Report

Wajih Arfaoui

3/9/2022

Logistic Regression

1. Main Objective

Logistic regression is a kind of parametric classification model. It uses a linear combination of features to come out with a probability to assign two values 0 and 1 fail or true and false to the response variable.
The middle value of probabilities is considered as threshold to establish what belong to the class 1 and to the class 0. In a general note, if the probability is greater that 0.5, then the observation belongs to class 1, otherwise it is belongs to class 0.

2. Logit Function

Logistic regression is expressed as:

$$log(\frac{p(X)}{1-p(X)}) = \beta_0 + \beta_1 X$$

Where the left-hand side is called the *log-odds* and $\frac{p(X)}{1-p(X)}$ is called *odds*, and it tells about the probability of success to probability of failure.

When taking the inverse of the logit function we will get:

$$p(X) = (\frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}})$$

This function is the **Sigmoid** function and it creates the *S-shaped* curve, and returns a probability value between 0 and 1.

3. Maximum Likelihood Estimation

Since **Maximum Likelihood** is the more general method of non-linear least squares and it has a better statistical properties, it is used to fit the logistic regression model.
The maximum likelihood estimation defines the coefficients for which the the probability of getting the observed data is maximized.

The *likelihood* function formalizing the stated intuition is:

$$L(\beta, y) = \prod_{i=1}^{N} (\frac{\pi_i}{1-\pi_i})^y (1-\pi_i)$$

for $y_i = [0, 1]$

π_i is the probability of success if y_i belongs to class 1

In order, to determine the parameters' values, we apply $log()$ to the likelihood function, since it does not change its initiatl propoerties. Then we apply **iterative** optimisation techniques such as **Gradient Descent**.

4. Pros & Cons

Pros	Cons
- Logistic Regression is easy to interpret and very efficient to train	- Logistic Regression may lead to overfitting when the number of features is greater than the number of observations
- In addition to providing coefficients' sizes, it tells the direction of association	- It assumes linearity between the dependent variable and the target
- It doesn't need hyperparameter tuning	- It is not considered as a very powerful algorithm and can be easily outperformed by other algorithms

https://www.sciencedirect.com/topics/medicine-and-dentistry/logistic-regression-analysis
https://towardsdatascience.com/logistic-regression-explained-9ee73cede081
https://medium.com/data-science-group-iitr/logistic-regression-simplified-9b4efe801389#:~:text=The%20idea%20of%20Logistic%20Regression,two%20values%2C%20pass%20and%20fail.
https://medium.com/analytics-vidhya/what-is-the-logistic-regression-algorithm-and-how-does-it-work-92f7394ce761

Linear Discriminant Analysis

1. Main Objective

The aim of LDA is to maximize the **between-class** variance and **minimize the within-class** variance through a linear discriminant function. It assumes that all classes are linearly separable, and the data in each class is described by a **Gaussian** probability density function which means that it has a bell-curve shape when plotted.

2. Linear Descriminante function

Since the LDA uses the Bayes' Theorem to make predictions based on the probability that the observation x belongs to each class. The class having the highest probability is designated as the output class, and then prediction is made by the LDA.
The *Bayes' theorem states that:

$$Pr(Y=k|X=x)=\frac{\pi_k f_k(x)}{\sum_{l=1}^{k}\pi_l f_l(x)}$$

Where x= input

k= output class

π_k = prior probability that an observation belongs to class k

$f_k(x)$ = estimated probability of x belonging to class k

Supposing that $f_k(x)$ follows a Gaussian Distribution, it takes the form:

$$f_k(x)=\frac{1}{\sqrt{2\pi\sigma_k^2}}\exp\left(-\frac{1}{2\sigma_k^2}(x-\mu_k)^2\right)$$

After plugging this into the probability equation and taking the log of it, we get the following discriminant function:

$$\delta_k(x)=x.\frac{\mu_k}{\sigma^2}-\frac{\mu_k^2}{2\sigma^2}+\log(\pi_k)$$

3. LDA assumptions

As already mentioned, LDA makes two important assumptions about the data:
- Each data variable is bell curve shaped
- The values of each variable vary around the mean by the same amount on the average.

