Introduction to Logistic Regression

A Tutorial

Koushik Khan

koushik.khan.ds@gmail.com linkedin.com/in/koushikkhan/ koushikkhan.github.io

March 2, 2025

Contents

Motivation

Binary classification problem

- Imagine situations where we would like to know,
  - the *eligibility of getting a bank loan* given the value of credit score (x_{credit_score}) and monthly income (x_{income}).
  - identifying a *tumor as benign or malignant* given its size (x_{tumor_size}).
  - classifying an *email as promotional* given the no. of occurrences for some keywords like {'win', 'gift', 'discount'} ($x_{n\_win}, x_{n\_gift}, x_{n\_discount}$).
  - finding a *monetary transaction as fraudulent* given the time of occurrence (x_{time_stamp}) and amount (x_{amount}).
- These problems occur frequently in real life & can be dealt with machine learning.
- All such problems come under the umbrella of what is known as *Classification*.
- In each scenario, only one of the two possible outcomes can occur, hence these are specifically known as *Binary Classification* problems.
- We need *special techniques* for such problems to handle with machine learning.

Datasets for binary classification problems

- An ML algorithm needs supported data to learn patterns for addressing classification problems.
- Any dataset containing *numerical* or *categorical* features can be used for this.
- However, the *target variable must be categorical* in nature.
- Specifically, for binary classification, a target variable (Y) *must have two distinct values* like {'benign', 'malignant'}
- To use with ML algorithms, values of the target variable are *encoded into numeric representations* e.g. `{"benign": 0, "malignant": 1}`, a.k.a 'class-0' & 'class-1' respectively.

Machine Learning for Classification

How does machine learning help us for classification?

- During inference, any ML model *predicts the class label* for an observation.
- A model estimates two probabilities $\hat{p}_0$ & $\hat{p}_1$ for the two classes where $\hat{p}_0 + \hat{p}_1 = 1$.
- The predicted label depends on $max\{\hat{p}_0, \hat{p}_1\}$ i.e. it's the one which is most probable based on the given features.
- Sometimes, $\hat{p}_1$ is *compared with a predefined threshold* (that comes from experience!) p^* to take decision about the predicted class like below:

$$\begin{aligned} \text{predicted class} &= 1;\ \hat{p}_1 \geq p^* \\ &= 0;\ \text{otherwise} \end{aligned}$$

Why NOT linear regression?

- Can't we really use linear regression to address classification? The answer is *NO!*
- Let's try to understand why:
  - To estimate p_1 using linear regression, we need:

$$\hat{p}_1 = \alpha + \beta x_{tumor\_size} \tag{1}$$

  - Eqn. (1) *doesn't seem to be feasible* as the R.H.S, in principle, belongs to $(-\infty, +\infty)$ & the L.H.S belongs to $[0, 1]$.
- Can we convert $(\alpha + \beta x_{tumor\_size})$ to something belonging to $[0, 1]$? That may work as an estimate of a probability! The answer is *YES!*
- We need a *converter (a function)*, say, $g(.)$ that would map $(\alpha + \beta x_{tumor\_size}) \in \mathbb{R}$ to some $p \in [0, 1]$.
- Fortunately, such functions do exist and they are often referred to as *link functions*.
- We are going to use a link function called *sigmoid function*, denoted by σ.

The sigmoid link function

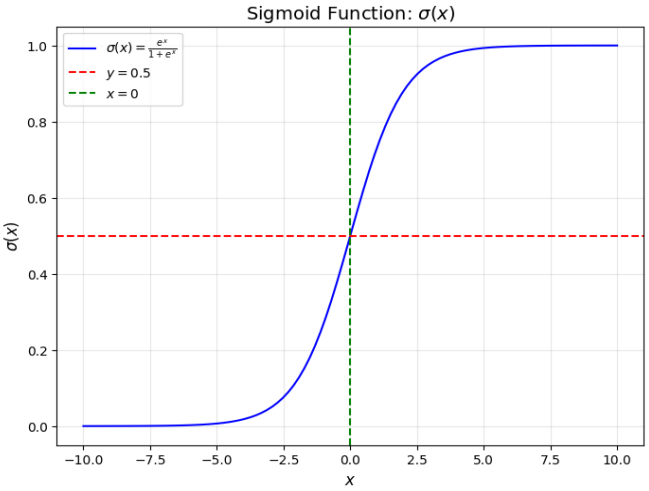

Figure: The sigmoid function

Math behind sigmoid

- The *sigmoid function,* $\sigma(.)$ is defined as below:

$$\sigma(z) = \frac{e^z}{1+e^z} = \frac{1}{1+e^{-z}};\ z \in \mathbb{R} \tag{2}$$

  and the following is *always true*: $\sigma(z) \in [0,1]\ \forall z \in \mathbb{R}$.
- It has a very nice & important property:

$$\frac{d\sigma}{dz} = \sigma'(z) = \sigma(z) \times (1 - \sigma(z)) \ \textit{(verify!)} \tag{3}$$

- It looks like an *elongated 'S'*, that is where it gets its name from.

Logistic Regression

The logistic regression setup

- In logistic regression, we try to find $P(Y = 1)$ (i.e. *probability that a tumor will be malignant*) as below:

$$P(Y = 1) = p_1 = \frac{e^{\alpha+\beta x_{tumor\_size}}}{1 + e^{\alpha+\beta x_{tumor\_size}}} \tag{4}$$

- And this simplifies to:

$$\log\left(\frac{p_1}{1 - p_1}\right) = \alpha + \beta x_{tumor\_size} \textit{ (verify!)} \tag{5}$$

where the L.H.S is called *log-odds* or *logit*.

It's a linear classifier

- The logistic regression equation (5) is actually a *straight line*.
- **Recall** the prediction rule:

$$\begin{aligned}\text{predicted class} = 1;\ p_1 \geq p^* &\Rightarrow \alpha + \beta x_{tumor\_size} \geq \log\left(\frac{p^*}{1-p^*}\right) \\ &= 0;\ \text{otherwise}\end{aligned}$$

- A simple logistic regression (the one we discussed) predicts the class label by *identifying the regions on either side of a straight line* (or hyperplane in general), hence it's a *linear classifier*.

Linear versus non-linear classification

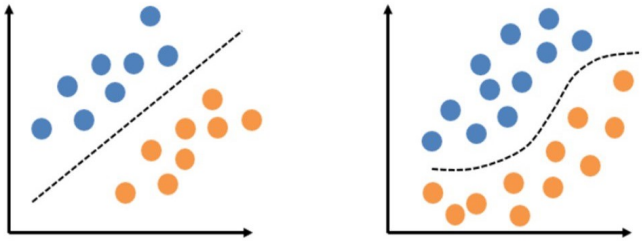

Figure: On the left, the classes are linearly separable, however they are not on the right

- Logistic regression works well for *linearly separable* classes.

Estimation of Parameters

Prerequisites

Bernoulli's Trial

A random experiment that results in one of the two possible outcomes, often called, a *success* and a *failure*, with a *constant probability* of success p.

Example

- tossing a fair coin - the coin shows either the 'HEAD' or the 'TAIL'
- performing COVID test - result will be either '+ve' or '-ve'
- detecting a tumor as 'benign' or 'malignant' etc.

- Bernoulli's trial can be expressed mathematically with a random variable, say, Y:

$$f(y|p) = P(Y = y) = p^y \times (1-p)^{1-y};\ y \in \{0, 1\} \tag{6}$$

  where eqn. (6) is known as *Probability Mass Function (PMF)*.
- In PMF *we don't know* what value Y would take but we assume *p is known* to us.

