1 logistic regression

Assignment 1 by Haakon Haaland Paaske

1.1 Description of algorithm

The goal of the logistic regression algorithm in machine learning is to predict the state of a dependant variable(in this case whether or not the point is 1 or 0, blue or orange), based on independent variables. Contrary to a linear regression, it utilizes a logistic function which it uses to model the probabilities. In general, logistic regression is a very effective model to apply to classification problems where the dependant variable is binary.

1.2 My implementation

I start off by initializing the learning rate, number of iterations, weights and bias. When I fit the model to the data I first loop through all the samples and creates a linear model based on each sample. I then pass the linear model through my sigmoid function which gives me the prediction of the dependant variable (Y).

$$z = dataframe * weights + bias$$

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

Further more I use gradiant decent to increase the accuracy of my predictions for each iteration bu updating the weights and the bias.

My predict function takes in the data set as a variable and creates a linear model from it and passes the model through the sigmoid function, the same as in my fit function. The difference is that we now predict whether or not the point is blue or orange. I decided to assign every point above 0.5 chance to 1 and every point under to 0.

My get centroids function just returns the centroids of the model to easily plot the centroids on a scatter plot.

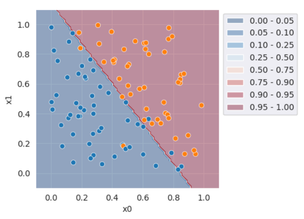

Figure 1: Logistic regression classifying points using the first data set

1.3 Inductive bias

The main inductive bias with my logistic regression model is that there has to be a boundary that can separate the samples based on its dependant variable. This of course means that the dependant variable has to be binary as well.

For a logistic regression model to generalize well on the training data, there also has to be very little correlation between the independent variable.

1.4 Second dataset

The difference between the first and the second data set is that one can easily draw a straight line which divides the plot of the data into two categories, orange points and blue points. In the second data set, it is not so easy to do that.

To get around this problem, I manipulated the data so that my implementation of the linear regression algorithm could more easily solve the problem. I created another feature in the data set which i called x2. The x2 feature is the absolute value of x1 +x2. I ran that data through the algorithm and it proved to give me good results. This made me go from 0.6 in accuracy and 15 in cross entropy to 0.992 in accuracy and o.2 in cross entropy.

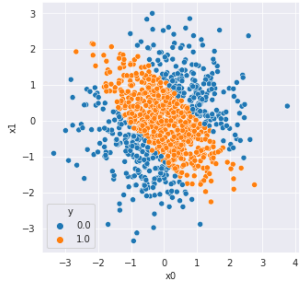

Figure 2: plot raw data

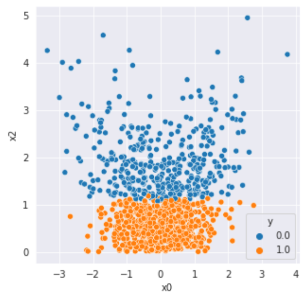

Figure 3: plot manipulated data

2 K-means

2.1 Description of K-means

K-means belongs to a category of learning algorithms called unsupervised learning algorithms. The objective of the k-means algorithm is to take an amount of data and organize them into clusters where each cluster contains a number of observations. Since the K-means algorithm belongs to the unsupervised learning family, it is well suited for problems where we do not have a variable to measure our features against. It tries on its own to find a pattern in the data its given. In today's world, K-means is widely used in crime detection, customer demographics and similar problems.

2.2 My implementation

My implementation starts by creating k=2 random centroids from the data. I decided to opt for a hard-coded k instead of letting my program find the best number to assign k to. This is for simplicity reasons. My algorithm then iterates through every point in the data frame and groups the points who are closest to each centroid into clusters. The distance between a point in the data frame and the centroid of a cluster is calculated using the euclidean distance function provided in the samplecode with the following formula.

$$d = \sqrt{(c_1 - c_2)^2 + (p_1 - p_2)^2}$$

Once all the points have been assign a centroid, I recalculate the centroid for each cluster based on the points in its respective cluster. For my algorithm, 4 iterations of this process seemed to work fine on both data set 1 and data set 2. Here, I decided again to hard-code the amount of iterations my K-means algorithm would run before it would stop. For further improvement, I would have checked if the algorithm converged to where the centroids don't move at all after one more iteration.

2.3 Inductive bias

For my K-means algorithm to generalize beyond the training data, there are some intrinsic properties that the data should have in order for the model to fit well to the data. The clusters have to be of similar size. If they are not, the algorithm will give bad results. In worst case, the points that visually belongs to one cluster will be placed in another cluster because the euclidean distance is shorter.

My K-means algorithm is also very sensitive to outliers. The centroid of a cluster can be shifted heavily if the outlier is far enough away and the density of the cluseter is low enough.

2.4 2nd data set

The first data set that I applied my K-means algorithm to only had two clusters. My algorithm ran fairly quick and I got good results in the beginning. My silhouette Score was 0.672 and my distortion: 8.837.

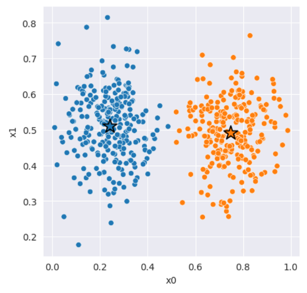

Figure 4: K-means applied to data set 1 with k=2

The second data set was a bit more difficult to solve because it was a bit more difficult to see how many clusters I would need get get good results on my algorithm. I modified my algorithm so it would handle k=10 clusters which proved to give good results in the end. My distortion score was 6.612 and my silhouette Score was 0.575.

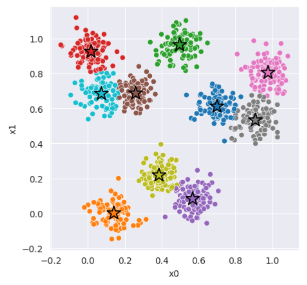

Figure 5: K-means applied to data set2 with k=10