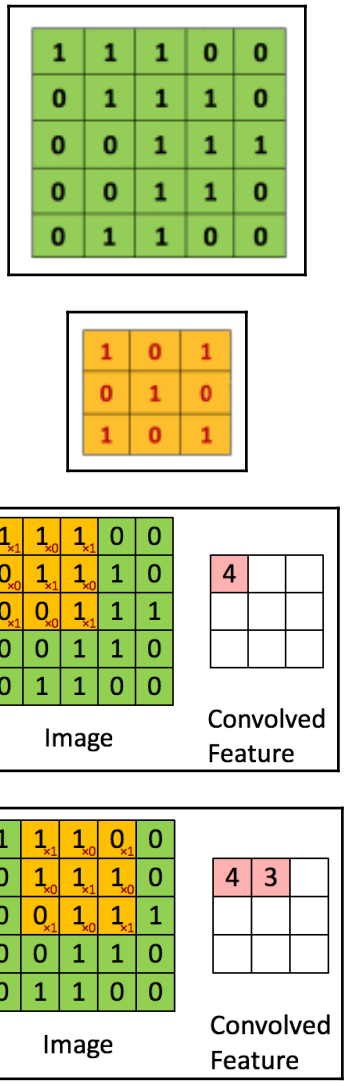

y
x
y=log(x)
3
2
1
-1
-2
-1
1
2
3
4

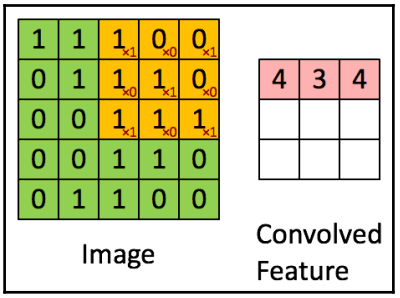

0
1
2
3
4
5
6
7
8
9

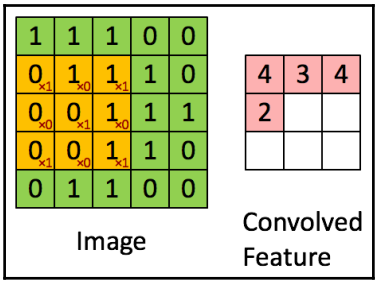

Sample: 0 Label: 7
0
5
10
15
20
25

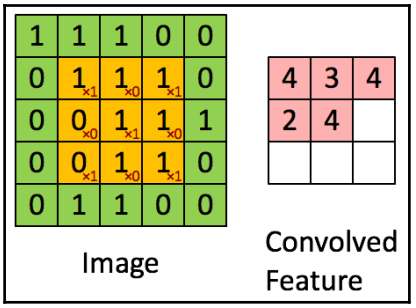

Prediction: 4 Label: 4
0
5
10
15
20
25
0
5
10
15
20
25

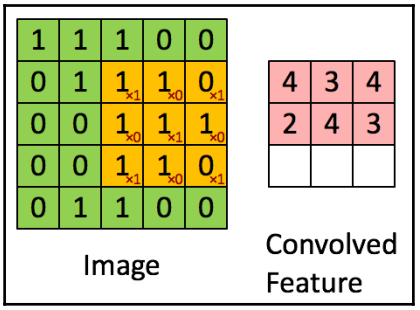
Chapter 7: Introduction to Convolutional Neural Networks

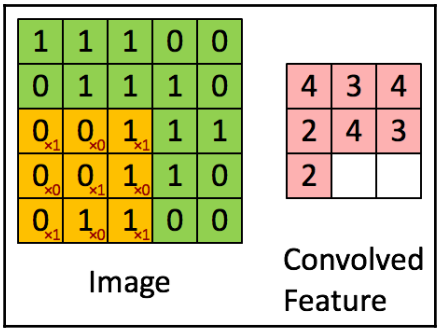


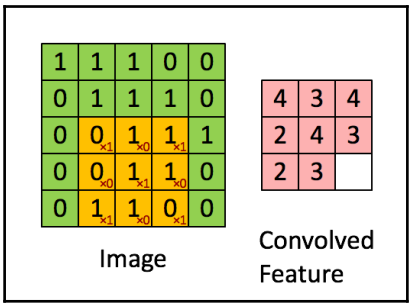

1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 0
0 1 1 0 0
Image
4 3 4
Convolved Feature

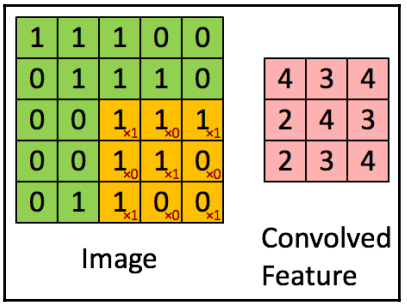

1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 0
0 1 1 0 0
Image
4 3 4
2
Convolved Feature

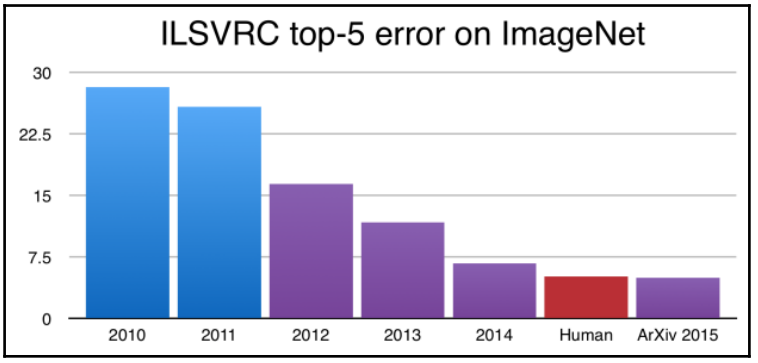

1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 0
0 1 1 0 0
Image
4 3 4
2 4
Convolved Feature

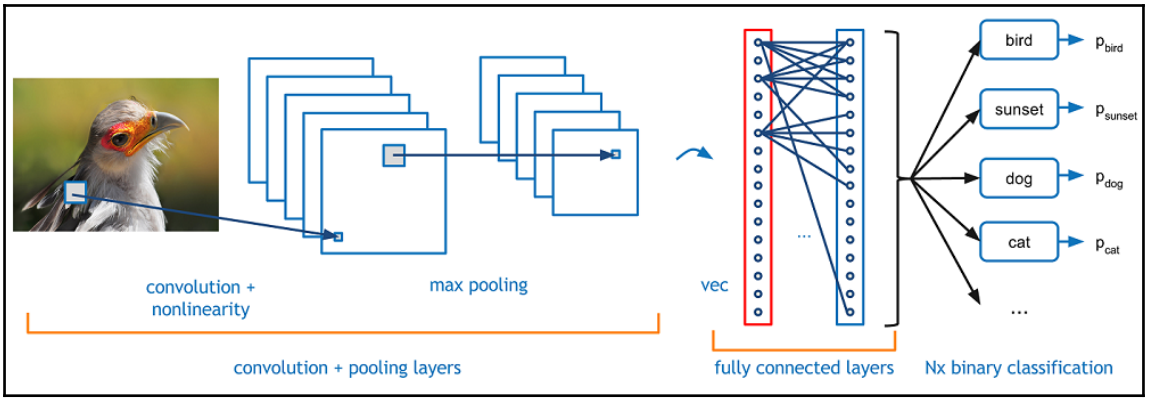

1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 0
0 1 1 0 0
Image
4 3 4
2 4 3
Convolved Feature

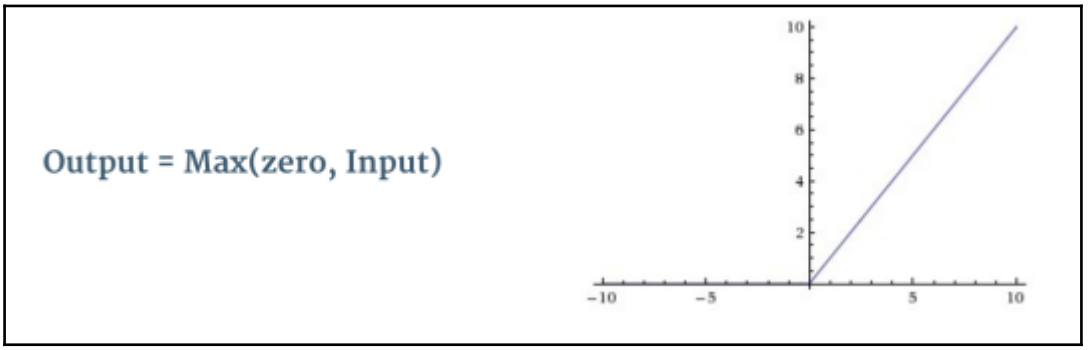

1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 0
0 1 1 0 0
Image
4 3 4
2 4 3
2
Convolved Feature

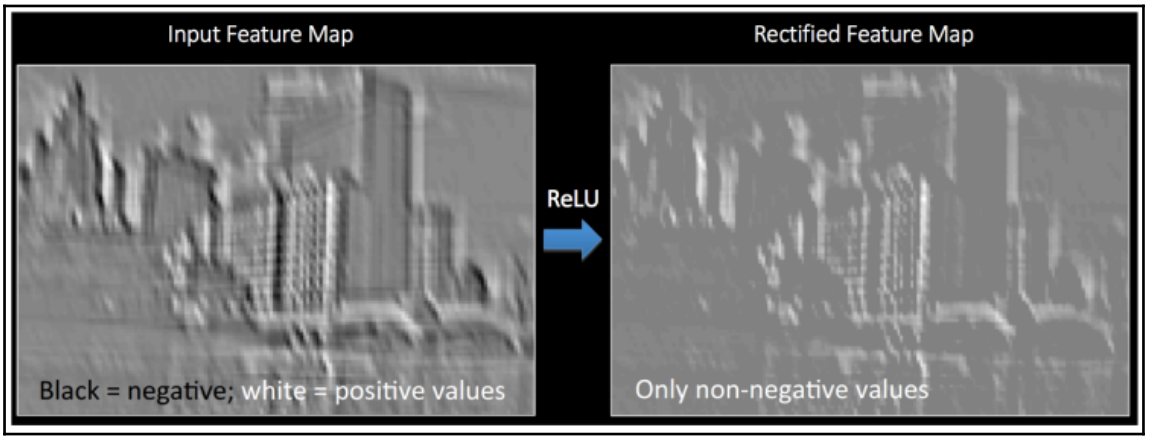

1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 0
0 1 1 0 0
Image
4 3 4
2 4 3
2 3
Convolved Feature

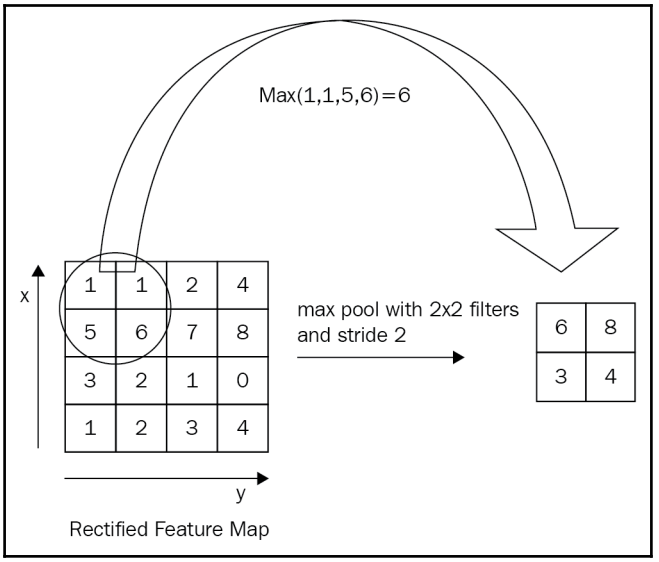

1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 0
0 1 1 0 0
Image
4 3 4
2 4 3
2 3 4
Convolved Feature

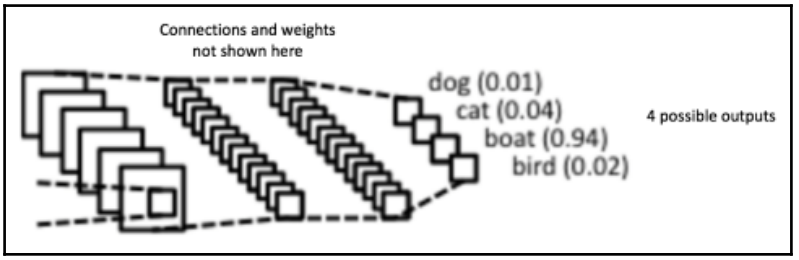

ILSVRC top-5 error on ImageNet
30
22.5
15
7.5
0
2010
2011
2012
2013
2014
Human
ArXiv 2015

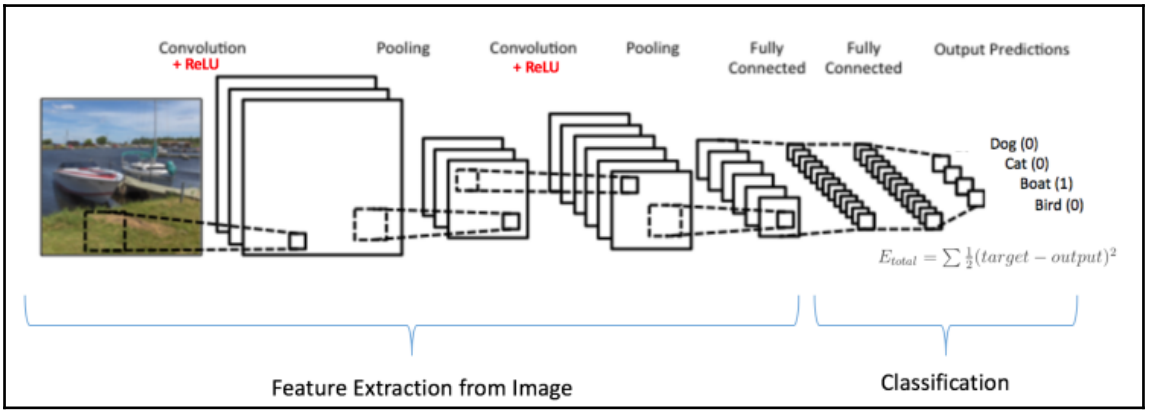

convolution + nonlinearity
max pooling
convolution + pooling layers
vec
fully connected layers
bird
sunset
dog
cat
...
Nx binary classification

