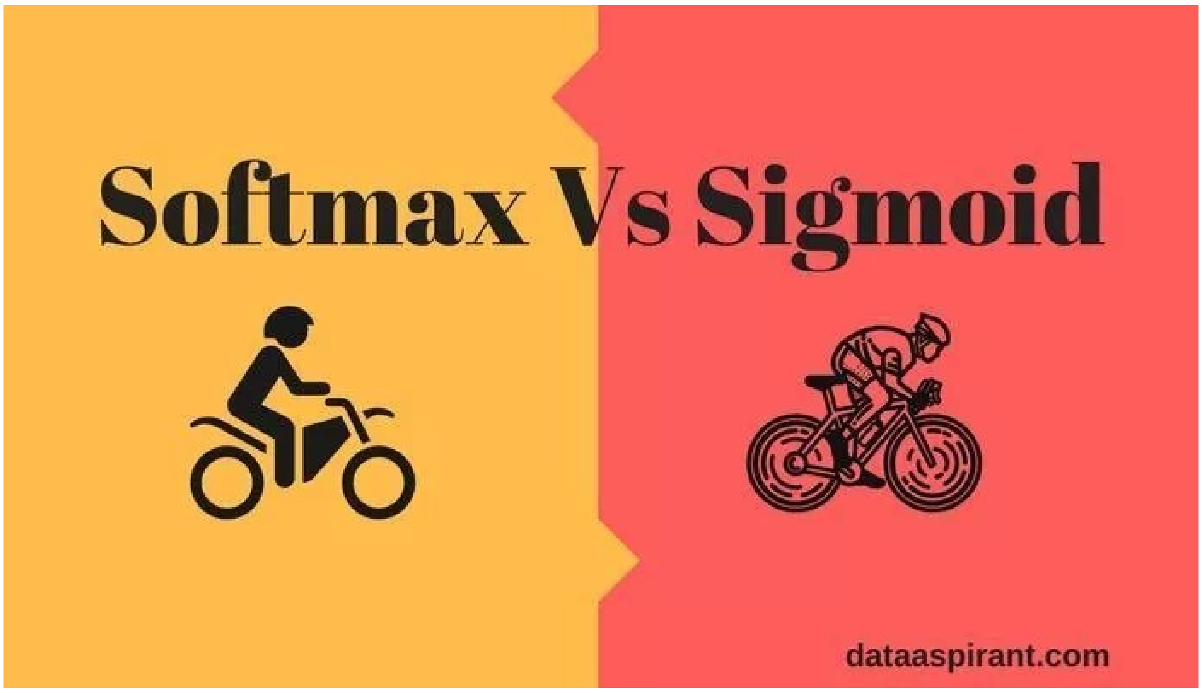
DIFFERENCE BETWEEN SOFTMAX FUNCTION AND SIGMOID FUNCTION

March 7, 2017 Saimadhu Polamuri 11 Comments Machine Learning



Softmax Function Vs Sigmoid Function

Softmax Function Vs Sigmoid Function

While learning the logistic regression concepts, the primary confusion will be on the functions used for calculating the probabilities. As the calculated probabilities are used to predict the target class in logistic regression model. The two principal functions we frequently hear are Softmax and Sigmoid function.

Even though both the functions are same at the **functional level.** (Helping to predict the target class) many noticeable mathematical differences are playing the vital role in using the functions in deep learning and other fields of areas.

So In this article, we were going to learn more about the fundamental differences between these two function and the usages.

