

ISI

Lab 9

Daniel Martínez Sánchez

9th May, 2023

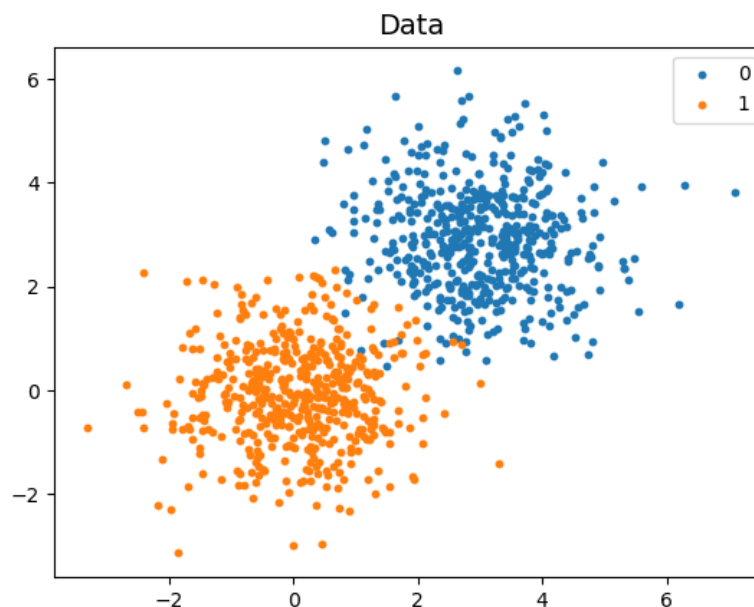
Perceptron

Dataset generation

Consider we create a dataset with the following code. What is the dataset like?
`X,y=make_blobs(n_samples=1000,n_features=2,centers=[[3,3],[0,0]],cluster_std=1,random_state=2)`

The given command creates a 2D dataset with 1000 data points with 2 features each and a class (either 0 or 1).

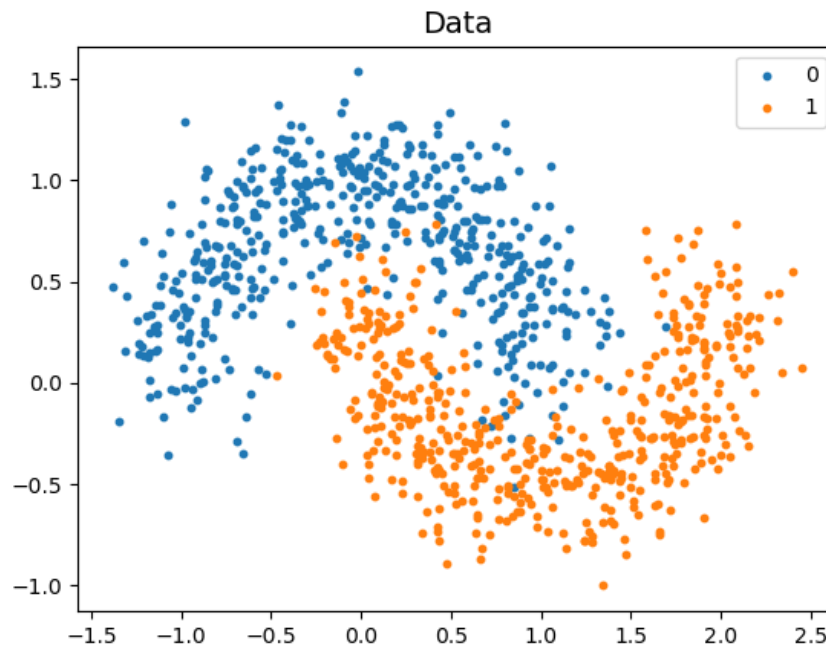
The data points are generated from 2 Gaussian clusters with centers located at [3,3] and [0,0], respectively, and a standard deviation (noise) of 1. We can clearly see in the image the two specified centers.



Consider we create a dataset with the following code. What is the dataset like? `X,y = make_moons(n_samples=1000,noise=0.2, random_state=0)`

The given command creates a 2D dataset with 1000 data points with 2 features each and a class (either 0 or 1).