Likelihood function

- The PMF maps the values taken by Y to probabilities e.g.

$$\text{when } y = 1 \Rightarrow P(Y = 1) = p$$
$$\text{when } y = 0 \Rightarrow P(Y = 0) = (1 - p)$$

- Data involve N patients with: $\{(x_{i,tumor\_size}, y_i)\}_{i=1}^{N},\ y_i \in \{0, 1\}$.
- We consider Y_i takes the value y_i with probability p_i (*as each patient can have different probability to have a malignant tumor*).
- In reality, *we don't know these p_i's*, so they are *estimated from data*.
- Interestingly, the expression for $P(Y_i = y_i) = p_i$ is same as before, eqn. (6),

$$f_i(p_i|y_i) = P(Y_i = y_i) = p_i^{y_i} \times (1 - p_i)^{1-y_i};\ y_i \in \{0, 1\}\ \forall i = 1, 2, \ldots, N \quad (7)$$

  The eqn. (7) is known as the likelihood for Y_i taking a value y_i.
- In Likelihood we know the data (*as we're observing them*), but the p_i's are *unknown* to us.

Joint likelihood function

- But what's the likelihood for *observing the entire dataset* i.e. $P(Y_1 = y_1 \cap \ldots \cap Y_N = y_N)$? Well, *joint likelihood* gives that answer.
- It's computed as below:

$$L = P(Y_1 = y_1 \cap \ldots \cap Y_N = y_N) = \Pi_{i=1}^{N} p_i^{y_i} \times (1 - p_i)^{1 - y_i} \tag{8}$$

  Eqn. (8) is called the *joint likelihood (L)*.
- It is much easier to work with joint likelihood after a *log-transformation*, also called the *log-likelihood (LL)*,

$$LL = \log(L) = \sum_{i=1}^{N} \Big\{ y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \Big\} \tag{9}$$

- Joint likelihood measures the probability of observing the underlying dataset i.e. having $\{Y_1 = y_1, \ldots, Y_N = y_N\}$ for some unknown set of probabilities $\{p_1, \ldots, p_N\}$.

Maximum likelihood estimation

- Let's assume, some process might have generated the observed dataset $\{(x_{i,tumor\_size}, y_i)\}_{i=1}^{N}$
- But NOT sure what values of p_i's the process would have considered to generate the dataset.
- We can imagine *several potential candidates* for each p_i (say, all belong to $\mathbb{P}$) that might have been used to generate the dataset.
- In principle, the best candidate for each p_i would be the one that makes the joint likelihood (L) or log likelihood (LL) (both are functions of p_i's) *maximum*.
- Mathematically, we would use,

$$\underset{p_1,\ldots,p_N \in \mathbb{P}}{\arg\max} \sum_{i=1}^{N} \Big\{ y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \Big\} \tag{10}$$

to find $\hat{p}_1, \ldots, \hat{p}_N$.

Going deeper

- Finding $\hat{p}_1, \ldots, \hat{p}_N$ using eqn. (10) is basically an *optimization problem*.
- **Recall** $p_i = \frac{e^{\alpha+\beta x_{i,tumor\_size}}}{1+e^{\alpha+\beta x_{i,tumor\_size}}} = \sigma(\alpha + \beta x_{i,tumor\_size})$
- The eqn. (10) can be simplified as:

$$\underset{p_1,\ldots,p_N \in \mathbb{P}}{\arg\max} \left[\sum_{i=1}^{N} \left\{ y_i \log\left(\sigma(\alpha + \beta x_{i,tumor\_size})\right) + (1 - y_i) \log\left(1 - \sigma(\alpha + \beta x_{i,tumor\_size})\right) \right\} \right] \tag{11}$$

- Here each p_i *is a function* of the parameters α & β and the known data $x_{i,tumor\_size}$.
- Finding $\hat{p}_1, \ldots, \hat{p}_N$ *is equivalent* to finding α and β with the help of eqn. (11).

Using gradient descent

- Specifically, it's a *maximization problem* here, however the ML community rather take *negative of LL, NLL* to *minimize* with the help of *Gradient Descent* !
- Actually a *scaled version* of NLL ($\frac{1}{N}NLL$) is used and that is known as *Binary Cross Entropy (BCE)* loss function.
- Now the problem boils down to:

$$\underset{p_1,\ldots,p_N \in \mathbb{P}}{\arg\min} \frac{1}{N} NLL = \underset{p_1,\ldots,p_N \in \mathbb{P}}{\arg\min} -\frac{1}{N} \sum_{i=1}^{N} \Big\{ y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \Big\} \tag{12}$$

or equivalently,

$$\begin{aligned} \underset{\alpha,\beta}{\arg\min} -\frac{1}{N} \sum_{i=1}^{N} \Big\{ & y_i \log\left(\sigma(\alpha + \beta x_{i,tumor\_size})\right) \\ & + (1 - y_i) \log(1 - \sigma(\alpha + \beta x_{i,tumor\_size})) \Big\} \end{aligned} \tag{13}$$