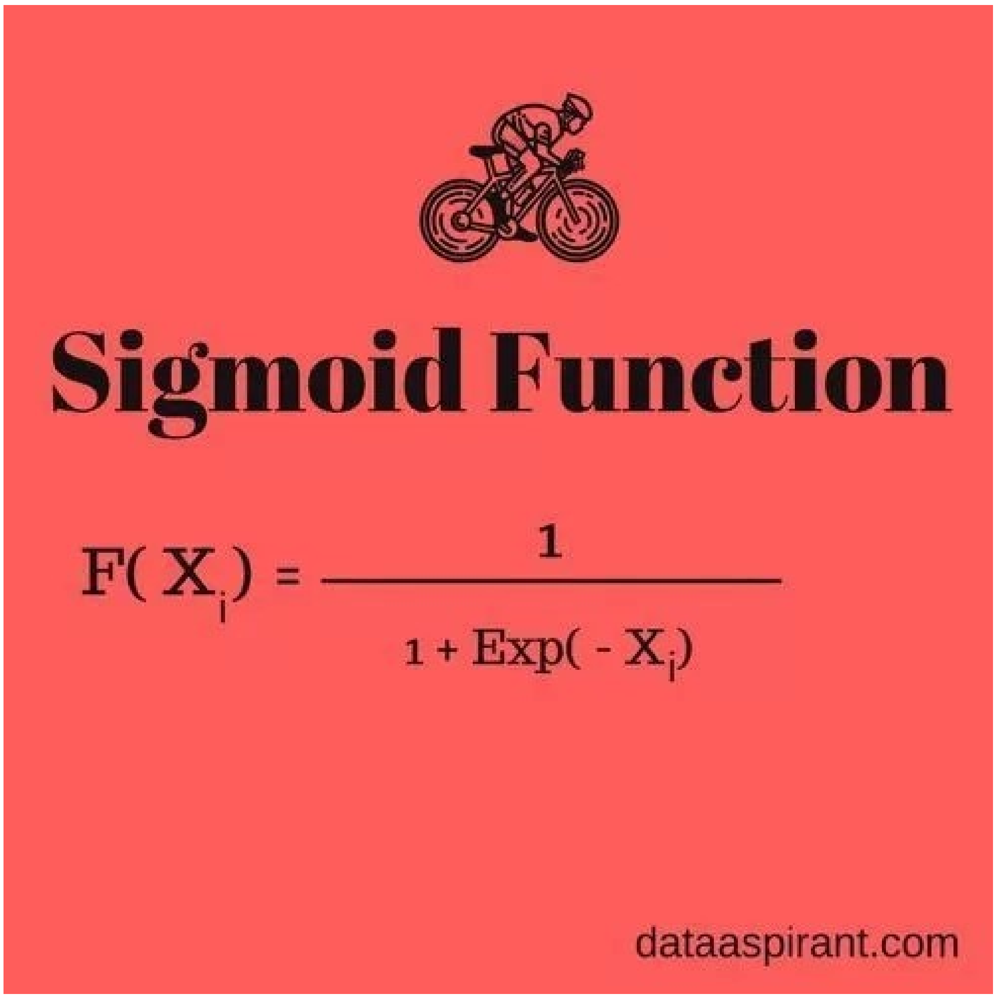
Before we begin, let's quickly look at the table of contents.

Table of Contents:

Sigmoid Function Vs Softmax Function #machinelearning

CLICK TO TWEET

What is Sigmoid Function?



Sigmoid Function

In mathematical definition way of saying the sigmoid function take any range real number and returns the output value which falls in the range of **0 to 1.** Based on the convention we can expect the output value in the range of **-1 to 1.**

The sigmoid function produces the curve which will be in the Shape **"S."** These curves used in the statistics too. With the cumulative distribution function (The output will range from 0 to 1)

Properties of Sigmoid Function

- The sigmoid function returns a real-valued output.
- The first derivative of the sigmoid function will be non-negative or non-positive.
    - **Non-Negative:** If a number is greater than or equal to zero.
    - **Non-Positive:** If a number is less than or equal to Zero.

Sigmoid Function Usage

- The Sigmoid function used for **binary classification** in logistic regression model.
- While creating artificial neurons sigmoid function used as the **activation function**.
- In statistics, the **sigmoid function graphs** are common as a cumulative distribution function.

Implementing Sigmoid Function In Python

Now let's implement the sigmoid function in Python

```python
# Required Python Package
import numpy as np

def sigmoid(inputs):
    """
    Calculate the sigmoid for the give inputs (array)
    :param inputs:
    :return:
    """
    sigmoid_scores = [1 / float(1 + np.exp(- x)) for x in inputs]
    return sigmoid_scores


sigmoid_inputs = [2, 3, 5, 6]
print "Sigmoid Function Output :: {}".format(sigmoid(sigmoid_inputs))
```

The above is the implementation of the sigmoid function.

- The function will take a list of values as an input parameter.
- For each element/value in the list will consider as an input for the sigmoid function and will calculate the output value.
- The code **1 / float(1 + np.exp(- x))** is the fucuntion is used for calcualting the sigmoid scores.
- Next, we take a list sigmiod_inputs having the values 2,3,5,6 as an input the function we implemented to get the sigmoid scores.