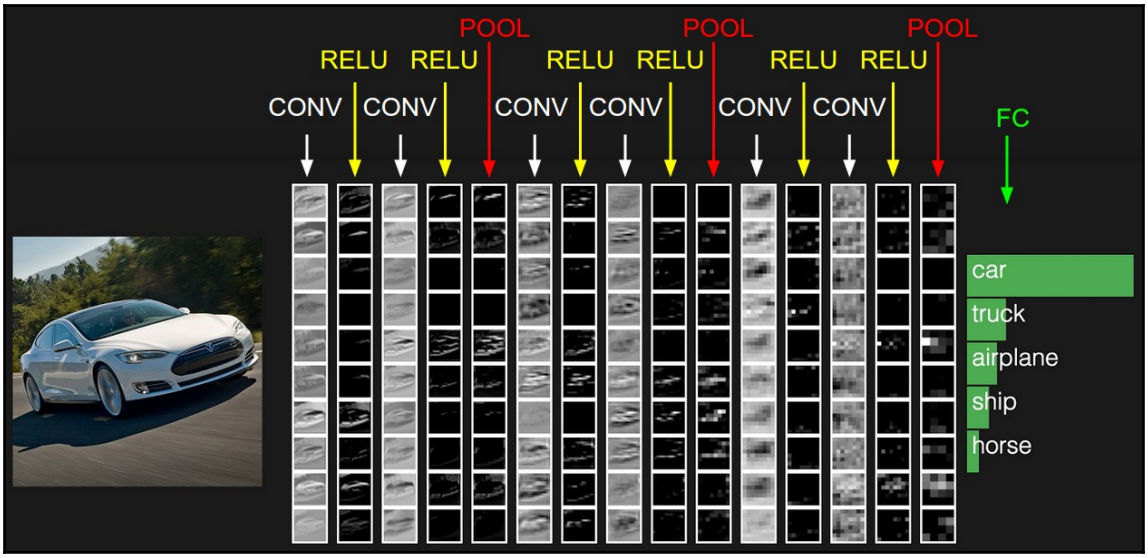

Output = Max(zero, Input)
10
8
6
4
2
-10
-5
5
10

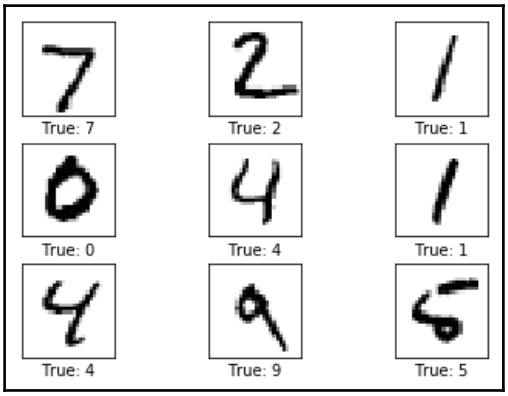

Input Feature Map
Rectified Feature Map
ReLU
Black = negative; white = positive values
Only non-negative values

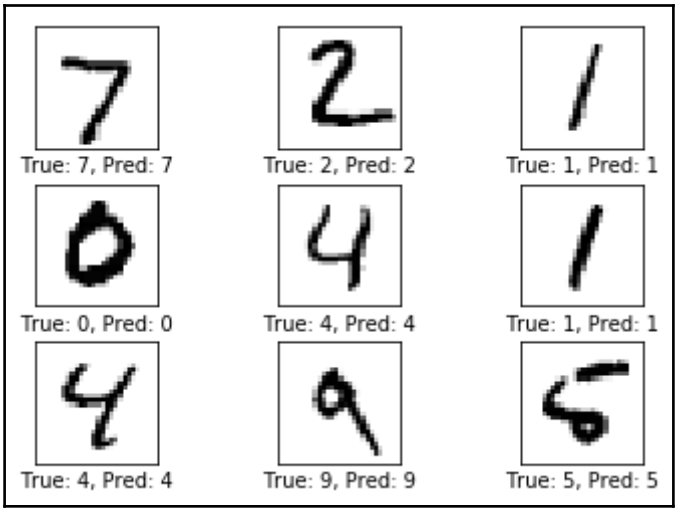

Max(1,1,5,6)=6
x
y
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
max pool with 2x2 filters
and stride 2
6 8
3 4
Rectified Feature Map

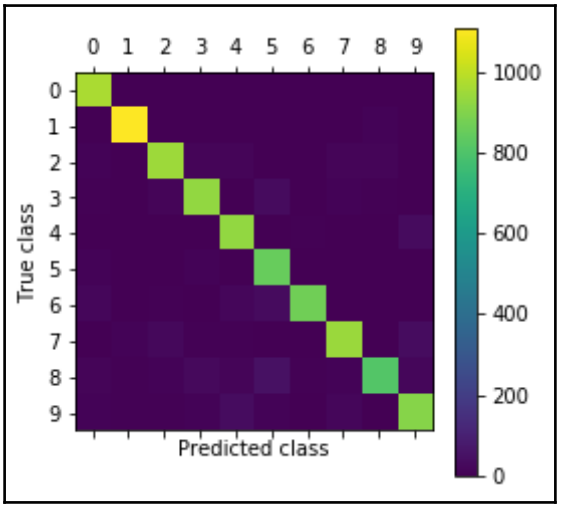

Connections and weights
not shown here
dog (0.01)
cat (0.04)
boat (0.94)
bird (0.02)
4 possible outputs


Convolution
+ ReLU
Pooling
Convolution
+ ReLU
Pooling
Fully
Connected
Fully
Connected
Output Predictions
Dog (0)
Cat (0)
Boat (1)
Bird (0)
$E_{total} = \sum \frac{1}{2}(target - output)^2$
Feature Extraction from Image
Classification

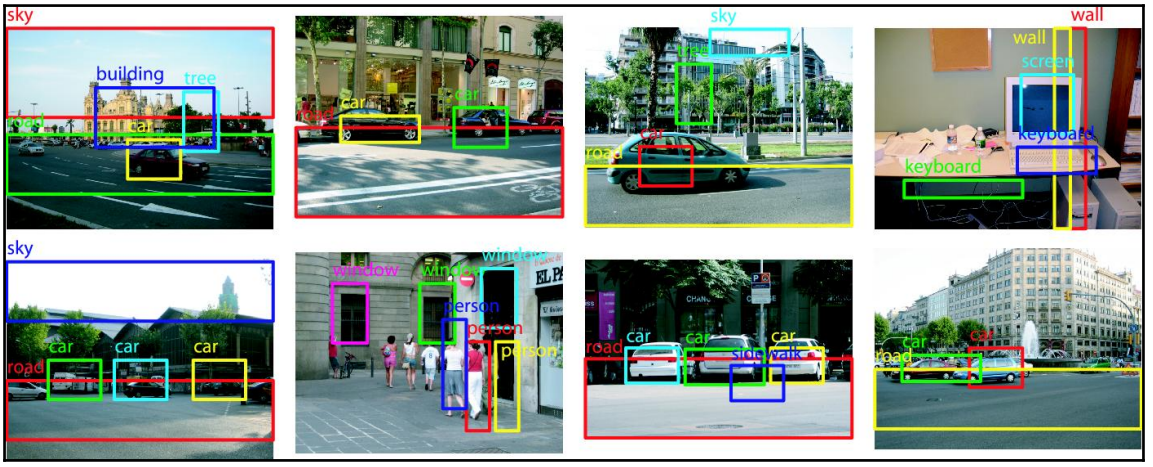

POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse

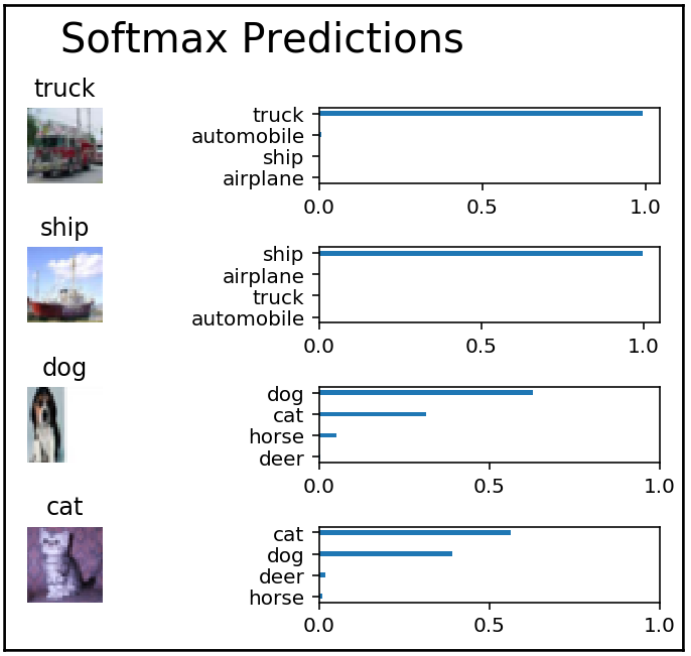

True: 7
True: 2
True: 1
True: 0
True: 4
True: 1
True: 4
True: 9
True: 5

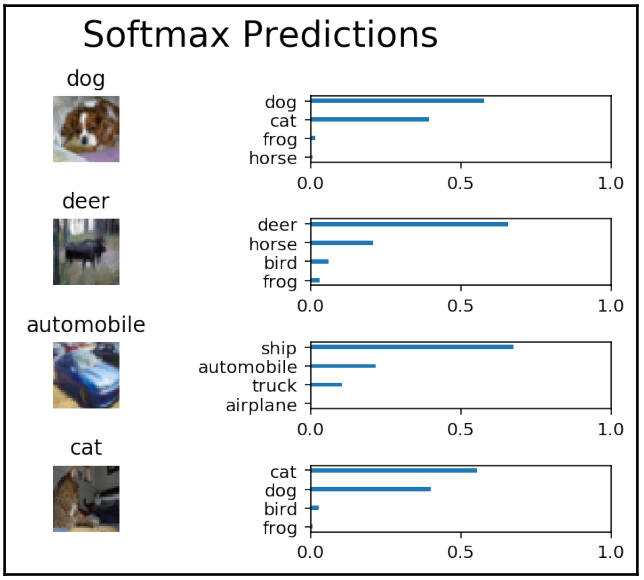

True: 7, Pred: 7
True: 2, Pred: 2
True: 1, Pred: 1
True: 0, Pred: 0
True: 4, Pred: 4
True: 1, Pred: 1
True: 4, Pred: 4
True: 9, Pred: 9
True: 5, Pred: 5


0 1 2 3 4 5 6 7 8 9
True class
Predicted class
1000
800
600
400
200
0

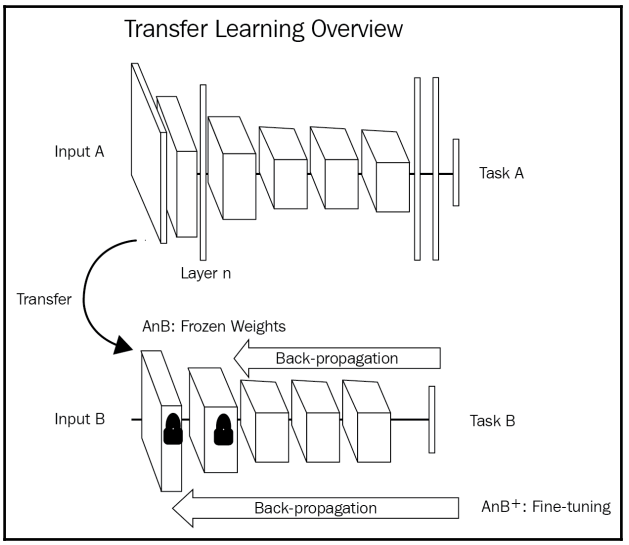
Chapter 8: Object Detection – CIFAR-10 Example

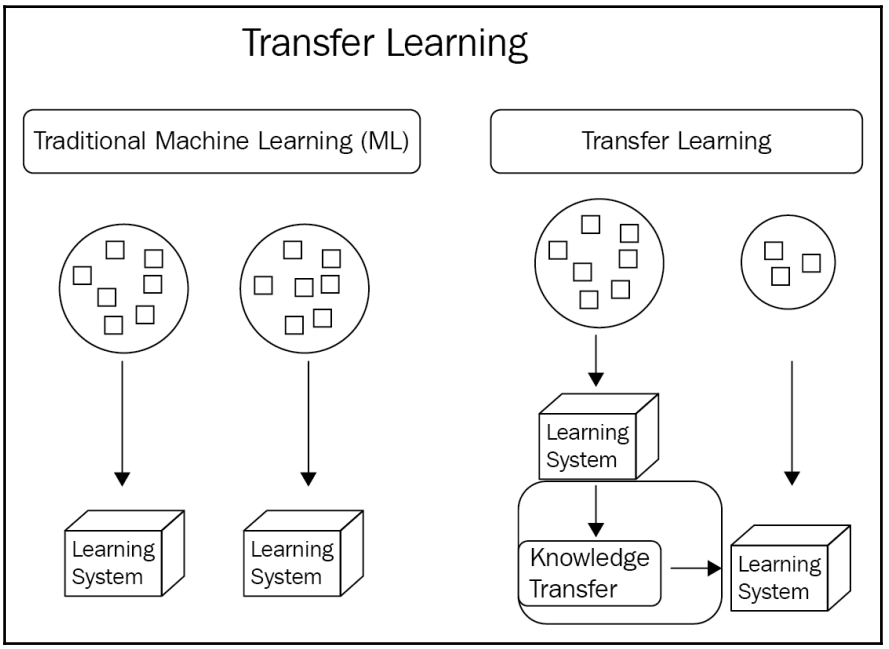


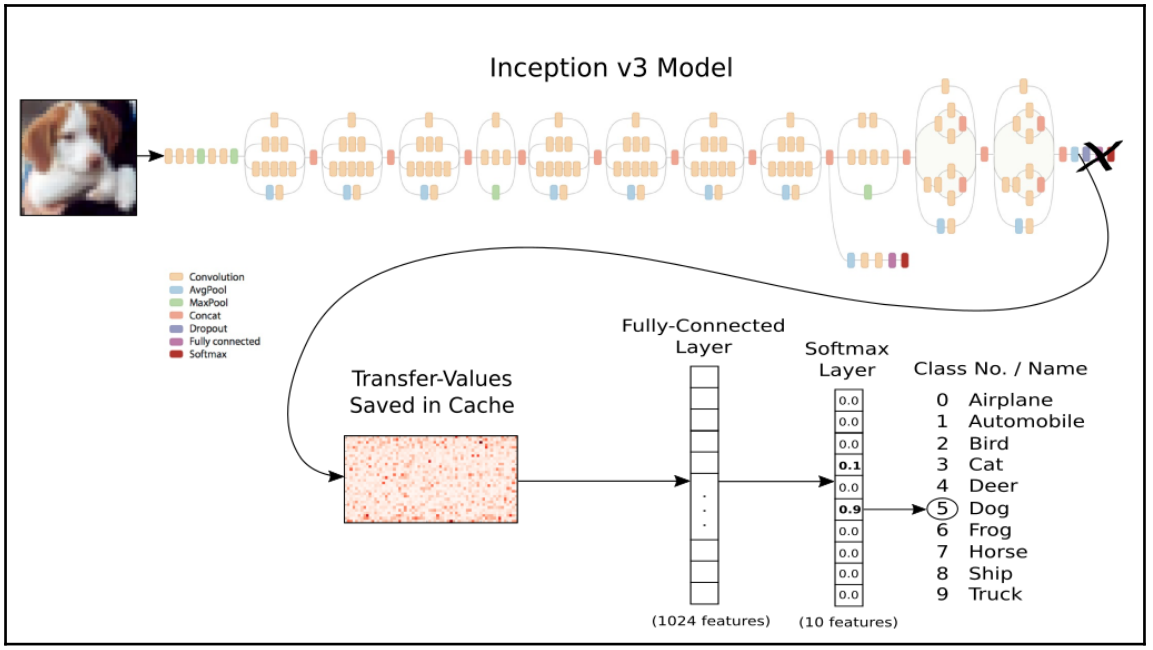

Softmax Predictions
truck
truck
automobile
ship
airplane
0.0
0.5
1.0
ship
ship
airplane
truck
automobile
0.0
0.5
1.0
dog
dog
cat
horse
deer
0.0
0.5
1.0
cat
cat
dog
deer
horse
0.0
0.5
1.0

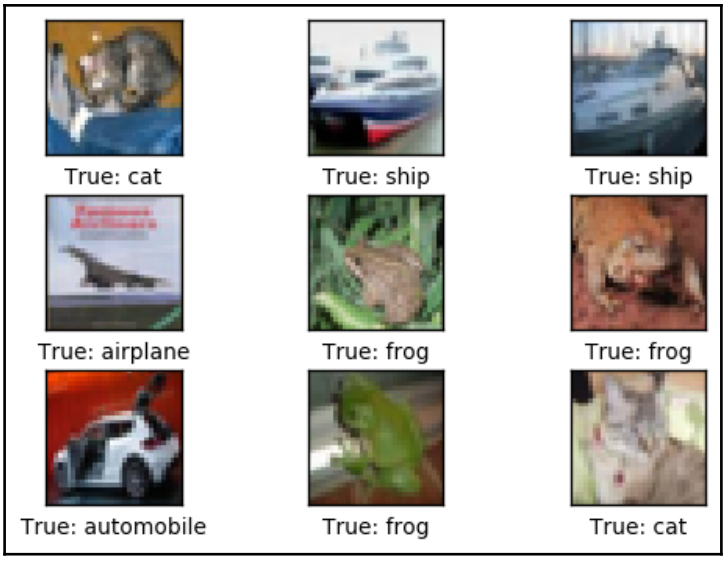

Softmax Predictions
dog
dog
cat
frog
horse
0.0
0.5
1.0
deer
deer
horse
bird
frog
0.0
0.5
1.0
automobile
ship
automobile
truck
airplane
0.0
0.5
1.0
cat
cat
dog
bird
frog
0.0
0.5
1.0

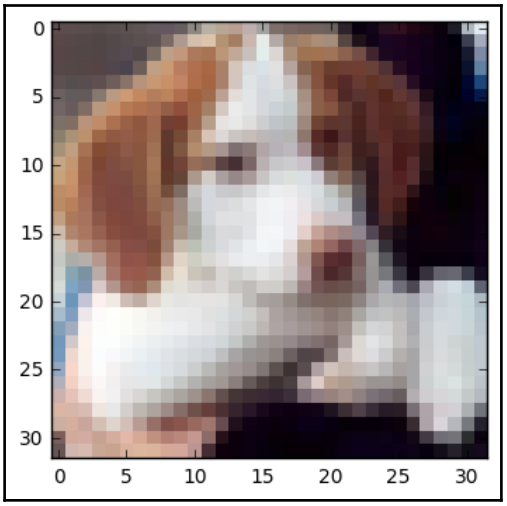
Chapter 9: Object Detection – Transfer Learning with CNNs

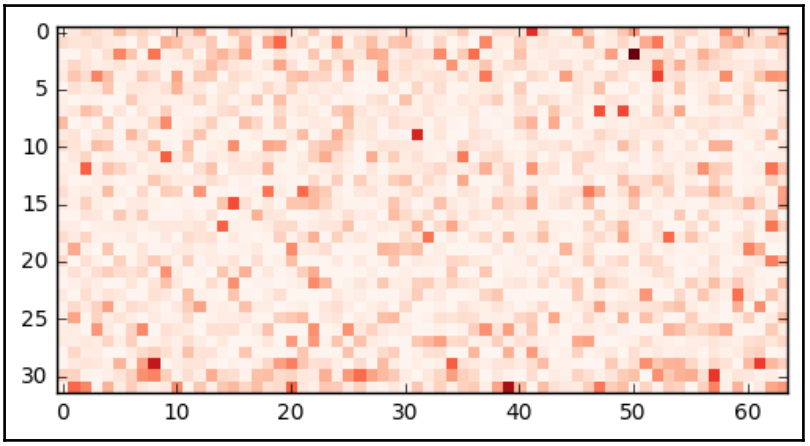


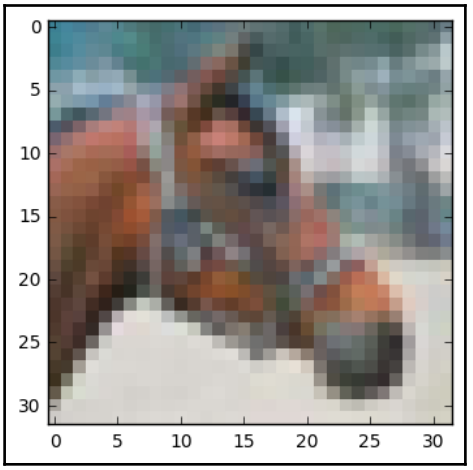


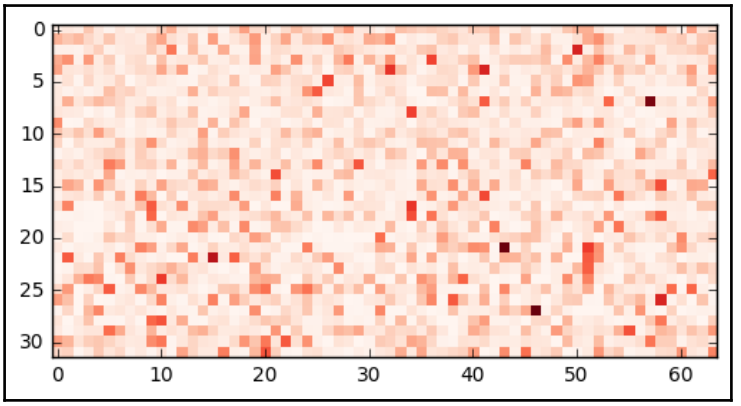

Inception v3 Model
Convolution
AvgPool
MaxPool
Concat
Dropout
Fully connected
Softmax
Transfer-Values
Saved in Cache
Fully-Connected
Layer
(1024 features)
Softmax
Layer
(10 features)
Class No. / Name
0 Airplane
1 Automobile
2 Bird
3 Cat
4 Deer
5 Dog
6 Frog
7 Horse
8 Ship
9 Truck

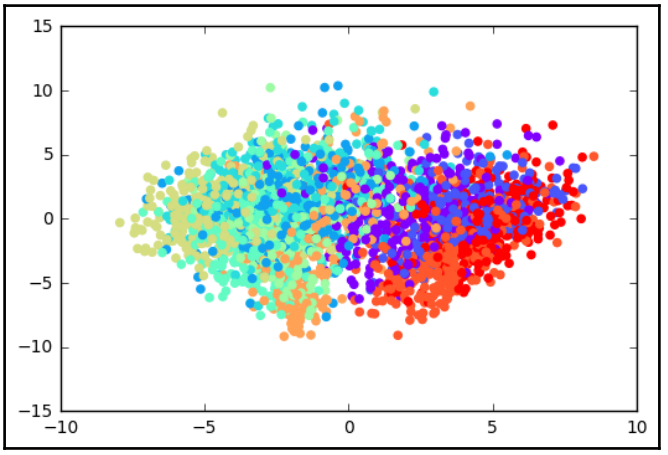

True: cat
True: ship
True: ship
True: airplane
True: frog
True: frog
True: automobile
True: frog
True: cat