Computing derivatives

- Gradient Descent *computes derivatives* of BCE w.r.t α and β.
- Here is how it works:
  Let us first consider $z_i = \alpha + \beta x_{i,tumor\_size}$
  we would like to compute:

$$\begin{aligned}\frac{\partial BCE}{\partial \alpha} &= -\frac{1}{N}\frac{\partial}{\partial \alpha}\sum_{i=1}^{N}\Big\{y_i \log\left(\sigma(z_i)\right) + (1-y_i)\log(1-\sigma(z_i))\Big\} \\ &= -\frac{1}{N}\sum_{i=1}^{N}\left\{y_i\frac{\partial}{\partial \alpha}\log\left(\sigma(z_i)\right) + (1-y_i)\frac{\partial}{\partial \alpha}\log\left(1-\sigma(z_i)\right)\right\}\end{aligned} \tag{14}$$

and similarly,

$$\frac{\partial BCE}{\partial \beta} = -\frac{1}{N}\sum_{i=1}^{N}\left\{y_i\frac{\partial}{\partial \beta}\log\left(\sigma(z_i)\right) + (1-y_i)\frac{\partial}{\partial \beta}\log\left(1-\sigma(z_i)\right)\right\} \tag{15}$$

The magic of using sigmoid

- We will compute the derivatives one by one.
- Here is how we proceed for α:

$$\begin{aligned}\frac{\partial \log\left(\sigma(z_i)\right)}{\partial \alpha} &= \overbrace{\frac{\partial \log\left(\sigma(z_i)\right)}{\partial \sigma(z_i)} \times \frac{\partial \sigma(z_i)}{\partial z_i} \times \frac{\partial z_i}{\partial \alpha}}^{\text{chain rule of differentiation}} \\ &= \frac{1}{\sigma(z_i)} \times \sigma(z_i) \times (1 - \sigma(z_i)) \times 1 \\ &= (1 - \sigma(\alpha + \beta x_{i,tumor\_size}))\end{aligned} \tag{16}$$

$$\begin{aligned}\frac{\partial \log\left(1 - \sigma(z_i)\right)}{\partial \alpha} &= \frac{\partial \log\left(1 - \sigma(z_i)\right)}{\partial (1 - \sigma(z_i))} \times \frac{\partial (1 - \sigma(z_i))}{\partial \sigma(z_i)} \times \frac{\partial \sigma(z_i)}{\partial z_i} \times \frac{\partial z_i}{\partial \alpha} \\ &= \frac{1}{(1 - \sigma(z_i))} \times (-1) \times \sigma(z_i) \times (1 - \sigma(z_i)) \times 1 \\ &= -\sigma(z_i) = -\sigma(\alpha + \beta x_{i,tumor\_size})\end{aligned} \tag{17}$$

Continuing...

- And similarly for β:

$$\begin{aligned}\frac{\partial \log\left(\sigma(z_i)\right)}{\partial \beta} &= \overbrace{\frac{\partial \log\left(\sigma(z_i)\right)}{\partial \sigma(z_i)} \times \frac{\partial \sigma(z_i)}{\partial z_i} \times \frac{\partial z_i}{\partial \beta}}^{\text{chain rule of differentiation}} \\ &= \frac{1}{\sigma(z_i)} \times \sigma(z_i) \times (1 - \sigma(z_i)) \times x_{i,tumor\_size} \\ &= (1 - \sigma(\alpha + \beta x_{i,tumor\_size})) \times x_{i,tumor\_size}\end{aligned} \tag{18}$$

$$\begin{aligned}\frac{\partial \log\left(1 - \sigma(z_i)\right)}{\partial \beta} &= \frac{\partial \log\left(1 - \sigma(z_i)\right)}{\partial (1 - \sigma(z_i))} \times \frac{\partial (1 - \sigma(z_i))}{\partial \sigma(z_i)} \times \frac{\partial \sigma(z_i)}{\partial z_i} \times \frac{\partial z_i}{\partial \beta} \\ &= \frac{1}{(1 - \sigma(z_i))} \times (-1) \times \sigma(z_i) \times (1 - \sigma(z_i)) \times x_{i,tumor\_size} \\ &= -\sigma(\alpha + \beta x_{i,tumor\_size}) \times x_{i,tumor\_size}\end{aligned} \tag{19}$$

