Project 1: Logistic Regression

UBIT Name: Prashi Khurana
UBIT Number: 50316796
prashikh@buffalo.edu

Abstract

In this project, we are supposed to train and build a classification model(Logistic Regression model) that estimates whether a given instance(FNA of breast mass) of data with characteristics of cell nuclei will be malignant(M) or benign(B).

1 Introduction

In this project we are focusing on a classification called Logistic Regression. In this we are expecting the output in binary form(either 0 or 1), or the output belongs to (0,1).

1.1 What is Logistic Regression and Why is it used

The logistic regression is a predictive analysis. Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable.[2] . The purpose of logistic regression is to estimate the probabilities of events, including determining a relationship between features and the probabilities of particular outcomes[3]. For example in our case we are trying to predict if we are given a set of features, we should be able to successfully classify it under class 0 or class 1. (where 0 stands for benign and 1 stands for malignant).

Types of Logistic Regression[4]

1. Binary Logistic Regression: The categorical response has only two 2 possible outcomes. Example: Malignant or Benign

2. Multinomial Logistic Regression : Three or more categories without ordering. Example: Predicting which food is preferred more (Veg, Non-Veg, Vegan)

3. Ordinal Logistic Regression: Three or more categories with ordering. Example: Movie rating from 1 to 5

2 Dataset Definition

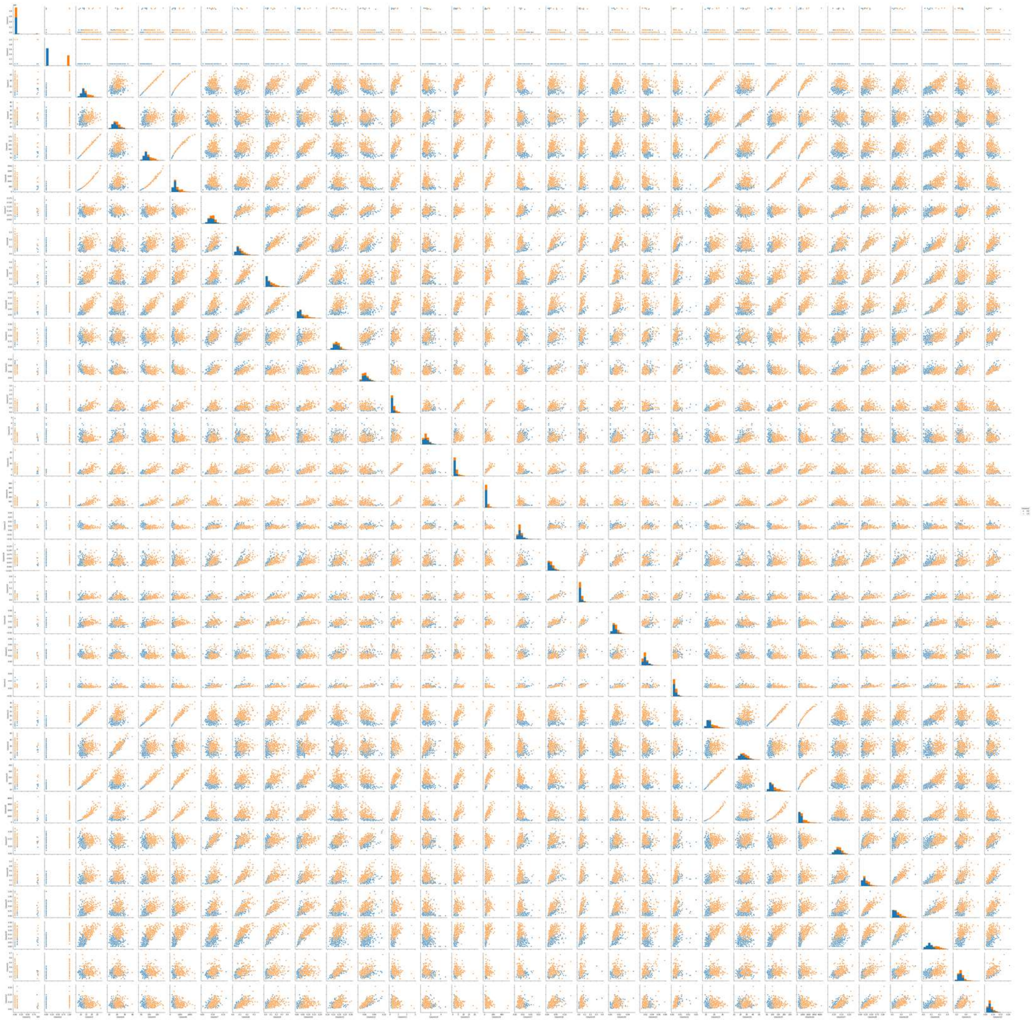
One instance of data set:
842302,M,17.99,10.38,122.8,1001,0.1184,0.2776,0.3001,0.1471,0.2419,0.07871,1.095,0.9053,8.589,153.4,0.006399,0.04904,0.05373,0.01587,0.03003,0.006193,25.38,17.33,184.6,2019,0.1622,0.6656,0.7119,0.2654,0.4601,0.1189