The data points are generated using the moons dataset generator which creates two circles that are intertwined with each other, forming a moon shape. The noise parameter controls the amount of random noise added to the data points, which in this case is 0.2.



Training and evaluating a Perceptron

What is the accuracy achieved on the test set?

The accuracy achieved on the test set is 0.9333333333333333.

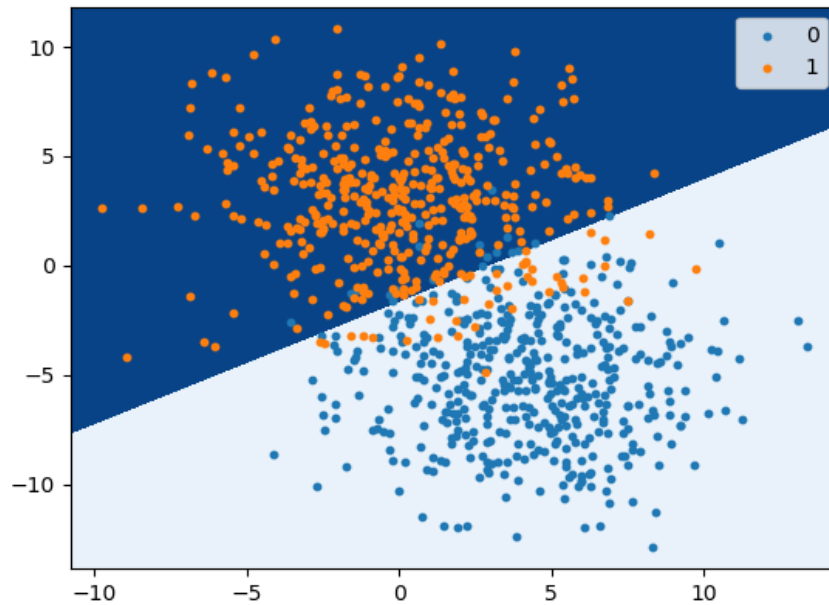
Do you consider we could train a model that classifies this dataset with accuracy close to 100%?

If we are talking about the simple Perceptron model, it is highly unlikely that this model is capable of classifying with near 100% accuracy. This is mainly due to the presence of noise and outliers that have a great impact on the performance of the Perceptron model.

Other models may have better luck with this dataset, specially more complex ones such as decision trees that can better handle noise.

What is the decision boundary like? Specify whether it is linear or nonlinear.

The decision boundary is linear as we can see in the image.

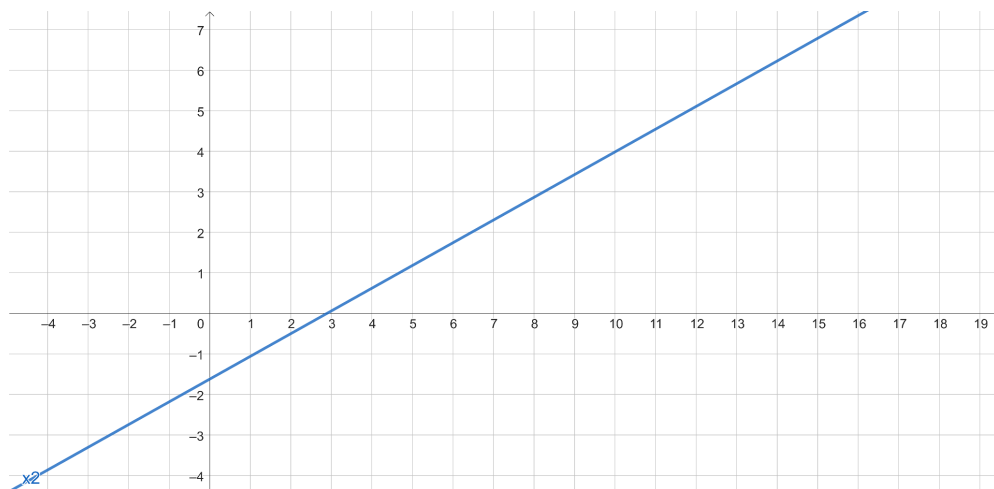



Weights of the perceptron after training:

- $W_0 = 11.0$
- $W_1 = -3.79868488$
- $W_2 = 6.77647054$

Compute manually the decision boundary and check it matches the one you get with `vis_classification_2D (X_test,y_test,clf)` in the code.

