Homework 1 of TDT4173 fall 2021

Elisa Tedde 567190

September 17, 2021

1 Introduction

In this assignment I have developed two machine learning algorithms: Logistic Regression and k-Means.

2 Logistic Regression

How does the algorithm work on a technical level and what kind of machine learning problems is it suited for?

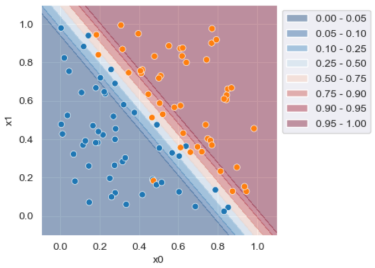
Logistic regression is a discriminative approach for classification. It is a supervised learning algorithm: the goal is to find a function h: X $\rightarrow$ Y where h(X) is a good predictor for the corresponding value of y. The model parameters are w and b. In this case we have a binary logistic model and we compute a linear combination of the inputs $y = w.T * x + b$ then, we pass it to the sigmoid function. A sigmoid function maps the linear combination of variables in value on to Bernoulli probability distribution with domain(0 to 1). We use the cost function to measure how close the model's predictions are to the actual outputs. In this process, we are trying to find different values of w and b and update them to reach the optimal ones which give the least classification error using gradient descent. Gradient descent is an iterative optimization technique which finds the minimum of a differentiable function by changing the values of coefficients repeatedly and its stopping criteria is the the number of iterations.

A predict function takes as input data x and will give the predicted value of y. This function will first create a linear model using using model parameters. Then we will pass this model to the sigmoid function which will return the y predicted values in the range of 0 to 1.

What is its inductive bias, i.e., what assumptions does it make about the data in order to generalize?

The inductive bias of the logistic function is based on a simple boundary that splits the samples that are classified as 1 from the samples classified as 0: its assumption is that decision rules are linear surfaces (hyperplanes) orthogonal to w. As you can see from the figure 1, the model for the first dataset computes a linear separation and the further you get from that boundary the more confident you can be: the misclassified samples are closed to the hyperplane and their probabilities are $p \geq 0.25$ and $p \leq 0.75$, while points with $p < 0.25$ and $p > 0.75$ are well classified. The accuracy that I have obtained is 0.930 and the cross entropy is 0.1745

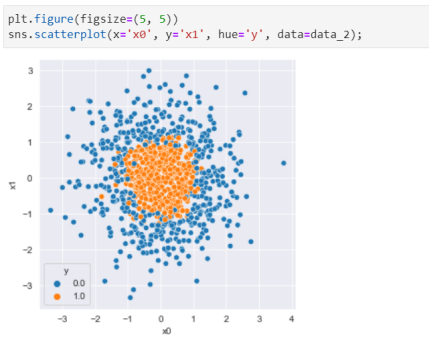
Figure 1: Linear Separation rule - First Dataset



What happens in the second dataset that makes it harder than the first and how does this problem relate to the algorithm's inductive bias?

For the first dataset data are linearly separable while for the second one, the linear separation does not work well because the samples follow a circular distribution: as we can see in the figure 2 there is the inner class (orange samples) and the outer class (blues samples)

Figure 2: Second Dataset Distribution



What modifications did you do to get around this problem?

I have decided to implement quadratic separation rules since the logistic regression model uses as feature vectors

$$\phi(\boldsymbol{x}) = \begin{bmatrix} \text{vec}\left(\boldsymbol{x}\boldsymbol{x}^T\right) \\ \boldsymbol{x} \end{bmatrix}$$

rather than X: the decision rules are linear separation surfaces in the space defined by the mapping of ϕ (I have used the method: computefi(x)). For this dataset I have obtained the following results:

Figure 3: Accuracy and Cross Entropy - Second Dataset

```
Train
Accuracy: 0.900
Cross Entropy:  0.222

Test
Accuracy: 0.908
Cross Entropy:  0.207
```

3 K-Means

How does the algorithm work on a technical level and what kind of machine learning problems is it suited for?

Clustering is an unsupervised method for finding similarity groups in data, called clusters. It groups samples that are similar to each other in one cluster and data instances that are very different from each other into different clusters. K-means clustering is a method that aims to partition n samples into k clusters: each cluster has a center called centroid. The algorithm is divided into four main steps:

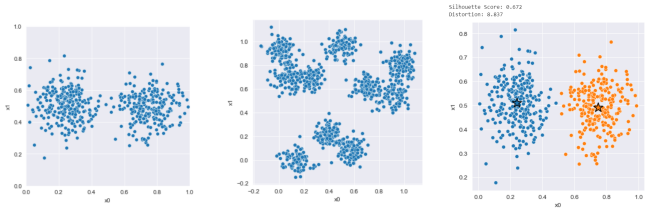
1. k samples are randomly selected to be the initial centroids
2. each data point is assigned to the closest centroid
3. new centroids are recomputed using the current clusters
4. the procedure is repeated until the convergence is not met

Finally, distortion and silhouette are computed to evaluate the quality of the classification: we would like to minimize the distortion and maximize the silhouette.

What is its inductive bias, i.e., what assumptions does it make about the data in order to generalize?

Firstly, we are trying to find homogeneous subgroups within the samples such that data points in each cluster are as similar as possible, according to a similarity measure such as euclidean-based distance or correlation-based distance. Secondly, the performance of a clustering algorithm may be affected by the chosen value of K and by the initial selected centroids. For the K value, instead of using a single predefined K, a set of values might be adopted. It is important for the number of values considered to be reasonably large, to reflect the specific characteristics of the dataset. At the same time, the selected values have to be significantly smaller than the number of objects in the datasets, which is the main motivation for performing data clustering. For the selection of initial centroids, it is clear that the algorithm is sensitive to the initial seeds: different initial centroids would give different clusters. I have decided to generate the initial random centroids for 4 times in order to select the ones that perform well. Running the same code multiple times I got an error that is “index out of bounds” because the algorithm sometimes generates empty clusters. So I have decided to fix my K-array = [2, 10] because for the first dataset the best k is 2 and for the second one is 10: this is an ”escape method” to reduce the probability of having empty clusters. In the figures below you can see the distribution of the two datasets and the clusters generated using the first dataset.

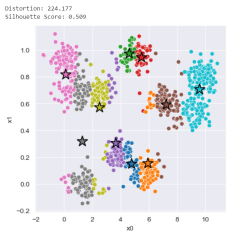
Figure 4: (a) First Dataset, (b) Second Dataset, (c) Clusters - First Dataset



(a) (b) (c)

What happens in the second dataset that makes it harder than the first and how does this problem relate to the algorithm’s inductive bias? For the second dataset, the problem is that the feature x0 is more or less 10 times the feature x1: computing the Euclidian distance x1 was irrelevant to x0 so my algorithms performs bad as you can see from the figure:

Figure 5: Second Dataset



What modifications did you do to get around this problem? I have normalized x0 dividing it by 10 in order to rebalance the weights of both features. I have obtained the results showed in the figure below:

Figure 6: Second Dataset