Maximum Entropy Classifier

Supervised Machine Learning

CSE538 - Spring 2024

Text Classification

The Buccaneers win it!

President Biden vetoed bill

Twitter to be acquired by Apple

She <u>will</u> drive to the office, to make sure the lawyer gives the <u>will</u> to the family.

<u>*will.n*</u> or <u>*will.v*</u> ?

noun or verb

I like the the movie.

The movie is like terrible.

Supervised Classification

Supervised Classification

X - features of N observations (i.e. words)

Y - class of each of N observations

GOAL: Produce a *model* that outputs the most likely class y_i, given features x_i.

$$f(X) = Y$$

Supervised Classification

X - features of N observations (i.e. words)

Y - class of each of N observations

GOAL: Produce a *model* that outputs the most likely class y_i, given features x_i.

$f(X) = Y$

$$
\begin{array}{ccc}
i & X & Y \\
\begin{array}{c} 0 \\ 1 \\ 2 \\ 3 \\ 4 \end{array} &
\begin{pmatrix} 0.0 \\ 0.5 \\ 1.0 \\ 0.25 \\ 0.75 \end{pmatrix} &
\begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}
\end{array}
$$

Supervised Classificati

X - features of N observations (

Y - class of each of N observations

Some function or rules to go from X to Y, as close as possible.

GOAL: Produce a *model* that outputs the most likely class y_i, given features x_i.

$f(X) = Y$

i	X	Y
0	0.0	0
1	0.5	0
2	1.0	1
3	0.25	0
4	0.75	1

Supervised Classification

Supervised Machine Learning: Build a model with examples of outcomes (i.e. Y) that one is trying to predict. (The alternative, *unsupervised* machine learning, tries to learn with only an X).

Classification: The outcome (Y) is a discrete class.

for example: $y \in$ {not-noun, noun}

$y \in$ {noun, verb, adjective, adverb}

$y \in$ {positive_sentiment, negative_sentiment}).

Classification as Producing a Probability

Binary classification goal: Build a model that can estimate $P(A=1|B=?)$

i.e. given B, yield (or “predict”) the probability that A=1

Classification as Producing a Probability

Binary classification goal: Build a “model” that can estimate $P(A=1|B=?)$

i.e. given B, yield (or “predict”) the probability that $A=1$

In machine learning, the tradition is to use **Y** for the variable being predicted and **X** for the features use to make the prediction.

Classification as Producing a Probability

Binary classification goal: Build a “model” that can estimate $P(Y=1|X=?)$

i.e. given X, yield (or “predict”) the probability that $Y=1$

In machine learning, the tradition is to use **Y** for the variable being predicted and **X** for the features use to make the prediction.

Classification as Producing a Probability

Binary classification goal: Build a "model" that can estimate $P(Y=1|X=?)$

i.e. given X, yield (or "predict") the probability that Y=1

In machine learning, the tradition is to use **Y** for the variable being predicted and **X** for the features use to make the prediction.

Example: Y: 1 if target is verb, 0 otherwise;
X: 1 if "was" occurs before target; 0 otherwise

I was <u>reading</u> for NLP. *We were <u>fine</u>.* *I am <u>good</u>.*

The cat was <u>very</u> happy. *We enjoyed the <u>reading</u> material.* *I was <u>good</u>.*

Classification as Producing a Probability

Binary classification goal: Build a "model" that can estimate P(Y=1|X=?)

i.e. given X, yield (or "predict") the probability that Y=1

In machine learning, the tradition is to use **Y** for the variable being predicted and **X** for the features use to make the prediction.

Example: Y: 1 if target is verb, 0 otherwise;
X: 1 if "was" occurs before target; 0 otherwise

I was reading for NLP. *We were fine.* *I am good.*

The cat was very happy. *We enjoyed the reading material.* *I was good.*

Classification as Producing a Probability

Example: Y: 1 if target is a part of a proper noun, 0 otherwise;
X: number of capital letters in target and surrounding words.

They attend Stony Brook University. *Next to the brook Gandalf lay thinking.*

The trail was very stony. *Her degree is from SUNY Stony Brook.*

The Taylor Series was first described by Brook Taylor, the mathematician.

Classification as Producing a Probability

Example: Y: 1 if target is a part of a proper noun, 0 otherwise;
X: number of capital letters in target and surrounding words.

They attend Stony Brook University. *Next to the brook Gandalf lay thinking.*

The trail was very stony. *Her degree is from SUNY Stony Brook.*

The Taylor Series was first described by Brook Taylor, the mathematician.

Classification as Producing a Probability

Example: Y: 1 if target is a part of a proper noun, 0 otherwise;
X: number of capital letters in target and surrounding words.