Based on that, LDA method approximates the Bayes classifier by plugging estimates for π_k, μ_k and σ^2 into the linear discriminant function.
The following estimates ares used:

$$\hat{\mu_k} = \frac{1}{n_k} \sum_{i:y=k} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n-k} \sum_{k=1}^{k} \sum_{i:y=k} (x_i - \hat{\mu_k})^2$$

$$\hat{\pi}_k = n_k / n$$

Where $\hat{\pi}_k$ = variance across all inputs x

n = number of instances

k = number of classes

$\hat{\mu_k}$ = mean for input x

4. Pros & Cons

Pros	Cons
- LDA is simple to implement and the classification is robust - It uses information from both the features to create a new axis which in turn minimizes the variance and maximizes the class distance of the two variables	- LDA's linear decision boundaries may not adequately separate the classes - It requires normal distribution assumption on features - It has a large time complexity

https://www.geeksforgeeks.org/ml-linear-discriminant-analysis/ https://www.knowledgehut.com/blog/data-science/linear-discriminant-analysis-for-machine-learning https://machinelearningmastery.com/linear-discriminant-analysis-for-machine-learning/ https://www.sciencedirect.com/topics/computer-science/linear-discriminant-analysis https://towardsdatascience.com/linear-discriminant-analysis-explained-f88be6c1e00b

K-Nearest Neighbors

1. Main Objective

K-nearest neighbors is a supervised machine learning algorithm that is used for both, regression and classification. It predicts the correct class for the test observation by measuring the distance between it and all the training data points. Then it tries to find the **nearest neighbors** by ranking points by increasing distance,and finally vote for the most frequent label to be assigned to the test data point, after selecting the specified number k of nearest neighbors to consider.

2. Calculating distance

The first step in the KNN application is to calculate the distance between the data point we want to classify and our training data points.
To do this, we need to choose one of the various methods used to measure the distance such as:

- **Euclidean Distance**: It is calculated as the square root of the sum of the squared differences between the test data point t and a train data point x.

$$\sqrt{\sum_{i=1}^{k}(t_i - x_i)^2}$$

- **Manhattan Distance**: It calculated the distance between two points in a N dimensional vector space by summing the absolute difference between the measures in all dimensions of two points.

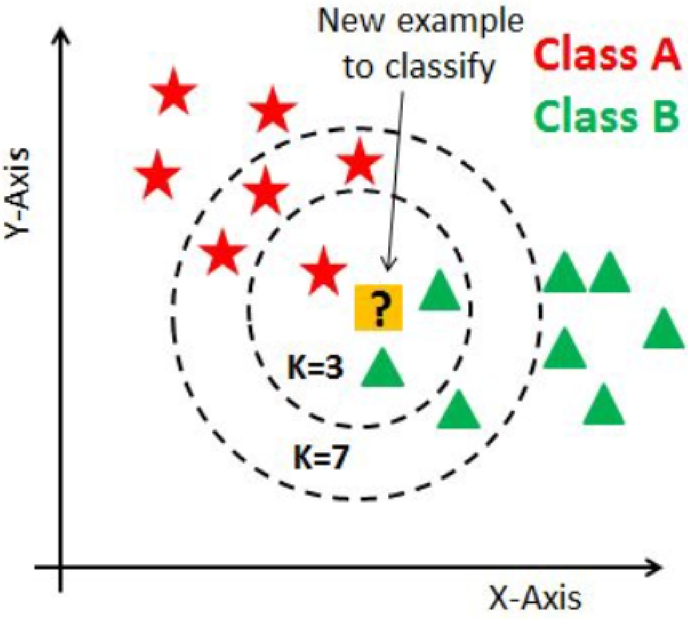
$$\sum_{i=1}^{k}|t_i - x_i|$$

The two stated methods above are applied for continuous variables, while in the case of categorical variables, we opt for:

- **Hamming Distance**: It counts the number of times the coordinates in two categorical vectors differ. It is mainly used when you one-hot encode your data and need to find distances between the two binary vectors.