- $W_1 * x_1 + W_2 * x_2 + W_0 = 0$
- $x_2 = (-W_1 * x_1 - W_0) / W_2$
- $x_2 = (3.79868488 * x_1 - 11) / 6.77647054$





Now, change the dataset using the `make_moons` function instead of the `make_blobs` one. Using `make_moons` create a dataset with 3000 samples and add noise with standard deviation of 0.3.

What is the accuracy achieved on the test set for the moons dataset?

The accuracy achieved on the test set is 0.8155555555555556.

Do you consider we could train a perceptron that performs much better on the moons dataset?

No. The moons dataset is a nonlinear dataset with overlapping clusters, therefore the accuracy using a simple Perceptron will always be low even if we optimize its hyperparameters because the classifier always assumes a linear decision boundary.

A MultiLayer Perceptron on the other hand can provide better results because it can be trained to classify non-linearly separable datasets.

**Do you consider we could find a model that performs better on the moons dataset?
What kind of decision boundary should this model define?**

As I previously stated, a nonlinear model such as a decision tree, random forest or a MultiLayer Perceptron should perform better on this particular dataset. The model we choose should be able to define a nonlinear decision boundary that captures the complex structure of the dataset.

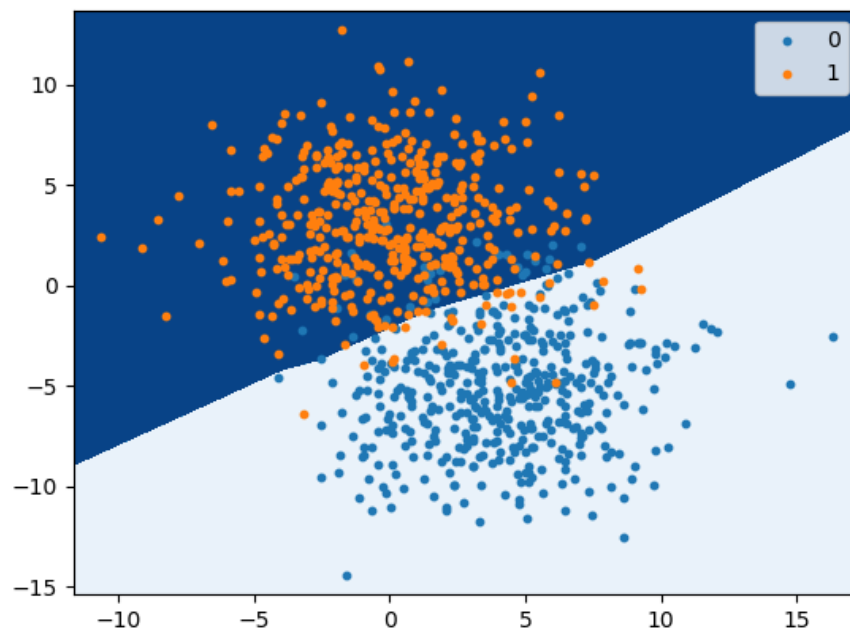
MultiLayer Perceptron

What is the accuracy achieved with the MLP on the test set for the gaussian data?

The accuracy achieved on the test set is 0.9388888888888889.

What is the decision boundary like? Specify whether it is linear or nonlinear.

The decision boundary is nonlinear because it cannot be represented by a linear equation.



Is the performance achieved with the MLP higher than the Perceptron performance computed in the previous section?