Here the first column is the id of the data set and the second column is the result. The rest 30 columns are the features of the data set.

The mean, standard error, and worst or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features.

1	radius (mean of distances from center to points on the perimeter)
2	texture (standard deviation of gray-scale values)
3	perimeter
4	area
5	smoothness (local variation in radius lengths)
6	compactness ($perimeter^2/area - 1.0$)
7	concavity (severity of concave portions of the contour)
8	concave points (number of concave portions of the contour)
9	symmetry
10	fractal dimension ("coastline approximation" - 1)

We can use a pair-plot to explain a relationship between two variables. Here the color orange and blue represent Class 0 and Class 1.



3. Pre-processing

3.1 Preprocessing the data

The data which is generally used for training models might be inconsistent, incomplete and needs pre-processing. In our project, the following pre-processing functions were required:

3.1.1 Removing Id and Result Column

The first column of the dataset had the id of the instances. These were to be removed. The second column of the dataset had the results of the instances. The results were in the form of ‘M’ and ‘B’, where ‘M’ is malignant and ‘B’ is Benign. We also need to set Malignant to ‘1’ and Benign to ‘0’.

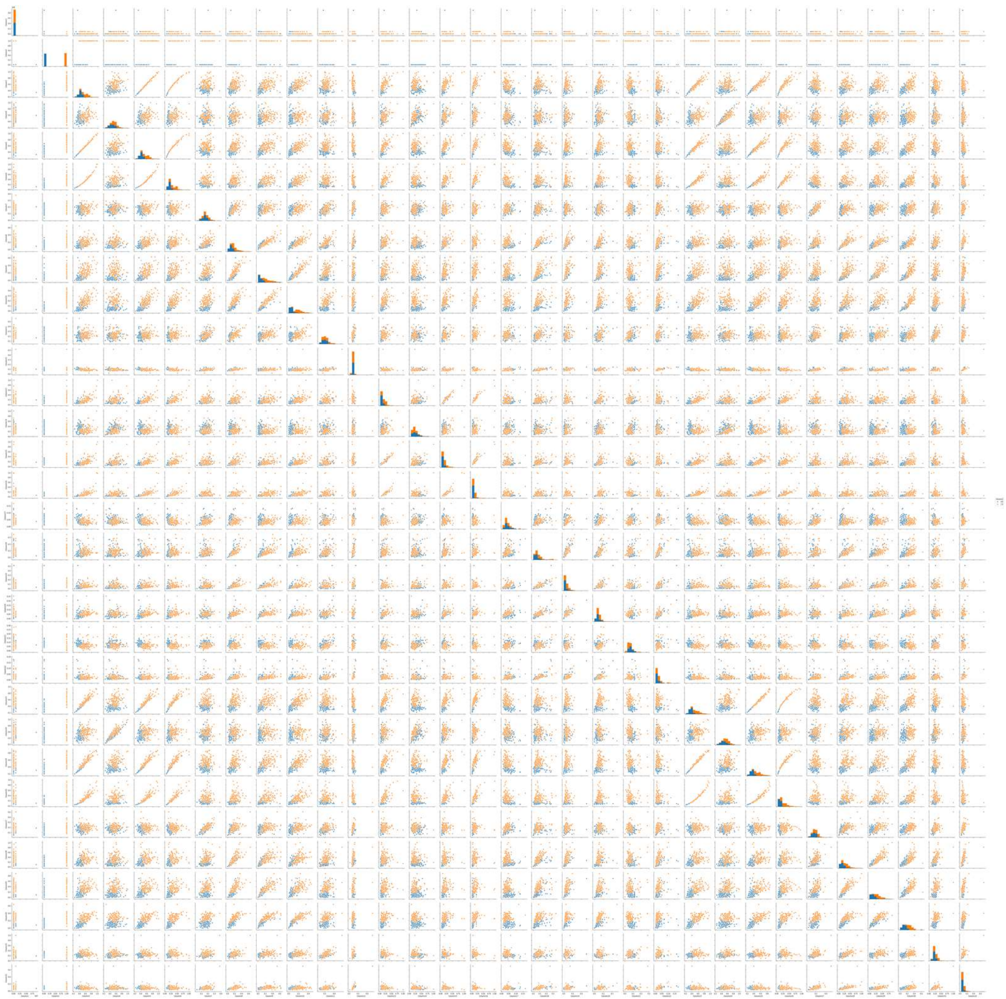
3.1.2 Divide the data Sets into 3 parts

The initial dataset is separated into 3 parts- Training set, Validation set and the Test Set. The training and validation set were 90% of the entire dataset and the last 10% is the test set.

3.1.3 Normalize the data

Normalization is an important part of the pre-processing of the data with machine learning. Normalization is the process of getting all the features of the of the instance in a common range. Gradient descent converges much faster when the features are normalized. The equation of normalization is : $x' = x - \min(x)/(\max(x)-\min(x))$