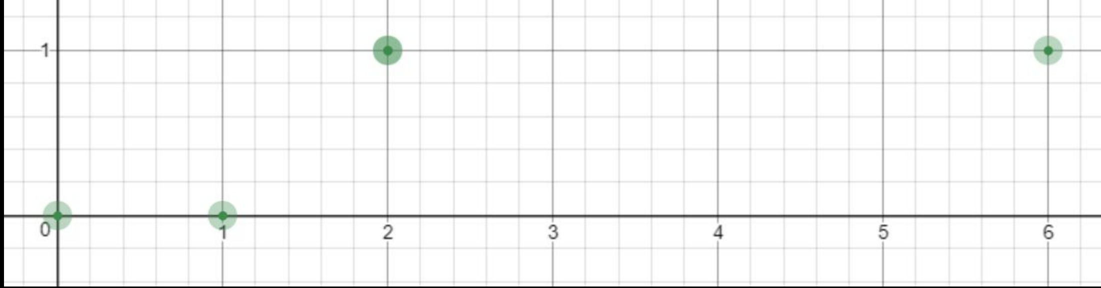
They attend Stony Brook University.　　*Next to the brook Gandalf lay thinking.*

The trail was very stony.　　*Her degree is from SUNY Stony Brook.*

The Taylor Series was first described by Brook Taylor, the mathematician.

x	y
2	1
1	0
0	0
6	1
2	1

Logistic Regression



Example: Y: 1 if target is a part of a proper noun, 0 otherwise;
X: number of capital letters in target and surrounding words.

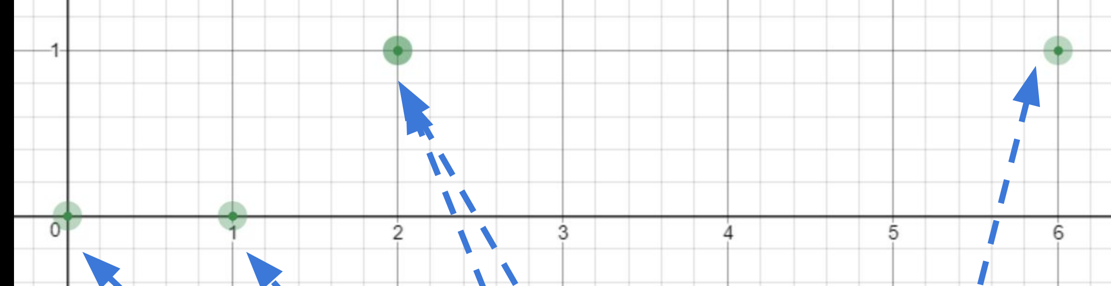
They attend Stony Brook University. *Next to the brook Gandalf lay thinking.*

The trail was very stony. *Her degree is from SUNY Stony Brook.*

The Taylor Series was first described by Brook Taylor, the mathematician.

x	y
2	1
1	0
0	0
6	1
2	1

Logistic Regression



Example: Y: 1 if target is a part of a proper noun, 0 otherwise;
X: number of capital letters in target and surrounding words.

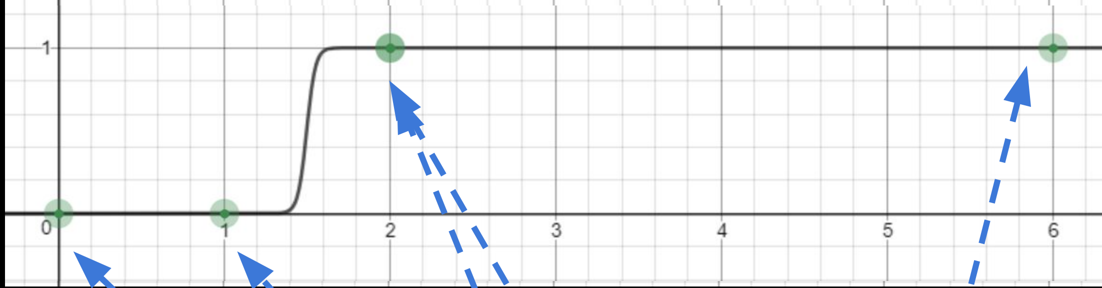
They attend Stony Brook University. *Next to the brook Gandalf lay thinking.*

The trail was very stony. *Her degree is from SUNY Stony Brook.*

The Taylor Series was first described by Brook Taylor, the mathematician.

x	y
2	1
1	0
0	0
6	1
2	1

Logistic Regression



Example: Y: 1 if target is a part of a proper noun, 0 otherwise;
X: number of capital letters in target and surrounding words.