Script Output

```
Sigmoid Function Output :: [0.8807970779778823, 0.9525741268224334, 0.9933071490757153
```

Creating Sigmoid Function Graph

Now let’s use the above function to create the graph to understand the nature of the sigmoid function.

- We are going to pass a list which contains numbers in the range 0 to 21.
- Will compute the sigmoid scores for the list we passed.
- Then we will use the outputs values to visualize the graph.

```
# Required Python Packages
import numpy as np
import matplotlib.pyplot as plt


def sigmoid(inputs):
    """
    Calculate the sigmoid for the give inputs (array)
    :param inputs:
    :return:
    """
    sigmoid_scores = [1 / float(1 + np.exp(- x)) for x in inputs]
    return sigmoid_scores


def line_graph(x, y, x_title, y_title):
    """
    Draw line graph with x and y values
    :param x:
    :param y:
    :param x_title:
    :param y_title:
    :return:
    """
    plt.plot(x, y)
    plt.xlabel(x_title)
    plt.ylabel(y_title)
    plt.show()


graph_x = range(0, 21)
graph_y = sigmoid(graph_x)

print "Graph X readings: {}".format(graph_x)
print "Graph Y readings: {}".format(graph_y)

line_graph(graph_x, graph_y, "Inputs", "Sigmoid Scores")
```

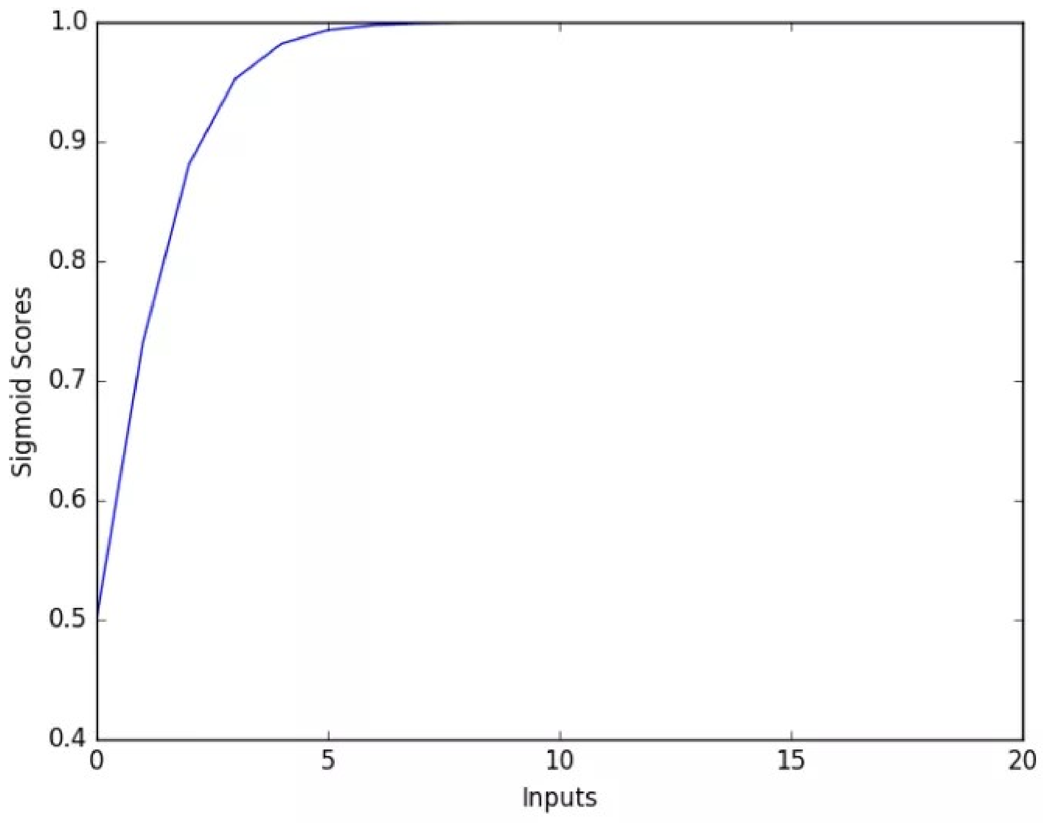
- Creating a **graph_x** list which contains the numbers in the range of 0 to 21.
- Next, in the **graph_y** list, we are storing the calculated **sigmoid scores** for the given graph_x inputs.
- Calling the **line_graph** function, which takes the x, y, and titles of the graph to create the line graph.

