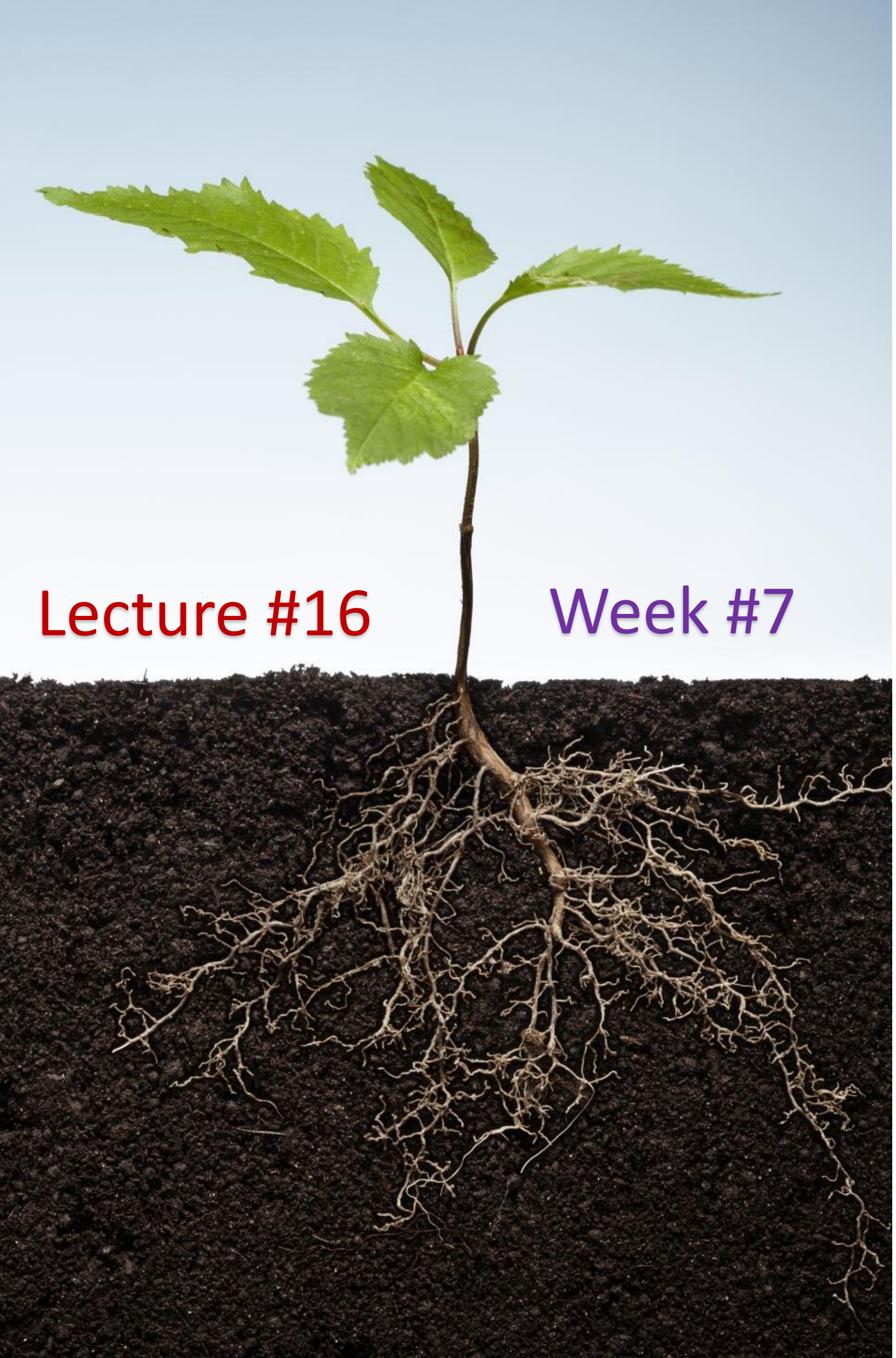



# Pattern Recognition

## ECSE 4410/6410 CAPA

## Spring 2021

# Logistic Regression +

Course Instructor - Thirimachos Bourlai

January to May 2021

Please Check the 2021 Syllabus

OVERVIEW

# Logistic Regression +

- The many names and terms used when describing logistic regression (like log odds and logit)
- The representation used for a logistic regression model
- Techniques used to learn the coefficients of a logistic regression model from data
- How to make predictions using a learned logistic regression model
- How to calculate the logistic function
- How to learn the coefficients for a logistic regression model using stochastic GD
- How to make predictions using a logistic regression model