The dataset after normalization:



3.1.4 Understanding the data after pre-processing and its dimensions

Let ‘m’ be the number of rows. (Number of instances in the set)
Let ‘n’ be the number of features

Size Of X – (m x (n+1))
Size Of Y- (mx1)
Size Of Weights – ((n+1)x1)

The instance given below:
842302,M,17.99,10.38,122.8,1001,0.1184,0.2776,0.3001,0.1471,0.2419,0.07871,1.095,0.9053,8.589,153.4,0.006399,0.04904,0.05373,0.01587,0.03003,0.006193,25.38,17.33,184.6,2019,0.1622,0.6656,0.7119,0.2654,0.4601,0.1189

Can be written as :
Y1 =1

X1f1=0.34,	X1f2=0.11,	X1f3=0.18,	X1f4=0.1,
X1f5=0.1184,	X1f6=0.2776,	X1f7=0.3001,	X1f8=0.1471,
X1f9=0.2419,	X1f10=0.07871,		
X1f11=0.095,	X1f12=0.9053,	X1f13=0.589,	X1f14=0.41,
X1f15=0.006399,	X1f16=0.04904,	X1f17=0.05373,	X1f18=0.01587,
X1f19=0.03003,	X1f20=0.006193,	X1f21=0.38,	X1f22=0.33,
X1f23=0.62,	X1f24=0.1,	X1f25=0.1622,	X1f26=0.6656,
X1f27=0.7119,	X1f28=0.2654,	X1f29=0.4601,	X1f30=0.1189

$$X \quad \begin{array}{cc} & \begin{array}{cccccccc} & f_1 & f_2 & f_3 & . & . & . & f_n \end{array} \\ \begin{array}{c} x_1 \\ x_2 \\ x_3 \\ . \\ . \\ . \\ x_m \end{array} & \left[\begin{array}{cccccccc} 1 & & & & & & & \\ 1 & & & & & & & \\ 1 & & & & & & & \\ 1 & & & & & & & \\ 1 & & & & & & & \\ 1 & & & & & & & \\ 1 & & & & & & & \end{array}\right] \end{array} \qquad Y \quad \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ . \\ . \\ . \\ y_m \end{bmatrix} \qquad W \quad \begin{bmatrix} b \\ w_1 \\ w_2 \\ . \\ . \\ . \\ w_{(n+1)} \end{bmatrix}$$

4 Model

The algorithm for logistic regression includes some basic concepts and functions like the sigmoid function, the cost function and the gradient descent. First we need to understand the different concepts used in this regression.

4.1 Hyper-parameters

As discussed in section 1, logistic regression requires some initial hyper-parameters. These include *weights(w), bias(b)* and the *learning rate(alpha)*. We need to initialize these values to get started and later we will see how our choice affects the accuracy of the test set.

4.2 Sigmoid Function and Genesis Equation

Logistic regression hypothesis is defined as: $\mathbf{h_\theta(x) = g(\theta^T x)}$, where function g is the sigmoid function. The sigmoid function is defined as: $\mathbf{g(z) = 1/(1+e^{-z})}$**.** The function maps any real value into another value between 0 and 1. In machine learning, we use sigmoid to map predictions to probabilities.

4.3 Cost Function

Instead of Mean Squared Error, we use a cost function called Cross-Entropy, also known as Log Loss.

Let us assume that[5]

$P(y = 1 \mid x; \theta) = h\theta(x)$ $P(y = 0 \mid x; \theta) = 1 - h\theta(x)$

Note that this can be written more compactly as: $p(y \mid x; \theta) = (h\theta(x))^{y} (1 - h\theta(x))^{1-y}$

Assuming that the m training examples were generated independently, we can then write down the likelihood of the parameters as: [5]

$$\begin{aligned} L(\theta) &= p(\vec{y} \mid X; \theta) \\ &= \prod_{i=1}^{m} p(y^{(i)} \mid x^{(i)}; \theta) \\ &= \prod_{i=1}^{m} \left(h_\theta(x^{(i)})\right)^{y^{(i)}} \left(1 - h_\theta(x^{(i)})\right)^{1-y^{(i)}} \end{aligned}$$

As before, it will be easier to maximize the log likelihood:

$$\begin{aligned} \ell(\theta) &= \log L(\theta) \\ &= \sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1 - y^{(i)}) \log(1 - h(x^{(i)})) \end{aligned}$$

$$J(\theta) = -\frac{1}{m} \sum \left[y^{(i)} \log(h\theta(x(i))) + \left(1 - y^{(i)}\right) \log(1 - h\theta(x(i))) \right]$$

With regularization the cost function equation is :

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \left[-y^{(i)} \log(h_\theta(x^{(i)})) - (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right] + \frac{\lambda}{2m} \sum_{j=1}^{n} \theta_j^2.$$

4.4 Gradient Descent

To minimize our cost, we use Gradient Descent. Gradient Descent of a function is the algorithm to find the minimum of a function.

$$\frac{\partial J(\theta)}{\partial \theta_0} = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} \qquad \text{for } j = 0$$

$$\frac{\partial J(\theta)}{\partial \theta_j} = \left(\frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)} \right) + \frac{\lambda}{m} \theta_j \qquad \text{for } j \geq 1$$

4.5 Decision Boundary

To predict which class a data belongs, a threshold can be set. Based upon this threshold, the obtained estimated probability is classified into classes.

Say, if predicted_value ≥ 0.5, then say that the feature set has Malignant else Benign.

4.6 Confusion Matrix

	1 - Predicted	0-Predicted
1 - Actual	True Positive	False Negative
0-Actual	False Positive	True Negative

True Positive (TP) : Observation is positive, and is predicted to be positive.
False Negative (FN) : Observation is positive, but is predicted negative.
True Negative (TN) : Observation is negative, and is predicted to be negative.
False Positive (FP) : Observation is negative, but is predicted positive.[6]

4.7 Understanding Accuracy, Recall and Precision through Confusion Matrix.

4.7.1 Accuracy:

Accuracy is defined as the quality or state of being correct or precise. This means how many instances of the test data set did the logistic regression was able to classify correctly.
Once we have the confusion matrix we can define the accuracy as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

4.7.2 Precision:

Precision (also called positive predictive value) is the fraction of relevant instances among the retrieved instances. Once we have the confusion matrix we can define the precision:[9]

$$Precision = \frac{TP}{TP + FP}$$

4.7.3 Recall:

While recall (also known as sensitivity) is the fraction of relevant instances that have been retrieved over the total amount of relevant instances. Once we have the confusion matrix we can define recall as:[9]

$$Recall = \frac{TP}{TP + FN}$$

5 Actual Implementation

5.1 Architecture

An Example of a logistic regression:[10]

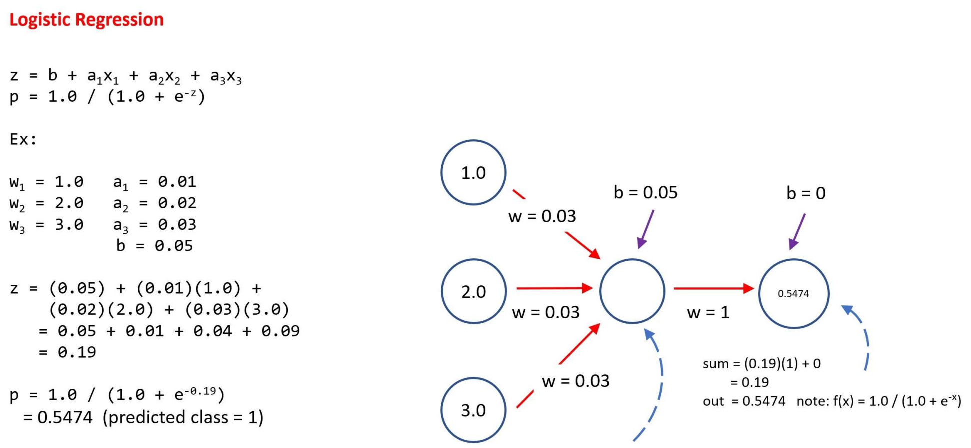


Image Source: [10]

5.2 Pseudocode for Logistic Regression

1. Pre-process the data (3)

2. Initialize the hyper-parameters (4.1)

3. Now that we have our training, validation and test data set, we can start training the data set.

3.1 *weights* = np.random.rand(features+1,1) (setting random weights in an array with bias included)

3.2 Adding a column of 1's to the instances to account for the bias (np.hstack([np.ones([xtrain.shape[0],1]), xtrain])

3.3 For different number of iterations/epochs:

3.3.1 Calculate the cost for the given number of iterations (4.3)

3.3.2 Calculate the gradient descent using formula (4.4)

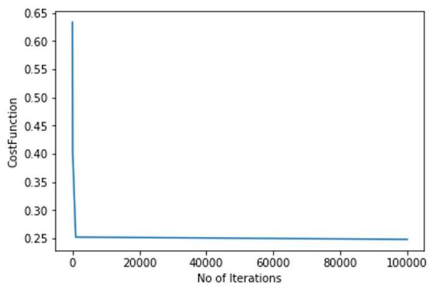
3.4 Test this on validation set (and choose the *weights,bias and learning rate* which has the maximum accuracy)

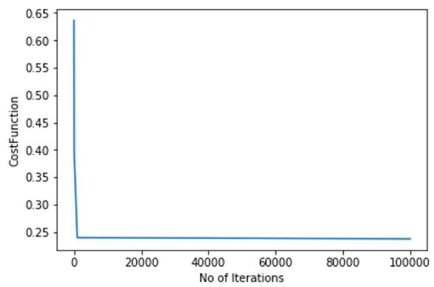
3.5 Test these parameters on the test set, and find the accuracy for the same.

6 Results

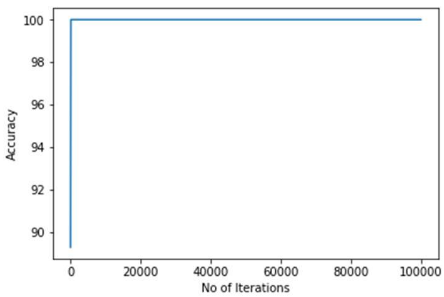
6.1 Graphs

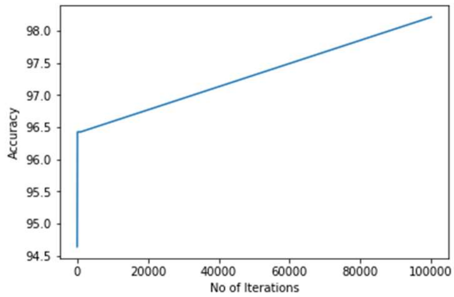
6.1.1 Cost Function(Training) Vs Iterations



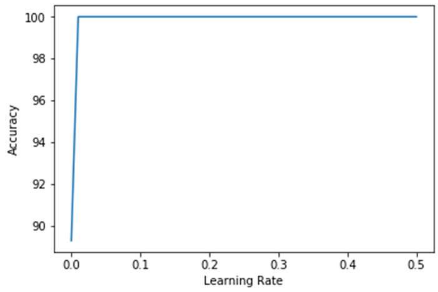


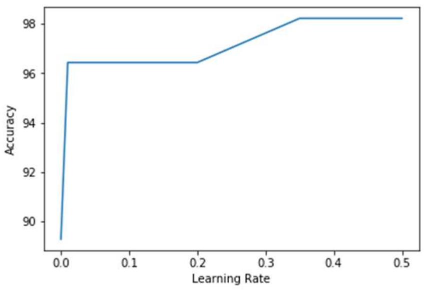
6.1.2 Iterations Vs Accuracy





6.1.3 Learning Rate Vs Accuracy





6.2 Confusion Matrix:

Set A:

	Predicted:No	Predicted:Yes
Actual: No	32	5
Actual: Yes	0	19

Set B:

	Predicted:No	Predicted:Yes
Actual: No	33	3
Actual: Yes	1	19

6.3 Accuracy,Precision And Recall:

SET A:

Accuracy : 94.64%
Precision : 0.875
Recall : 1.0

SET B:

Accuracy : 91.07%
Precision : 0.8571428571428571
Recall : 0.96

7 Conclusion

Once we have trained the logistic regression model, we need to understand the results of the model.

- Logistic regression is a simple algorithm that can be used for binary/multivariate classification tasks.[8]
- Even though we are using the same dataset to train the model, it is not necessary that we get the same accuracy each time. This is because each time we use a

different set of the training data(a different 80% each time of the entire dataset -as mentioned in 3.1.2). This is the reason we have two sets of results.

- Understanding the graphs in **6.1.1.1** and **6.1.2.1**
    - *Cost Function Vs Iteration*s:
      We calculate the cost function and plot the graph against the cost function and the number of iterations. As the number of iterations increase we see that the cost function decreases. This is because as you train the data, the weights become more accurate and the loss function or cost function decreases. Decreasing of the cost function means your trained logistic regression will have more accuracy.
    - *Iterations Vs Accuracy*:
      As we can see the graph, initially the accuracy increases with the number of iterations, but it stabilizes later and does not increase any further with the number of iterations.
    - *Learning Rate Vs Accuracy*:
      We see that the learning rate vs accuracy graph is similar to the iterations vs accuracy graph.
- Understanding Accuracy, Precision and Recall as in **6.1.1.3** and **6.1.2.3**.
    - Accuracy:
      As mentioned in 4.7.1, accuracy is a measure to show how correct is the regression model. In this case we have achieved an accuracy of 93%(approximately).
    - Precision:
      As mentioned in 4.7.2, precision is a measure of the fraction of relevant instances among the retrieved instances. In our code we have received a recall of around 0.85
    - Recall:
      As mentioned in 4.7.3 recall is a measure) is the fraction of relevant instances that have been retrieved over the total amount of relevant instances. In our code we have achieved a recall of around 0.95

- So we can conclude that logistic regression is a good algorithm to classify instances of data into 2 classes.

References

[1]https://ml-cheatsheet.readthedocs.io/en/latest/logistic_regression.html

[2] https://en.wikipedia.org/wiki/Logistic_regression#Logistic_regression_vs._other_approaches

[3] https://searchbusinessanalytics.techtarget.com/definition/logistic-regression

[4] https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc

[5] http://cs229.stanford.edu/notes/cs229-notes1.pdf

[6] https://www.geeksforgeeks.org/confusion-matrix-machine-learning/

[7]https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/logistic-regression-analysis-r/tutorial/

[8]https://hackernoon.com/introduction-to-machine-learning-algorithms-logistic-regression-cbdd82d81a36

[9] https://en.wikipedia.org/wiki/Precision_and_recall

[10]https://jamesmccaffrey.wordpress.com/2017/07/01/a-neural-network-equivalent-to-logistic-regression/