Script Output

```
Graph X readings: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 1

Graph Y readings: [0.5, 0.7310585786300049, 0.8807970779778823, 0.9525741268224334, 0.
```

Graph

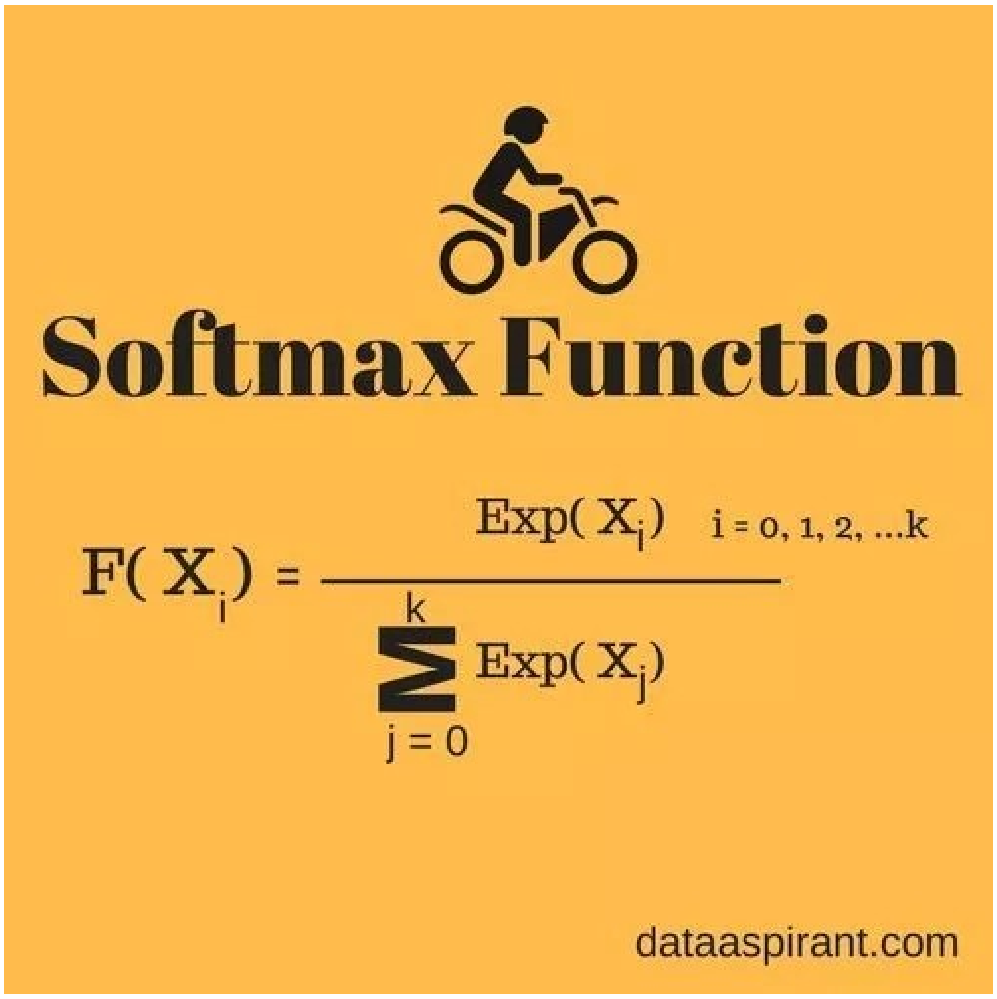
On successfully running the above code the below image will appear on your screen. If the above code failed in your system. Check the machine learning packages setup.



Sigmoid graph

From the above graph, we can observe that with the increase in the input value the sigmoid score increase **till 1**. The values which are touching at the top of the graph are the values in the range of **0.9 to 0.99**

What is Softmax Function?