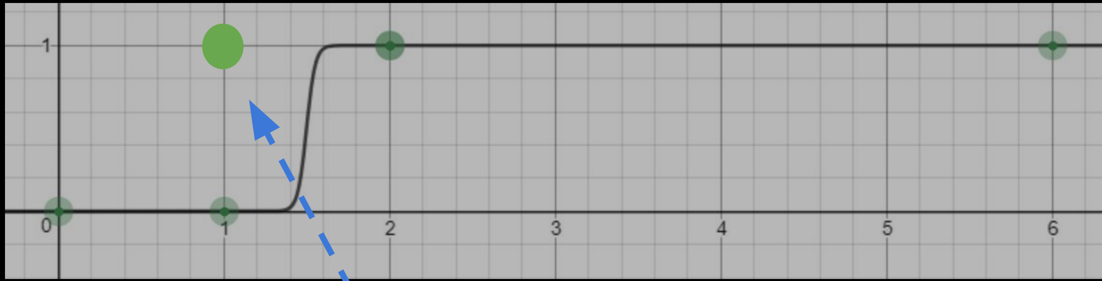
They attend Stony Brook University. *Next to the brook Gandalf lay thinking.*

The trail was very stony. *Her degree is from SUNY Stony Brook.*

The Taylor Series was first described by Brook Taylor, the mathematician.

x	y
2	1
1	0
0	0
6	1
2	1

Logistic Regression



Example: Y: 1 if target is a part of a proper noun, 0 otherwise;
X: number of capital letters in target and surrounding words.

They attend Stony Brook University. *Next to the brook Gandalf lay thinking.*

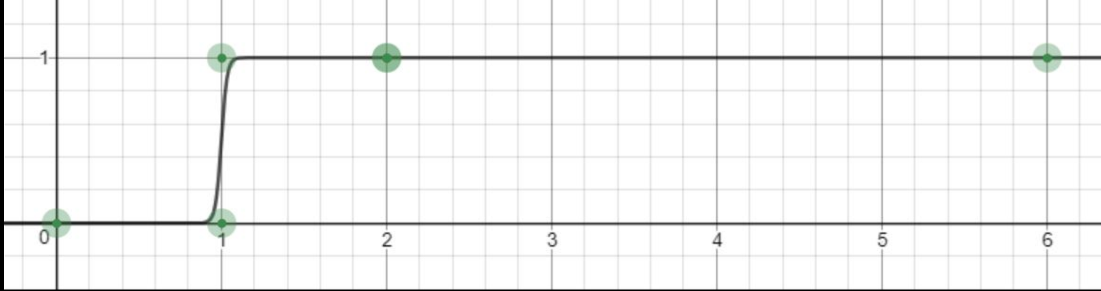
The trail was very stony. *Her degree is from SUNY Stony Brook.*

The Taylor Series was first described by Brook Taylor, the mathematician.

They attend Binghamton.

x	y
2	1
1	0
0	0
6	1
2	1
1	1

Logistic Regression



Example: Y: 1 if target is a part of a proper noun, 0 otherwise;
X: number of capital letters in target and surrounding words.

They attend Stony Brook University. *Next to the brook Gandalf lay thinking.*

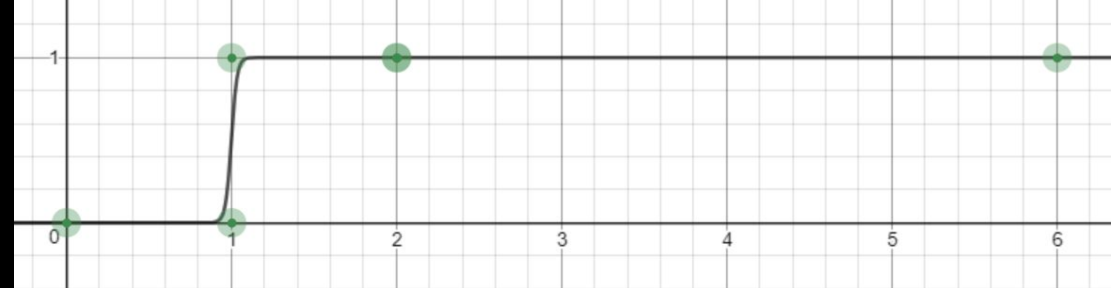
The trail was very stony. *Her degree is from SUNY Stony Brook.*

The Taylor Series was first described by Brook Taylor, the mathematician.

They attend Binghamton.

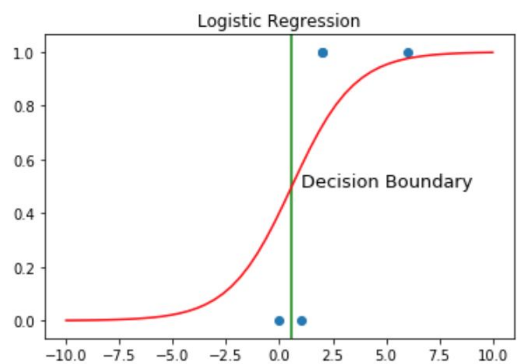
x	y
2	1
1	0
0	0
6	1
2	1
1	1

Logistic Regression



Example: Y: 1 if target is a part of a proper noun, 0 otherwise;
X: number of capital letters in target and surrounding words.

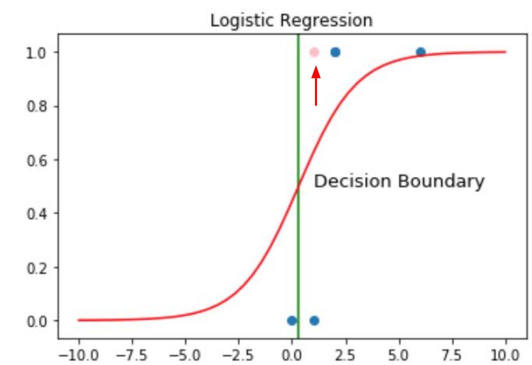
Out[43]: [<matplotlib.lines.Line2D at 0x116e68d68>]



In [78]:
```
-b_0/b_1
```
Out[78]: 0.5824799517820446

In [28]:
```
logisticRegr.predict(x)
```
Out[28]: array([1, 1, 0, 1, 1])

Out[80]: [<matplotlib.lines.Line2D at 0x11a60f160>]

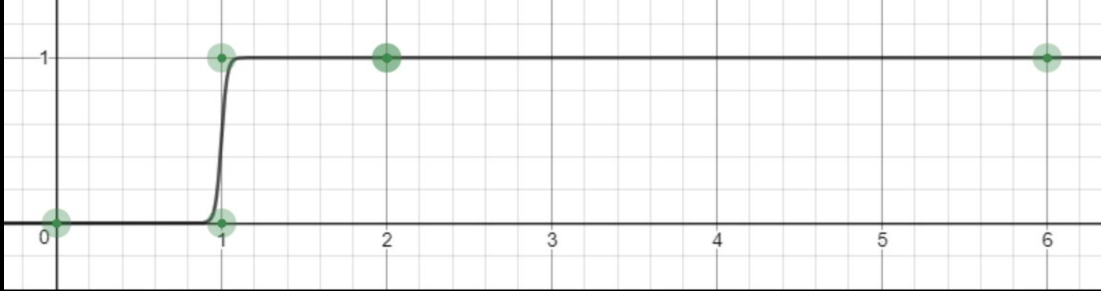


In [81]:
```
-b2_0/b2_1
```
Out[81]: 0.31089309388058134

In [82]:
```
logisticRegr2.predict(x2)
```
Out[82]: array([1, 1, 0, 1, 1, 1])

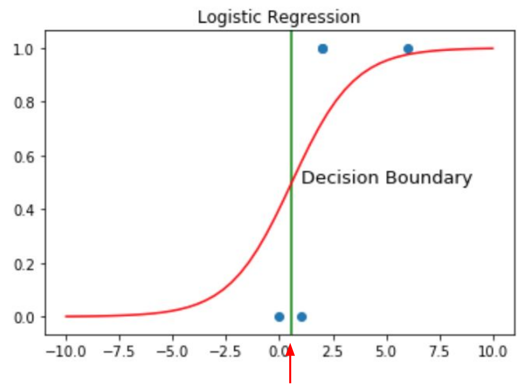
1	1

Logistic Regression



Example: Y: 1 if target is a part of a proper noun, 0 otherwise;
X: number of capital letters in target and surrounding words.

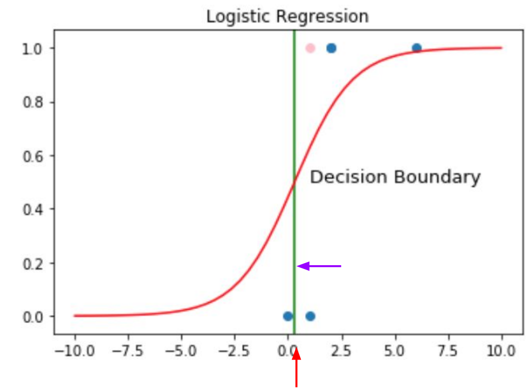
```
Out[43]: [<matplotlib.lines.Line2D at 0x116e68d68>]
```




```
In [78]: -b_0/b_1
Out[78]: 0.5824799517820446

In [28]: logisticRegr.predict(x)
Out[28]: array([1, 1, 0, 1, 1])
```

```
Out[80]: [<matplotlib.lines.Line2D at 0x11a60f160>]
```