The iterative rule of gradient descent

- We can now compute $\frac{\partial BCE}{\partial \alpha}$ and $\frac{\partial BCE}{\partial \beta}$.
- And finally make use of the Gradient Decent *update rule*:

$$\overbrace{\begin{pmatrix}\hat{\alpha}^{(t+1)}\\ \\ \hat{\beta}^{(t+1)}\end{pmatrix}}^{\text{estimates at } (t+1)^{th} \text{ step}} = \overbrace{\begin{pmatrix}\hat{\alpha}^{(t)}\\ \\ \hat{\beta}^{(t)}\end{pmatrix}}^{\text{estimates at } t^{th} \text{ step}} - \eta \cdot \overbrace{\begin{pmatrix}\frac{\partial BCE}{\partial \alpha}\Big|_{\hat{\alpha}^{(t)}}\\ \\ \frac{\partial BCE}{\partial \beta}\Big|_{\hat{\beta}^{(t)}}\end{pmatrix}}^{\text{gradient adjustments at } t^{th} \text{ step}} \tag{20}$$

Here η is the *learning rate*.

Evaluating Model Performance

Confusion matrix & related metrics

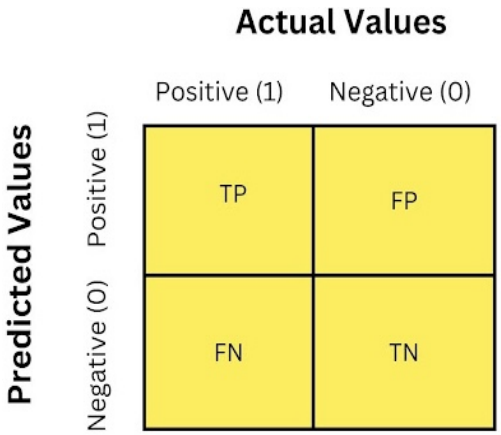

Figure: Confusion Matrix

- All cases = $TP + TN + FP + FN$
- Correctly classified cases = $TP + TN$
- Misclassified cases = $FP + FN$
- Precision = $\frac{TP}{TP+FP}$
- Recall (Sensitivity) = $\frac{TP}{TP+FN}$
- Specificity = $\frac{TN}{TN+FP}$
- Accuracy = $\frac{TP+TN}{TP+TN+FP+FN}$
- F1-score = $\frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$
- All the above metrics except the first one *depends on the threshold* p^*.

Understanding precision

- Precision (a.k.a **True Positive Rate**) measures the *probability of predicting a true positive case* by a fitted model.
- According to the formula: $\frac{TP}{TP+FP}$, the *lesser* the number of false positive cases, the *higher* will be the precision.
- Precision is important where is making *false positive mistakes* is *risky*.
- In a email spam detection system, it's crucial that a *non-spam* email is *not getting tagged* as a spam email, otherwise an user may miss an important email. - *expecting a high precision*
- In medical diagnosis (e.g. cancer detection) high precision gives *confidence* to the doctors to start treatment without further tests. - *expecting a high precision*
- Precision is also important when the dataset is highly imbalanced (e.g. credit fraud detection, where getting a fraudulent transaction is rare). If the precision is low, even if the accuracy is very high, the model would probably raise many *false alarms*, which is misleading. - *expecting a high precision*