Softmax Function
$$F(X_i) = \frac{Exp(X_i)}{\sum_{j=0}^{k} Exp(X_j)} \quad i = 0, 1, 2, ...k$$
dataaspirant.com

Softmax Function

Softmax function calculates the probabilities distribution of the event over 'n' different events. In general way of saying, this function will calculate the probabilities of each target class over all possible target classes. Later the calculated probabilities will be helpful for determining the target class for the given inputs.

The main advantage of using Softmax is the output probabilities range. The range will **0 to 1**, and the sum of all the probabilities will be **equal to one**. If the softmax function used for multi-classification model it returns the probabilities of each class and the target class will have the high probability.

The formula computes the **exponential (e-power)** of the given input value and the **sum of exponential values** of all the values in the inputs. Then the ratio of the exponential of the input value and the sum of exponential values is the output of the softmax function.

Properties of Softmax Function

Below are the few properties of softmax function.

- The calculated probabilities will be in the range of 0 to 1.
- The sum of all the probabilities is equals to 1.

Softmax Function Usage

- Used in multiple classification logistic regression model.
- In building neural networks softmax functions used in different layer level.

Implementing Softmax Function In Python

Now let’s implement the softmax function in Python

```python
# Required Python Package
import numpy as np


def softmax(inputs):
    """
    Calculate the softmax for the give inputs (array)
    :param inputs:
    :return:
    """
    return np.exp(inputs) / float(sum(np.exp(inputs)))


softmax_inputs = [2, 3, 5, 6]
print "Softmax Function Output :: {}".format(softmax(softmax_inputs))
```

Script Output

```
Softmax Function Output :: [ 0.01275478  0.03467109  0.25618664  0.69638749]
```

If we observe the function output for the input value 6 we are getting the high probabilities. This is what we can expect from the softmax function. Later in classification task, we can use the high probability value for predicting the target class for the given input features.

Creating Softmax Function Graph

Now let’s use the implemented Softmax function to create the graph to understand the behavior of this function.

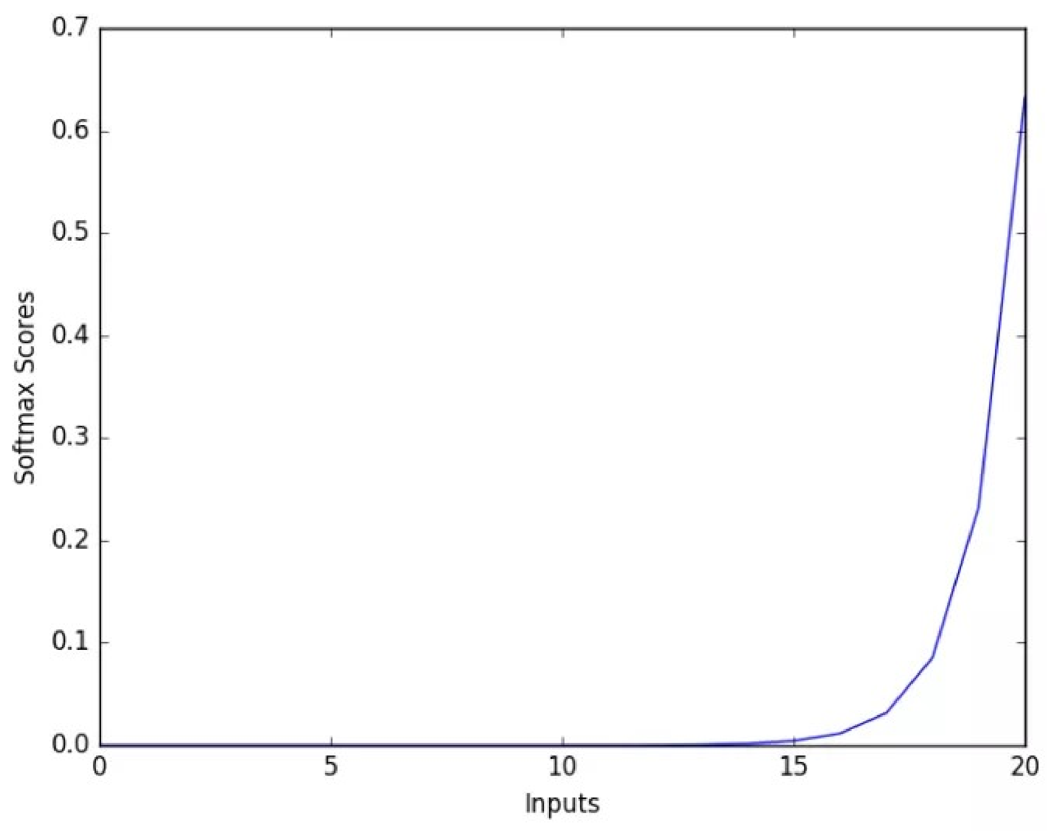
- We are going to create a list which contains values in the range of 0 to 21.
- Next, we are going to pass this list to calculate the scores from the implemented function.
- To create the graph we are going to use the list and the estimated scores.

```python
# Required Python Packages
import numpy as np
import matplotlib.pyplot as plt


def softmax(inputs):
    """
    Calculate the softmax for the give inputs (array)
    :param inputs:
    :return:
    """
    return np.exp(inputs) / float(sum(np.exp(inputs)))


def line_graph(x, y, x_title, y_title):
    """
    Draw line graph with x and y values
    :param x:
    :param y:
    :param x_title:
    :param y_title:
    :return:
    """
    plt.plot(x, y)
    plt.xlabel(x_title)
    plt.ylabel(y_title)
    plt.show()


graph_x = range(0, 21)
graph_y = softmax(graph_x)

print "Graph X readings: {}".format(graph_x)
print "Graph Y readings: {}".format(graph_y)

line_graph(graph_x, graph_y, "Inputs", "Softmax Scores")
```

