Predicting Survivors of the Titanic

Fermín Moreno

***Abstract*— In this document I present an implementation for the Titanic competition in Kaggle along with the various techniques, and methods used in the attempt to solve this problem.**
Link to GitHub repository: https://github.com/FerminAMC/cancer-detection

I. Indtroduction

This project was a submission for the Titanic competition in Kaggle.
On April 15, 1912, during her maiden voyage, the Titanic sank after colliding with an iceberg, killing 1502 out of 2224 passengers and crew.
One of the reasons that the shipwreck led to such loss of life was that there were not enough lifeboats for the passengers and crew. Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class.
The objective of this challenge was to predict, based on a passengers characteristics, whether he or she survived using logistic regression.

II. Logistic Regression

Logistic regression is a statistical model used to describe data and to explain the relationship that exists between one dependent binary variable and one or more independent variables, which may or may not be binary [1].

II-A. Sigmoid Function

The main function in the logistic regression for the regression analysis is the sigmoid function. The sigmoid functions represents a smooth curve that can map any real value into a value between 0 and 1 inside the curve.

$$S(z) = \frac{1}{1 + e^{-z}}$$

Where S(z) gives an output p between 0 and 1, and then is classified to one of either value with a threshold. If $p \geq 0.5$, p = 1, and if $p < 0.5$, then p = 0.

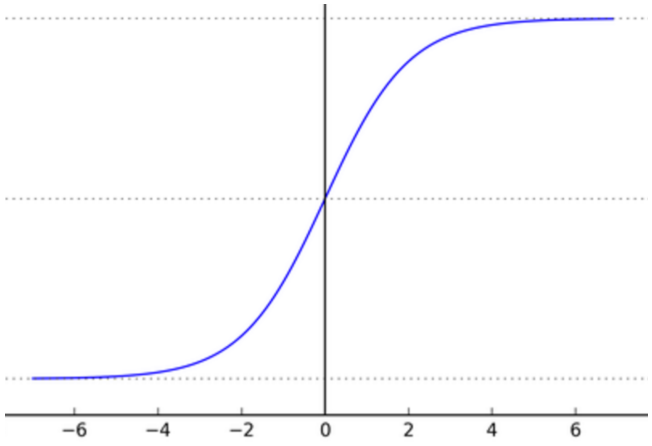

Fig. 1. Graph of the sigmoid function.

II-A.1. Multiple Linear Regression: In the sigmoid function presented in Fig. 1, the z value represents the multiple linear regression formula, which happens to be the same one used for linear regression.

$$z = \theta_0 b + \theta_1 x_1 + \theta_2 x_2 + ... + \theta_n x_n$$

Where b represents the bias, and each θ_n represents the coefficient for the independent variable x_n.
If the value for z is replaced with the multiple linear regression formula in the sigmoid function, we end up with the following equation, which will be used for making the predictions:

$$S(z) = \frac{1}{1 + e^{-(\theta_0 b + \theta_1 x_1 + \theta_2 x_2 + ... + \theta_n x_n)}}$$

III. Gradient Descent

Gradient descent is an optimization algorithm used for minimizing some function by iteratively moving in the direction of the steepest descent, as defined by the negative of the gradient. In the case of machine learning, and, specifically, logistic regression, it is used to update the coefficients of the model [2]. The formula for gradient descent looks as follows:

$$\theta_j = \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^i) - y^i) x_j^i$$

Where θ is the coefficient to be updated, α is the learning rate, $h_\theta(x^i)$ represents the prediction done with the sigmoid function, and y^i is the actual value.

III-A. Learning Rate

The size of the steps taken in the gradient descent method is called the learning rate, and is represented by α in the gradient descent function. These steps are shown in Fig. 2. When we use a high learning rate, each step can cover more ground on each step, but it comes at the risk of overshooting the lowest point. If we use a smaller learning rate we can move in the direction of the negative gradient more precisely, but it comes at the cost of longer training times if this value is too small.

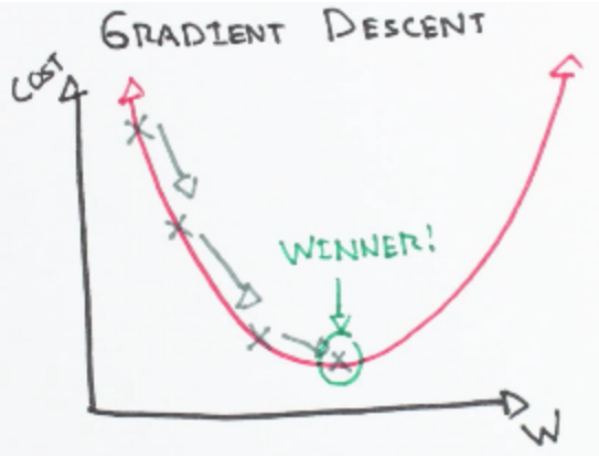

Fig. 2. Steps taken in the gradient descent algorithm.

IV. Cross-Entropy

Cross-Entropy is the alternative cost-function to mean-squared-error (MSE) for logistic regression. If we used the MSE function, we would end up with a non-convex function, with a lot of local minimums, and the gradient descent algorithm wouldn't be able to find the global minimum of the function.
The value of this cost function is dependent on the value of y, the actual value of the prediction.
The formulas look as follows:
$y = 1$:

$$Cost(h(x), y) = -log(h(x))$$

$y = 0$:

$$Cost(h(x), y) = -log(1 - h(x))$$

V. Testing

The implementation was tested on the Kaggle competition mentioned before [3], with the dataset provided for training and testing. This dataset can be downloaded from: https://www.kaggle.com/c/titanic/data
The image shown in Fig. 3 shows the submission made to the Kaggle competition, achieving an accuracy score of 0.6555.

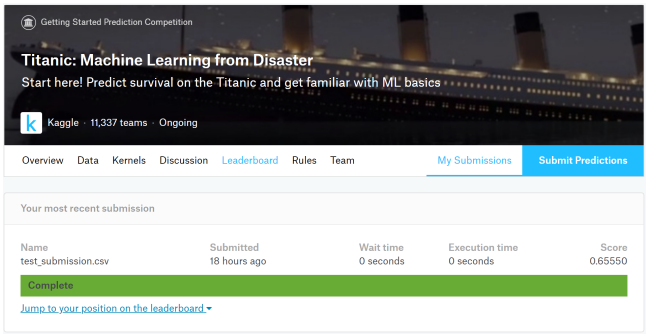

Fig. 3. Submission made to the Kaggle competition.

References

[1] ML Cheatsheet. Logistic Regression. https://ml-cheatsheet.readthedocs.io/en/latest/logistic_regression.html. Accessed: 2019-05-06.
[2] ML Cheatsheet. Gradient Descent. https://ml-cheatsheet.readthedocs.io/en/latest/gradient_descent.html. Accessed: 2019-05-06.
[3] Kaggle. Titanic: Machine Learning from Disaster. https://www.kaggle.com/c/titanic. Accessed: 2019-05-06.