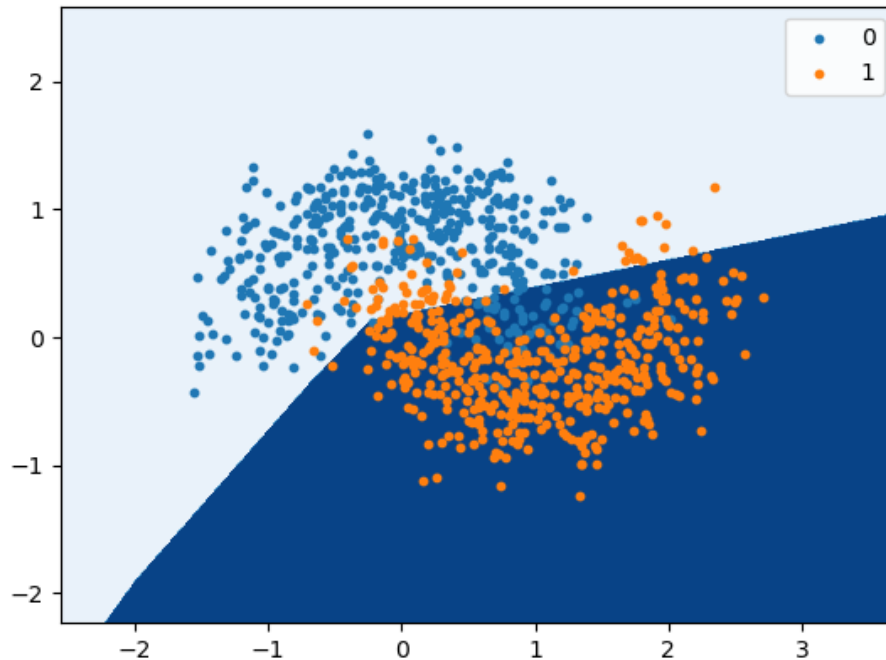
The performance is marginally better for the MLP, but this is expected because the gaussian dataset is a linear dataset and the simple Perceptron is designed to handle this kind of datasets.

Now, modify the code changing this dataset for the moons one and use the MLP to classify the moons dataset. What is the accuracy achieved with the MLP on the test set for the moons dataset?

The accuracy achieved on the test set is 0.8666666666666667.

What is the decision boundary like? Specify whether it is linear or nonlinear.

The decision boundary is nonlinear because it cannot be represented by a linear equation.



Could you propose any other classifier based on MLP that gives better performance?

```
clf = MLPClassifier(  
    hidden_layer_sizes=(5,),  
    max_iter=1500,  
    tol=1e-5,  
    activation="relu",  
    solver="adam",  
    verbose=True,  
    random_state=1,  
)
```

Improved accuracy: 0.93

Perceptron Mushroom

How many examples does the whole dataset have? How many classes are there?. Indicate the number of examples for each class.

8124 examples in total.

There are two classes:

- Edible: 4208.
- Poisonous: 3916.

Based on the previous result, explain the content of the variables “X_train”, “X_test”, “y_train”, “y_test”.

- The X_train and X_test variables contain information about the characteristics of mushrooms used for training and testing the Perceptron classifier.
- The y_train and y_test variables contain the labels (edible or poisonous) of the mushrooms in the training and test sets.
- The data was divided into training and test sets with 30% of the data reserved for testing.
- This split was performed in a way that keeps the proportion of edible and poisonous mushrooms consistent in both sets (stratified).

Is the perceptron trained during the maximum number of iterations?

Yes, the training stopped after 24 epochs because it achieved convergence.

Based on the results indicate:

- Accuracy = 0.9089417555373257
- TP (True Positives) = 975
- TN (True Negatives) = 1241
- FP (False Positives) = 200
- FN (False Negatives) = 22

MultiLayer Mushroom

Is the MLP trained during the maximum number of iterations?

No, it reaches the maximum number of iterations (150) and the optimization hasn't converged yet, so it stops.

Based on the results indicate:

- Accuracy = 0.9495488105004102
- TP (True Positives) = 1057
- TN (True Negatives) = 1258
- FP (False Positives) = 5
- FN (False Negatives) = 118

Try to improve the performance for the MLP classifier by modifying the network architecture, number of training cycles or solver. Indicate the best configuration you have found:

- hidden_layer_sizes: (5, 5)
- max_iter: 500
- activation: "relu"
- solver: "adam"
- tol: 1e-5

Based on the results indicate:

- Accuracy = 1.0
- TP (True Positives) = 1175
- TN (True Negatives) = 1263
- FP (False Positives) = 0
- FN (False Negatives) = 0