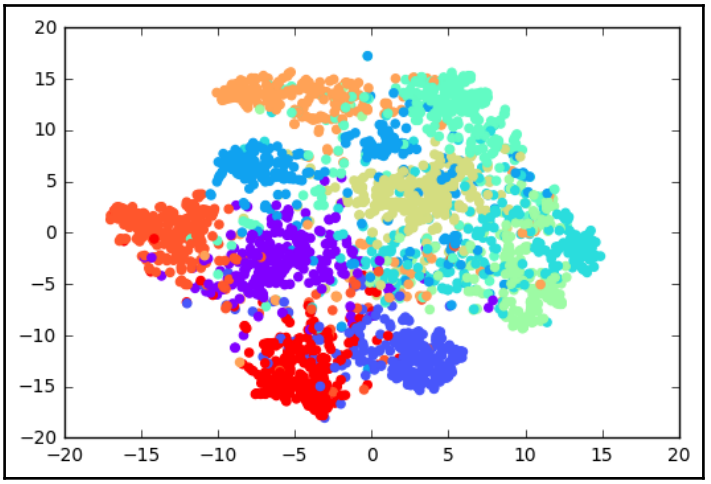

0
5
10
15
20
25
30
0
5
10
15
20
25
30

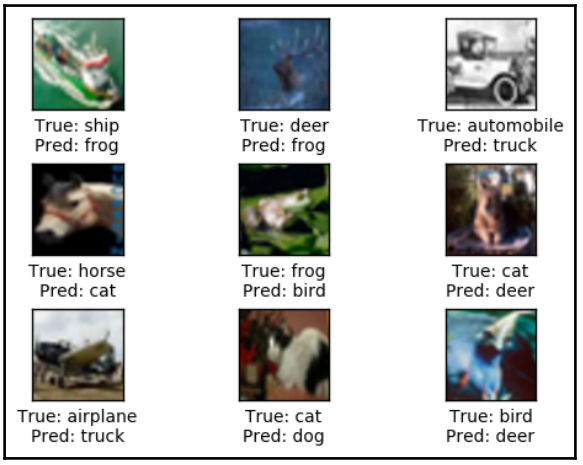

0
5
10
15
20
25
30
0
10
20
30
40
50
60


0
5
10
15
20
25
30
0
5
10
15
20
25
30

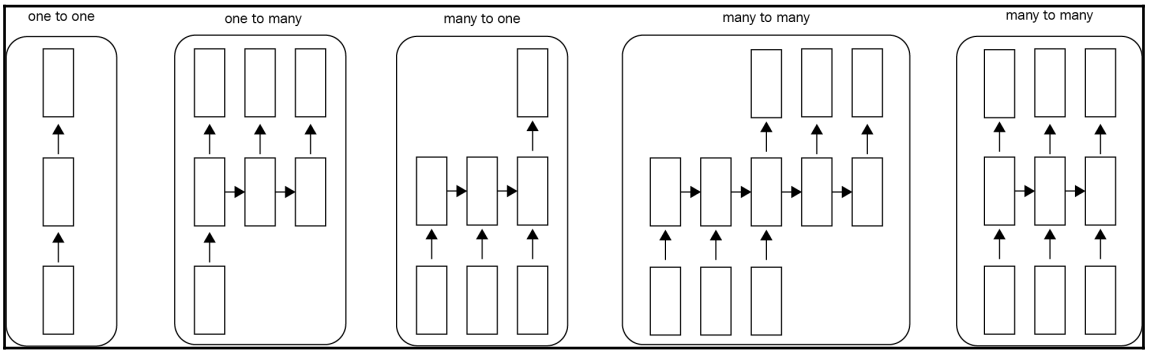

0
5
10
15
20
25
30
0
10
20
30
40
50
60

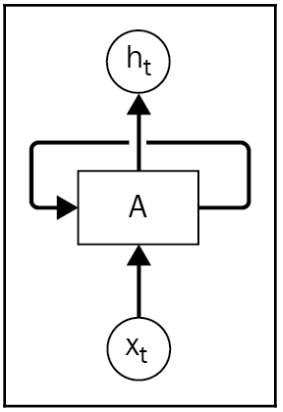

15
10
5
0
−5
−10
−15
−10
−5
0
5
10

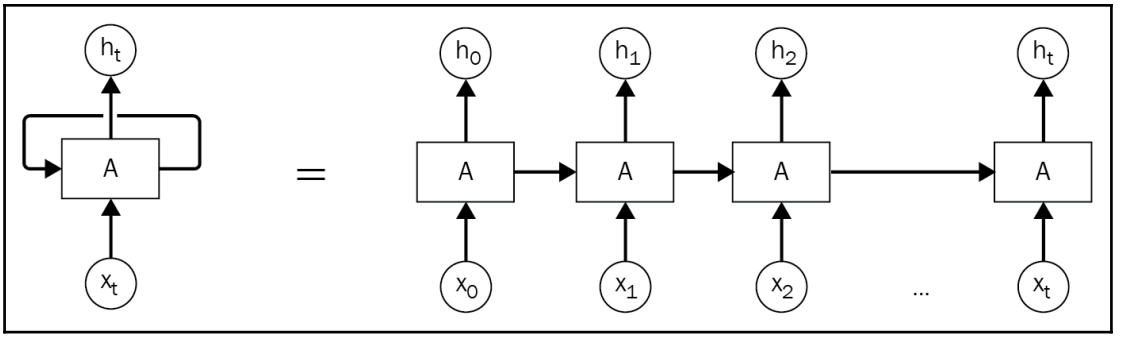

20
15
10
5
0
−5
−10
−15
−20
−20
−15
−10
−5
0
5
10
15
20

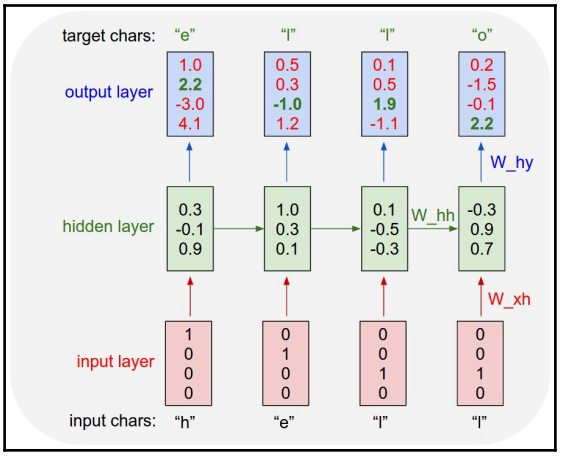

True: ship
Pred: frog
True: deer
Pred: frog
True: automobile
Pred: truck
True: horse
Pred: cat
True: frog
Pred: bird
True: cat
Pred: deer
True: airplane
Pred: truck
True: cat
Pred: dog
True: bird
Pred: deer

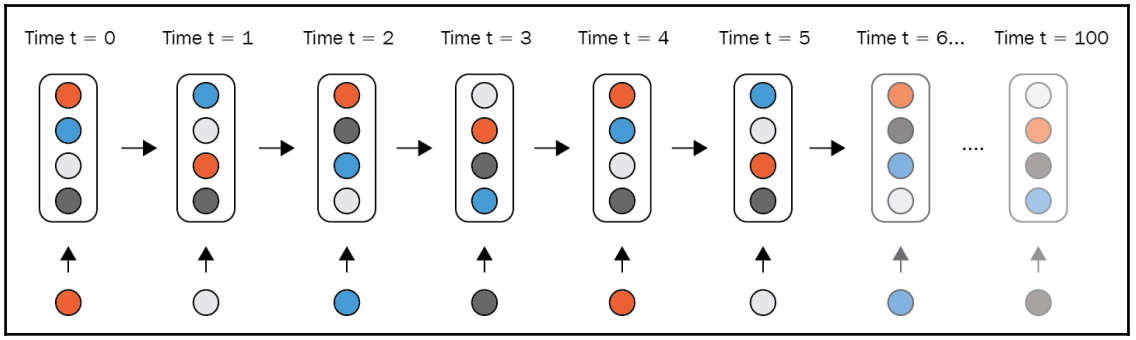
Chapter 10: Recurrent-Type Neural Networks - Language Modeling

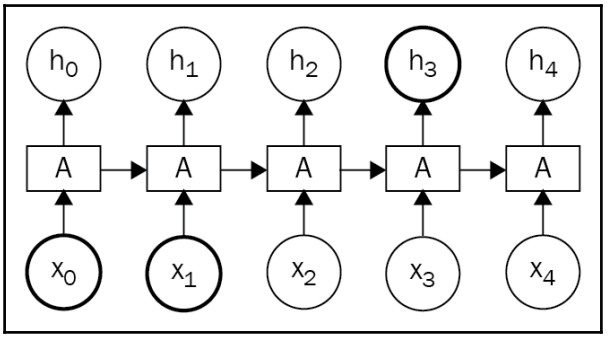


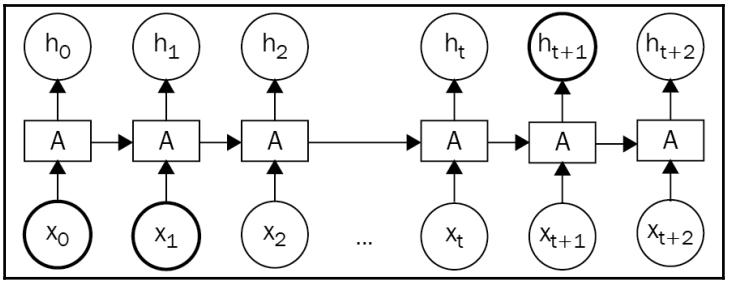


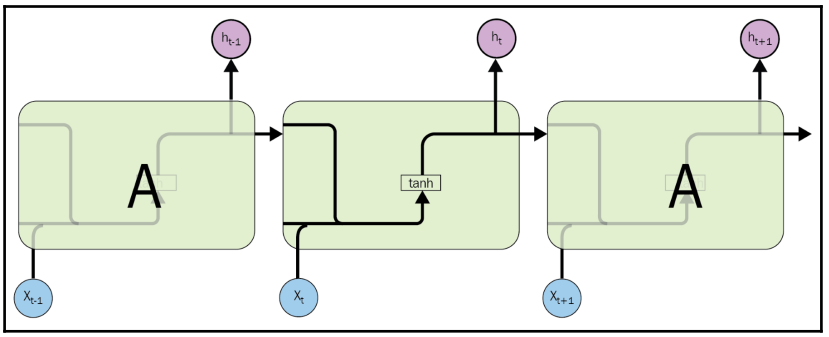


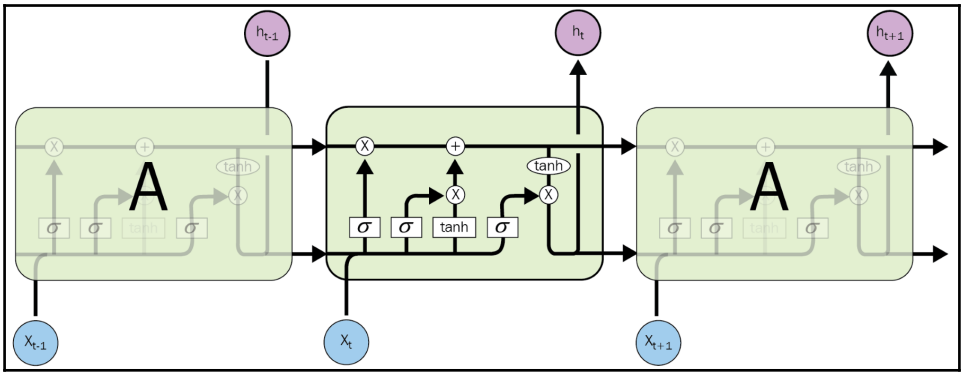

target chars: "e" "l" "l" "o"
output layer
1.0
2.2
-3.0
4.1
0.5
0.3
-1.0
1.2
0.1
0.5
1.9
-1.1
0.2
-1.5
-0.1
2.2
W_hy
hidden layer
0.3
-0.1
0.9
1.0
0.3
0.1
0.1
-0.5
-0.3
W_hh
-0.3
0.9
0.7
W_xh
input layer
1
0
0
0
0
1
0
0
0
0
1
0
0
0
1
0
input chars: "h" "e" "l" "l"

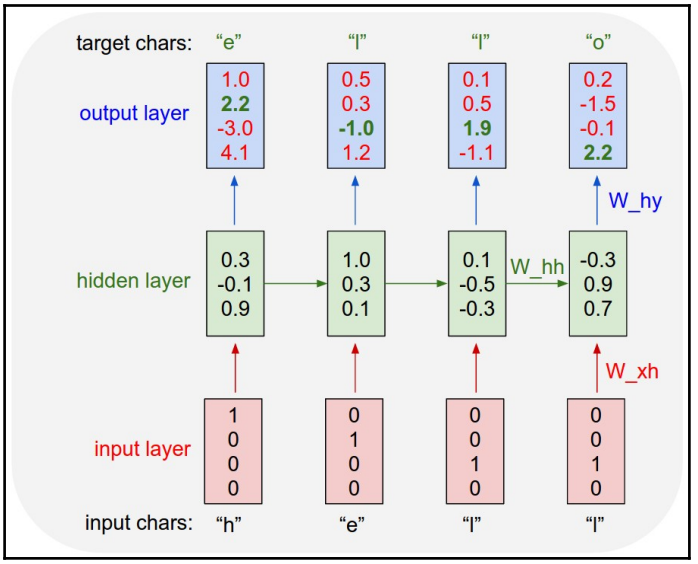

Time t = 0
Time t = 1
Time t = 2
Time t = 3
Time t = 4
Time t = 5
Time t = 6...
Time t = 100
....

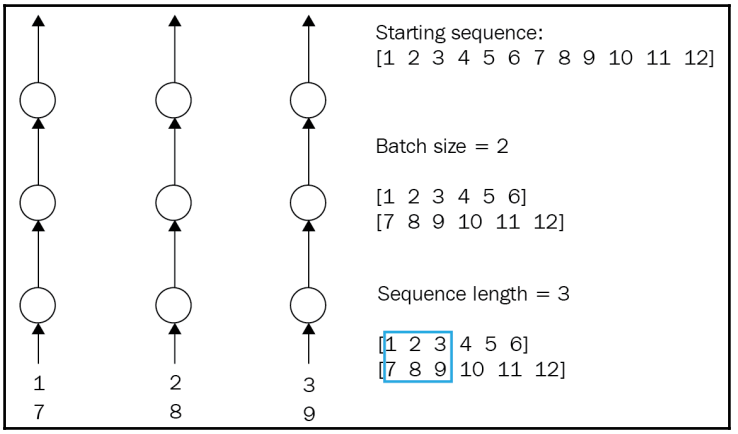

h0
h1
h2
h3
h4
A
A
A
A
A
x0
x1
x2
x3
x4

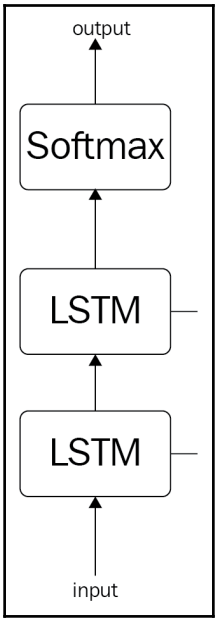

h0
h1
h2
ht
ht+1
ht+2
A
A
A
A
A
A
x0
x1
x2
...
xt
xt+1
xt+2

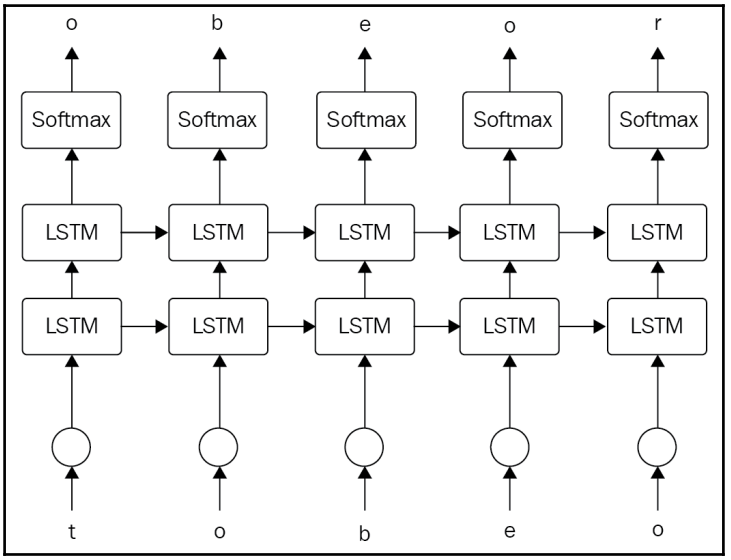

ht-1
ht
ht+1
A
tanh
A
xt-1
xt
xt+1


ht-1
ht
ht+1
A
tanh
σ
σ
tanh
σ
A
xt-1
xt
xt+1

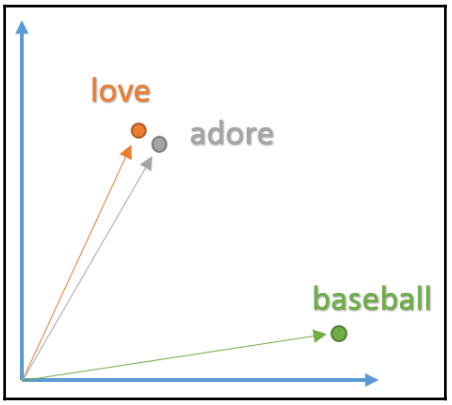

target chars:
"e"
"l"
"l"
"o"
output layer
1.0
2.2
-3.0
4.1
0.5
0.3
-1.0
1.2
0.1
0.5
1.9
-1.1
0.2
-1.5
-0.1
2.2
W_hy
hidden layer
0.3
-0.1
0.9
1.0
0.3
0.1
0.1
-0.5
-0.3
W_hh
-0.3
0.9
0.7
W_xh
input layer
1
0
0
0
0
1
0
0
0
0
1
0
0
0
1
0
input chars:
"h"
"e"
"l"
"l"