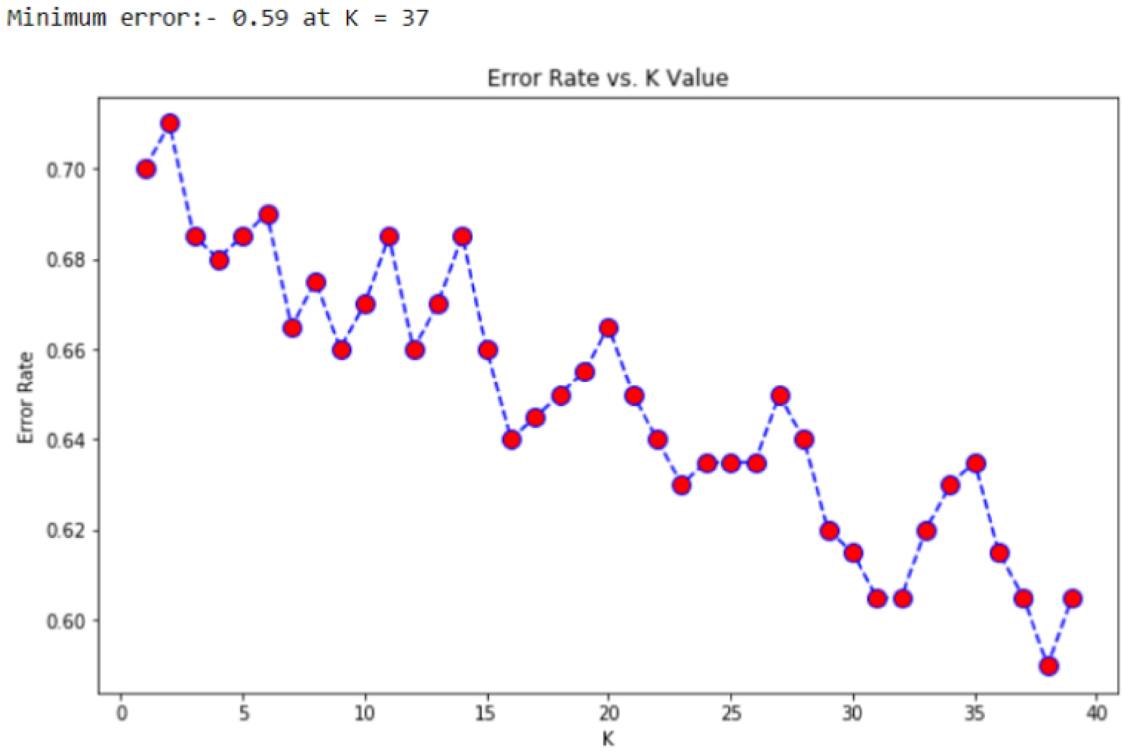
3. Choosing the right value for K

As **K** value indicates the number of nearest neighbors, we need to look for the optimal value to ensure we get the most accurate classification.
From the photo below, if we choose K=3, we predict that test observations belongs to class B, but it belongs to class A if we choose K to be 7.



So, to select the K that is right for the data, we don't have pre-defined statistical methods to find the most favorable one, but instead we need to run the KNN algorithm multiple times with a different K each time.

Then, we choose the K that reduces the number of errors we encounter while maintaining the algorithm's ability to accurately make predictions when it's given data it hasn't seen before as show in the plot below.



```
Minimum error:- 0.59 at K = 37
```

4. Pros & Cons

Pros	Cons
- It is simple to implement since it has only one parameter *k*	- It is a lazy learner, it takes longer time for inference than training
- It allows dealing with complex objects, such as time series, graphs and any distance metric even defined by the user	- It is prone to overfitting as it is a distance based approach and can be affected by outliers
- It is versatile, since it can be used for classification, regression, recommendations etc.	- It has the curse of dimensionality

https://www.saedsayad.com/k_nearest_neighbors.htm#:~:text=K%20nearest%20neighbors%20is%20a,as%20a%20non%2Dparametric%20technique. https://medium.com/analytics-vidhya/pros-and-cons-of-popular-supervised-learning-algorithms-d5b3b75d9218 https://www.kdnuggets.com/2020/11/most-popular-distance-metrics-knn.html#:~:text=The%20Hamming%20distance%20method%20looks,between%20the%20two%20binary%20vectors. https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761 https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4

Classification Decision Trees

1. Main Objective

Classification trees is used in order to create a training model that leads to a decision about the class of an object by learning simple decision rules inferred from the training dataset.

In Decision trees, in order to predict a class label for the test observation, we start from the root of the tree, then we compare the values of the root attribute to the observation that we want to classify attribute. Once the comparaison is done, we follow the branch corresponding to that value and we move on to the next node.

4. Pros & Cons

Pros	Cons
- It is inexpensive to construct, and it is fast at classifiying unknown observations	
- It allows dealing with complex objects, such as time series, graphs and any distance metric even defined by the user	- It is prone to overfitting as it is a distance based approach and can be affected by outliers
- It is versatile, since it can be used for classification, regression, recommendations etc.	- It has the curse of dimensionality