logistic regression

November 20, 2022

1 logistic Regression

Logistic regression is a statistical and machine-learning technique for categorizing records in a dataset based on their input field values. It is a classification algorithm for categorical variables. Logistic regression is analogous to linear regression but tries to predict a categorical or discrete target field instead of a numeric one. In logistic regression we predict a binary such as yes/no, true,false or multi-class. In logistic regression independent variables should be continuous. if categorical, they should be dummy or indicator coded (i.e, transform them to some continuous value).

Applications of logistic regression

- Predicting the probability of a person having a heart attack
- Estimating mortality rates for injured patients
- Predicting a customer's propensity to purchase a product or halt a subscriptio
- Predicting the probability of failure of a given process or product

When should we use a logistic regression algorithm?

1. If the target field is binary (0 or 1) (true or false), (Yes, No)
2. If you need a probability result
3. If your data is linearly separable
4. If you need to understand the impact of a feature

Logistic Regression vs Linear Regression The equation that used for linear regression is

$$y' = \theta^T X$$

where Θ is called the weights factor or confidences of the equation

$$\theta^T = [\theta_0, \theta_1, \theta_2, ..., \theta_n]$$

and the x represents the independent variables

$$X = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ ... \\ x_n \end{bmatrix}$$

The Correlation-Pearson equation is used to compute the theta, as shown below.

$$\theta_{of_independent_variable} = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2(y_i - \overline{y})^2}}$$

And the mean is computed using the following equation:

$$\bar{x} = \frac{\sum x_i}{n}$$

To compute the intercept according to equation

$$\theta_0 = \frac{\sum_{i=1}^{n}(y_i - (\theta_i \times x_i))}{n}$$

To predict categorical classes using linear regression, we need to set a threshold. Any value above the threshold will be considered (one, or true) even if it is one thousand, and any value below the threshold will be considered zero or false. And this will not give us the probability of belonging to a class that is very desirable. Logistic regression is preferable for solving a linear regression problem with categorical data. The logistic regression algorithm is based on the sigmoid function it also called logistic function, which calculates the likelihood (probability) of belonging to a class instead to reurn the value of y. Sigmoid function it always returns value between 0 and 1.

$$sigmoid = \frac{1}{1 + e^{-\theta^T X}}$$

How do you calculate theta (Θ)? Theta can be found by following the training process outlined below.

- initialize Theta
- determine the predicted y
- Compare the predicted and actual y values and record the difference.
- compute the error for all y
- Determine the model cost j(Θ).
- Change the (Θ)
- Go to step 2 and compute the cost model again.

How to compute the model cost ? In linear regression the cost function is

$$J(\hat{y}, y) = \frac{1}{n}\sum_{i=0}^{n}(\hat{y}_i - y_i)^2$$

For the logistic function, the above equation is not proper since it is not convex when applied the gradient descent method to calculate the best theta in case of logistic regression. Therfore the log function is more poper for logistic (sigmoid function)

$$J(\hat{y}, y) = -\frac{1}{n}\sum_{i=0}^{n} y_i log(\hat{y}_i) + (i - y_i) log(1 - \hat{y}_i)$$

How to chose the best parameters for our model (theta)?. This can be achieved by using a gradiant descent approche.

What is the gradiant descent approche? A technique to use the derivative of a cost function to change the parameter values (thetas) to minimize the cost or error

How the gradiant descent work?

1. Initialize the parameters randomly

$$\theta^T = [\theta_0, \theta_1, \theta_2, ..., \theta_n]$$

2. Feed the cost function with training set, and calculate the error

$$J(\hat{y}, y) = -\frac{1}{n}\sum_{i=0}^{n} y_i log(\hat{y_i}) + (i - y_i) log(1 - \hat{y}_i)$$

3. Calulate the gradiant of cost function. If there are multiple thetas, take one as a variable and return the rest as a constant. then calculate the best value of theta.

$$\nabla J = [\frac{\partial J}{\partial \theta_1}, \frac{\partial J}{\partial \theta_2}, \frac{\partial J}{\partial \theta_3}, ..., \frac{\partial J}{\partial \theta_n}]$$

4. Update the weights with new values. Where is the learning rate or (steps) and means the difference between the predicted and actual values in the cost function.

$$\theta_{new} = \theta_{old} - \eta \nabla J$$

5. Go to step 2 until the cost is small enough

   There a thing called Stochastic Gradiant Descent that uses a randomly selected subset of the data at every step rather than the full dataset. This reduce the time to calculate the derivatives of the Loss function

References

1. Youtube video
2. Machine learning with Python
3. Run Math equations