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

# Logistic Function

- **Logistic regression** is <u>named for the function used</u> at the core of the method → the **logistic function**
- **Logistic function**
  - **Also called the Sigmoid function**
  - Was developed by statisticians to *describe properties of population growth* in ecology
  - **S-shaped curve:**
    - **Input:** any real-valued number
    - **Output**: map it into a value between 0 and 1 - *never exactly at those limits*

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

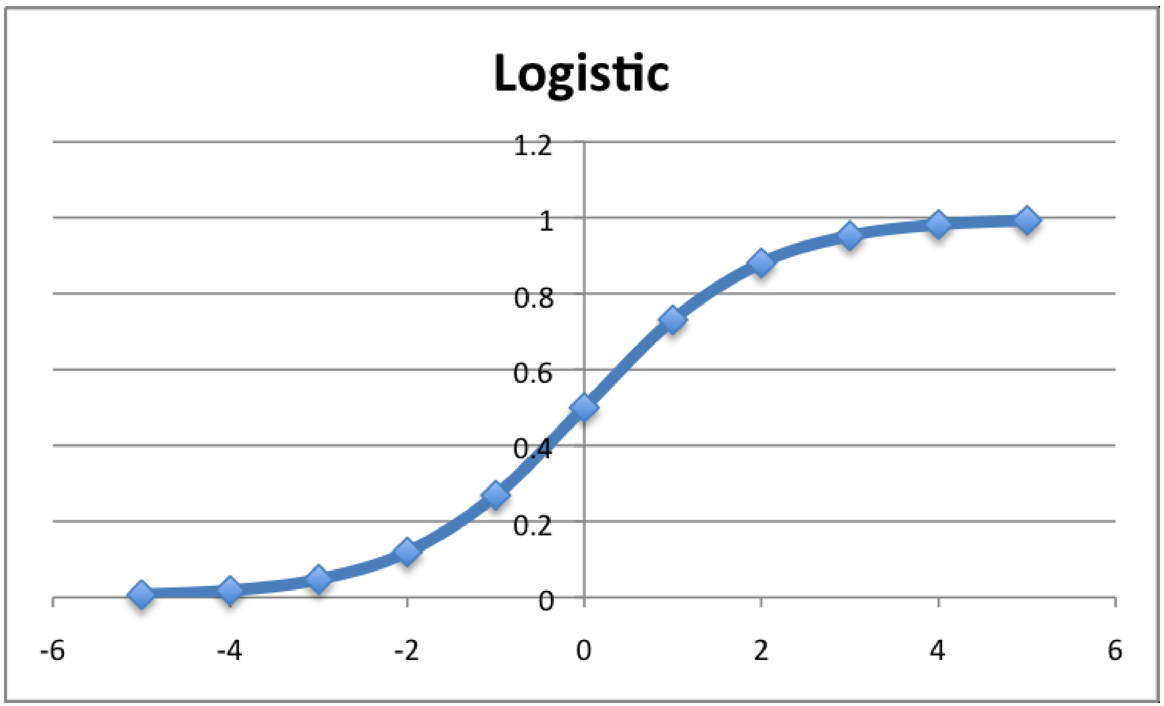
Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

# Logistic Function

$$\frac{1}{1 + e^{-value}}$$

# Logistic Function

- **E** = base of the natural logarithms (Euler's number or the EXP() function in your spreadsheet)
- **Value =** the numerical value that we want to transform




$$\frac{1}{1+e^{-value}}$$

Below is a plot of the numbers between -5 and 5 transformed into the range 0 and 1 using the logistic function.

# Logistic Regression Representation

- Logistic regression uses an equation as the representation, very much like linear regression.
- Input values (x) are combined linearly using weights or coefficient values to predict an output value (y).
- Key difference between LR and LogR:
  - The output value being modeled is a binary value (0 or 1) rather than a numeric value.

# Logistic Function

Here is an **example logistic regression equation**:

$$y = \frac{e^{B0+B1\times x}}{1+e^{B0+B1\times x}}$$

- y = the predicted output
- B0 = bias or intercept term
- B1 = coefficient for the single input value (x)

- **Each column in your input data →** has an associated B coefficient (a constant real value) that must be learned from your training data
- Th **model representation** that we would **store** (in memory or in a file) are the **coefficients in the equation** (the **beta value** or B's).

# Logistic Function: Predicting Probabilities

- **Logistic regression** <u>models the probability of the default class</u> (e.g. the first class)

For example:

- if we are modeling people’s gender as male or female from their height, then:
  - **Class 1**: **male** and
  - The **LogR model** could be written as the **probability of male** <u>given</u> **a person's height**, or more formally:

P(GERNDER= Male / Height)

# Logistic Function: Predicting Probabilities

The **LogR model** could be written as the **probability of male** given **a person's height**, or more formally:

$$P(GERNDER = Male / Height)$$

Written another way: we are modeling the probability that an input (X) belongs to the default class (Y = 1)

$$P(X) = P(Y = 1 / X)$$

- **Question**: Are we predicting probabilities?
- **Answer**: the probability prediction **must be transformed** into a binary values (0 or 1) in order **to make a crisp prediction**.

# Logistic Function

- Logistic regression is a linear method
- However, the predictions are transformed using the logistic function.
- Thus, we can no longer understand the predictions as a linear combination of the inputs as we can with linear regression

for example, the model

$$P(X) = P(Y = 1 / X)$$

Can be stated as:

$$p(X) = \frac{e^{B0+B1\times X}}{1 + e^{B0+B1\times X}}$$

Next, we can remove the **e** from the right side by adding a ln() to the other:

$$ln(\frac{p(X)}{1 - p(X)}) = B0 + B1 \times X$$

- **INPUT**: on the left is a natural logarithm of the probability of the default class
- **OUTPUT**: The calculation is linear again (just like linear regression)

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

# Logistic Function

- This ratio on the left is called the odds of the default class
- Odds are calculated as a ratio of the probability of the event divided by the probability of not the event, e.g. 0.8/[1-0.8] which has the odds of 4.

We can re-write:

$$ln(\frac{p(X)}{1-p(X)}) = B0 + B1 \times X$$

$$ln(odds) = B0 + B1 \times X$$

$$ln(odds) = B0 + B1 \times X$$

$$odds = e^{B0+B1\times X}$$

# Learning the Logistic Regression Model

- The **coefficients** of the LogR algorithm must be **estimated** from the **training data**
- This is done **using** the maximum-likelihood estimation (**MLE**)
- MLE = a **common learning algorithm** used by a variety of ML algorithms
  - Downside: it does make assumptions about the distribution of our data

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

# Logistic Function

- The best coefficients would result in a model that would predict a value very close to 1 (e.g. male) for the default class and a value very close to 0 (e.g. female) for the other class

- The intuition for MLE for LogR
  - A search procedure seeks values for the coefficients that minimize the error in the probabilities predicted by the model to those in the data (e.g. probability of 1 if the data is the primary class).

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

# Logistic Function

- We will talk about MLE later in more details
- At this time we can say that this minimization algorithm is used to optimize the best values for the coefficients for your training data
- This is often implemented in practice using efficient numerical optimization algorithm (Quasi-newton method)

# Logistic Regression: Making Predictions

- Making predictions with a LogR model is very simple
- Its like plugging in # into the LogR equation and calculating a result

**Example:**

- Say we have a model that **predicts** *Male vs. Female* based on their **height**
- If height (H) = 150 cm is the person male or female?
- We have learned the coefficients of:

B0 = -100 and B1 = 0.6.

