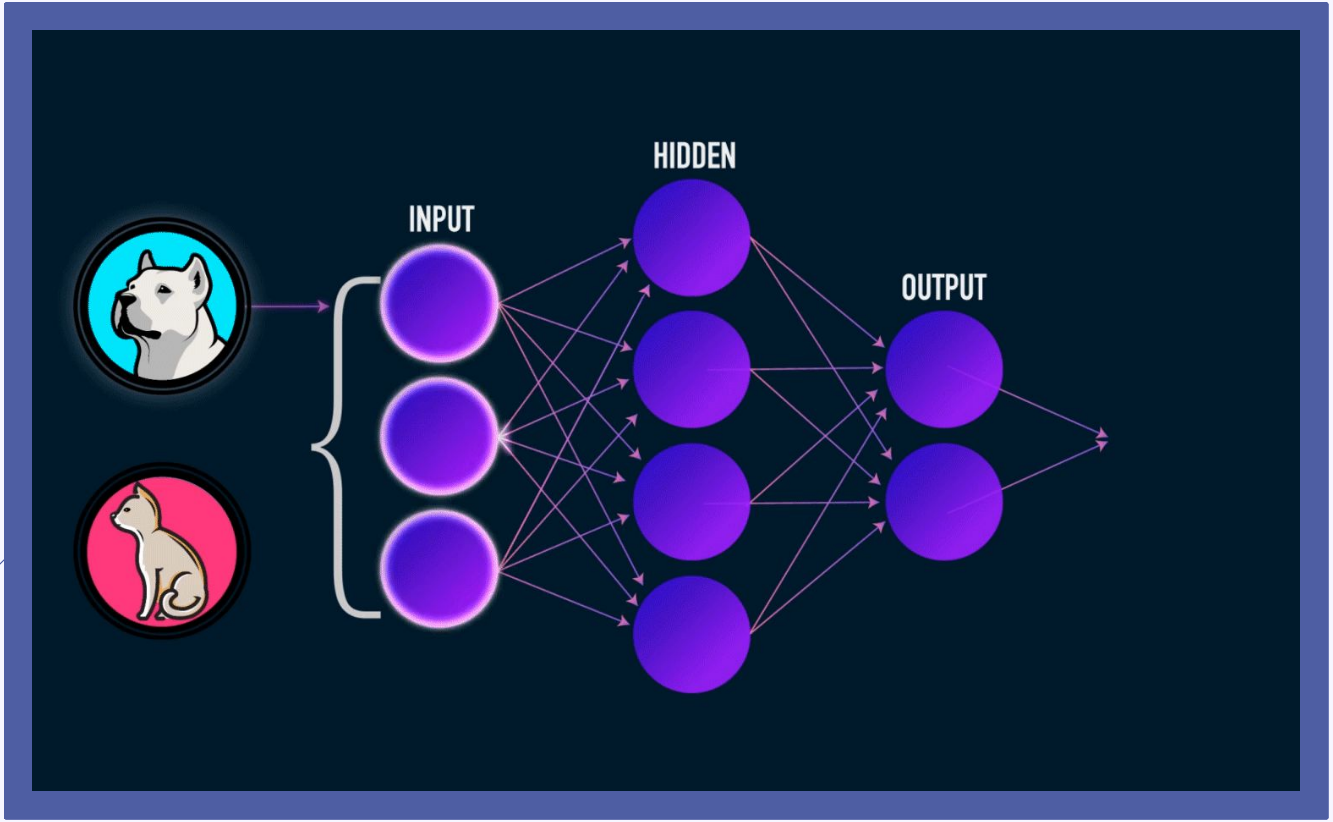
Multi-layer Perceptron **(MLP)**

Feedforward Artificial Neural Network (ANN)

Summary

1. Introduction
2. MLP implementation
3. Results
4. Conclusion



Example of an articial neural network

https://149695847.v2.pressablecdn.com/wp-content/uploads/2018/12/nural-network-banner.gif

Multi-layer Perceptron

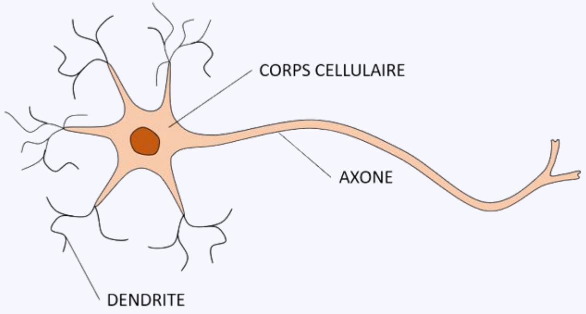


Image 1 : Representation of a biological neurone

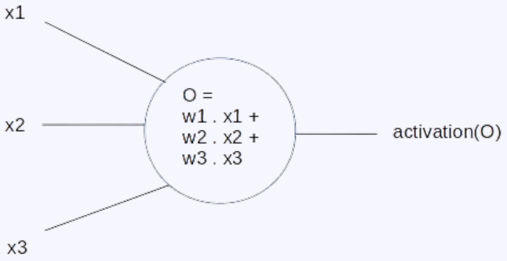


Image 2 : Representation of an artificial neurone

An MLP is an **Artificial Neural Networks (ANN)**

Inspired by **biological neural networks**

<u>Applications:</u>
Speech and image recognition, text processing…

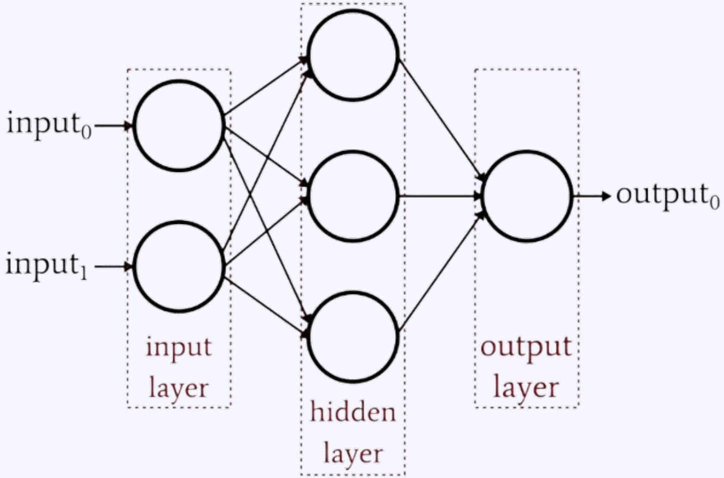
At least **3 layers** of nodes: an **input** layer, a **hidden** layer and a **output** layer.

Learns by **training** on a dataset

Image 1: https://svtdiderot.fr/cordewener/4eme-cordewener/le-systeme-nerveux/

Image 2: https://cdancette.fr/assets/neuron.png

Multi-layer Perceptron



<https://www.allaboutcircuits.com/uploads/articles/an-introduction-to-training-theory-for-neural-networks_rk_aac_image2.jpg>

An MLP is an **Artificial Neural Networks (ANN)**

Inspired by **biological neural networks**

Applications:
Speech and image recognition, text processing...

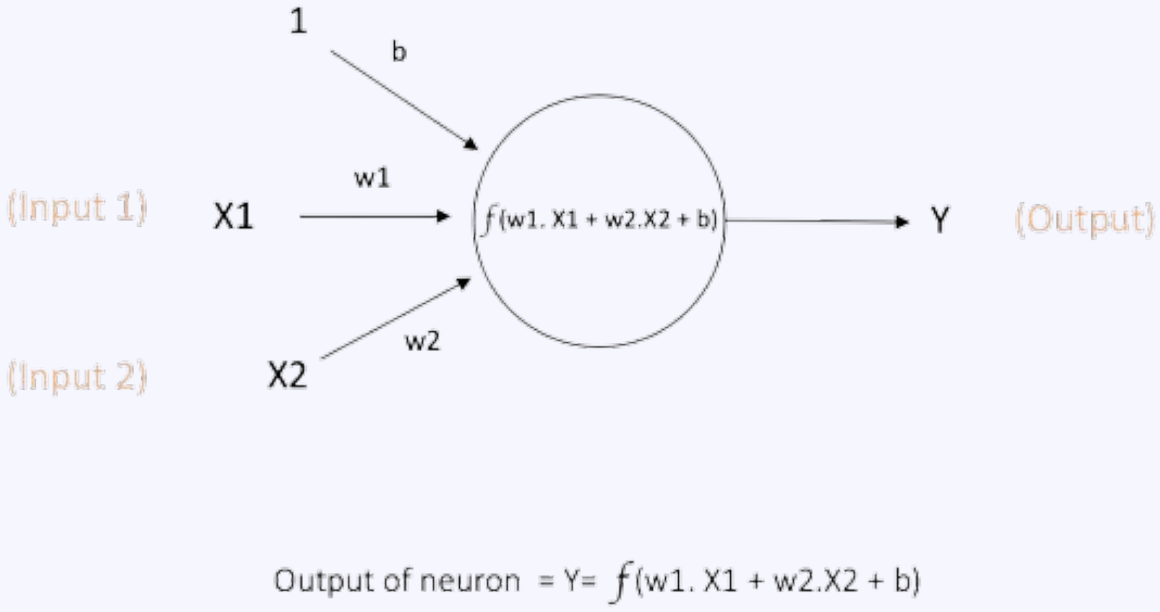
At least **3 layers of nodes**: an **input** layer, a **hidden** layer and a **output** layer.

Learns by **training** on a dataset

A single neurone

Also called **node** or **unit**

- **Inputs**: X1, X2, Bias 1
- **Weights**: w1, w2, b
- **Output**: Y
- **Activation function**: non-linear function



https://ujjwalkarn.me/2016/08/09/quick-intro-neural-networks/

1. Introduction

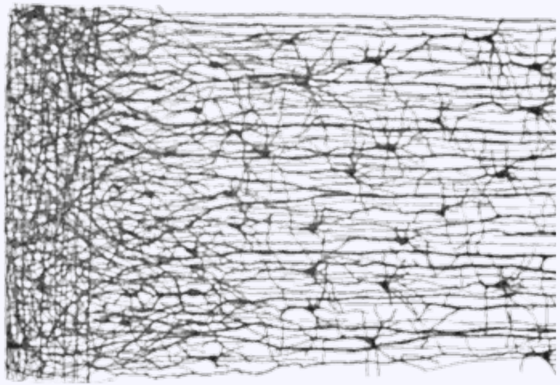


Image 4: Multiple layers in a biological neural network

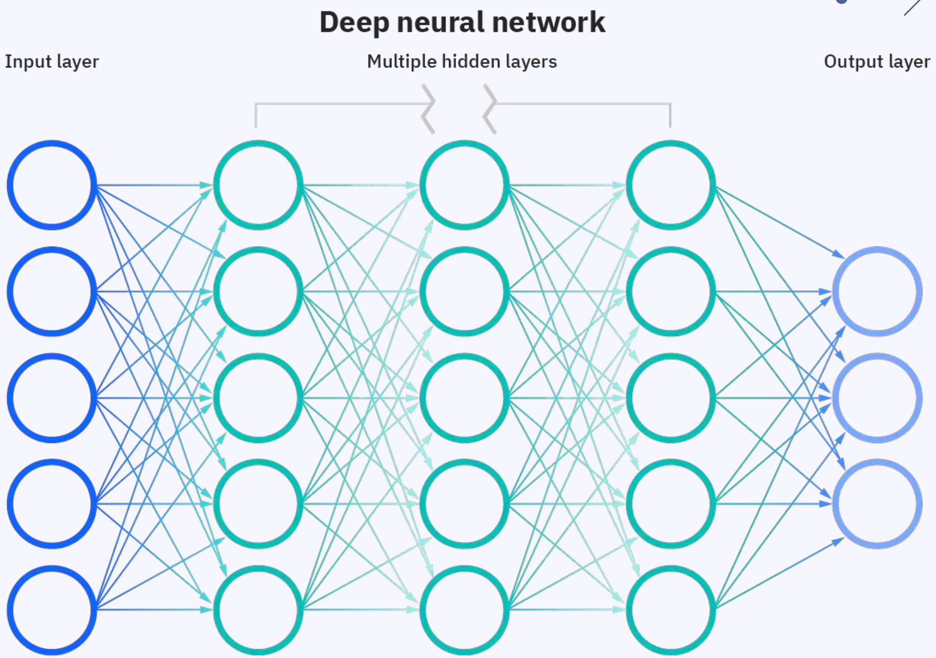
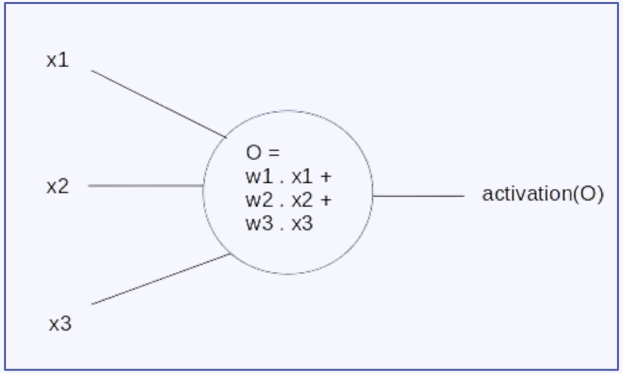


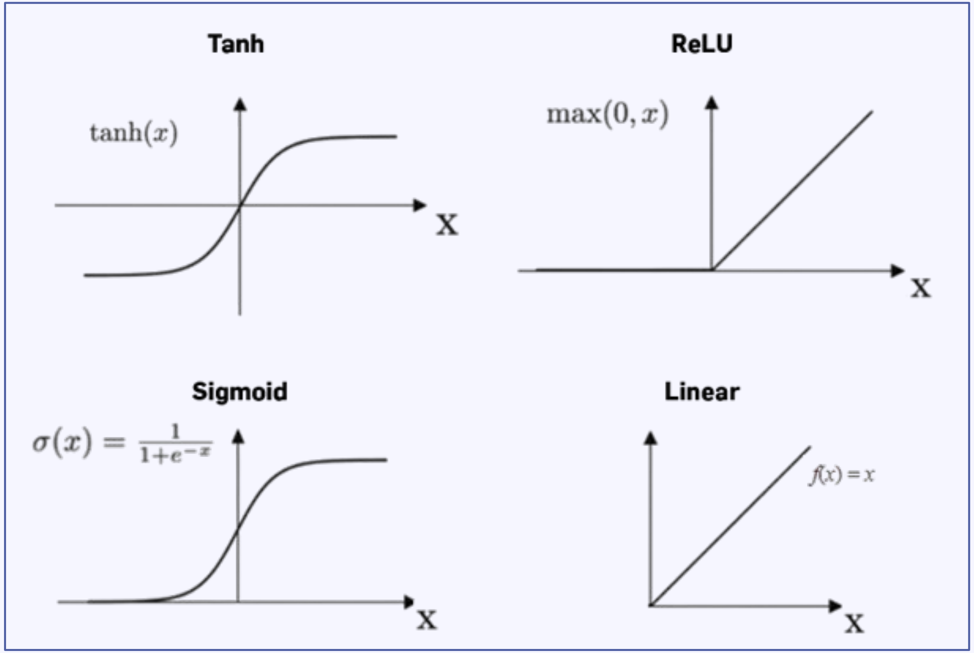
Image 5: Multiple layers in an artificial neural network

https://www.oreilly.com/api/v2/epubs/9781492037354/files/assets/mlst_1002.png

https://www.ibm.com/cloud/learn/neural-networks

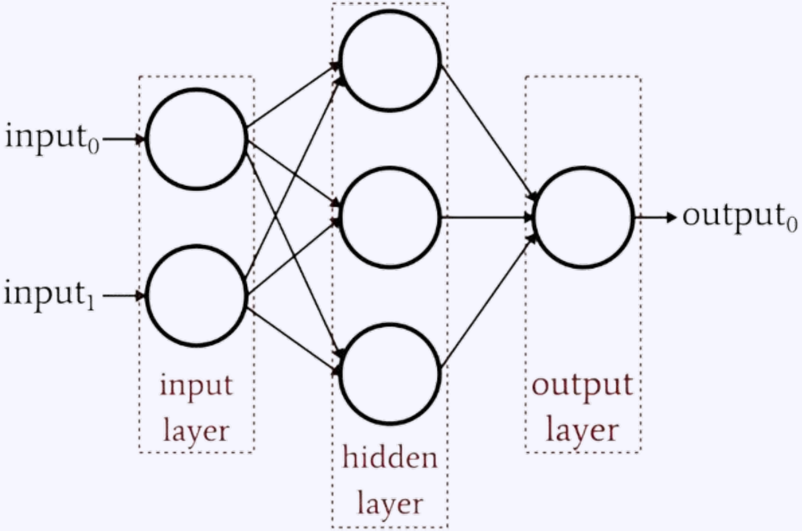
Activation function

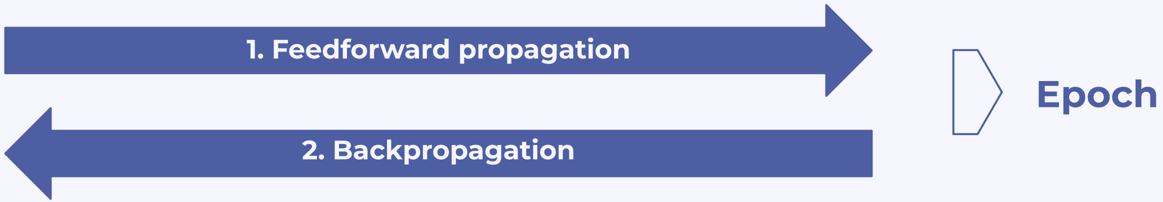




https://machine-learning.paperspace.com/wiki/activation-function

Feedforward and Backpropagation





https://www.allaboutcircuits.com/uploads/articles/an-introduction-to-training-theory-for-neural-networks_rk_aac_image2.jpg

Requirements

Pandas

Open csv file

Numpy

Operation and matrix

Scikitlearn

Model implementation
Cross-validation

Matplotlib

Data visualization

Seaborn

Heat Map

Pam50 dataset: It contains the expression of 50 known human genes with patterns according to a subtype of breast cancer.

Data importation

```
Data_Pam = pd.read_csv("data/Data_Pam50.csv")
```

	Unnamed: 0	subtype	ACTR3B	ANLN	BAG1	BCL2	BIRC5	BLVRA
0	Normal.Breast.10	Normal	-1.151	-3.736	0.260	1.300	-2.860	-0.569
1	Normal.Breast.2	Normal	-0.485	-3.739	0.591	1.580	-3.250	-0.533
2	Normal.Breast.3	Normal	0.298	-2.848	0.359	1.292	-2.493	-0.687
3	Normal.Breast.4.Custom	Normal	1.153	-4.717	0.098	1.954	-3.237	-0.535
4	Normal.Breast.7	Normal	-0.287	-3.681	0.441	1.911	-2.156	-0.965
...	...	...	...	...	...	...	...	...
67	H1AUNC.1319.C	LumB	-0.267	-0.803	-0.409	0.329	0.179	1.542
68	H1AUNC.1323.C	LumB	-0.938	-1.999	-0.910	0.472	-1.564	0.352
69	H1AUNC.1462.C	LumB	-0.950	-1.757	0.250	0.099	-1.289	1.219
70	H1AUNC.1471.C	LumB	0.039	-1.937	0.095	0.137	-0.948	2.833
71	H1AUNC.1474.C	LumB	-1.441	-2.706	-1.142	0.393	-1.330	0.333

72 rows × 52 columns

Portion of the data frame

Slicing

```
label_raw = Data_Pam.iloc[:,1]
data_raw = Data_Pam.iloc[:,2:]
```

0	Normal
1	Normal
2	Normal
3	Normal
4	Normal
	...
67	LumB
68	LumB
69	LumB
70	LumB
71	LumB

Raw Labels

+

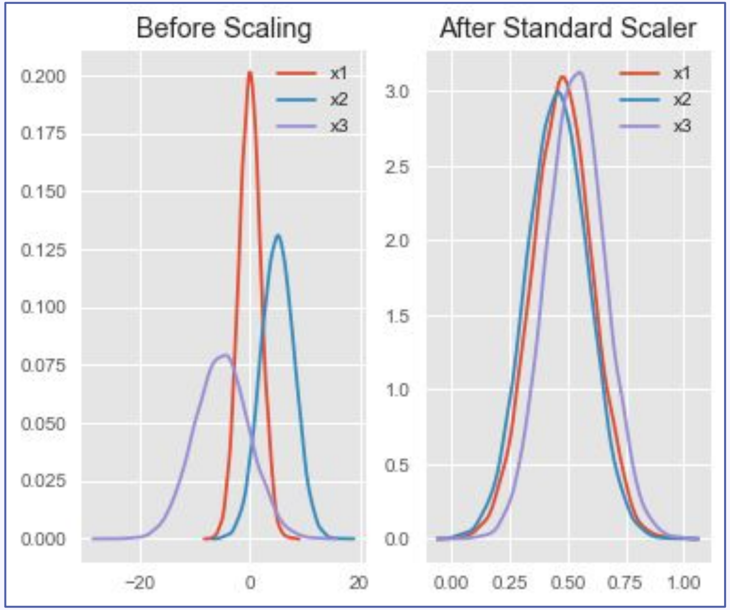
	ACTR3B	ANLN	BAG1	BCL2	BIRC5	BLVRA	CCNB1	CCN
0	-1.151	-3.736	0.260	1.300	-2.860	-0.569	-2.981	-1.9
1	-0.485	-3.739	0.591	1.580	-3.250	-0.533	-2.935	-1.0
2	0.298	-2.848	0.359	1.292	-2.493	-0.687	-2.810	-1.1
3	1.153	-4.717	0.098	1.954	-3.237	-0.535	-3.558	-2.1
4	-0.287	-3.681	0.441	1.911	-2.156	-0.965	-2.869	-1.1
...	...	...	...	...	...	...	...	
67	-0.267	-0.803	-0.409	0.329	0.179	1.542	-0.854	-1.1
68	-0.938	-1.999	-0.910	0.472	-1.564	0.352	-1.842	-1.5
69	-0.950	-1.757	0.250	0.099	-1.289	1.219	-1.135	-1.2
70	0.039	-1.937	0.095	0.137	-0.948	2.833	-1.669	-0.9
71	-1.441	-2.706	-1.142	0.393	-1.330	0.333	-1.415	-1.3

72 rows × 50 columns

Raw Data

Data Scaling

```
from sklearn.preprocessing import StandardScaler
sc_X = StandardScaler()
X_trainscaled=sc_X.fit_transform(X_train)
X_testscaled=sc_X.transform(X_test)
```

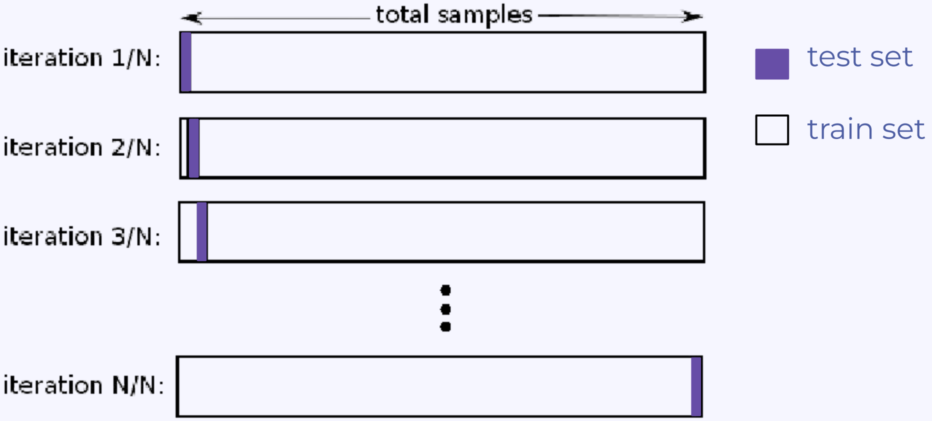


Example of scaling with StandardScaler

http://benalexkeen.com/feature-scaling-with-scikit-learn/

Cross-validation

```
loo = LeaveOneOut()
for train, test in loo.split(data_scaled):
    clf = MLPClassifier(hidden_layer_sizes=layer,
        random_state=1, max_iter=epochs, tol = tols).fit(data_scaled[train],
                                                          label_target[train])
```



https://www.researchgate.net/publication/266617511_Comparison_of_Statistical_Methods_for_Genomic_Signature_Extraction

Parameters and GridSearchCV

```
layer = [(100,1),(200,1),(100,10),(200,10),(100,20),(200,20),(50,10)]
tols = [10e-4, 10e-5, 10e-3, 10e-2, 10e-6, 10e-7]
epochs = [100,50,20,200,300]
```

```
clf = GridSearchCV(
    MLPClassifier(),
    param_grid= {
        'hidden_layer_sizes': layer,
        'tol' : tols,
        'max_iter' : epochs
        },
    refit='True',
    cv=2,
    n_jobs=-1,
)
```

MLPClassifier parameters:

- *hidden_layer_sizes*
- *max_iter*
- *tol* (tolerance)

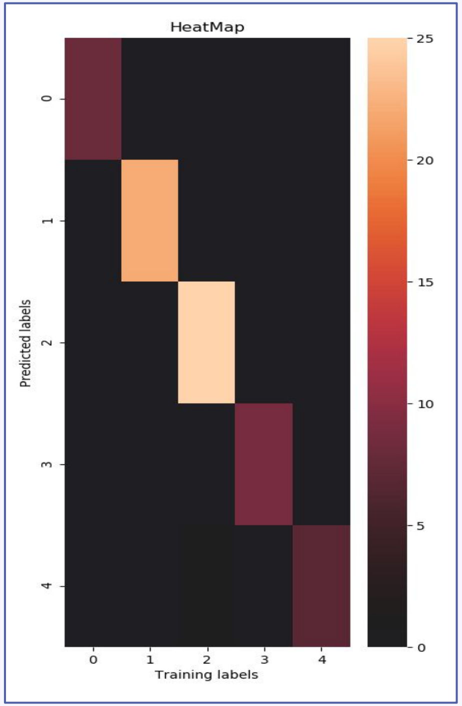
Heat Map

Repartition of the labels

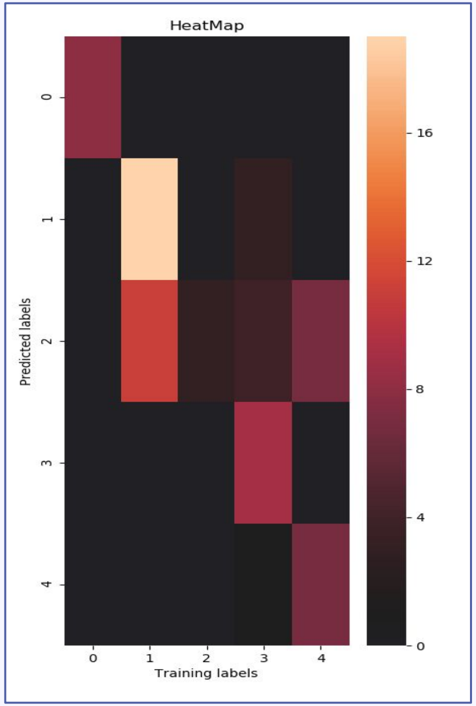
→ Each number is a category for a label

A Heat Map with high accuracy:

- First predicted labels should match with first true labels, etc.



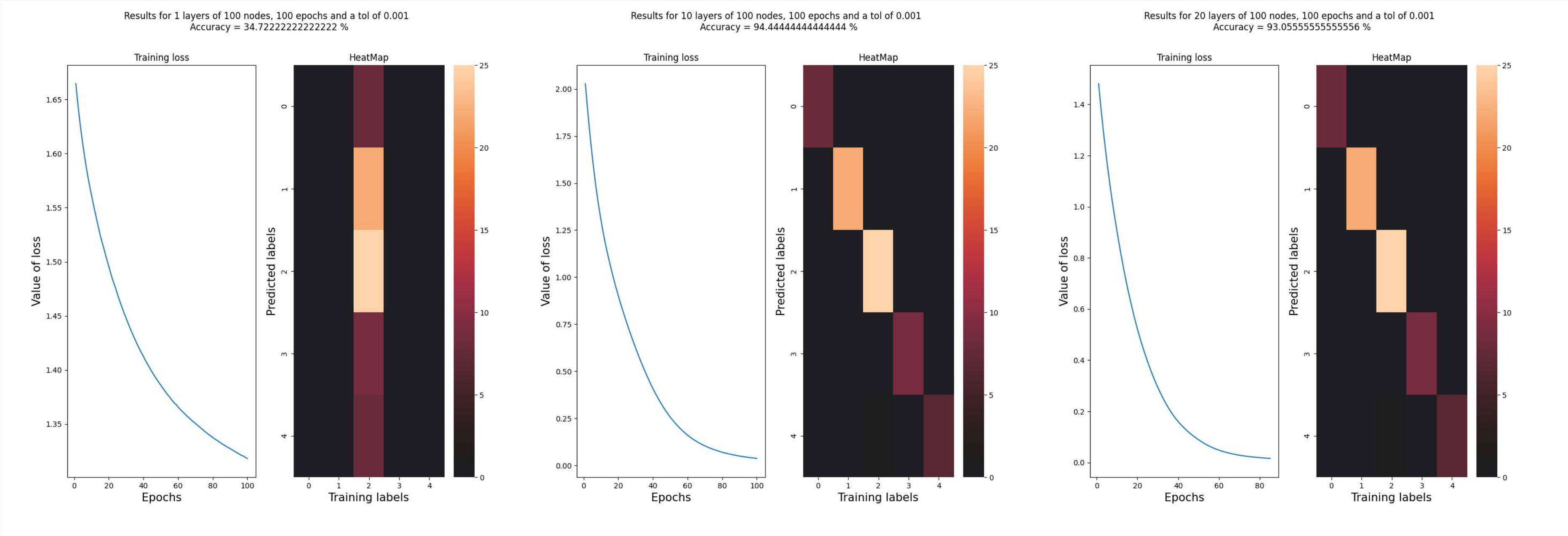
Heat map with high accuracy



Heat map with low accuracy

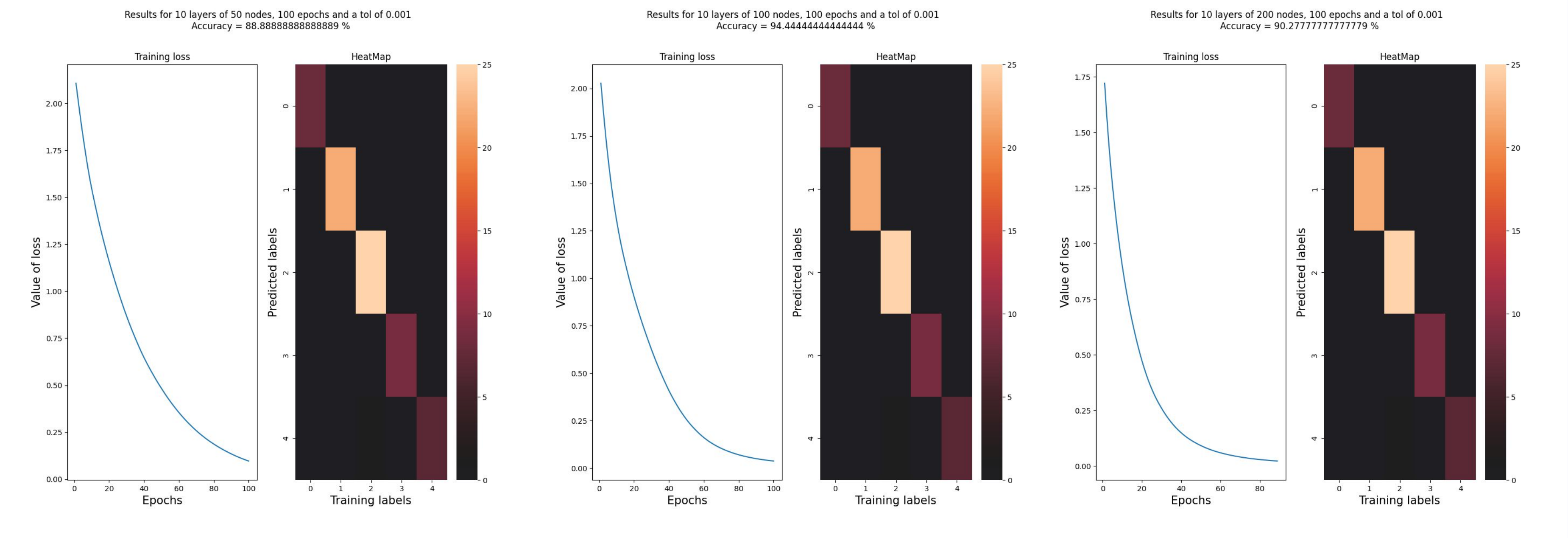
3. Results

1st parameter : size of network



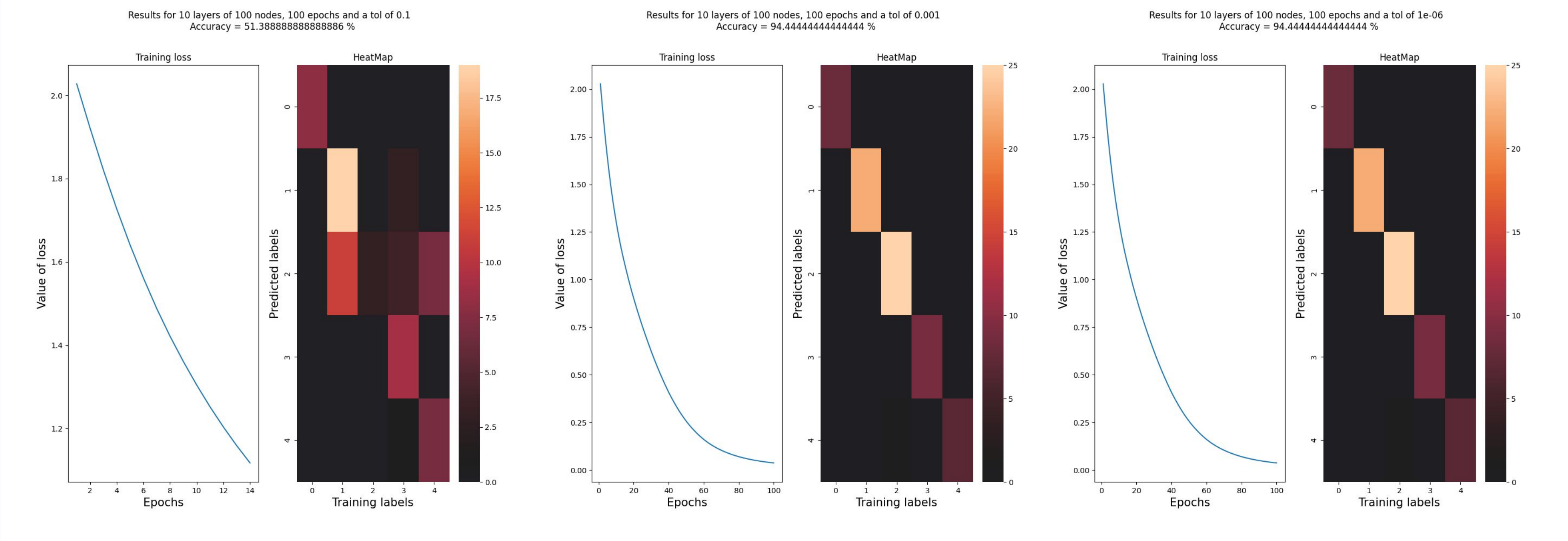
3. Results

1st parameter : size of network

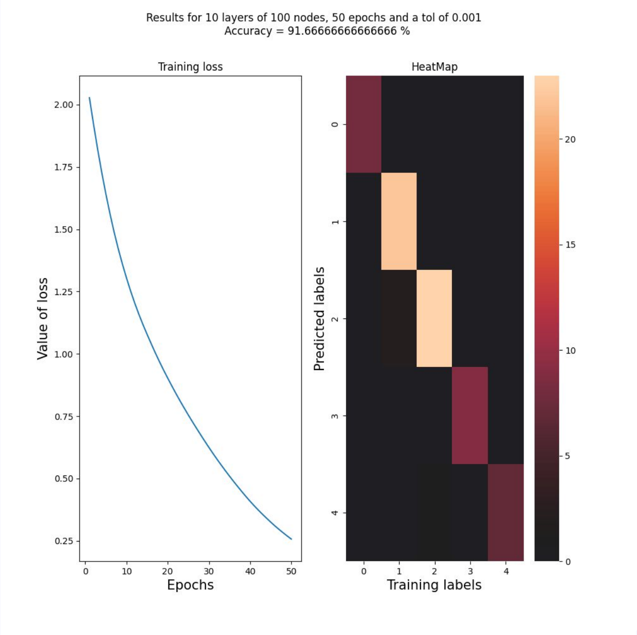


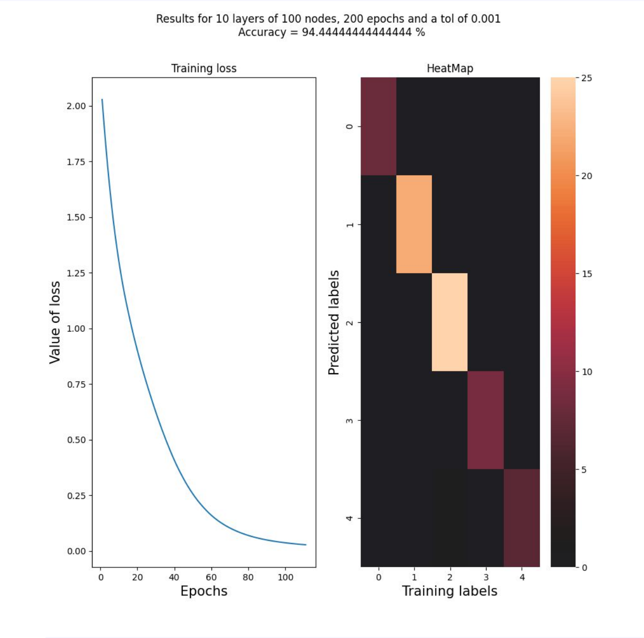
3. Results

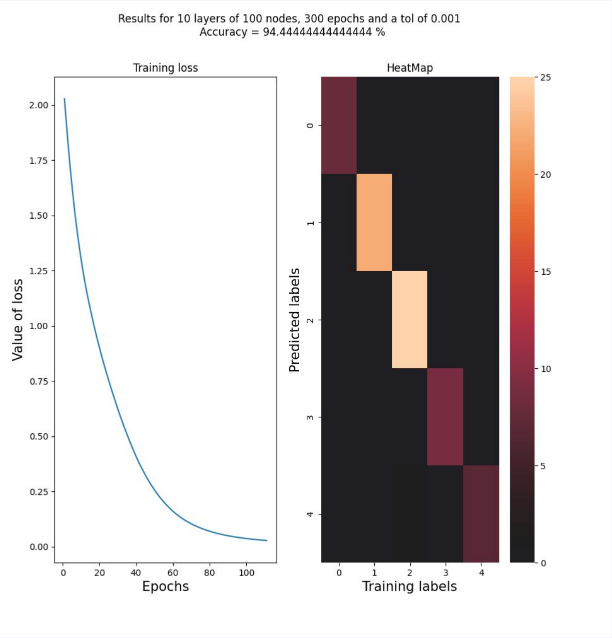
2nd parameter : tolerance



3^{rd} parameter : number of epochs

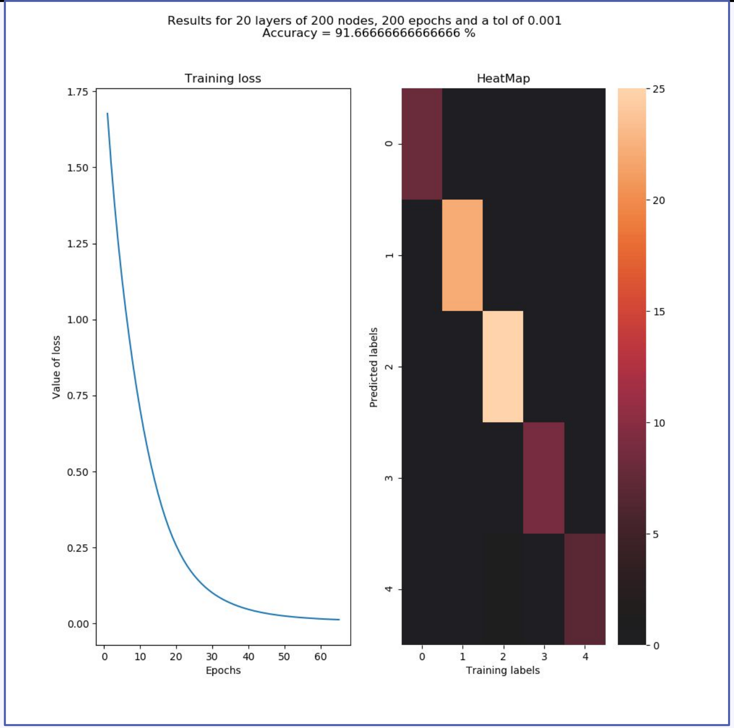






GridSearchCV

```
{'hidden_layer_sizes': (200, 20), 'max_iter': 200, 'tol': 1e-05}
```



4. Conclusion

Multi-layer Perceptron : at least 3 layers (input, hidden, output)

Epoch : feedforward propagation and backpropagation

Scaling is important

Cross-validation used for small datasets

Multiple **parameters** for estimator

Avoid overfitting (100% accuracy)

Bibliography

Scikit-learn documentation :
https://scikit-learn.org/stable/modules/neural_networks_supervised.html
https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html

Other :
https://ujjwalkarn.me/2016/08/09/quick-intro-neural-networks/
https://www.ibm.com/cloud/learn/neural-networks
https://en.wikipedia.org/wiki/Multilayer_perceptron

Git :
https://github.com/TexierLouis/Multilayer-perceptron