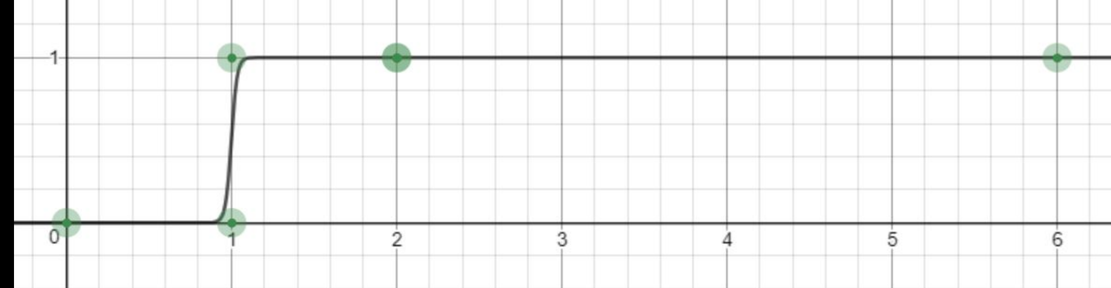



```
In [81]: -b2_0/b2_1
Out[81]: 0.31089309388058134

In [82]: logisticRegr2.predict(x2)
Out[82]: array([1, 1, 0, 1, 1, 1])
```

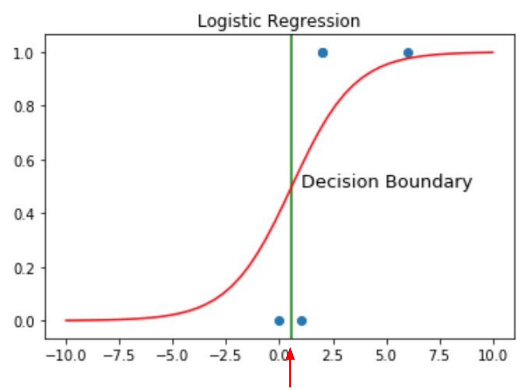
1	1

Logistic Regression



Example: Y: 1 if target is a part of a proper noun, 0 otherwise;
X: number of capital letters in target and surrounding words.

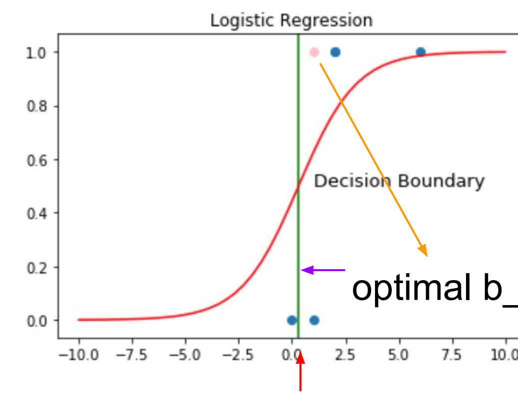
```
Out[43]: [<matplotlib.lines.Line2D at 0x116e68d68>]
```




```
In [78]: -b_0/b_1
Out[78]: 0.5824799517820446

In [28]: logisticRegr.predict(x)
Out[28]: array([1, 1, 0, 1, 1])
```

```
Out[80]: [<matplotlib.lines.Line2D at 0x11a60f160>]
```



optimal b_0, b_1 changed!

```
In [81]: -b2_0/b2_1
Out[81]: 0.31089309388058134

In [82]: logisticRegr2.predict(x2)
Out[82]: array([1, 1, 0, 1, 1, 1])
```

1	1

Logistic Regression on a single feature (*x*)

$Y_i \in \{0, 1\}$; X is a **single value** and can be anything numeric.

$$P(Y_i = 1 | X_i = x) = \frac{e^{\beta_0 + \beta_1 x_i}}{1 + e^{\beta_0 + \beta_1 x_i}}$$

Logistic Regression on a single feature (*x*)

$Y_i \in \{0, 1\}$; X is a **single value** and can be anything numeric.

$$P(Y_i = 1 | X_i = x) = \frac{e^{\beta_0 + \beta_1 x_i}}{1 + e^{\beta_0 + \beta_1 x_i}}$$

$$= \frac{1}{1 + e^{-(\beta_0 + \sum_{j=1}^{m} \beta_j x_{ij})}}$$

Logistic Regression on a single feature (*x*)

$Y_i \in \{0, 1\}$; X can be anything numeric.

$$P(Y_i = 1 | X_i = x) = \frac{e^{\beta_0 + \beta_1 x_i}}{1 + e^{\beta_0 + \beta_1 x_i}}$$

The goal of this function is to: <u>take in the variable *x*</u> and
<u>return a probability that *Y* is 1</u>.