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

# Logistic Regression: Making Predictions

- Using the equation @Slide 10 we calculate the probability of Male given a H>=150cm as

P(male / height=150)

- We will use EXP(), because that is what you can use if you type this example into your spreadsheet:

$$y = \frac{e^{B0+B1\times X}}{1+e^{B0+B1\times X}}$$

$$y = \frac{EXP(-100+0.6\times 150)}{1+EXP(-100+0.6\times 150)}$$

$$y = 0.0000453978687$$

Near 0 probability to be a male

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

# Logistic Regression: Making Predictions

**In practice** we can use the probabilities directly

Because this is **classification** and we want a crisp answer, we **can snap the probabilities** to a binary class value, for example:

- **prediction = 0 ---** IF p(male) < 0.5
- **prediction = 1 ---** IF p(male) >= 0.5

Thus, now that we know how to make predictions using LogR

NEXT: how we can prepare our data in LogR?

# Logistic Regression: Preparing Data

- The assumptions made by LogR about the distribution and relationships in data → same to the assumptions made in LR Much study has gone into defining these assumptions and precise probabilistic and statistical language is used
- In predictive modeling ML projects, we are not as focused on making accurate predictions but more focused on interpreting the results
- Thus, we can break some assumptions if the model is robust and performs well

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

# Logistic Regression: Preparing Data

**#1 Binary Output Variable**

- LogR is intended for binary (2-class) classification problems
- LogR → predicts the probability of an instance belonging to the class #1 (0), or class #2 (1)

**#2 Remove Noise**

- LogR assumes no error in the output variable (y)
- Consider removing outliers
- Consider removing possibly misclassified instances from your training data

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

# Logistic Regression: Preparing Data

**#3 Gaussian Distribution**

- LogR is a **linear algorithm** (with a **nonlinear transform** on output)
- It **assumes a linear I/O relationship**. i.e. between the input variables with the output
- **Data transforms** of our **input variables** that <u>better expose this linear relationship</u> *can result in a more accurate model*
  - E.g. we can use **log**, **root**, **Box-Cox** and *other univariate transforms* to *better expose this relationship*

# Logistic Regression: Preparing Data

**#4 Remove Correlated Inputs**

- LogR models can overt if you have multiple highly-correlated inputs (just like LR).
- **Calculating the pairwise correlations** between all inputs and **removing highly correlated inputs** can be beneficial

**#5 Fail to Converge**

- It is possible for the expected likelihood estimation process that learns the coefficients to fail to converge. This can happen if there are many highly correlated input data or very sparse data (e.g. when having lots of zeros in our input data).

# Logistic Regression – Key Learning Points

SUMMARY

- What the logistic function is and how it is used in LogR
- That the key representation in LogR are the coefficients (same in LR)
- That LogR coefficients are estimated using a process called MLE. Can use Gradient Descent!
- That making predictions using LogR is so easy that you can do it in a spreadsheet
- Data preparation for LogR is like LR

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

# Logistic Regression Variables

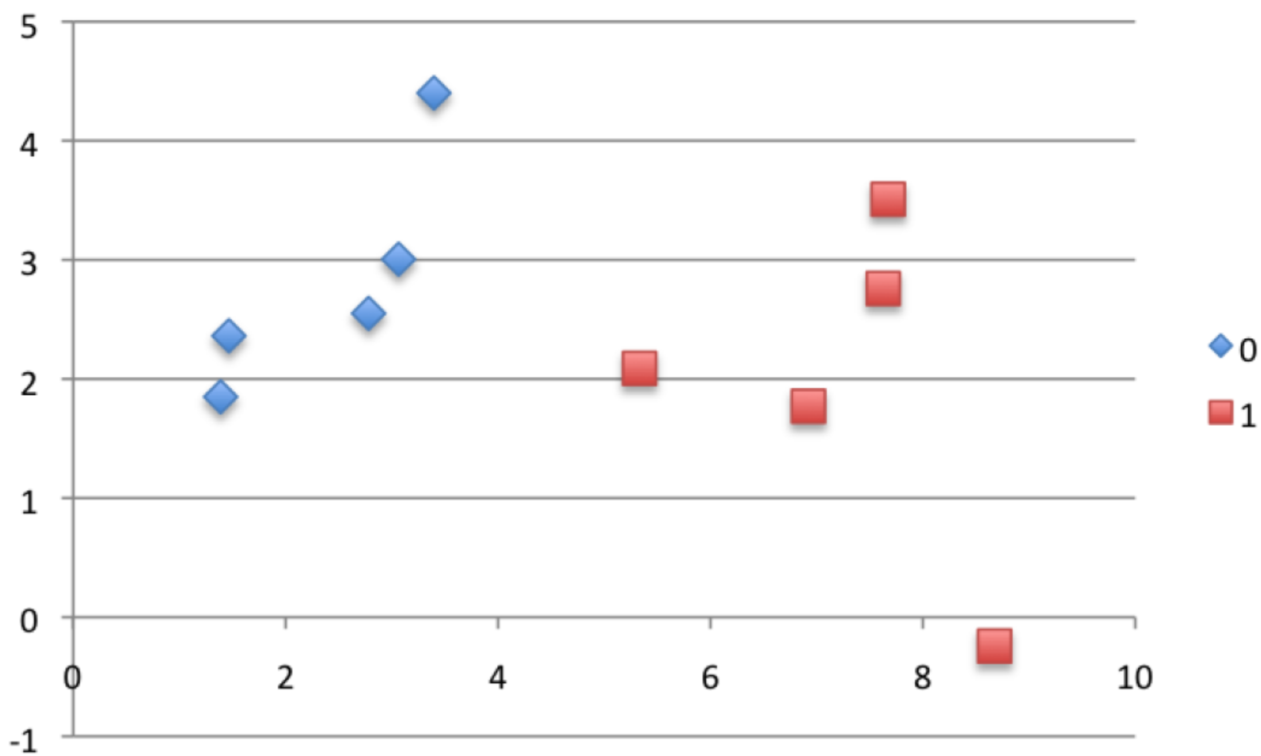
- DS1: two input variables (X1 and X2); one output variable (Y )
- X1/X2: real-valued random numbers drawn from a Gaussian distribution
- Y: has two values (binary classification problem

| X1 | X2 | Y |
|---|---|---|
| 2.7810836 | 2.550537003 | 0 |
| 1.465489372 | 2.362125076 | 0 |
| 3.396561688 | 4.400293529 | 0 |
| 1.38807019 | 1.850220317 | 0 |
| 3.06407232 | 3.005305973 | 0 |
| 7.627531214 | 2.759262235 | 1 |
| 5.332441248 | 2.088626775 | 1 |
| 6.922596716 | 1.77106367 | 1 |
| 8.675418651 | -0.242068655 | 1 |
| 7.673756466 | 3.508563011 | 1 |

Source: ML-Mastery – Master Machine Learning Algorithms
2020 by J. Brownlee

# Logistic Regression Dataset

- We can easily draw a line to separate the classes, as we did before
- This is exactly what we are going to do with the the LogR model




Logistic Regression Dataset.

# Logistic Regression Model

A LogR model:

- Inputs: real-valued numbers

- Makes a prediction: as to the **probability of the input** *belonging to the default class (class0)*.

- If the probability is **greater than 0.5** we can take the output as a prediction for the default class (**class0**)
- If otherwise the **prediction** is for the other class (**class1**).

# Logistic Regression Model

For this dataset, the LogR has three coefficients just like LR, for example:

$$output = B0 + B1\ X1 + B2\ X2$$

| X1 | X2 | Y |
|---|---|---|
| 2.7810836 | 2.550537003 | 0 |
| 1.465489372 | 2.362125076 | 0 |
| 3.396561688 | 4.400293529 | 0 |
| 1.38807019 | 1.850220317 | 0 |
| 3.06407232 | 3.005305973 | 0 |
| 7.627531214 | 2.759262235 | 1 |
| 5.332441248 | 2.088626775 | 1 |
| 6.922596716 | 1.77106367 | 1 |
| 8.675418651 | -0.242068655 | 1 |
| 7.673756466 | 3.508563011 | 1 |

