PROJECT STATUS

Problem Statement:

Gender Recognition system from audio files using FFT with Artificial Neural Networks.

Status:

8. Parameter Tuning

Parameter tuning is an important part in machine learning or deep learning , it helps to improve the accuracy and leads to better results with less loss of data and it is important that classifier creates less loss in classification because more loss may leads to misclassification even when the classification accuracy is high enough.

There are 3 type of parameter tuning that i performed which are as follows

1. Optimizers
2. Loss
3. Metrics

8.1 Optimizers

An optimizer is one of the two arguments required for compiling a Keras model.

You can either instantiate an optimizer before passing it to `model.compile()` , as in the above example, or you can call it by its name. In the latter case, the default parameters for the optimizer will be used.

The types of optimization algorithms are as folllows:

1.SGD: `keras.optimizers.SGD(lr=0.01, momentum=0.0, decay=0.0,nesterov=False)`

2.RMSPROP: `keras.optimizers.RMSprop(lr=0.001, rho=0.9, epsilon=None, decay=0.0)`

3.ADAGRAD : `keras.optimizers.Adagrad(lr=0.01, epsilon=None, decay=0.0)`

4.ADADELTA : `keras.optimizers.Adadelta(lr=1.0, rho=0.95, epsilon=None, decay=0.0)`

5.ADAM : `keras.optimizers.Adam(lr=0.001, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0, amsgrad=False)`

6.ADAMAX : `keras.optimizers.Adamax(lr=0.002, beta_1=0.9, beta_2=0.999, epsilon=None, decay=0.0)`

7.NADAM : `keras.optimizers.Nadam(lr=0.002, beta_1=0.9, beta_2=0.999, epsilon=None, schedule_decay=0.004)`

After tuning the different optimization algorithms , i have plotted the scatterplot to vizualise better for selection :

Scatterplot 1:

x-axis : accuracy
y-axis : Loss

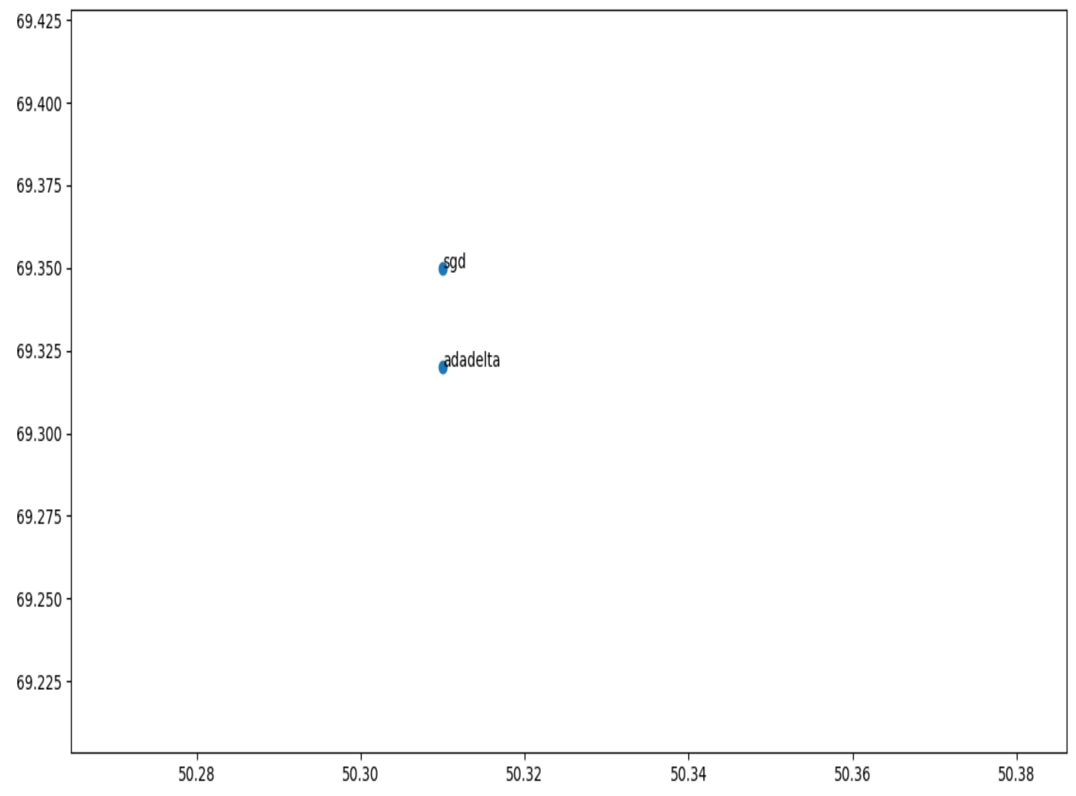


Illustration 1: optimiser_1

Note : the above scatterplot is zoomed section of complete scatterplot

x-axis:accuracy
y-axis: loss

Scatterplot 2 :

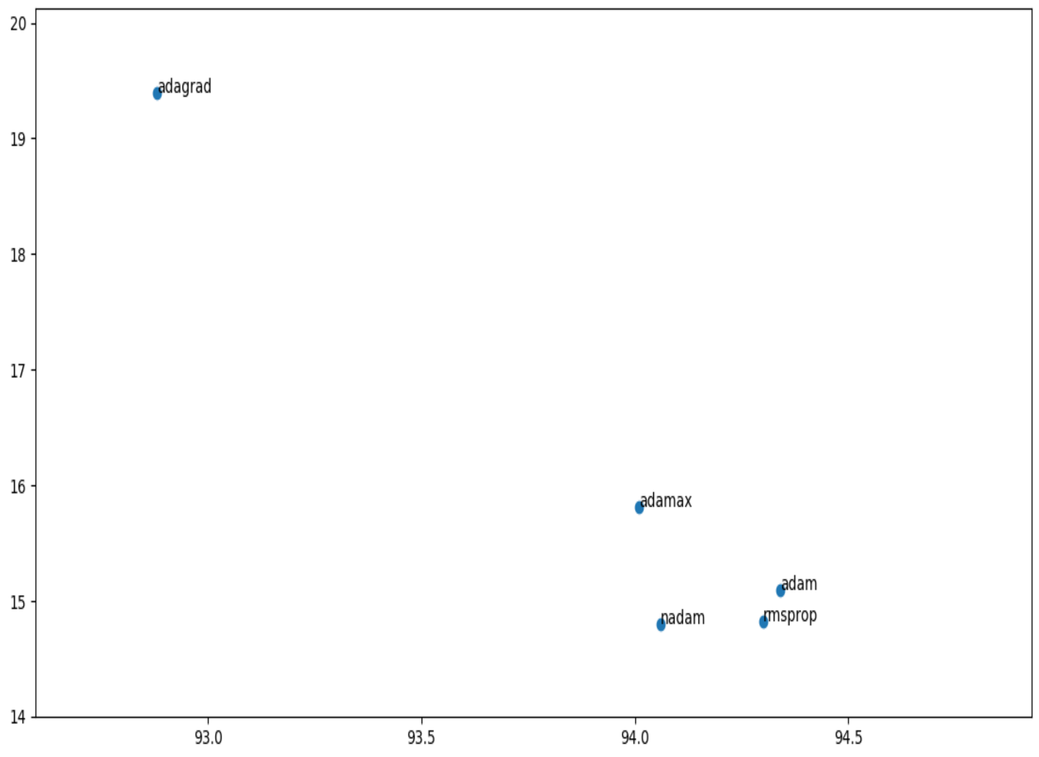


Illustration 2: optimiser_2

Observation result:

By the above charts it is easy to sort out the reliable algorithms which are:
adamax,adam,nadam,rmsprop amongs them the best suitable algorithm that i find useful is :
adam (because of higher accuracy with acceptable loss and rmsprop is better for RNN)

8.2 Loss

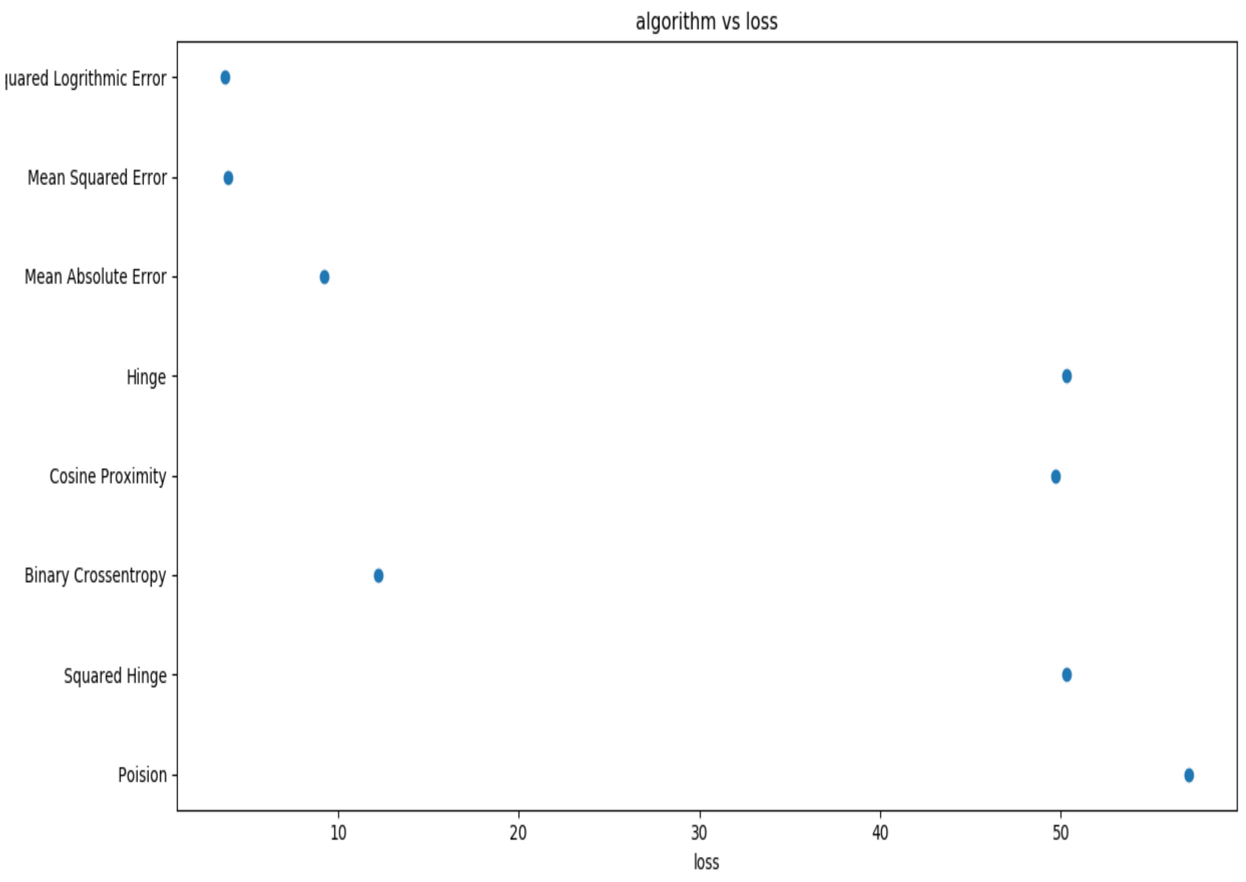
A loss function. This is the objective that the model will try to minimize. It can be the string identifier of an existing loss function (such as categorical_crossentropy or mse), or it can be an objective function.