1
7
2
8
3
9
Starting sequence:
[1 2 3 4 5 6 7 8 9 10 11 12]
Batch size = 2
[1 2 3 4 5 6]
[7 8 9 10 11 12]
Sequence length = 3
[1 2 3 4 5 6]
[7 8 9 10 11 12]

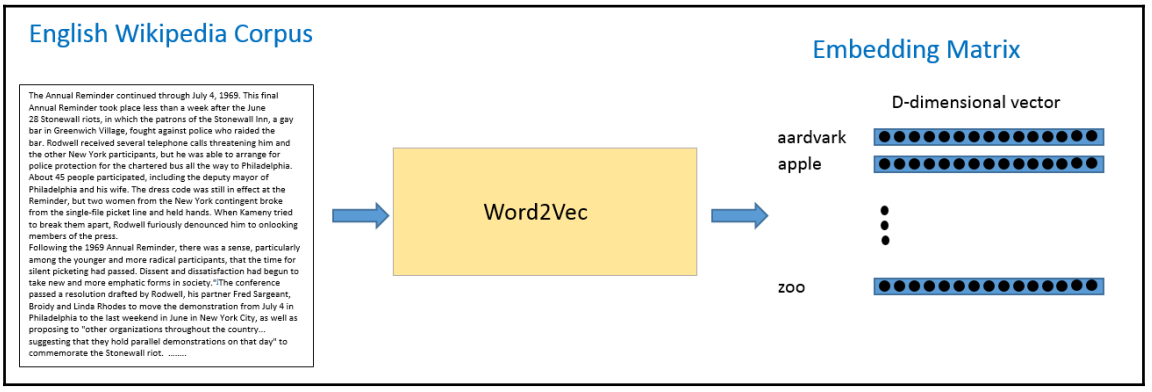

output
Softmax
LSTM
LSTM
input

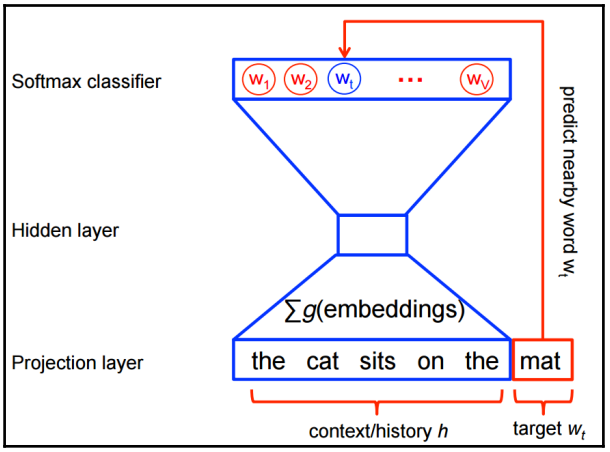

o
b
e
o
r
Softmax
Softmax
Softmax
Softmax
Softmax
LSTM
LSTM
LSTM
LSTM
LSTM
LSTM
LSTM
LSTM
LSTM
LSTM
t
o
b
e
o

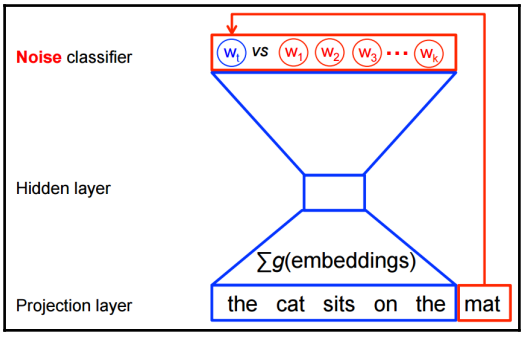
Chapter 11: Representation Learning - Implementing Word Embeddings

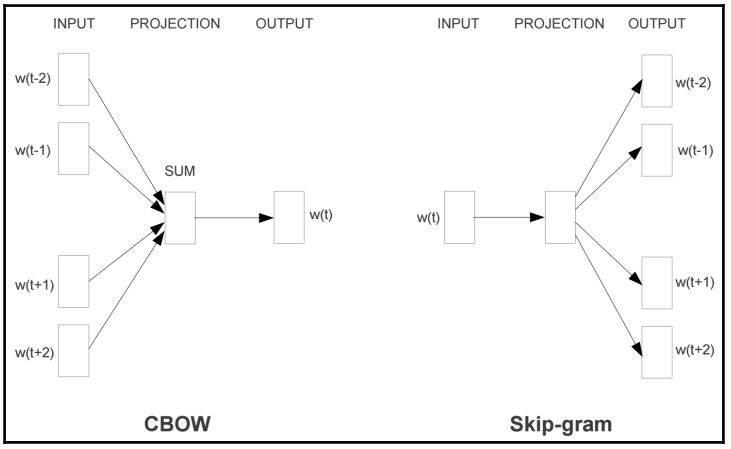


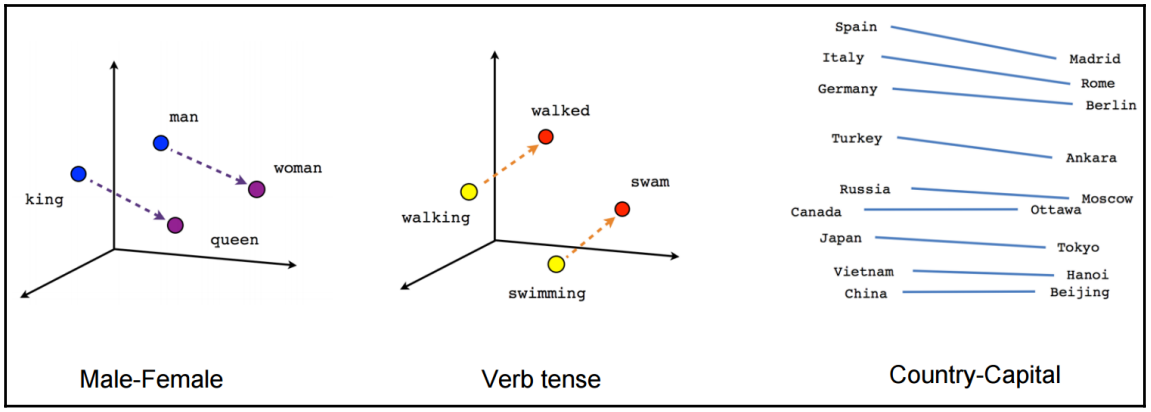
I love taking long walks on the beach.
My friends told me that they love popcorn.
⋮
The relatives adore the baby's cute face.
I adore his sense of humor.

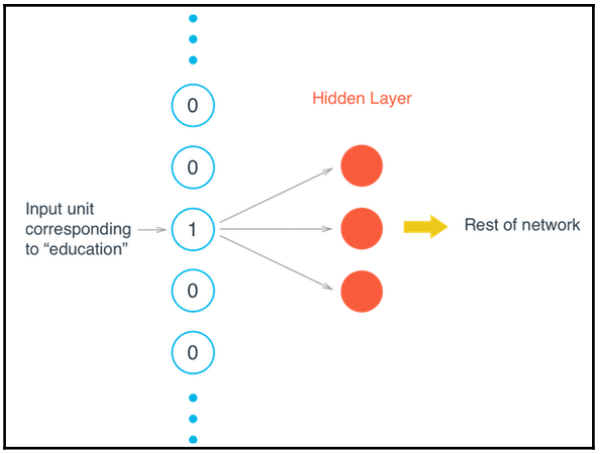



Softmax classifier
Hidden layer
Projection layer
∑g(embeddings)
the cat sits on the mat
predict nearby word w_t
context/history h
target w_t

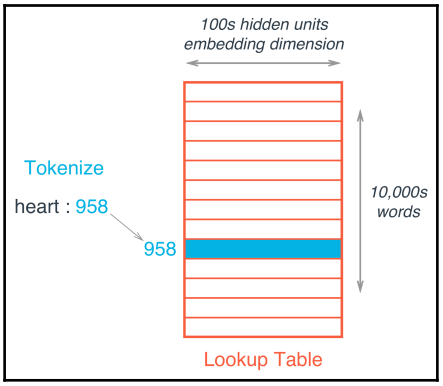

Noise classifier
vs
Hidden layer
Projection layer
∑g(embeddings)
the cat sits on the mat

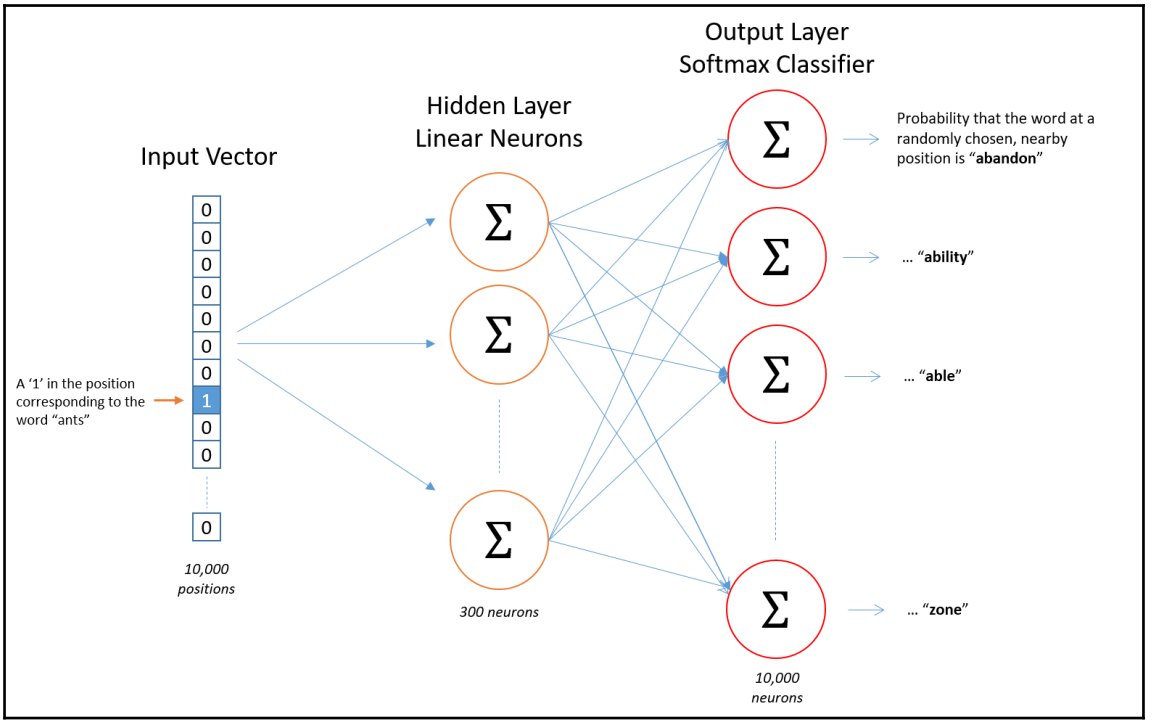

INPUT
PROJECTION
OUTPUT
w(t-2)
w(t-1)
SUM
w(t)
w(t+1)
w(t+2)
CBOW
INPUT
PROJECTION
OUTPUT
w(t)
w(t-2)
w(t-1)
w(t+1)
w(t+2)
Skip-gram

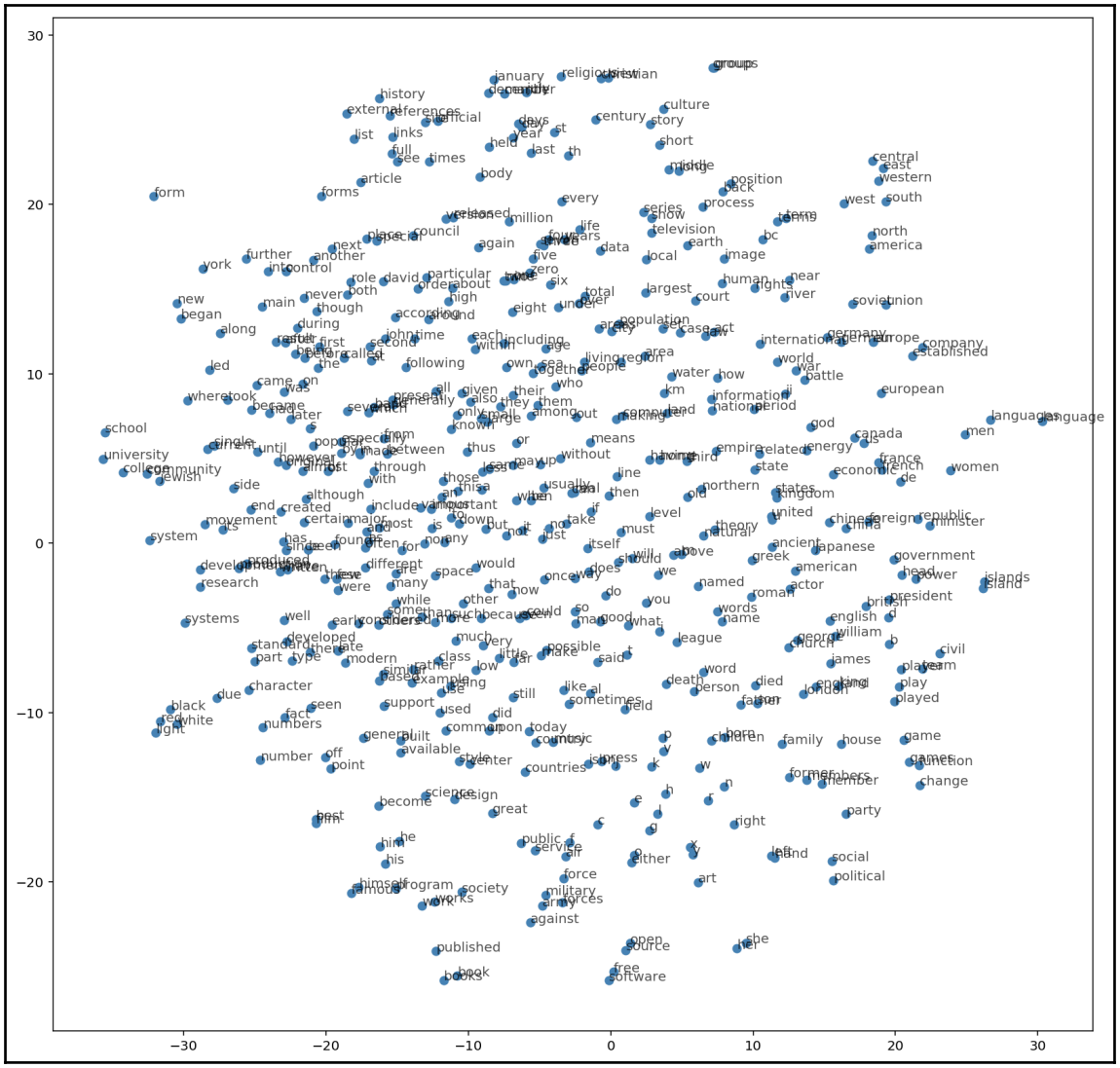




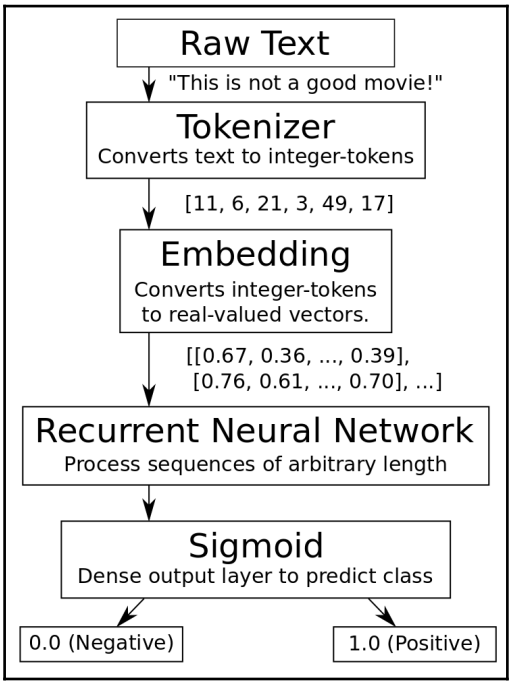
$$\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 8 & 2 & 1 & 9 \\ 6 & 5 & 4 & 0 \\ 7 & 1 & 6 & 2 \\ 1 & 3 & 5 & 8 \\ 0 & 4 & 9 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 3 & 5 & 8 \end{bmatrix}$$

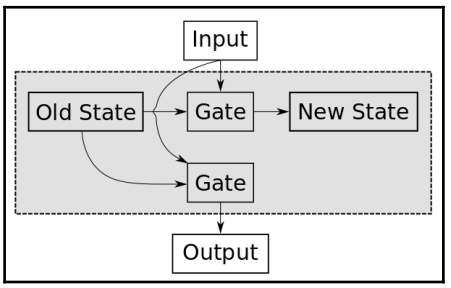

100s hidden units
embedding dimension
Tokenize
heart : 958
958
10,000s
words
Lookup Table

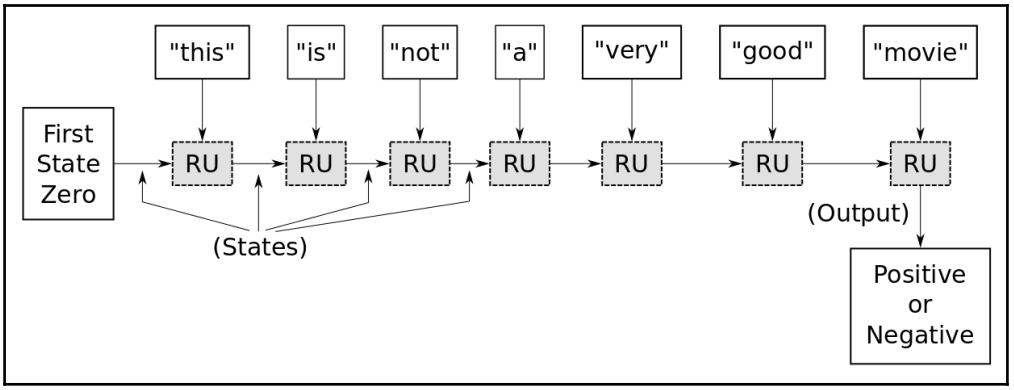

Output Layer
Softmax Classifier
Hidden Layer
Linear Neurons
Input Vector
Probability that the word at a randomly chosen, nearby position is "abandon"
... "ability"
... "able"
... "zone"
A '1' in the position corresponding to the word "ants"
0
0
0
0
0
0
0
1
0
0
0
10,000
positions
300 neurons
10,000
neurons

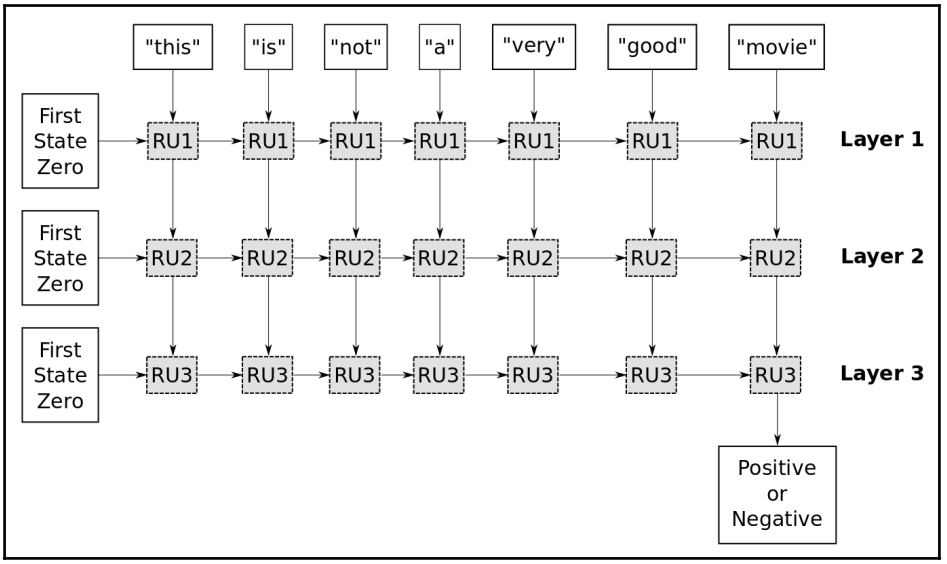


Chapter 12: Neural Sentiment Analysis

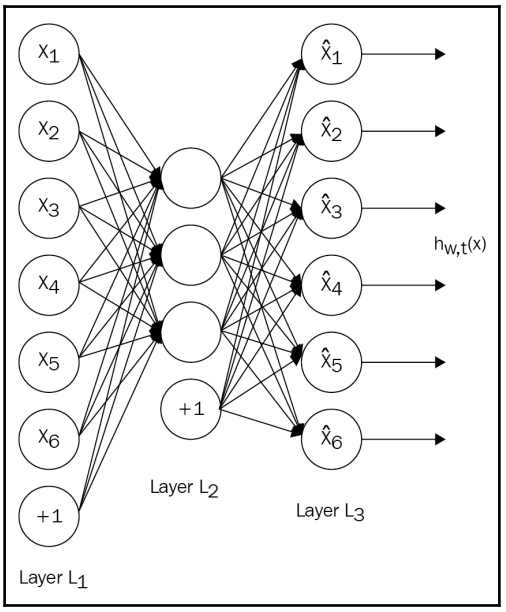


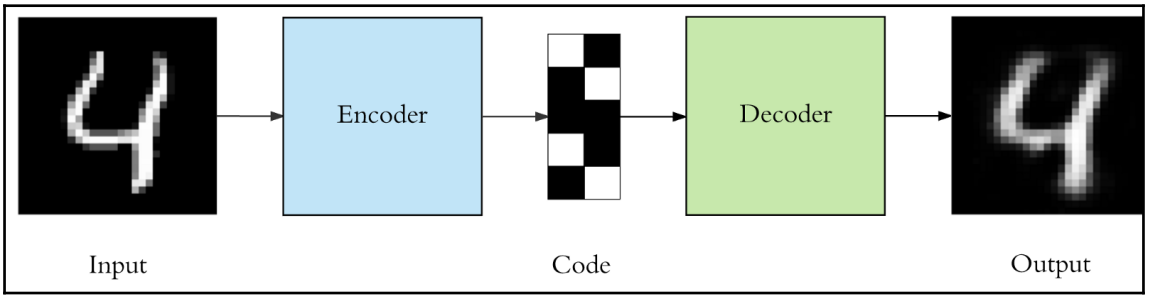


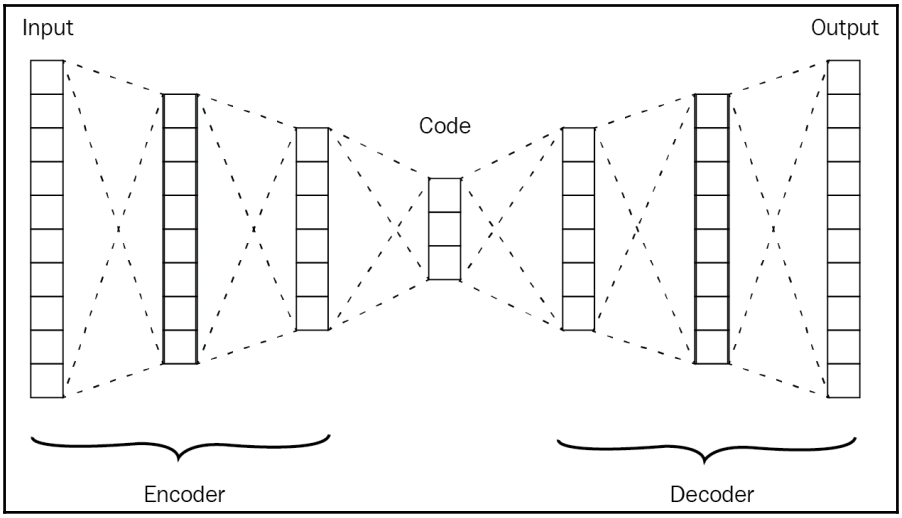

"this"
"is"
"not"
"a"
"very"
"good"
"movie"
First State Zero
RU
(States)
(Output)
Positive or Negative

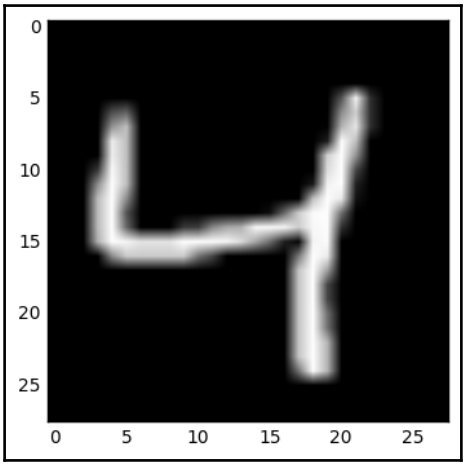

"this"
"is"
"not"
"a"
"very"
"good"
"movie"
First State Zero
RU1
RU2
RU3
Layer 1
Layer 2
Layer 3
Positive or Negative

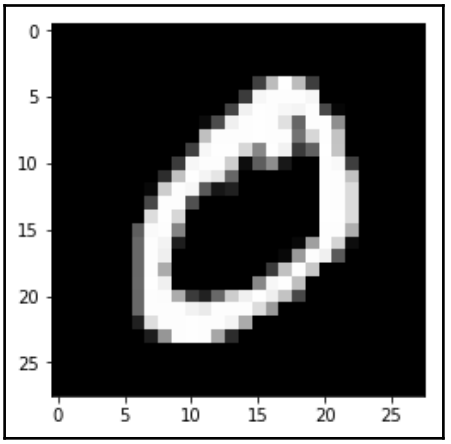
Chapter 13: Autoencoders – Feature Extraction and Denoising