Understanding recall

- Recall (a.k.a **Sensitivity**) measures the *probability of detecting a true positive case* when it's *actually positive*.
- According to the formula: $\frac{TP}{TP+FN}$, the *lesser* the number of false negative cases, the *higher* will be the recall.
- Recall is important where making *false negative mistakes* is *risky*.
- In case of cancer detection diagnosis, it's very important that a cancer is *getting detected* in the body if it is *actually there*, otherwise it will be a *life risk*. - *expecting a high recall*
- While using medical kit for detecting COVID, it's important that a person is *NOT tagged* as "COVID -VE" when he is *actually* "COVID +VE". Having such cases will *infect* many other people. - *expecting a high recall*
- Detecting as many threats as possible is important for an airport security system. Having higher values of recall will make sure almost all positive cases are identified. - *expecting a high recall*

Understanding F1 score

- The idea of using F1-score is to keep *a balance* between precision and recall.
- According to the formula: $\frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$, F1-score calculates *harmonic mean* of precision and recall.
- Harmonic mean *penalizes the extreme values* of both precision and recall.
- For example, in case of credit fraud detection, actual fraudulent cases (positive) are very rare and this may make the model biased towards *legitimate cases* with a *very high accuracy*, however it *may not* make any sense.
- Here the model should actually:
  - have *high precision* i.e. lower chance of raising false alarm by identifying a legitimate case as fraudulent
  - have *high recall* i.e. lower chance of missing fraudulent transaction

  The F1-score keeps a balance between these and gives a much more *realistic evaluation* of how well a model is performing in detecting the *minority class (fraud)*.

Receiver operating characteristic curve (ROC)

- It's a *graphical tool* depending on two metrics derived from the *confusion matrix*:
  - True Positive Rate = P(Predicted Positive | Actually Positive) = $\frac{TP}{TP+FP}$
  - False Positive Rate = P(Predicted Positive | Actually Negative) = $\frac{FP}{FP+TN}$
- By *varying* p^* within a *permissible range* a set of $\{(FPR_k, TPR_k)\}_{k=1}^{K}$ are obtained, and are plotted to form what is called *ROC Curve*.
- The FPR varies *along the X-axis* and TPR varies *along the Y-axis*.
- Both TPR and FPR vary within $[0, 1]$ resulting the *total area* of the plotting canvas to be 1.
- The *diagonal line*, connecting the coordinates $(0, 0)$ and $(1, 1)$ indicates $FPR = TPR$, which is how a *random model* would behave.
- It's always good to have a model which produces *TPR values on the higher side* and *FPR values on the lower side*.

Area under the curve (AUC)

- It's one of the robust measures to *compare* different models or model configurations.
- The diagonal line (– – –) divides the plotting canvas in to *two halves* having an *area of* 0.5 each. This line indicates a random classifier which is *equally good and bad*.
- Any model better than the random one will *cover an area* > 0.5.
- The *higher* the AUC the model achieves, the *better* its performance.

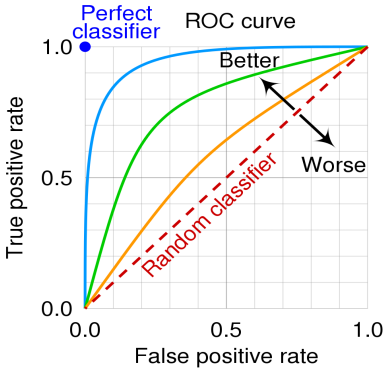

Figure: Comparing models with ROC curves and AUC values [1]

[1] Image Source: Wikipedia

Interpretations of parameters

- α: When x_{tumor_size} *is zero*, it's the *value of the log-odds*. It's often called the *baseline log-odds*.
- β: It's the *change in log-odds* for an *unit change in* x_{tumor_size}.
- Interpretations are just like the *linear regression* as the alternate form of logistic regression (eqn. (5)) is exactly a linear regression w.r.t the log-odds.
- log-odds $\in (-\infty, +\infty)$ *(verify!)*
- Even though logistic regression is used for *classification*, it actually *estimates a probability*, which is *continuous* within $[0, 1]$ - it's a *bridge* between *continuous modeling* and *discrete outcomes*, classification is just a *practical application* of this model.

References

- Generalized Linear Models by by P. McCullagh, J. A. Nelder
- Categorical Data Analysis by Alan Agresti

Thank You