Logistic Regression on a single feature (*x*)

$Y_i \in \{0, 1\}$; X can be anything numeric.

$$p_i \equiv P(Y_i = 1 | X_i = x) = \frac{e^{\boxed{\beta_0}+\boxed{\beta_1}\boxed{x_i}}}{1 + e^{\boxed{\beta_0}+\boxed{\beta_1}\boxed{x_i}}}$$

The goal of this function is to: take in the variable *x* and return a probability that *Y* is 1.

Note that there are only three variables on the right: X_i, B_0, B_1

Logistic Regression on a single feature (x)

$Y_i \in \{0, 1\}$; X can be anything numeric.

$$p_i \equiv P(Y_i = 1 | X_i = x) = \frac{e^{\boxed{\beta_0} + \boxed{\beta_1}\boxed{x_i}}}{1 + e^{\boxed{\beta_0} + \boxed{\beta_1}\boxed{x_i}}}$$

The goal of this function is to: take in the variable *x* and return a probability that *Y* is 1.

Note that there are only three variables on the right: X_i, B_0, B_1

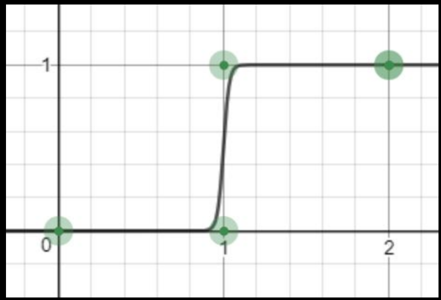
X is given. B_0 and B_1 must be learned.

Logistic Regression on a single feature (*x*)

$Y_i \in \{0, 1\}$; X can be anything numeric.

$$p_i \equiv P(Y_i = 1|X_i = x) = \frac{e^{\boxed{\beta_0}+\boxed{\beta_1}x_i}}{1 + e^{\boxed{\beta_0}+\boxed{\beta_1}x_i}}$$

HOW? Essentially, try different B_0 and B_1 values until “best fit” to the training data (example X and Y).



X is given. B_0 and B_1 must be **learned**.

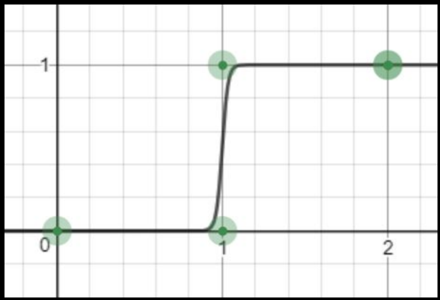
“best fit” : whatever maximizes the likelihood function:

$$L(\beta_0, \beta_1 | X, Y) = \prod_{i=1}^{n} p(x_i)^{y_i} (1 - p(x_i))^{1-y_i}$$

$$p_i \equiv P(Y_i = 1 | X_i = x) = \frac{e^{\boxed{\beta_0} + \boxed{\beta_1} x_i}}{1 + e^{\boxed{\beta_0} + \boxed{\beta_1} x_i}}$$

HOW? Essentially, try different B_0 and B_1 values until “best fit” to the training data (example X and Y).

X is given. B_0 and B_1 must be **learned**.



“best fit” : whatever maximizes the likelihood function:

$$\boxed{L(\beta_0, \beta_1 | X, Y) = \prod_{i=1}^{n} p(x_i)^{y_i} (1 - p(x_i))^{1-y_i}}$$

"best fit" : whatever maximizes the likelihood function:

$$\boxed{L(\beta_0, \beta_1 | X, Y) = \prod_{i=1}^{n} p(x_i)^{y_i} (1 - p(x_i))^{1-y_i}}$$

"best fit" : whatever maximizes the *likelihood* function:

$$\boxed{L(\beta_0, \beta_1 | X, Y) = \prod_{i=1}^{n} p(x_i)^{y_i} (1 - p(x_i))^{1-y_i}}$$

"best fit" : more efficient to maximize *log likelihood* :

"best fit" : whatever maximizes the *likelihood* function:

$$\boxed{L(\beta_0, \beta_1 | X, Y) = \prod_{i=1}^{n} p(x_i)^{y_i} (1 - p(x_i))^{1-y_i}}$$

"best fit" : more efficient to maximize *log likelihood* :

$$\ell(\beta) = \sum_{i=1}^{N} y_i log\, p(x_i) + (1 - y_i) log\, (1 - p(x_i))$$

“best fit” : whatever maximizes the *likelihood* function:

$$\boxed{L(\beta_0, \beta_1 | X, Y) = \prod_{i=1}^{n} p(x_i)^{y_i}(1 - p(x_i))^{1-y_i}}$$

“best fit” : more efficient to maximize *log likelihood* :

$$\ell(\beta) = \sum_{i=1}^{N} y_i log\, p(x_i) + (1 - y_i) log\, (1 - p(x_i))$$

“best fit” for neural networks: software designed to **minimize** rather than maximize (typically, normalized by N, the number of examples.)