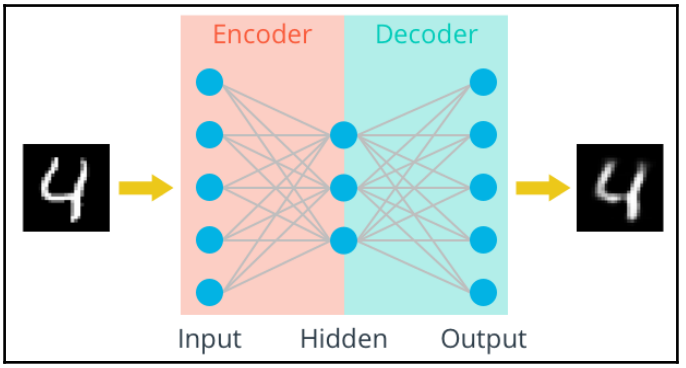


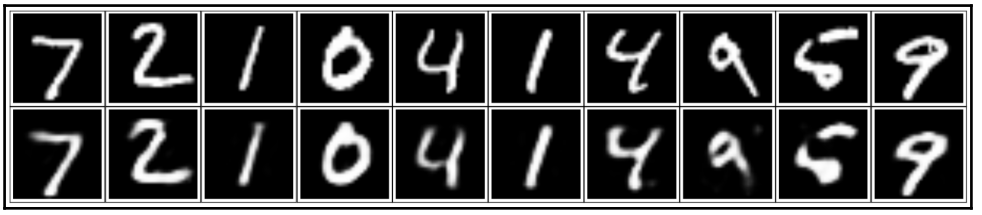


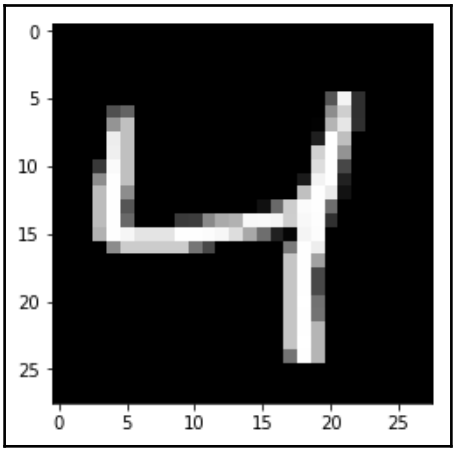

Input
Output
Code
Encoder
Decoder

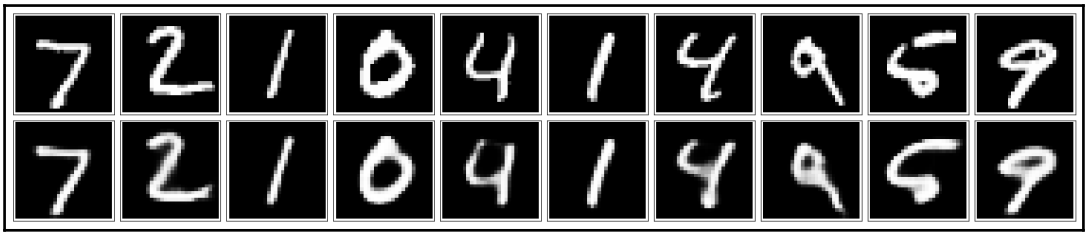

0
5
10
15
20
25
0
5
10
15
20
25

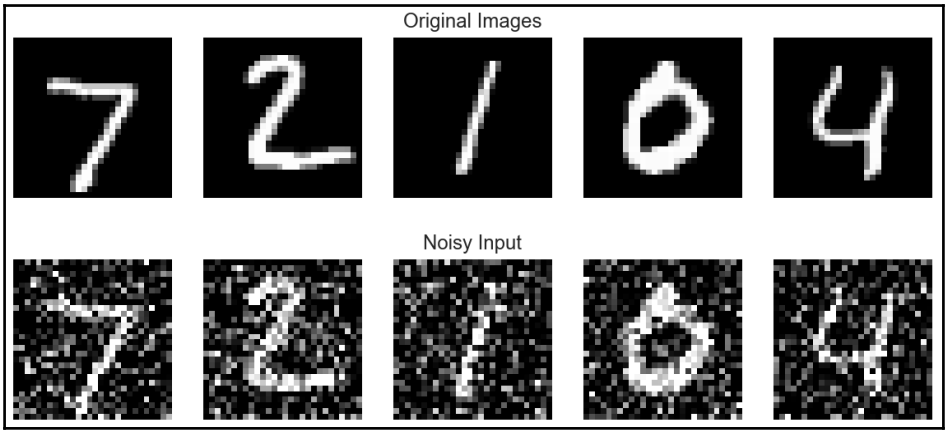

0
5
10
15
20
25
0
5
10
15
20
25

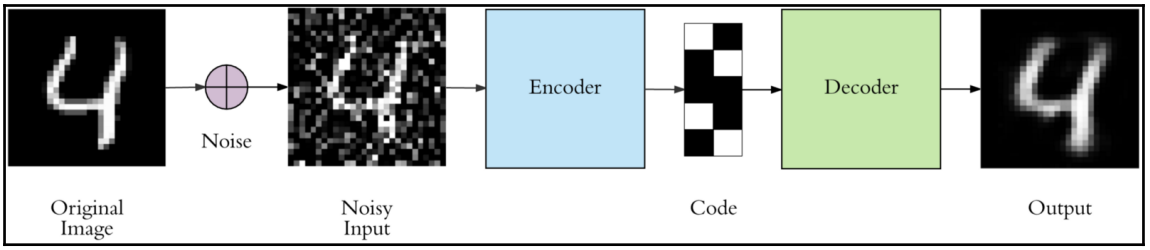

Encoder
Decoder
Input
Hidden
Output

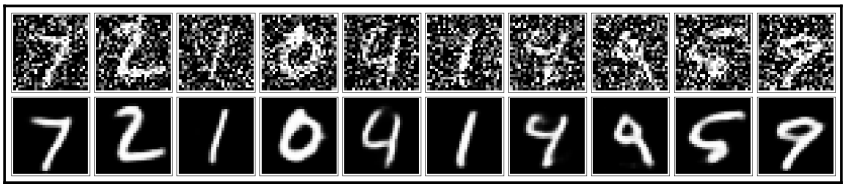


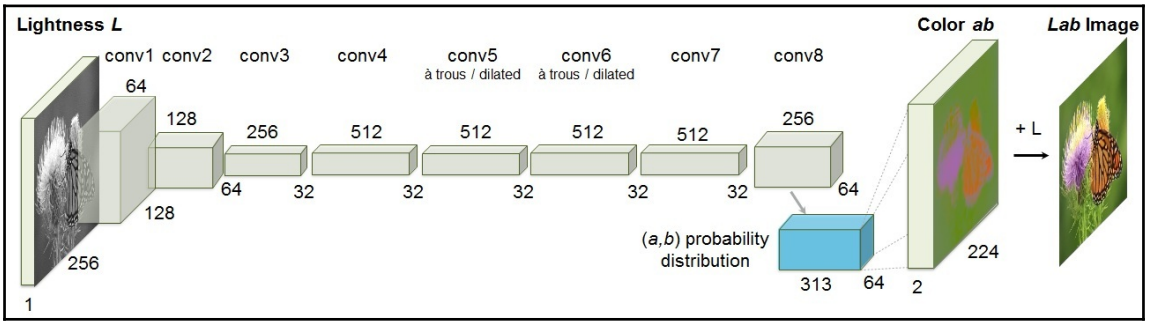

0
5
10
15
20
25
0
5
10
15
20
25



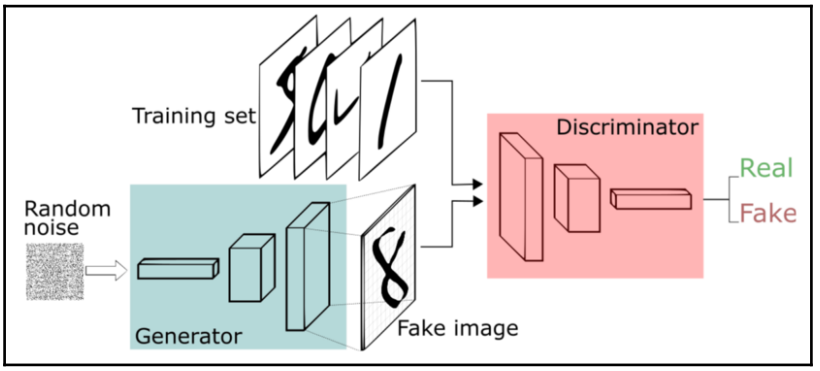

Original Images
Noisy Input

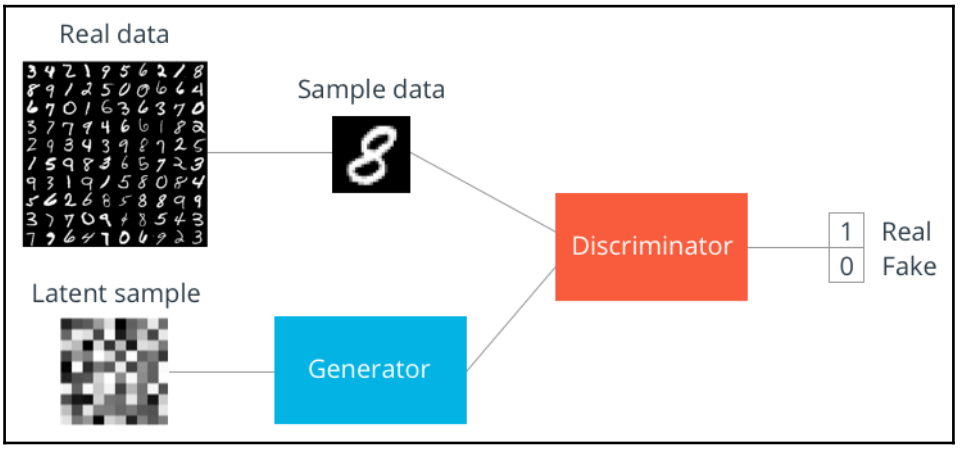

Noise
Encoder
Decoder
Original Image
Noisy Input
Code
Output

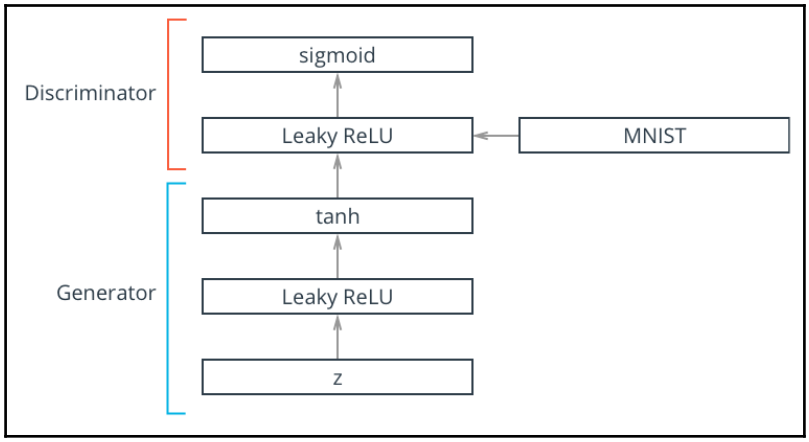


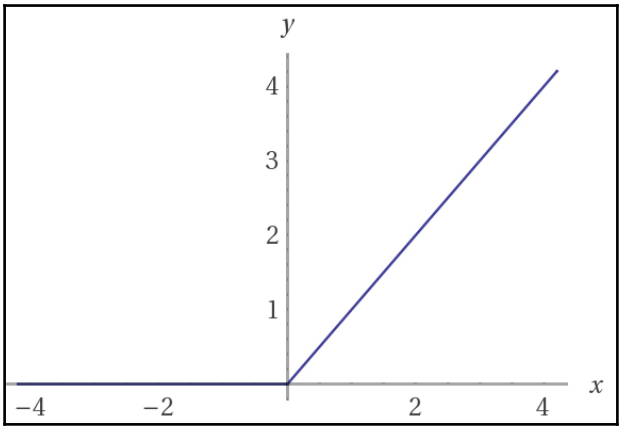

Lightness L
Color ab
Lab Image
conv1 conv2 conv3 conv4 conv5 conv6 conv7 conv8
à trous / dilated
à trous / dilated
64
128
256
512
512
512
512
256
64
32
32
32
32
32
64
128
256
1
(a,b) probability distribution
313
64
2
224
+ L

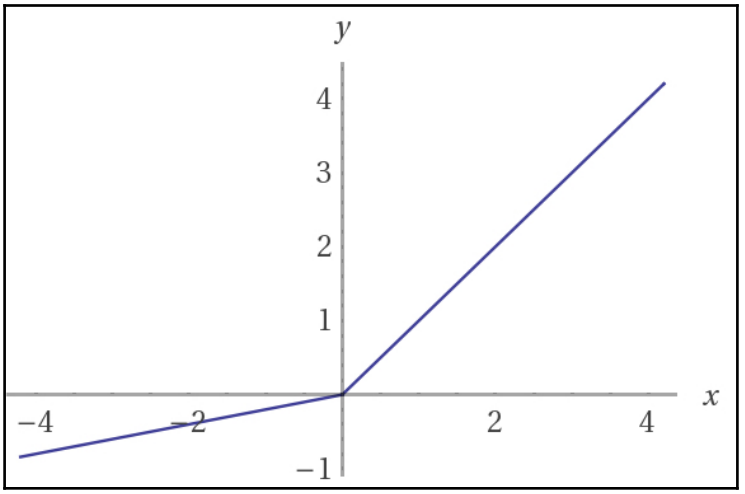
Chapter 14: Generative Adversarial Networks