The ML algorithm will be to **discover the values** for the coefficients (B0, B1 and B2) based on the training data

# Logistic Regression Model

Unlike linear regression, the **output** is **transformed** into a **probability** using the **logistic function**:

$$p(class = 0) = \frac{1}{1 + e^{-output}}$$

OR

$$p(class = 0) = \frac{1}{1 + EXP(-output)}$$

# Logistic Regression by Stochastic GD

The coefficients values of the LogR can be estimated using stochastic GR

**STEP 1 – Calculate the Prediction**

- Assign 0.0 to each coefficient
- Calculating the probability of the first training instance that belongs to **class 0 (default class)**.

$$B0 = 0.0$$
$$B1 = 0.0$$
$$B2 = 0.0$$

- The 1st training instance is:
  - X1 = 2.78, X2 = 2.55, Y = 0.0
- Using the equation $p(class = 0) = \frac{1}{1 + e^{-output}}$ **output = B0 + B1 X1 + B2 X2**

  we can plug in all these numbers and calculate a prediction

$$prediction = \frac{1}{1 + e^{-(B0+B1\times X1+B2\times X2)}}$$

$$prediction = \frac{1}{1 + e^{-(0.0+0.0\times 2.7810836+0.0\times 2.550537003)}}$$

$$prediction = 0.5$$

# Logistic Regression by Stochastic GD

The coefficients values of the LogR can be estimated using stochastic GR

## STEP 2 – Calculate New Coefficients

$$b = b + alpha \times (y - prediction) \times prediction \times (1 - prediction) \times x$$

- $b$ = the coefficient we are updating
- **prediction** = output when using the model (initiate 0.5)
- *alpha* = learning rate (to be selected at the beginning) ➔ controls how much the coefficients (and therefore the model) changes or learns each time it is updated
    - *alpha* (in this example) is set to be large, i.e., 0.3
- $X$ = input
- $Y$ = 0
- $B_0$ = bias (has no value = 1)

| X1 | X2 | Y |
|---|---|---|
| 2.7810836 | 2.550537003 | 0 |
| 1.465489372 | 2.362125076 | 0 |
| 3.396561688 | 4.400293529 | 0 |
| 1.38807019 | 1.850220317 | 0 |
| 3.06407232 | 3.005305973 | 0 |
| 7.627531214 | 2.759262235 | 1 |
| 5.332441248 | 2.088626775 | 1 |
| 6.922596716 | 1.77106367 | 1 |
| 8.675418651 | -0.242068655 | 1 |
| 7.673756466 | 3.508563011 | 1 |

$$B0 = B0 + 0.3 \times (0 - 0.5) \times 0.5 \times (1 - 0.5) \times 1.0$$
$$B1 = B1 + 0.3 \times (0 - 0.5) \times 0.5 \times (1 - 0.5) \times 2.7810836$$
$$B2 = B2 + 0.3 \times (0 - 0.5) \times 0.5 \times (1 - 0.5) \times 2.550537003$$