“best fit” : whatever maximizes the *likelihood* function:

$$\boxed{L(\beta_0, \beta_1 | X, Y) = \prod_{i=1}^{n} p(x_i)^{y_i} (1 - p(x_i))^{1-y_i}}$$

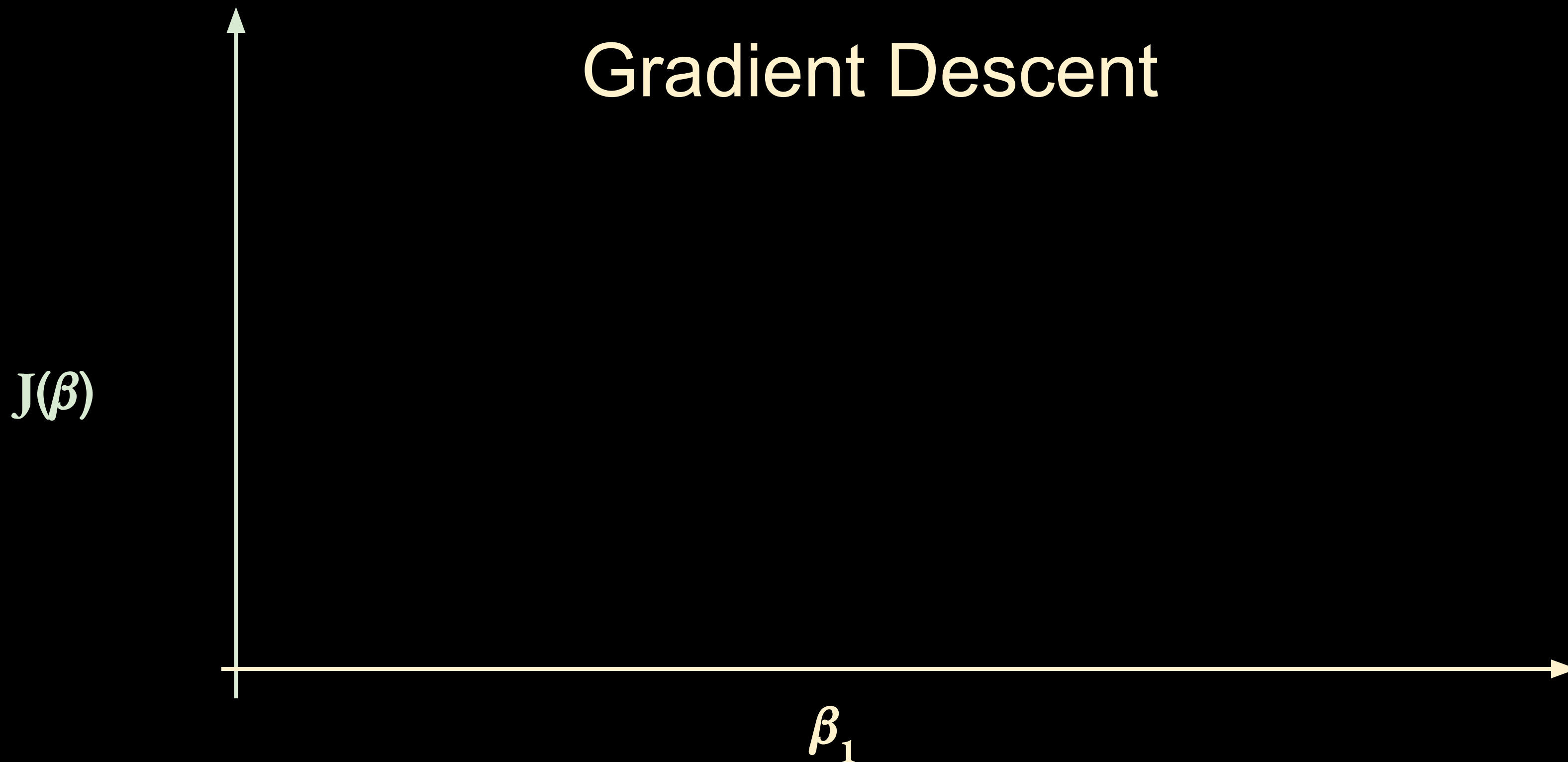
“best fit” : more efficient to maximize *log likelihood* :

$$\ell(\beta) = \sum_{i=1}^{N} y_i log\, p(x_i) + (1 - y_i) log\, (1 - p(x_i))$$

“best fit” for neural networks: software designed to **minimize** rather than maximize (typically, normalized by N, number of examples.) "*log loss*" or *"normalized log loss":*

