

Submission by **Matan Ben Nagar & Yaara Bark**

[The entire project can be found on our Github repository:  
<shorturl.at/lprzY>]

Added to the submission are two files:  
1. **Logistic Regression.ipynb**  
2. **MLP.ipynb**  
You can press “Run all” and get the results that we got here

**Handling Dataset**

In this assignment, my partner and I have tried to establish a connection between different physical attributes of a person, to predict his sex. We were using the Cardiovascular Disease dataset from Kaggle (provided in this link: <https://www.kaggle.com/sulianova/cardiovascular-disease-dataset> ).

Dataset:

* Our dataset contains 70,000 examples
* At first we took the only columns that were relevant for our research such as: [age, gender, height, weight, smoke]

First steps we implemented:

1. Data cleaning:  
   - Removing unnecessary features that supposedly don’t have impact or relation to what we were trying to predict (such as cholesterol, gluc and so on… )

- Detect duplicates rows in our dataset and remove them

1. תמונה שמכילה שולחן

   התיאור נוצר באופן אוטומטיData preprocessing:

- Detecting Null values, and if such exist remove the entire row

- Standarization: Converting column data so that it ranges around the same values as other columns. Bring the values close together.

1. Normalization: Making sure all data in columns if of the same type (integer)

**Logistic Regression**

Gender classification is a classification problem, because of that we chose Logistic Regression as our predictive algorithm.

**Understanding the data**Our first challenge was to realize how our data dimensions are supposed to look like. We also struggeled to under what the tf.placeholder, tf.Variable functions do. After we gained some deeper understandig it became clear.

**Y\_data**

**X\_data**

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי  
  
We then used pandas to split our database into the feature data and label data.

**Loss and Updating**For the loss function we used the cross entropy function which measures **the probability error in discrete classification tasks in which each class is independent and not mutually exclusive**.  
Our update function used the GradientDescentOptimizer which aims to minimise the value of the error Variable, which is defined earlier as the square of the differences (a common error function). The 0.01 is the step it takes to try learn a better value.

**Training – Steps and Learning rate**

* Finally we started training our model, while printing the loss we get each round. To our surprise, sometime the loss was going down and sometimes it was going up. We realized that our step was too big, and so we attempted to make it smaller. The issue was fixed and the loss was steadily going down.
* The pace at which the loss was going down was too slow and we had to increase the total number of steps in order to get a smaller loss.

**Testing**At the end of the training, we then split the remaining data into X\_test, Y\_test and sampled the results. To create a confusion matrix, we saved our predictions inside an array that held 2 values: {0,1}. Everytime the logistic regression returned a values, we checked if it was bigger or smaller than 0.5.  
We have been at this proccess a few times, each time trying to change some of the variables, in order to get better results.

**תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטיResults**Confusion Matrix: **תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי**



**Neural Network**

In this part we chose to use MLP(multilayer perceptrons).

**Number of Neurons**  
As for the Neural Network, we used the same learning rate, optimizer function and loss function, with additional a hidden layer - with 10 neurons, for each one of the neurons we created number of weights as the number of features (In this case we have 4 features)  
**We had different attempts at the hidden layer variable** and from what we gathered, it is recommened to place more neurons than the numbers of features, but less than 2 times the features. So we were training our model with 10,8,7,6,5 neuron.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 10 Neurons | 8 Neurons | 6 Neurons | 5 Neurons |
| Accuracy | 0.6684 | 0.67415 | 0.6702 | 0.67475 |
| Loss | 0.6168465 | 0.6122621 | 0.6203663 | 0.6173222 |
| Confusion  Matrix |  | תמונה שמכילה שולחן  התיאור נוצר באופן אוטומטי | תמונה שמכילה שולחן  התיאור נוצר באופן אוטומטי |  |

**Keeping it fair:**  
While training this MLP model, we used a similar number of steps and learning rate (same as we did in the Logistic Regression part) all so that we could make a comparison based on the algorithm itself and not because we trained this model more than the other.

**Results**

**תמונה שמכילה טקסט

התיאור נוצר באופן אוטומטי**Best results were achieved with 7 neurons

תמונה שמכילה שולחן

התיאור נוצר באופן אוטומטי

|  |  |  |
| --- | --- | --- |
| Model | Test acuracy | Train loss |
| Logistic Regression | 67% | 0.8832 |
| MLP | 70% | 0.5947 |

**Logistic Regression VS MLP:**