$$\Rightarrow$$

$$B0 = -0.0375$$
$$B1 = -0.104290635$$
$$B2 = -0.095645138$$

# Logistic Regression by Stochastic GD

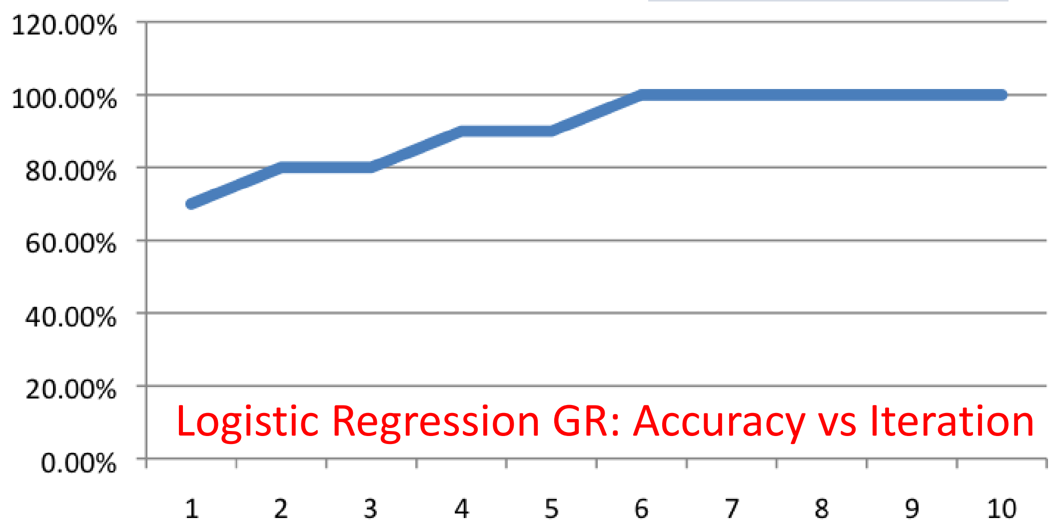
## STEP 3 – Repeat Process

- We repeat the process → update the model <u>for each training instance</u> in the dataset
- A single iteration through the training dataset is called an epoch
- It is common to repeat the Stochastic GD procedure for a fixed number of epochs
- At the end of each epoch, we calculate the error values for the model
- This is a classification problem → need to determine accuracy per iteration

The coefficients calculated after 10 epochs of Stochastic GD are:

$$B0 = -0.406605464$$
$$B1 = 0.852573316$$
$$B2 = -1.104746259$$

**Accuracy** *On the Training Set*



# Logistic Regression by Stochastic GD

## STEP 4 – Make Predictions

- The model is trained on the original/given data
  - We calculated the coefficients after 10 epochs
- Make predictions for each training instance

| X1 | X2 | Prediction |
|---|---|---|
| 2.7810836 | 2.550537003 | 0.298756986 |
| 1.465489372 | 2.362125076 | 0.145951056 |
| 3.396561688 | 4.400293529 | 0.085333265 |
| 1.38807019 | 1.850220317 | 0.219737314 |
| 3.06407232 | 3.005305973 | 0.247059 |
| 7.627531214 | 2.759262235 | 0.954702135 |
| 5.332441248 | 2.088626775 | 0.862034191 |
| 6.922596716 | 1.77106367 | 0.971772905 |
| 8.675418651 | -0.242068655 | 0.999295452 |
| 7.673756466 | 3.508563011 | 0.905489323 |

$$prediction = \text{IF } (output < 0.5) \text{ Then } 0 \text{ Else } 1$$

| Crisp |
|---|
| 0 |
| 0 |
| 0 |
| 0 |
| 0 |
| 1 |
| 1 |
| 1 |
| 1 |
| 1 |

$$accuracy = \frac{CorrectPredictions}{TotalPredictions} \times 100$$

$$accuracy = \frac{10}{10} \times 100$$

$$accuracy = 100\%$$

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

# Logistic Regression - Matlab

% returns a matrix, B, of coefficient estimates for a multinomial logistic regression of the nominal responses in Y on the predictors in X.
**B = mnrfit(X,Y)**

% The column vector, species, consists of iris flowers of **three** different **species**, setosa, versicolor, virginica. **The double matrix <u>meas</u>** consists of **four types of measurements on the flowers**, the length and width of sepals and petals in centimeters, respectively.
**load fisheriris**

% Define the nominal response variable using a categorical array.
**sp = categorical(species);**

% Fit a multinomial regression model to predict the species using the measurements.
**[B, dev, stats] = mnrfit(meas,sp);**
**B**

# Logistic Regression Example in R

https://www.machinelearningplus.com/machine-learning/logistic-regression-tutorial-examples-r/

# What have we learned

- We discovered how you can implement step-by-step LogR
- How to calculate the logistic function
- How to learn the coefficients for a LogR model using Stochastic GD
- How to make predictions using a LogR model

Source: ML-Mastery – Master Machine Learning Algorithms 2020 by J. Brownlee

# Questions?

**THANK YOU!**