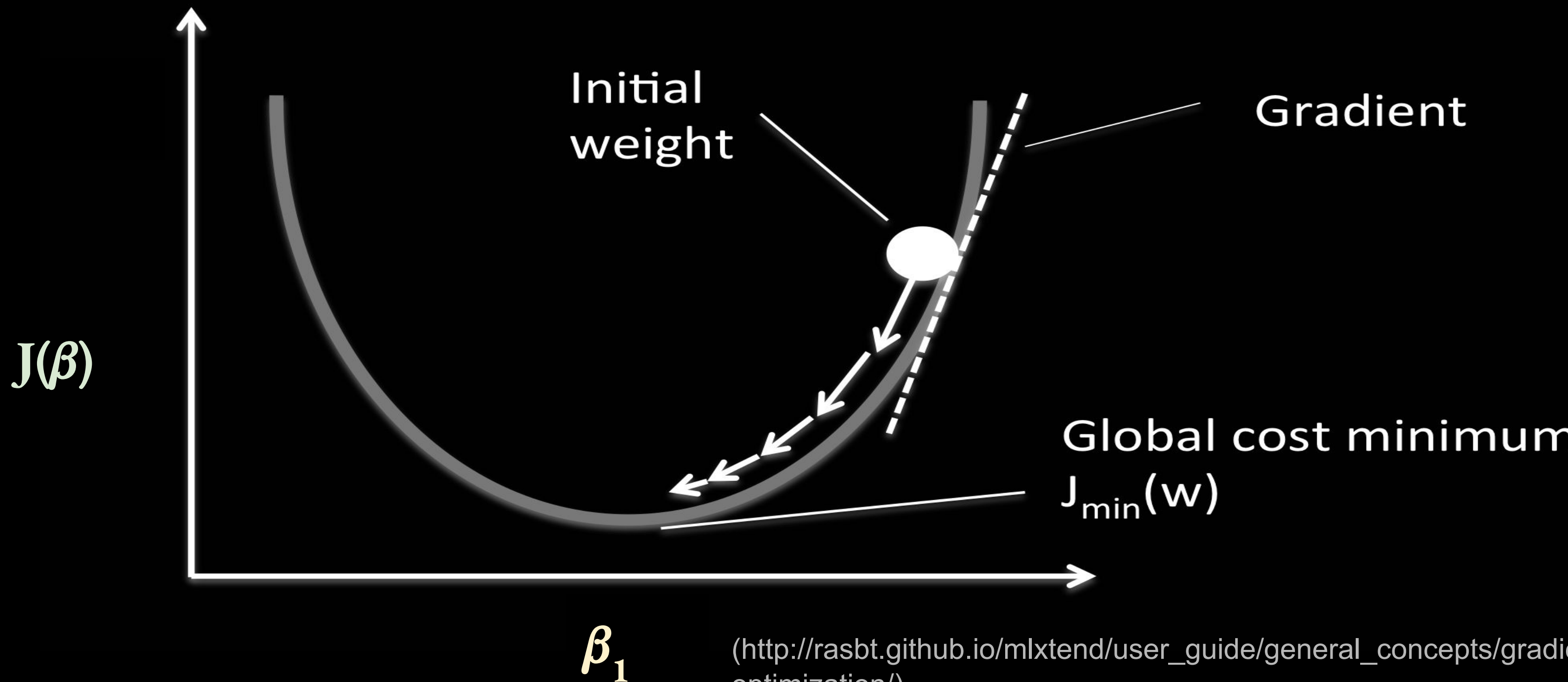
$$\boxed{J(\beta) = -\frac{1}{N} \sum_{i=1}^{N} y_i log\, p(x_i) + (1 - y_i) log\, (1 - p)(x_i))}$$

Gradient Descent



"log loss" or *"normalized log loss":*

$$\boxed{J(\beta) = -\frac{1}{N}\sum_{i=1}^{N} y_i log\ p(x_i) + (1 - y_i) log\ (1 - p)(x_i))}$$



(http://rasbt.github.io/mlxtend/user_guide/general_concepts/gradient-optimization/)

"log loss" or *"normalized log loss":*

$$\boxed{J(\beta) = -\frac{1}{N}\sum_{i=1}^{N} y_i log\ p(x_i) + (1 - y_i) log\ (1 - p)(x_i))}$$