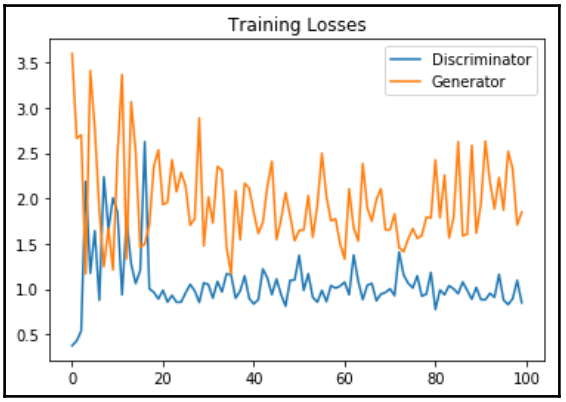


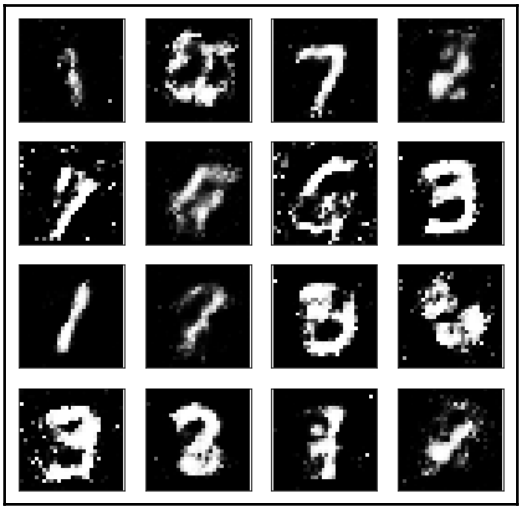


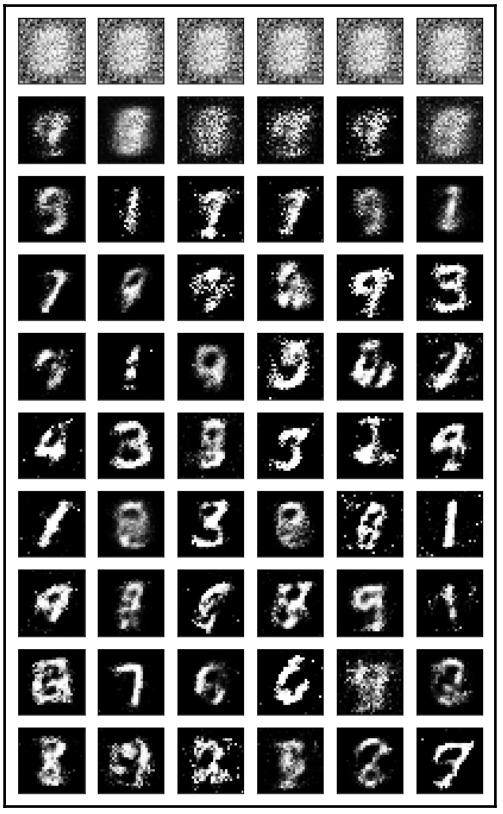

Discriminator
sigmoid
Leaky ReLU
MNIST
Generator
tanh
Leaky ReLU
z

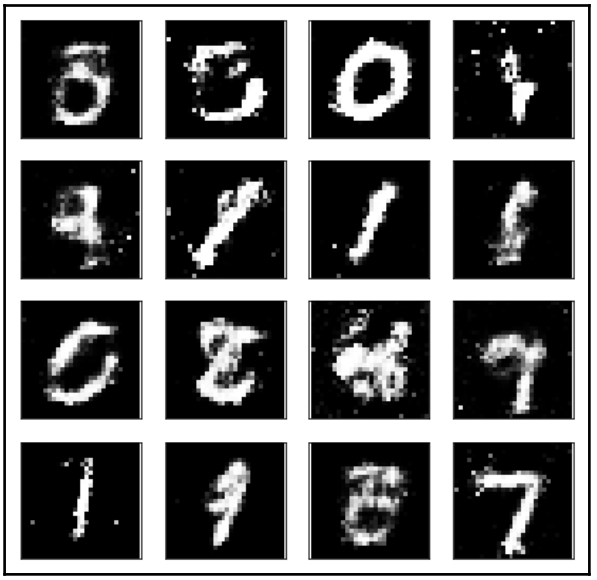

y
4
3
2
1
x
−4
−2
2
4


y
4
3
2
1
−4
−2
2
4
−1
x

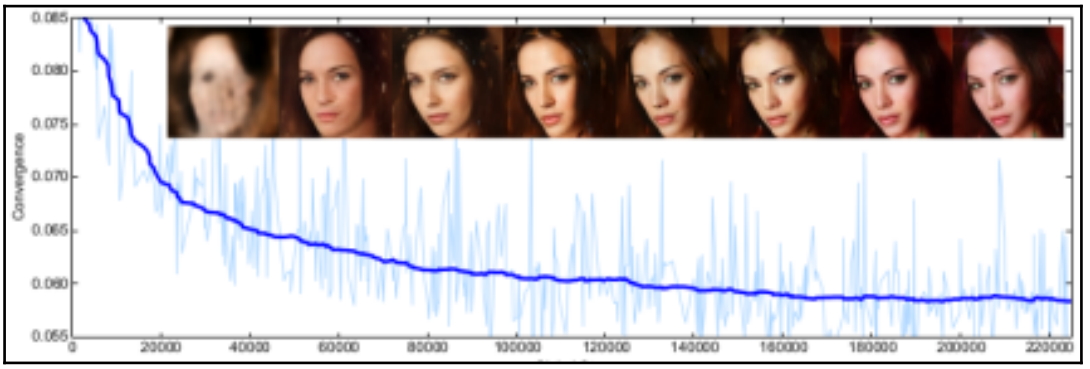

Training Losses
Discriminator
Generator
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0
20
40
60
80
100

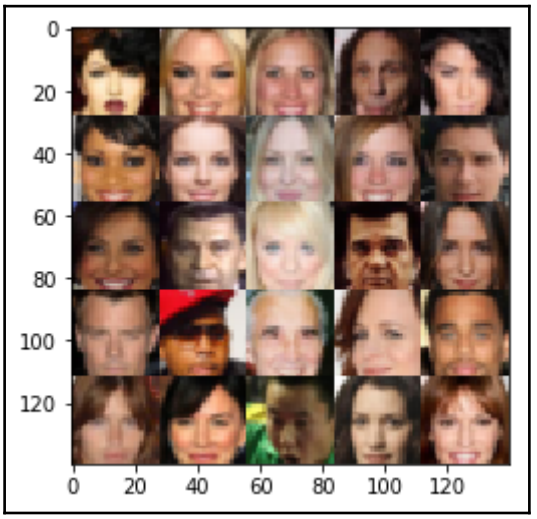


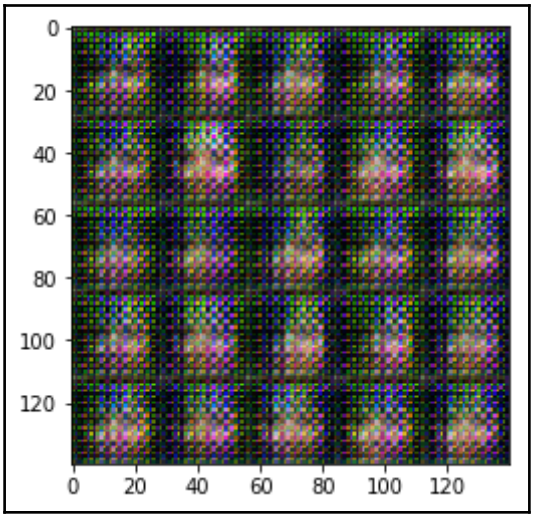


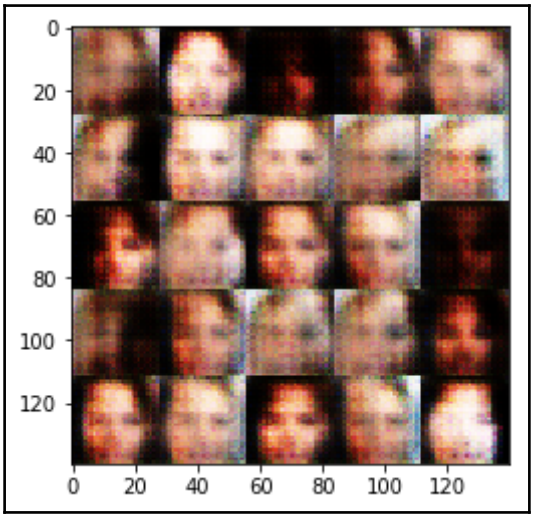


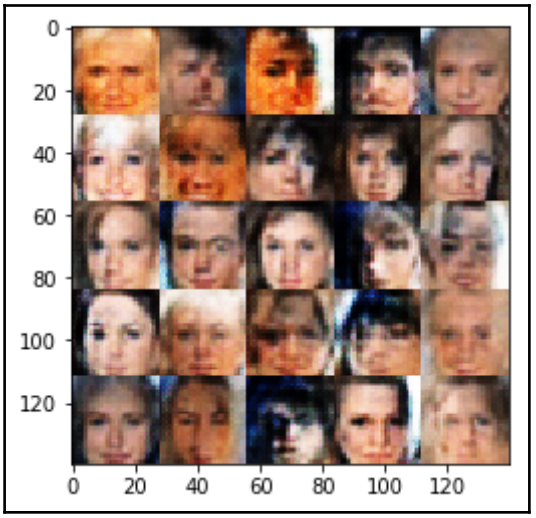
Chapter 15: Face Generation and Handling Missing Labels

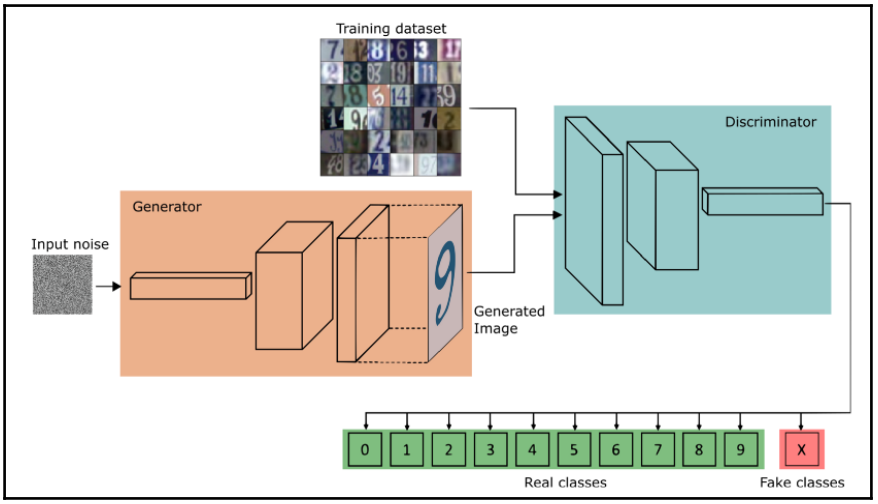


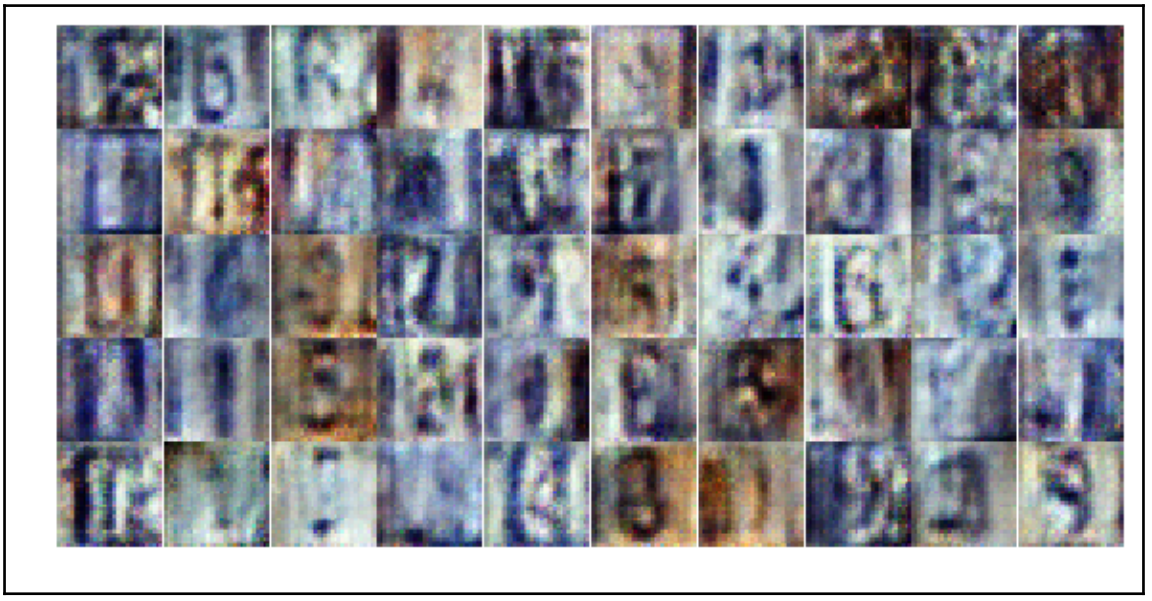


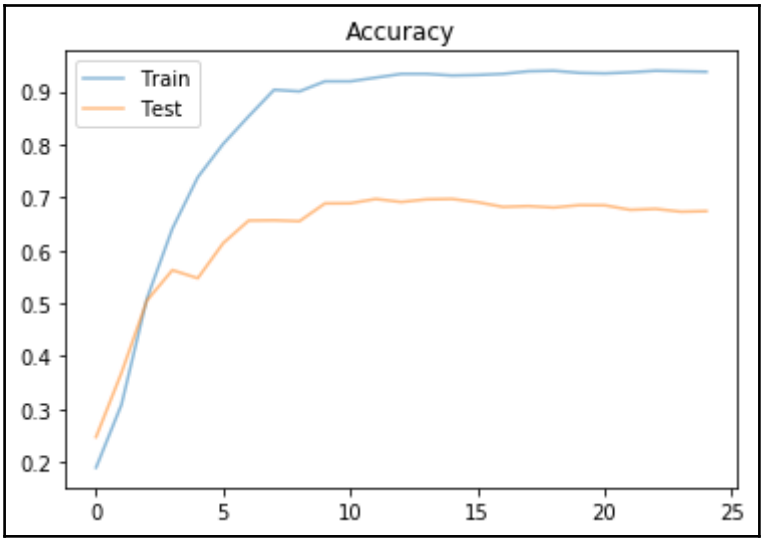

0
20
40
60
80
100
120
0
20
40
60
80
100
120


0
20
40
60
80
100
120
0
20
40
60
80
100
120


0
20
40
60
80
100
120
0
20
40
60
80
100
120


Training dataset
Discriminator
Generator
Input noise
Generated
Image
0
1
2
3
4
5
6
7
8
9
X
Real classes
Fake classes




Accuracy
Train
Test
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0
5
10
15
20
25

Index