Adaline Algorithm – NN

We create two-dimension random data <x, y> in size 1,000, with range (-1, 1).
Than classified each sample to "1" or "-1" such that if x and y both bigger than 0.5 they get the value "1", if-else, "-1".

To implements the Adaline algorithm we use the function:

- **fit()**
  Function to train our model, for every iteration we update the weights.

- **update_weighte()**
  Updating weights vector by LMS algorithm:

  $Wi_{(new)} = Wi_{(old)} + a * (y - output) * Xi$

  - y – desired output
  - output – sum Xi, Wi
  - a - Learning rate

- **shuffle()**
  Shuffle the training data.

- **net_input()**
  Calculate net input with matrix multiplication.

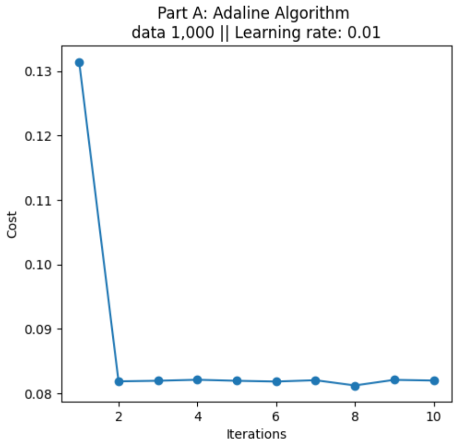
- **activation()**
  Passive function for linear activation.

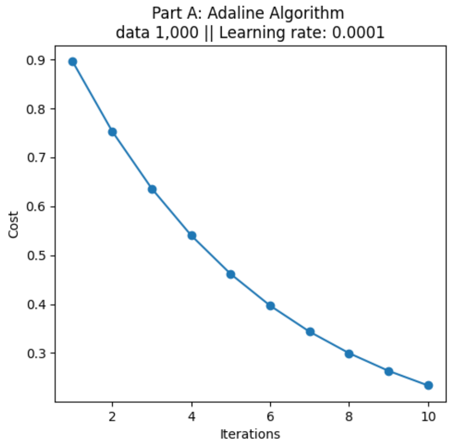
- **predict()**
  Use the model to predict the correct answer, by multiply Wi with Xi.

- **score()**
  Return our success rate of our model by percent.

Part A:

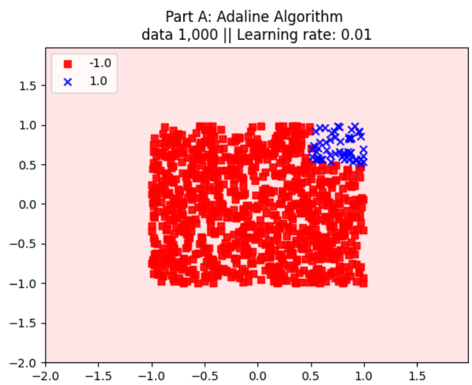
In this part we will use data with 1000 samples and compare different learning rates:

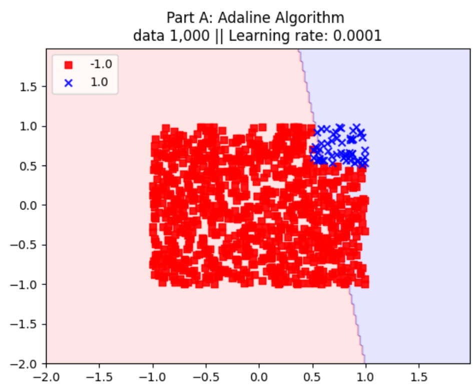


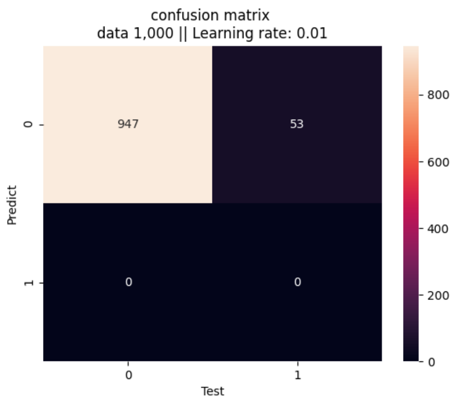


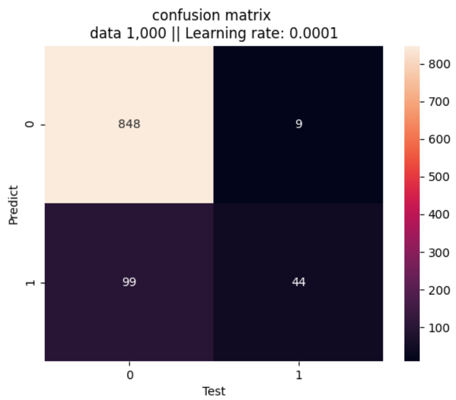
We can see the importance of choosing the correct Learning rate. As much as the Learning rate is smaller we'll need more iteration to decrees the cost (error^2 /2).

Real data output vs our model prediction.
We can see that with a learning rate of - 0.01 our model got **96.7%** of good prediction by classification all of the data as "-1". At the same time with a learning rate of – 0.001 our model got **89.2%** of good prediction by classification 85.6% as "-1" and the rest as "1".



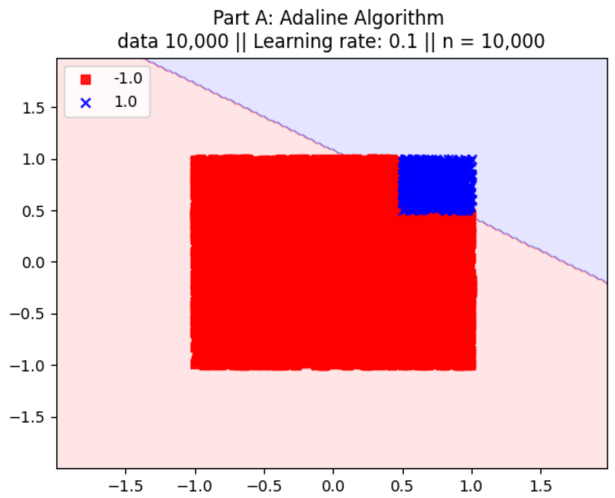


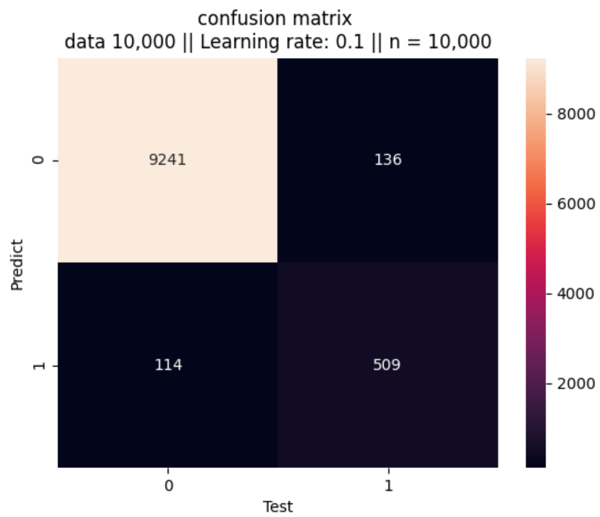




If the data would be of the size of n/10,000 (with -10000 <= n <= 10,000) we will choose bigger data set and a bigger learning rate with the same number of iteration.

For example, we increase our data to 10,000 instead of 1,000 and our learning rate to 0.1 instead of 0.01, both with 10 iterations. We can see that as much as our distribution is bigger we need bigger data to get closer to the average distribution. That way we need to increase our data and learning rate to train on a larger range of possible values.





The score of this model is: **97.5%**.

Part B:

After we change the problem,
so that points such that <x, y> has value 1 only if 1/2 <= x**2 + y**2 <= 3/4.

The best results we obtain using an Adaline are **80.16%** by using data size = 100,000 instead of **79.2%** of success by using data size = 1,000. The results become better when we used more data because our model classified all data as "-1". Because there is no average distribution as more as we have more data we have more values in the "-1" area.
In conclusion, our model doesn't become any better while using more data. Adaline is a linear model and there is no possible way to classified values with high success rates, which are scattered in a nonlinear way.

In this part we will use learning rate 0.0001 and changing data size: