Project Report

CISC 867 Project 1: the Leaf Classification dataset using a neural network architecture

By

Ahmed Mohamed Moussa Mohamed

Supervised by:

Prof. Hazem Abbas

**Objective of the Report:**

There are estimated to be nearly half a million species of plant in the world. Classification of species has been historically problematic and often results in duplicate identifications.

**Problem Formulation**

• **Input**: - Features collected from half a million species of plant in the world.0

• **Output**: - Predicted species for leaves.

• **Deep Learning Function**: - Manipulating, analyzing, preprocessing the data, and training the data.

• **Problems**: - Classification of species has been historically problematic and often results in duplicate identifications.

• **Objective**: - The objective of this playground competition is to use binary leaf images and extracted features, including shape, margin & texture, to accurately identify 99 species of plants. Leaves, due to their volume, prevalence, and unique characteristics, are an effective means of differentiating plant species.

**Data Description:-**

• The dataset consists of approximately 1,584 images of leaf specimens (16 samples each of 99 species) which have been converted to binary black leaves against white backgrounds. Three sets of features are also provided per image: a shape contiguous descriptor, an interior texture histogram, and a fine-scale margin histogram. For each feature, a 64-attribute vector is given per leaf sample and finally, it contains 193 Features.

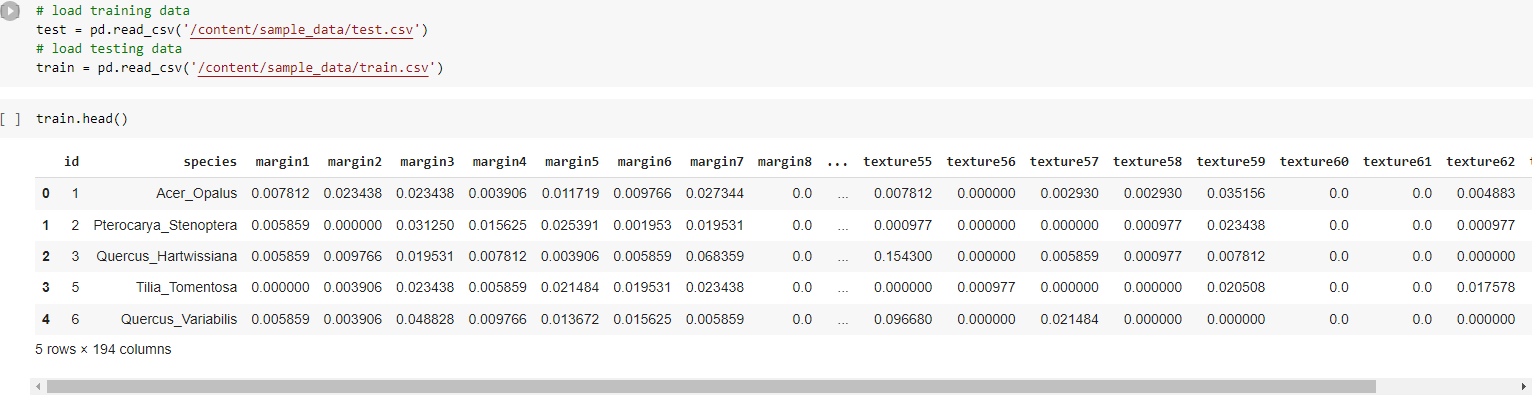
• Note that of the original 100 species, we have eliminated one on account of incomplete associated data in the original dataset.

**The libraries**

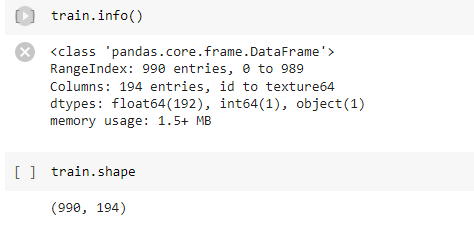
We used some important libraries in python to help us for building the classifiers :



**Data processing**

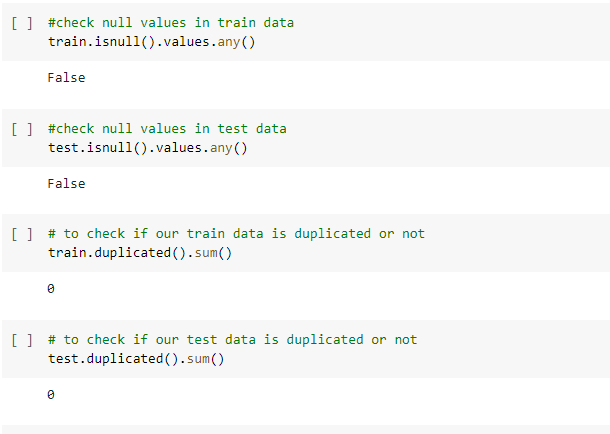
****

As we saw after running the data it consists of 990rows × 194 columns…



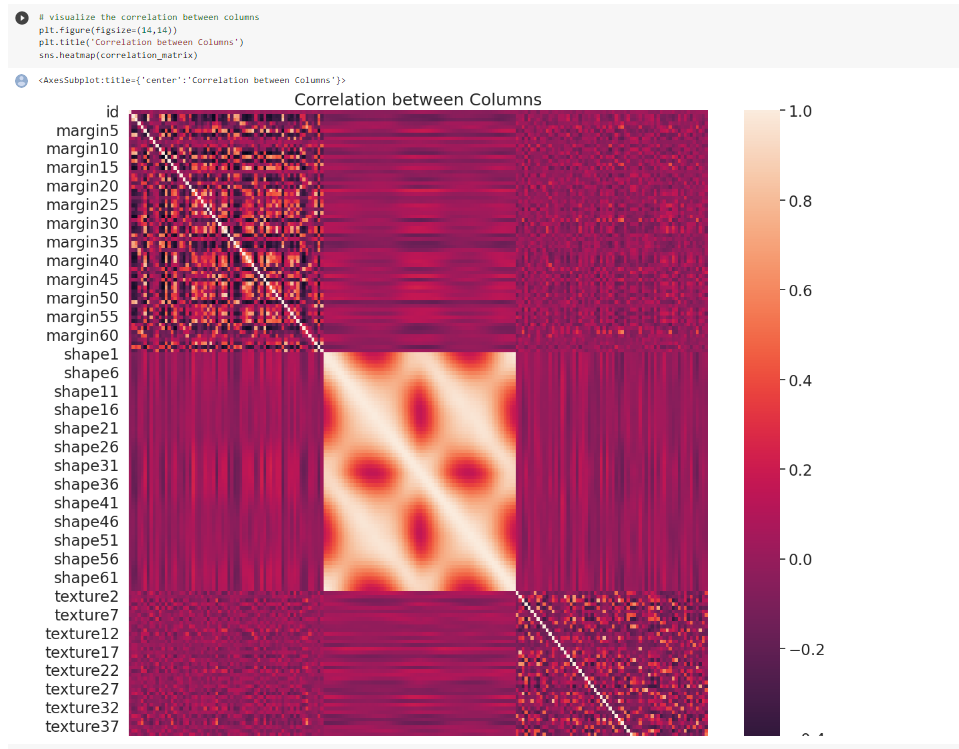
• Check duplication: - We found no duplicates.

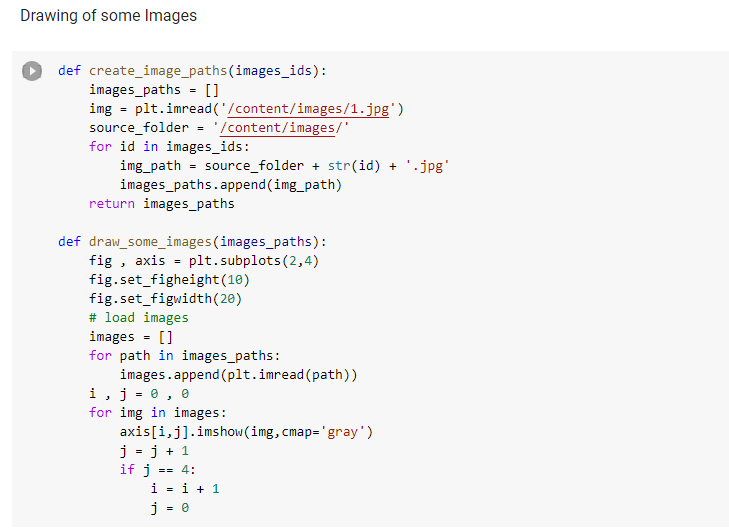
• Check missing values: We found no null values.

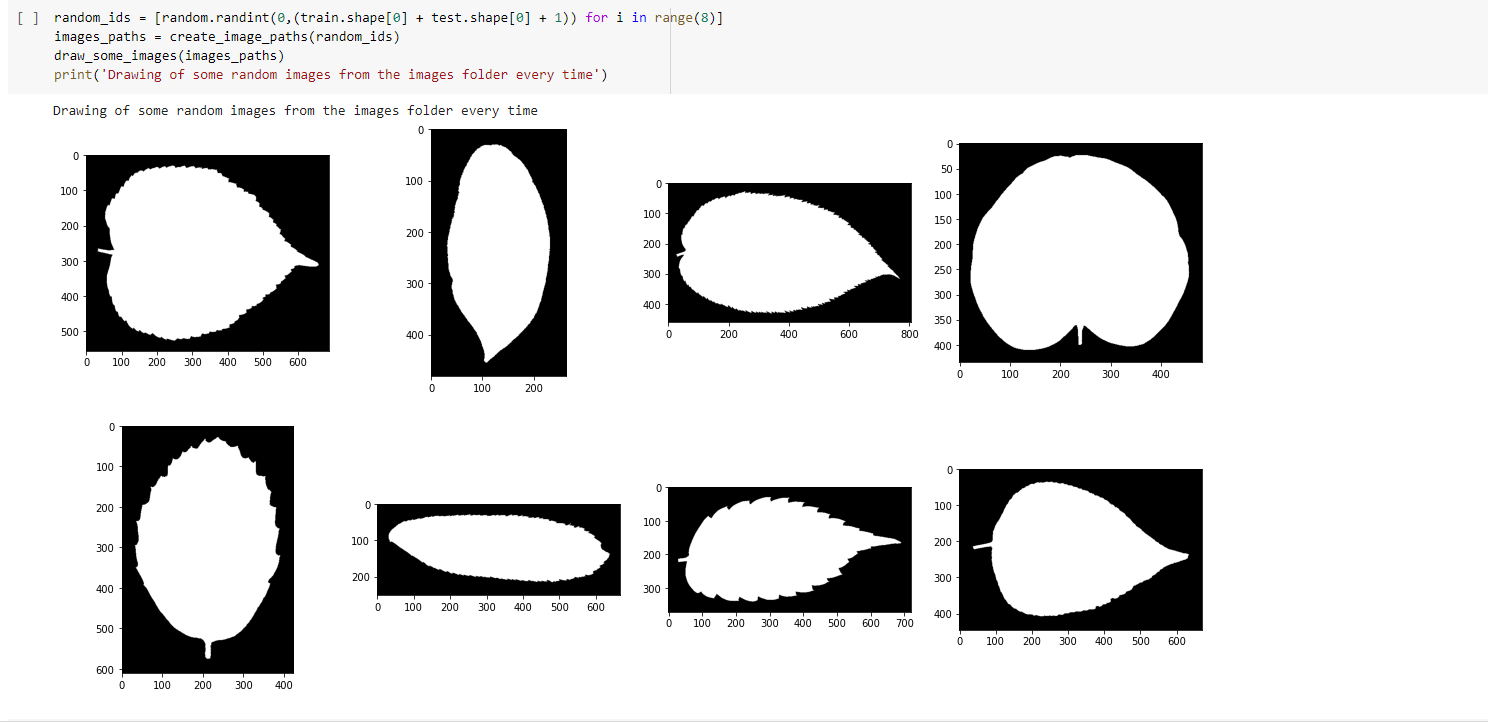


**Visualizing dataset…**

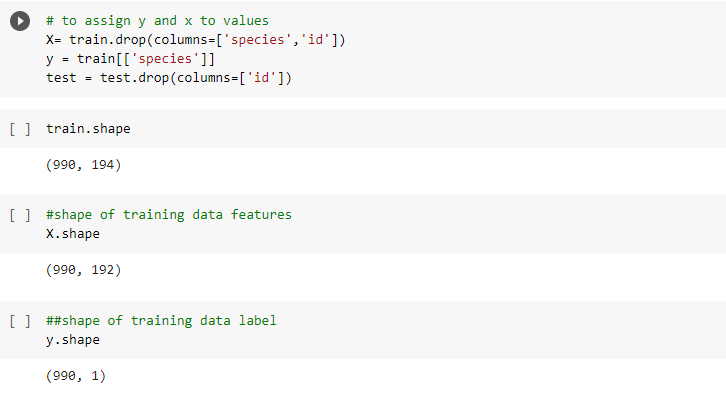
One of the most important skills in data science is data visualization. We need to understand the underlying dataset before we can start creating viable models. You'll never be an expert on the data you're dealing with, and you'll always need to go deep into the variables before moving on to developing a model or doing something else with it. The most crucial tool in your arsenal for accomplishing this is effective data visualization



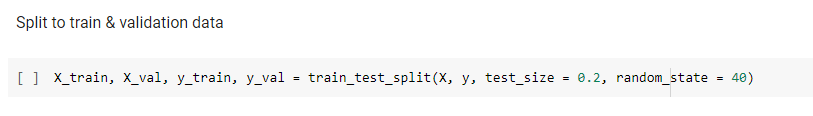




**Splitting data to x , y**

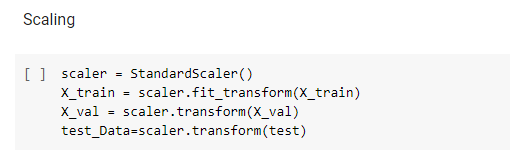


**Split to train & validation data**



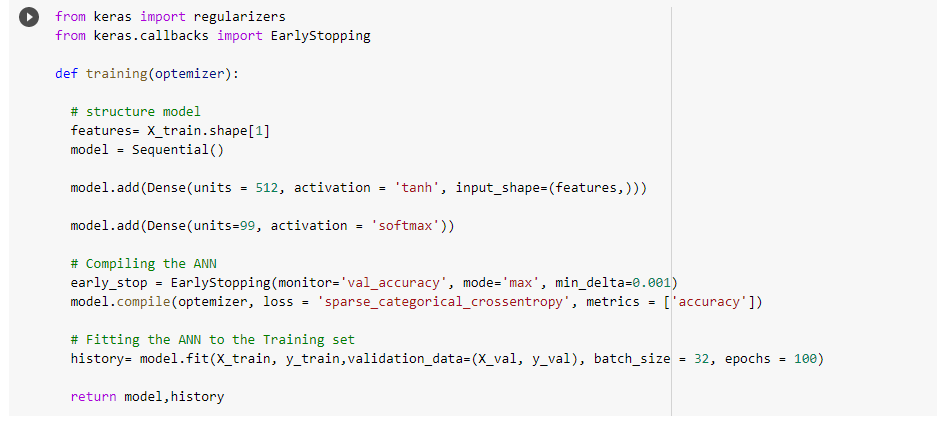
By scikit-learn library we used the train-test split evaluation procedure via the train\_test\_split() function, that takes a loaded dataset as input and returns the dataset split into two subsets(train and test subsets). We split the data into 20% for the testing and 80% for the training.

**Scaling**

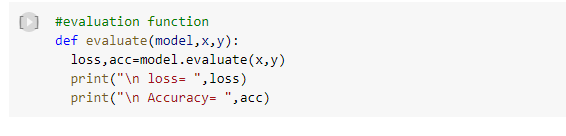


**Model**

Training function of choose optimizer:

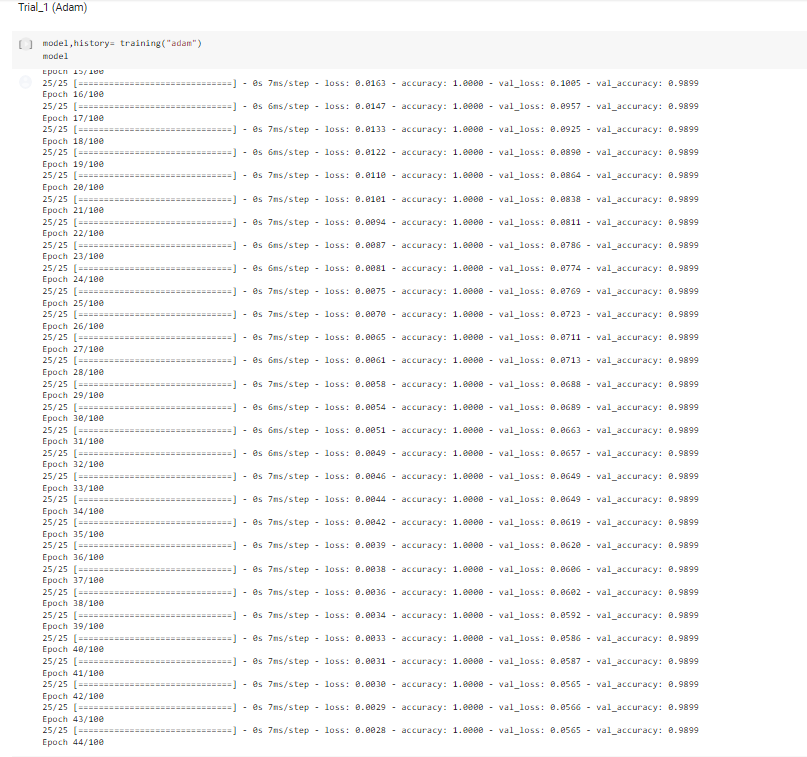


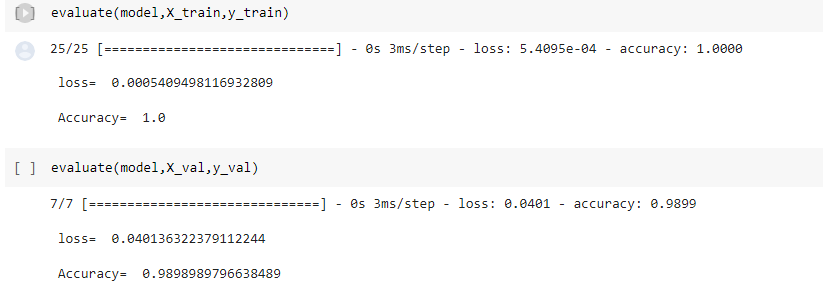
Evaluation function

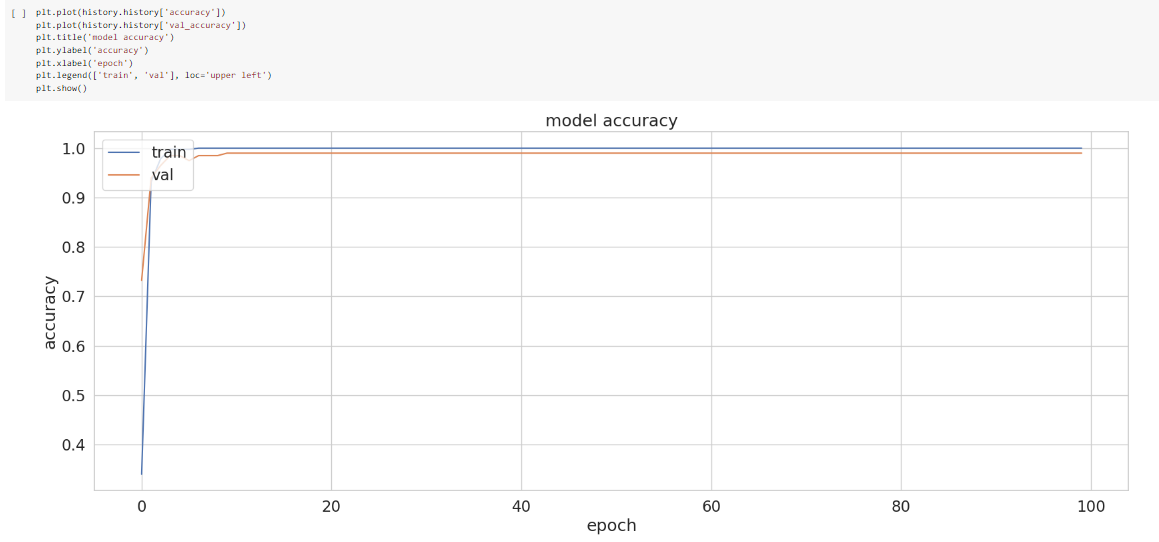


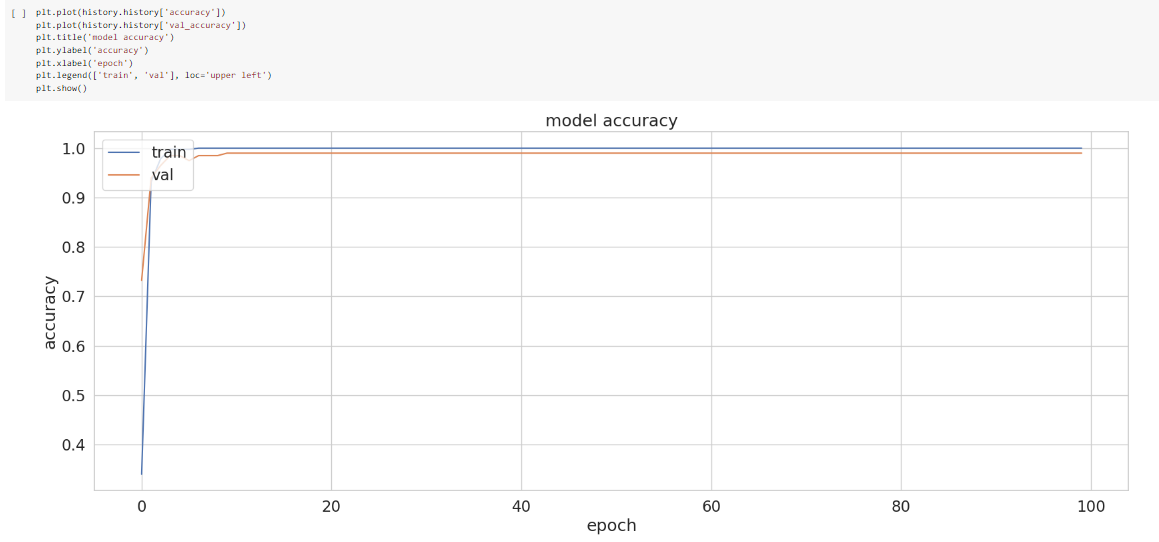
Some trials using different hyperparameters

Optimizer trial 1 (Adam)

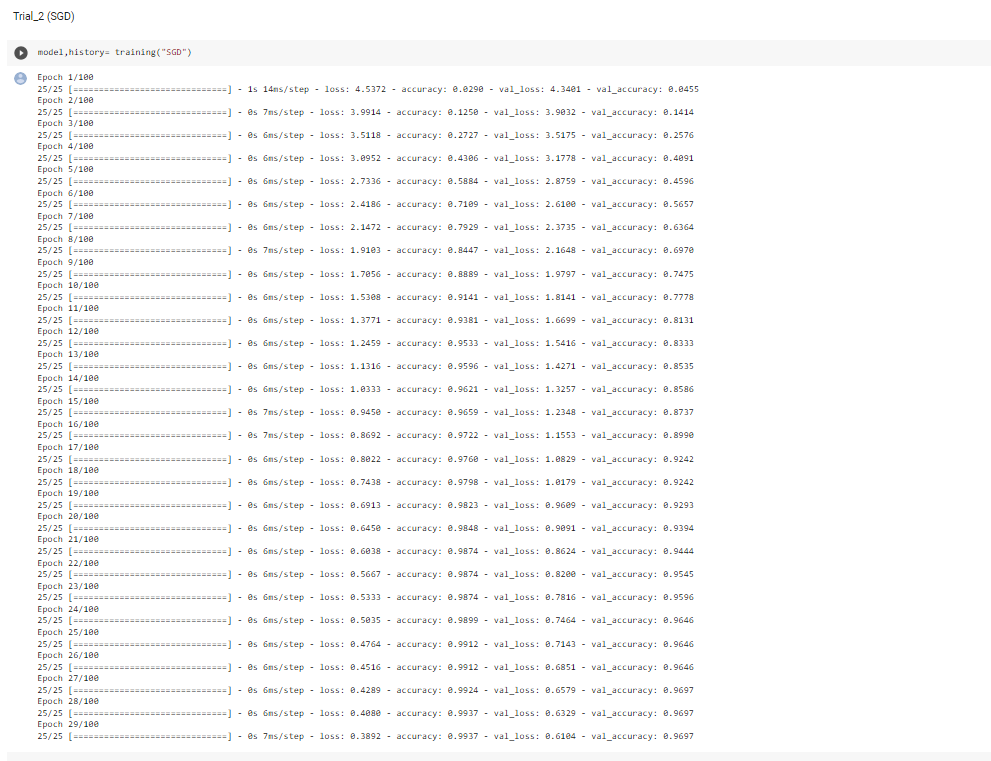


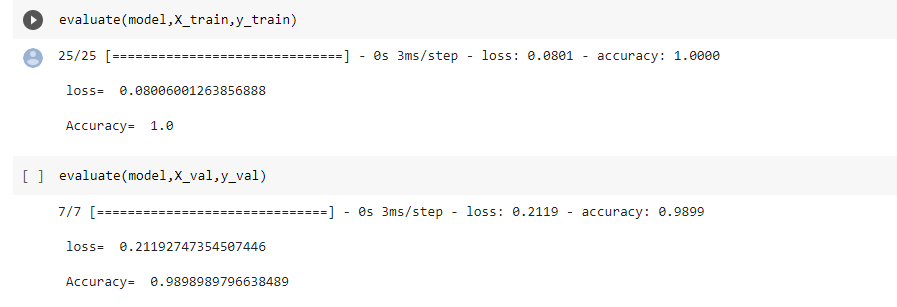


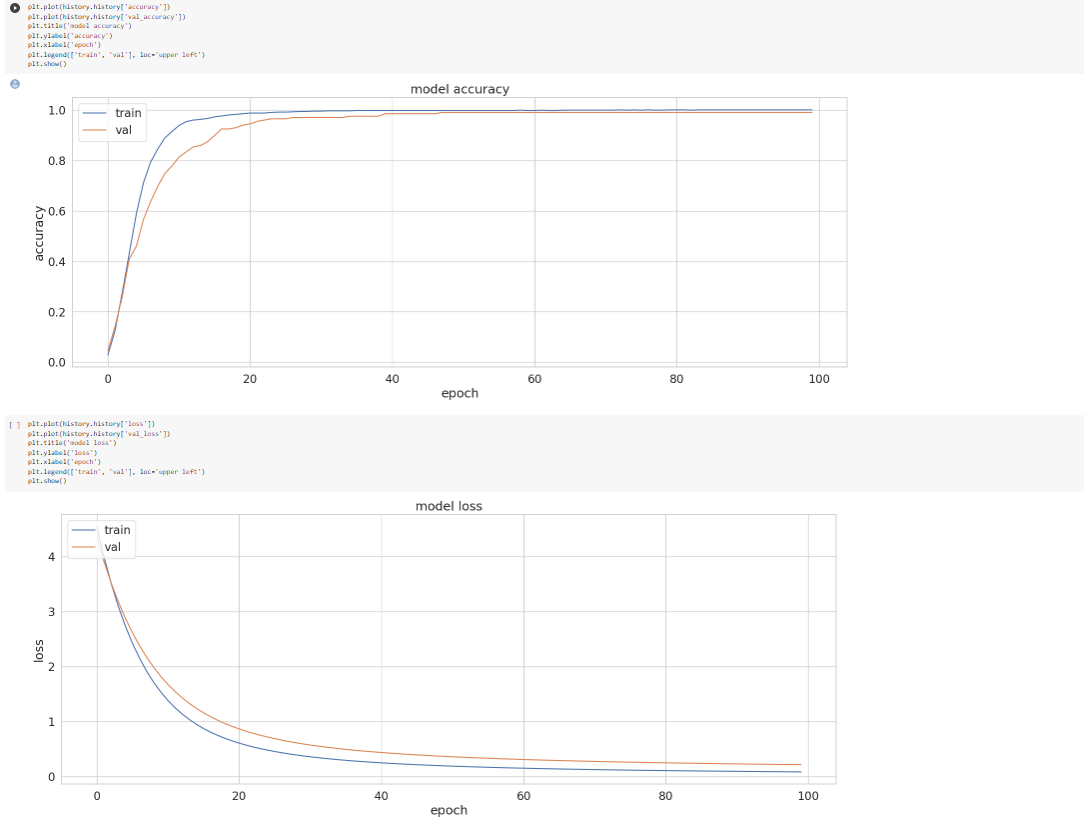




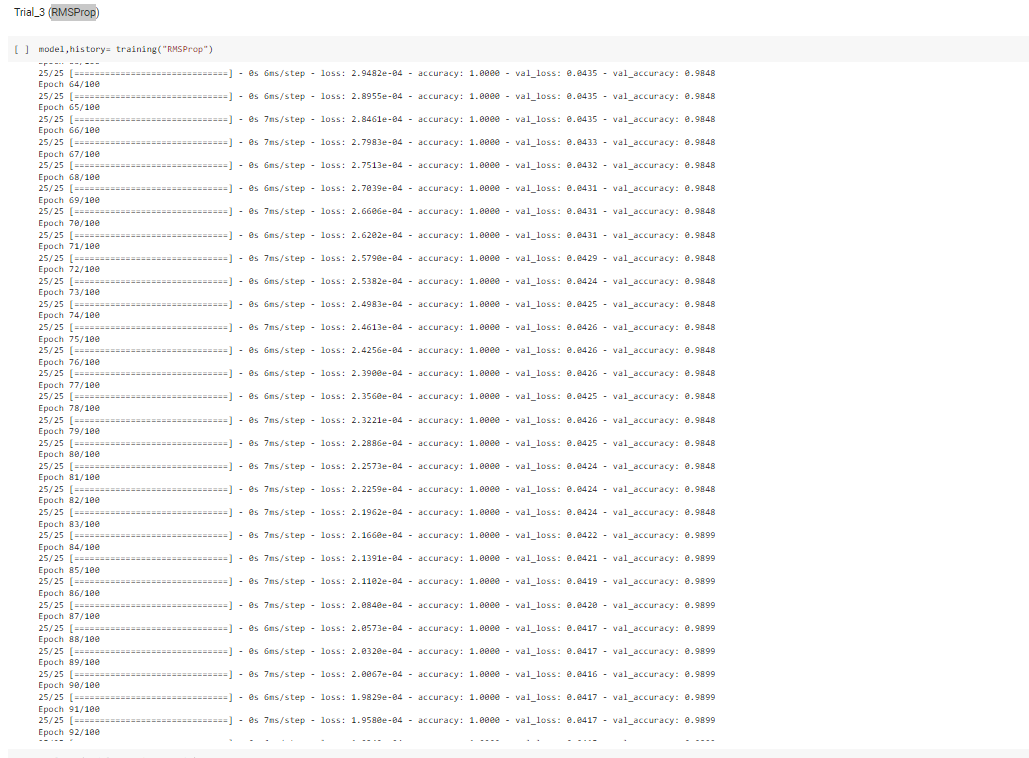
Optimizer trial 2 (SGD)

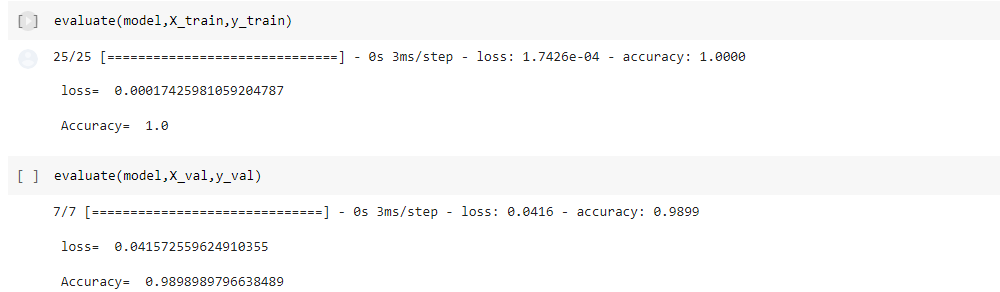


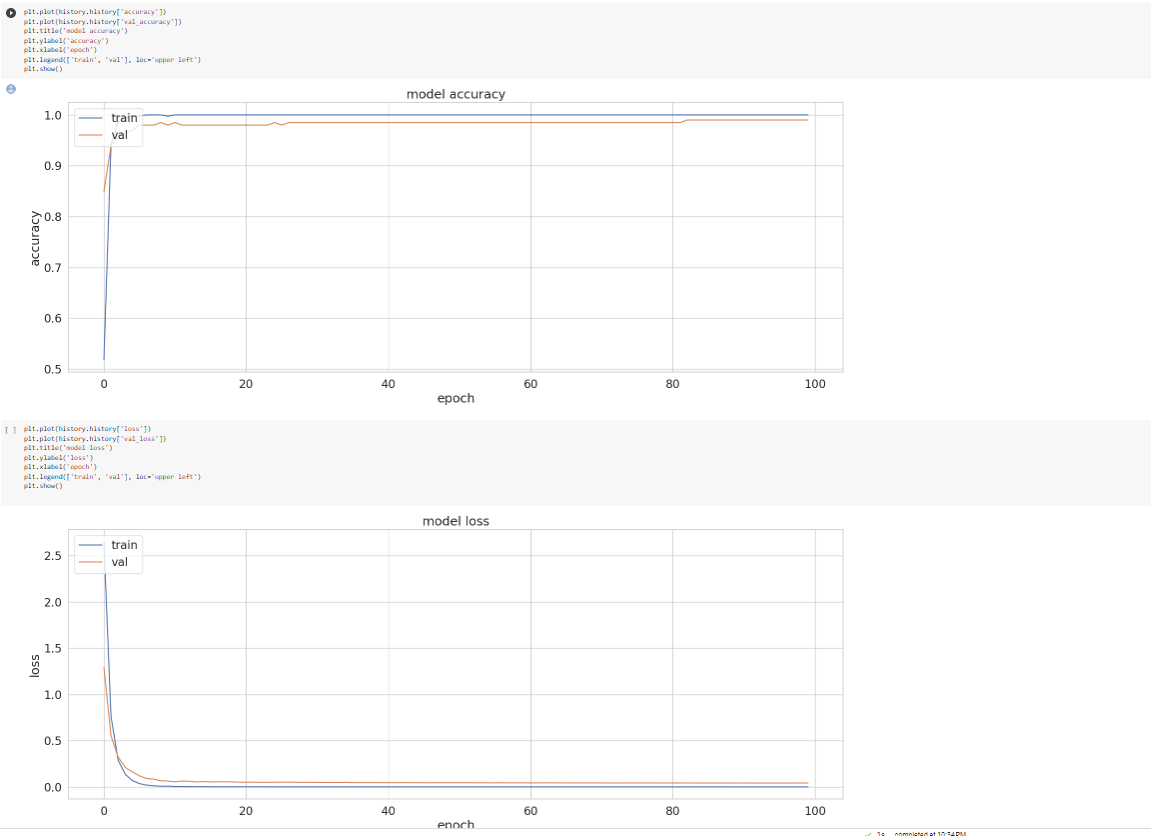




Optimizer trial 3 (RMSProp)







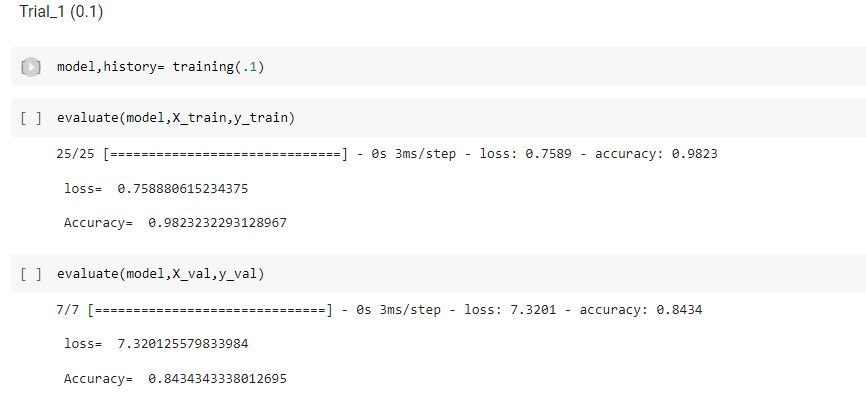
**Observation**: From the previous trials, we discovered that Adam optimizer is the best optimizer

**Learning rate hyperparameter**

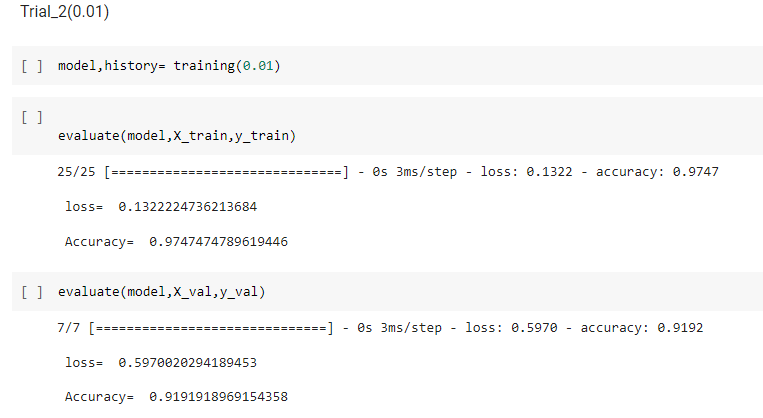
Model learning rate



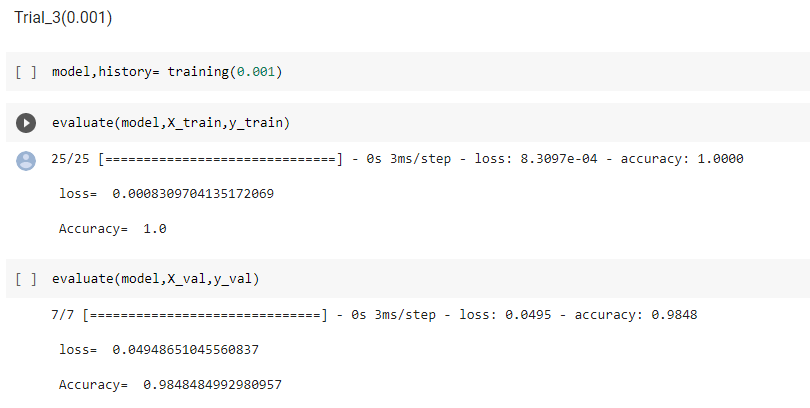
learning rate trial 1 (0.1)

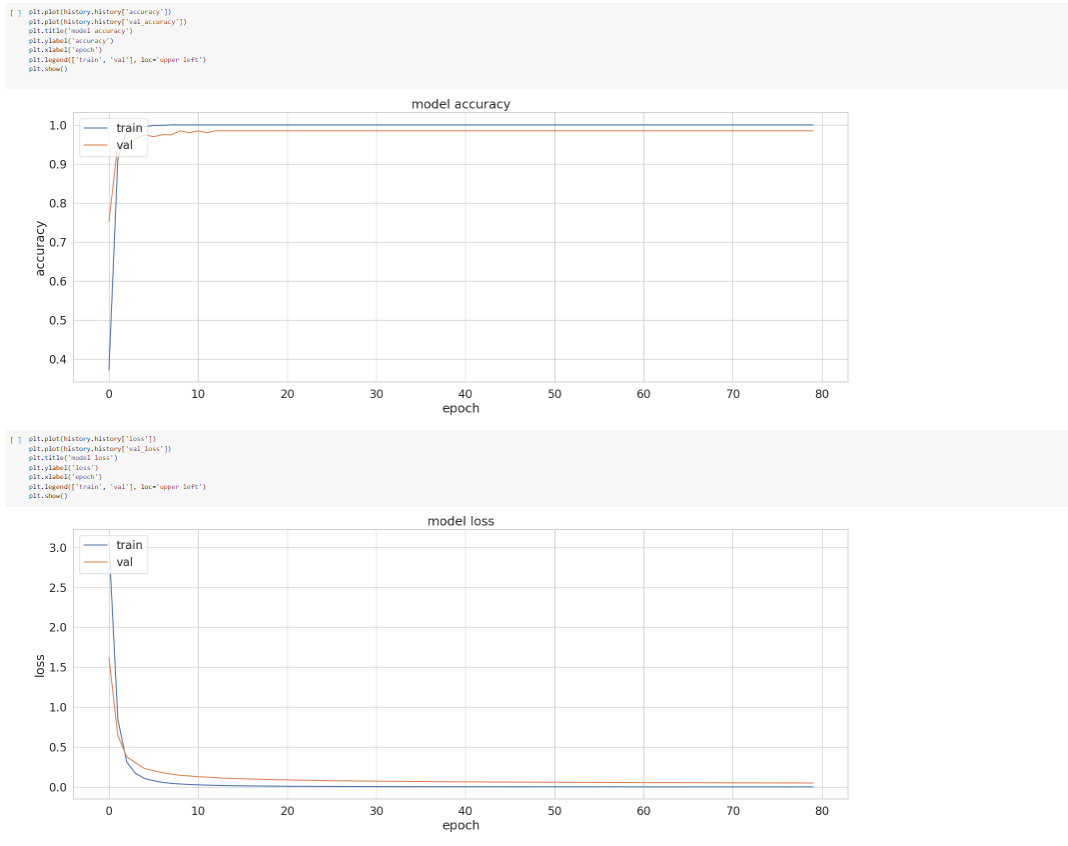




learning rate trial 2 (0.01) 



learning rate trial (0.001) 

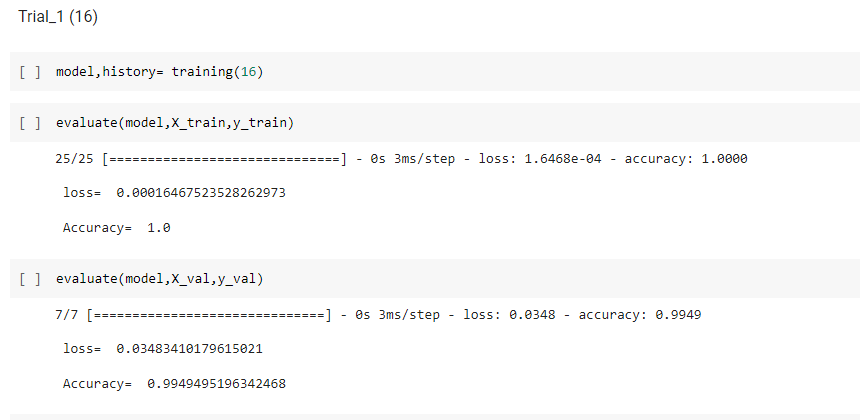


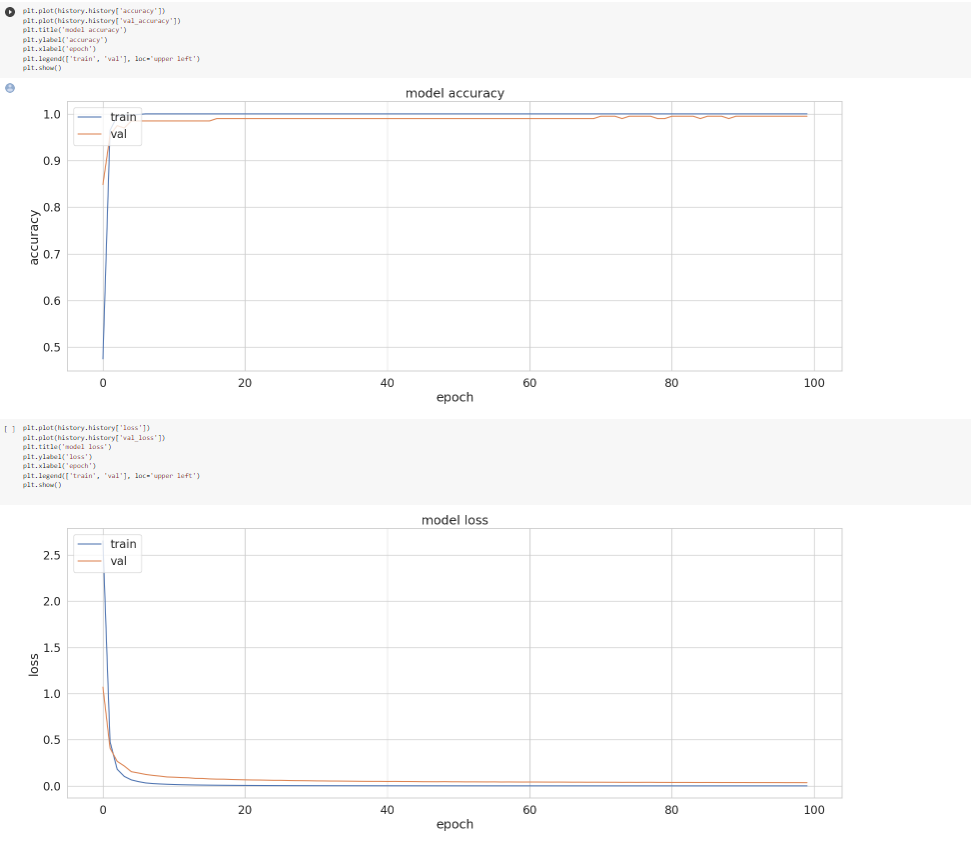
**Observation**: From the previous trials, we discover that the best learning rate =0.001

**Model Batch\_size**

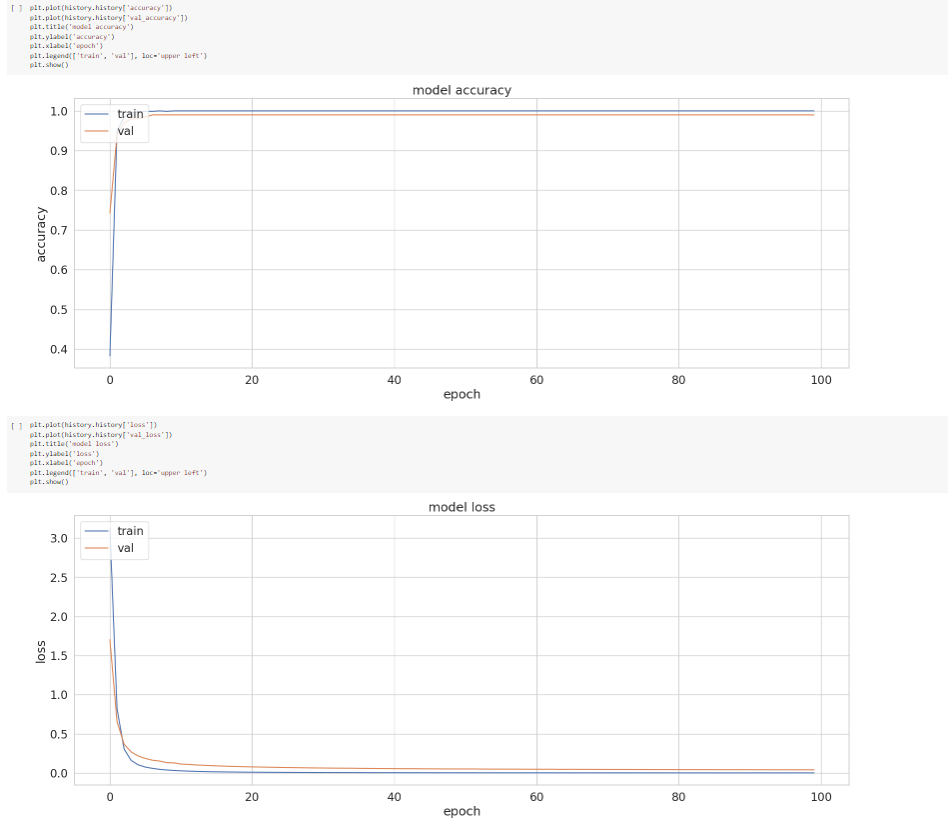
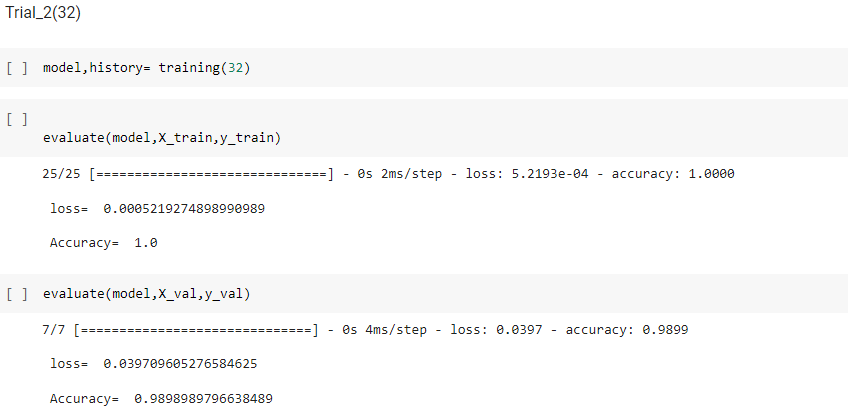


Batch\_size trial 1 (16)

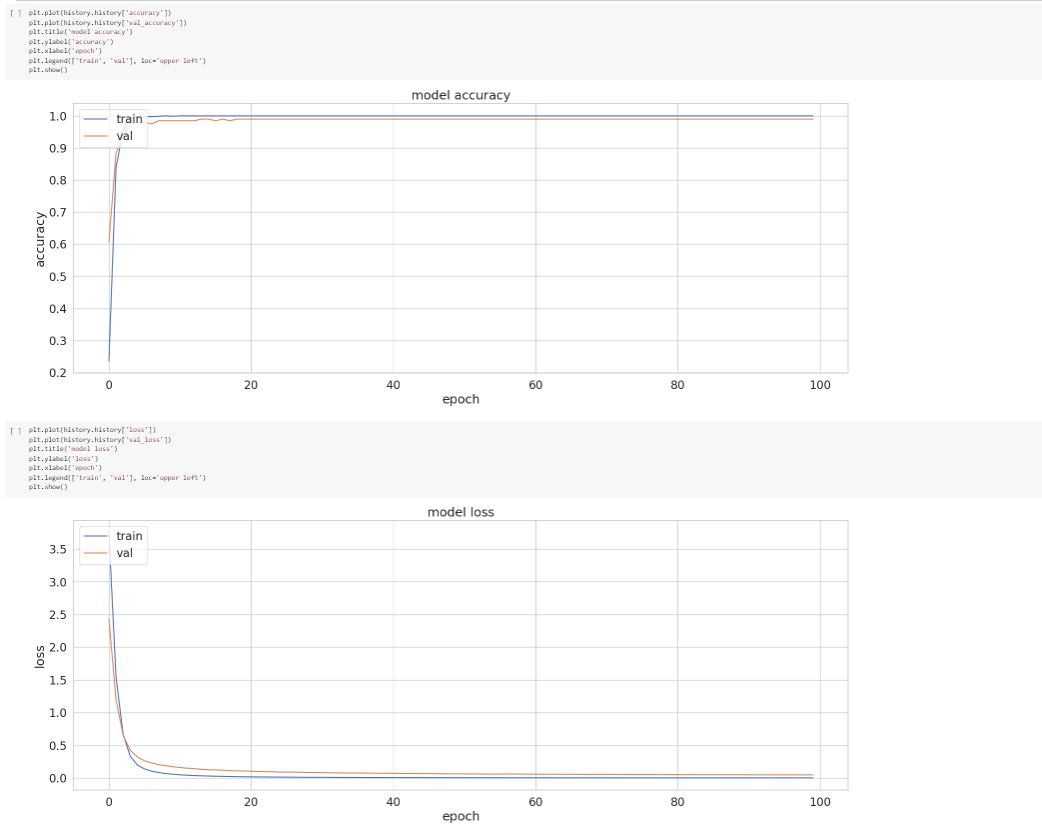
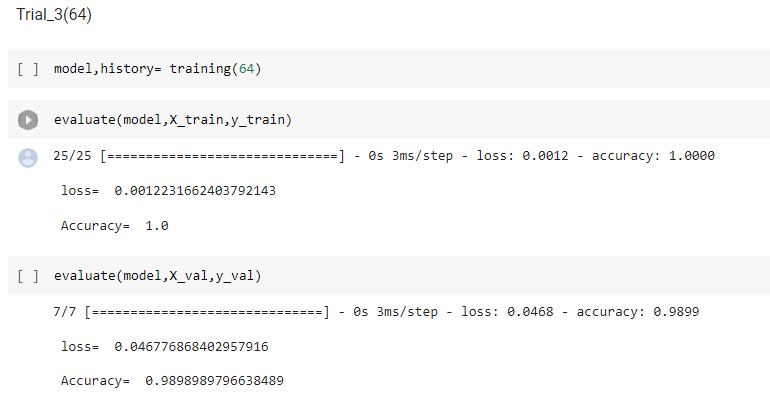
****

****

Batch\_size trial 2 (32)



Batch\_size trial 3 (64)

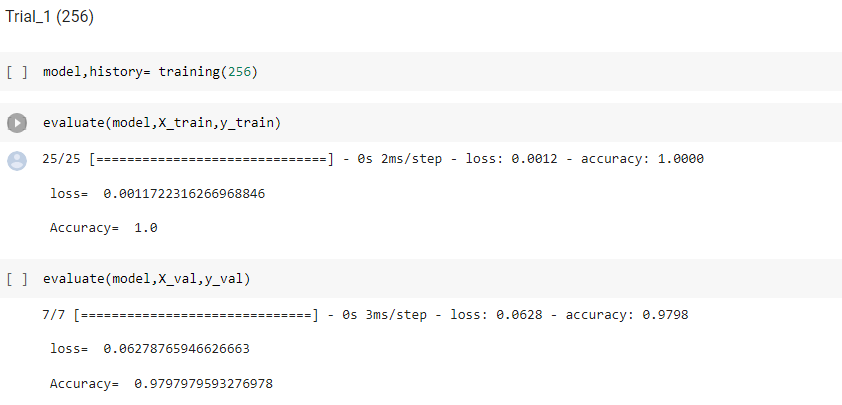


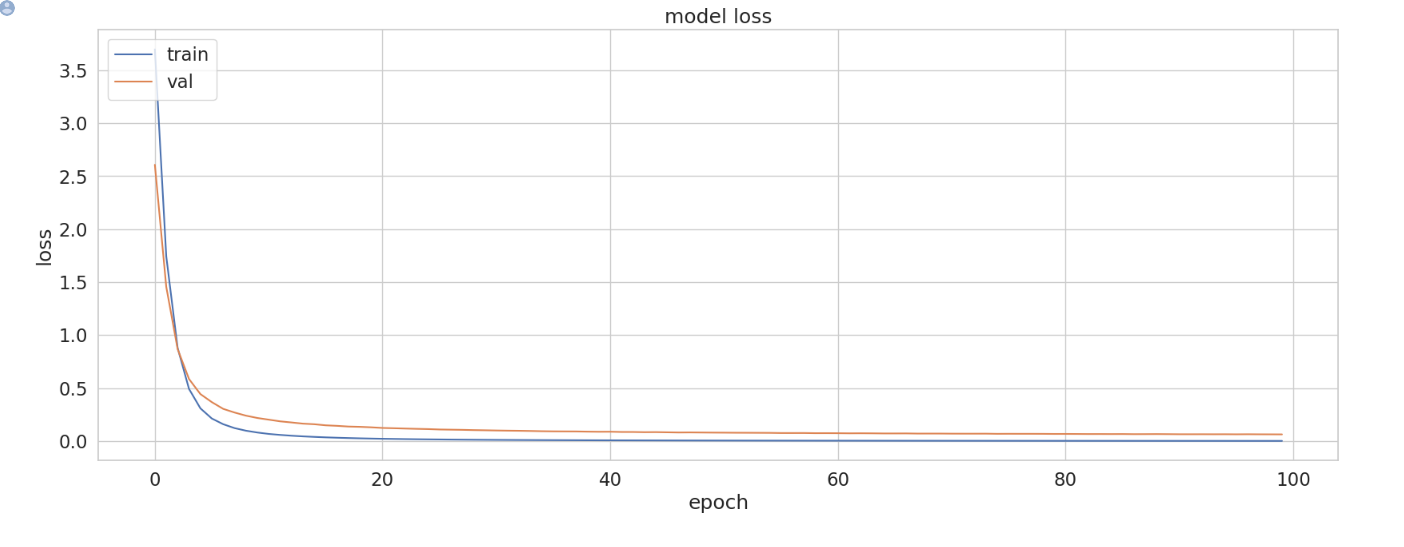
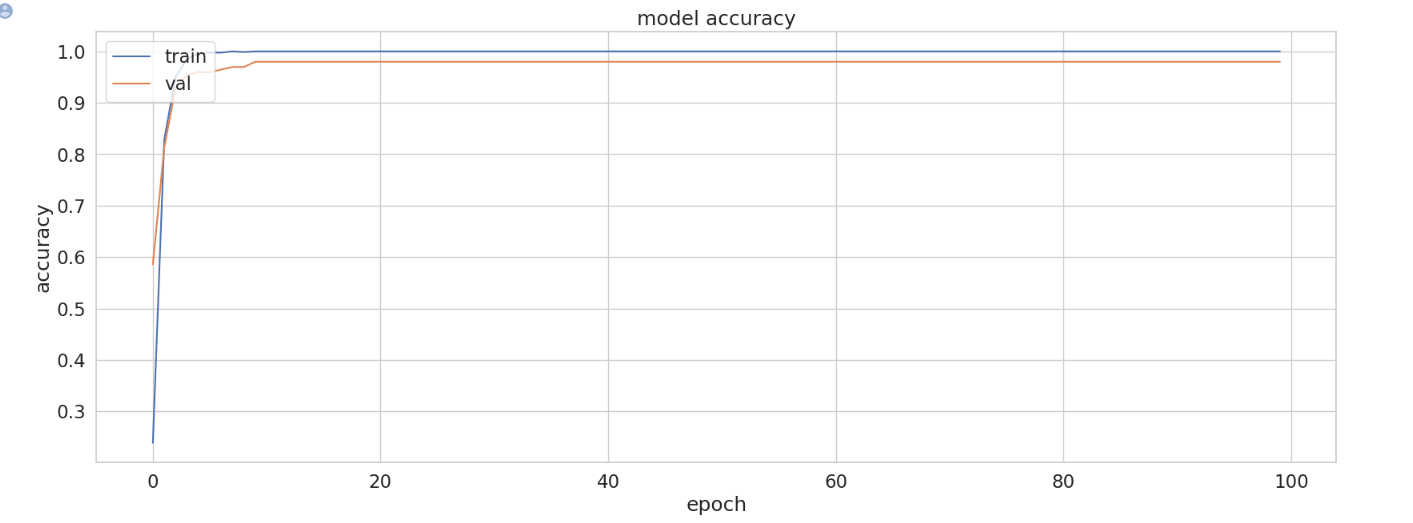
Observation: From the previous trials, we discovered that the best batch\_size=32

**Model hidden units**

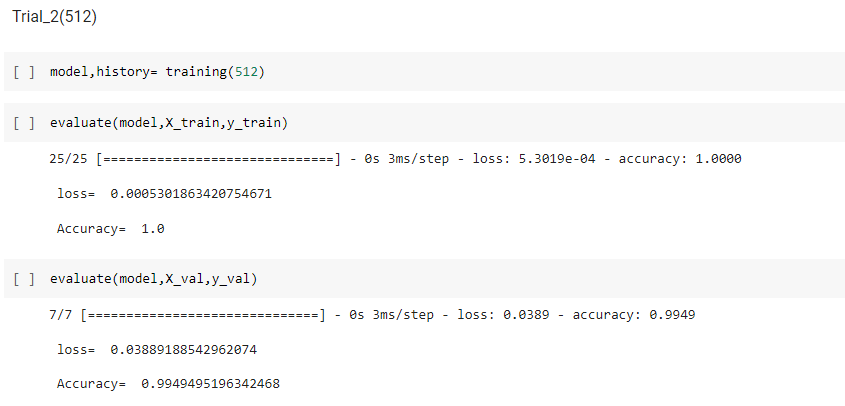


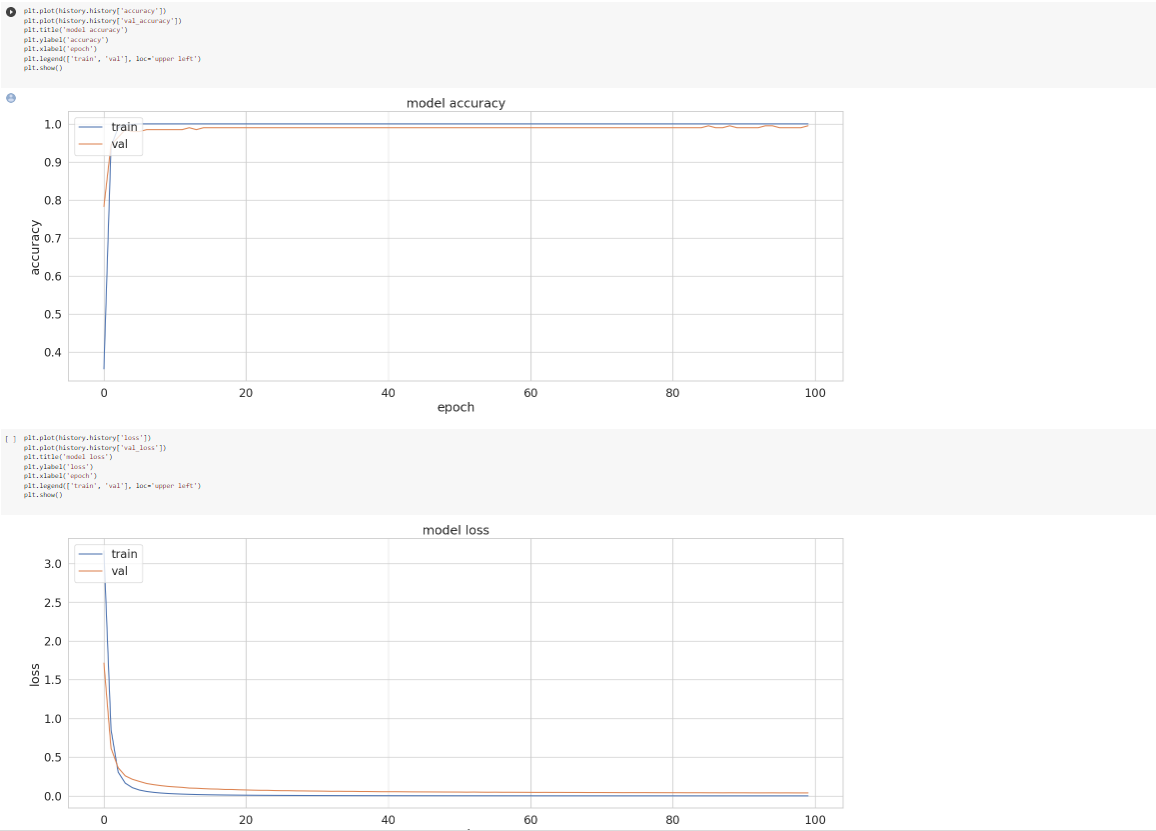
hidden unit trial 1(256)

****

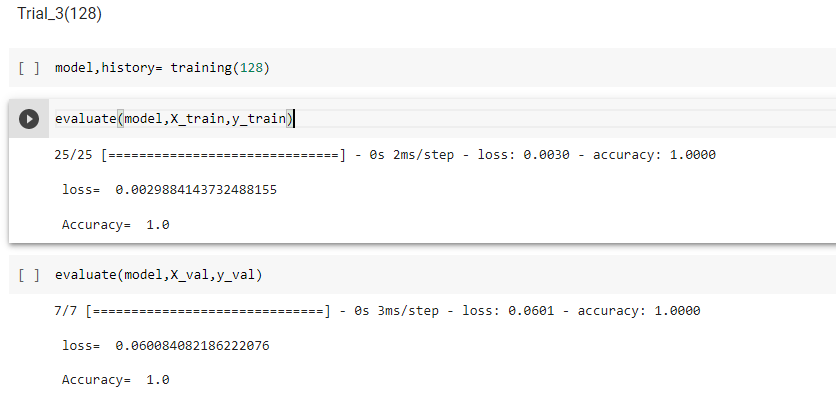
****

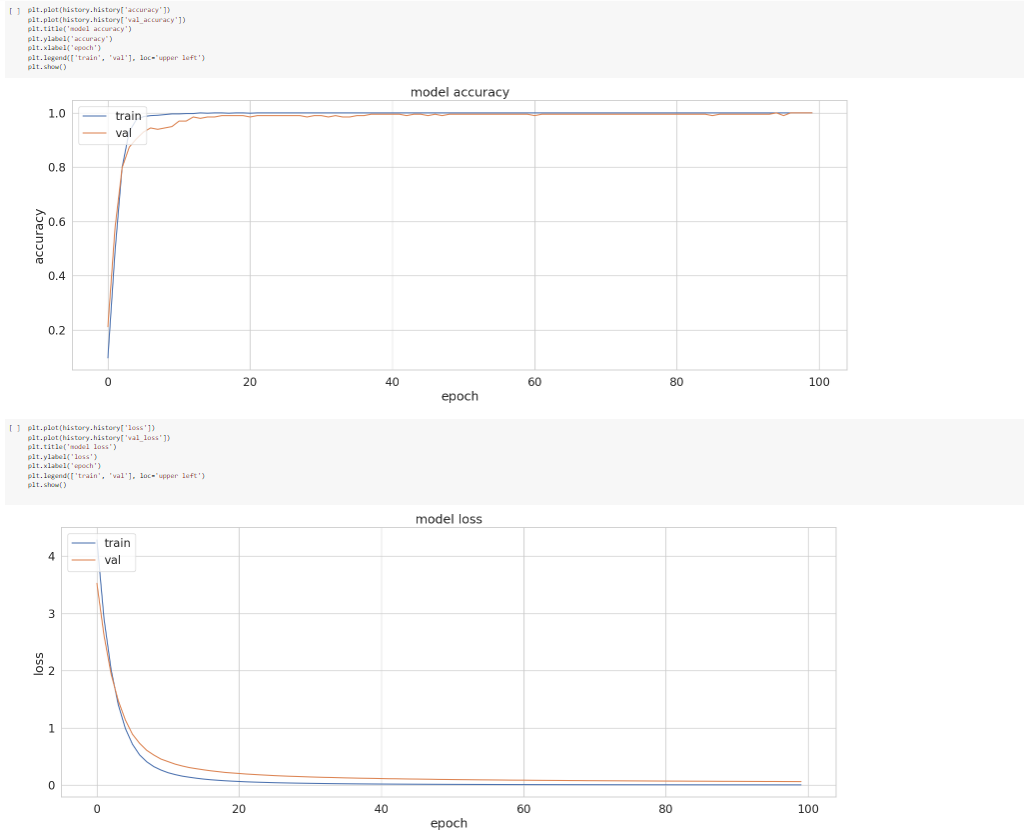
hidden unit trial 2 (512)





hidden unit trial 3 (128)





**Observation**: From the previous trials, we discovered that best hidden units=128

**Conclusion**

I tried different trials with different hyperparameters and chose the best trial as the end.

The trials were from these hyperparameters values:

1) optimizer (Adam, SGD, RMSprop)

2)learning rate (.1, .01, .001)

3)Batch size (16, 32, 64)

4)Hidden units (128, 256, 512)

From these trials I discovered that the best trial of them is (Optimizer: Adam) , (Learning rate : 0.001) , (batch size : 32) and (Hidden unit : 128)

With

validation loss = 0.0667441263794899

and validation accuracy = 0.979797959327697