Script Output

```
Graph X readings: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 1
Graph Y readings: [ 1.30289758e-09 3.54164282e-09 9.62718331e-09 2.61693975e-08 7.1135
```

Graph



Softmax Graph

The figure shows the fundamental property of softmax function. The **high value will have the high probability**.

Difference Between Sigmoid Function and Softmax Function

The below are the tabular differences between Sigmoid and Softmax function.

	Softmax Function	**Sigmoid Function**
1	Used for multi-classification in logistic regression model.	Used for binary classification in logistic regression model.
2	The probabilities sum will be 1	The probabilities sum need not be 1.
3	Used in the different layers of neural networks.	Used as activation function while building neural networks.
4	The high value will have the higher probability than other values.	The high value will have the high probability but not the higher probability.

Conclusion

In this article, you learn in details about two functions which determine the logistic regression model. Just for a glance.

- **Softmax:** Used for the multi-classification task.
- **Sigmoid:** Used for the binary classification task.

Follow us:

FACEBOOK| QUORA |TWITTER| GOOGLE+ | LINKEDIN| REDDIT | FLIPBOARD | MEDIUM | GITHUB

I hope you like this post. If you have any questions, then feel free to comment below. If you want me to write on one particular topic, then do tell it to me in the comments below.

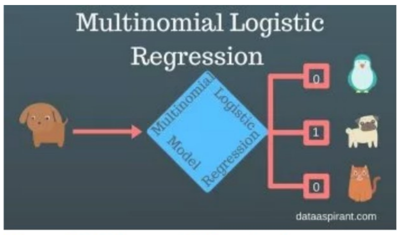
Related Data Science Courses

- Using softmax in deep learning neural networks.
- Machine learning classification concepts for beginners.
- Logistic regression model implementation with Python.
- Applying machine learning classification techniques case studies.

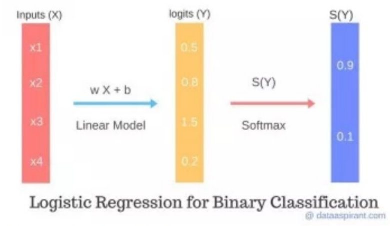
Share this:

166

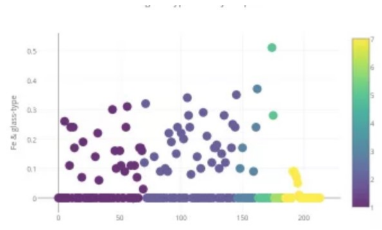
Related


Multinomial Logistic Regression
Multinomial Logistic Model Regression
0
1
0
dataaspirant.com

How Multinomial Logistic Regression Model Works In Machine Learning
March 14, 2017
In "Machine Learning"



How the logistic regression model works
March 2, 2017
In "Machine Learning"



2 Ways to Implement Multinomial Logistic Regression In Python
May 15, 2017
In "Machine Learning"

classification algorithms

logistic regression