The list of loss functions are :

1.Binary Crossentropy .
2.Mean Squared Error.
3.Mean Absolute Error.
4.Mean Squared Logrithmic error.
5.Squared Hing.
6. Hinge.
7.Poision.
8.Cosine Proximity.

The below plot will be of help to vizualise and sort out the best function to minimize the losses.

Scatterplot 3:



Observation result:

From the above plot it is clear the **Mean Squarred Error and Mean Squared Logrithmic Error** both will be useful but depends of training accuracy.

8.3 Metric

A metric is a function that is used to judge the performance of your model. Metric functions are to be supplied in the metrics parameter when a model is compiled.

A list of metrics. For any classification problem you will want to set this to metrics=['accuracy']. A metric could be the string identifier of an existing metric or a custom metric function.

The list of metric function used to test are as follows :

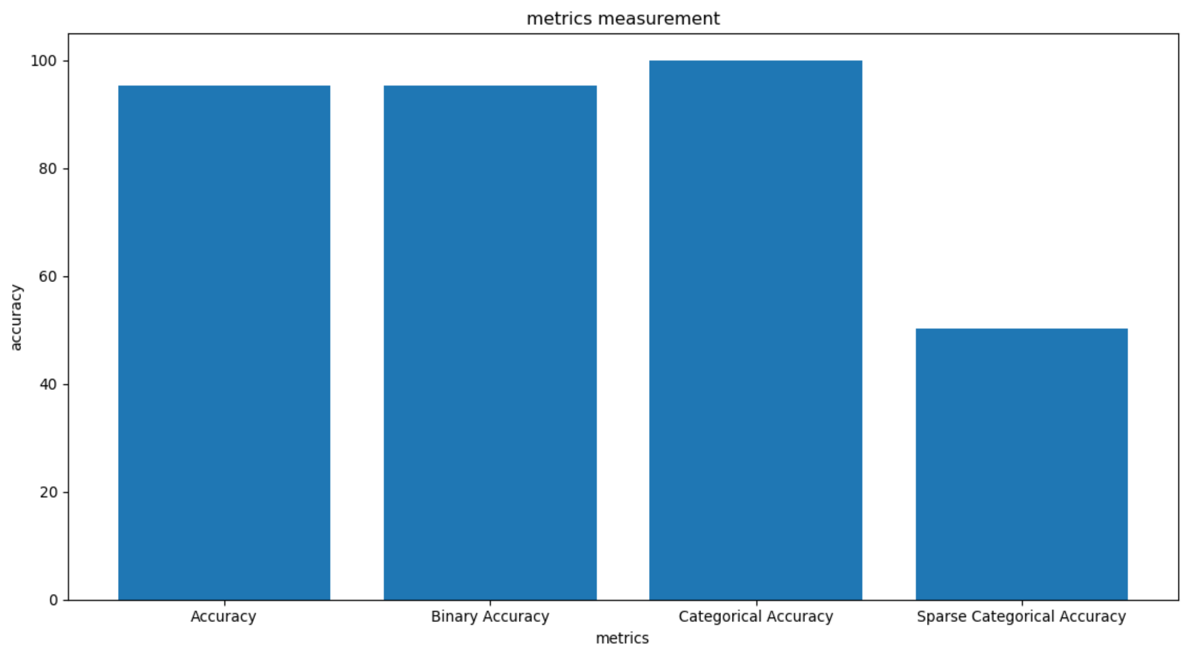
1. Accuracy.
2. Binary Accuracy.
3.Categorical Accuracy.
4. Sparse Categorical Accuracy.

Keras defination to set metric function :

```
from keras import metrics

model.compile(loss='mean_squared_error',
              optimizer='sgd',
              metrics=[metrics.mae, metrics.categorical_accuracy])
```

Bar chart will help in visualization better in this case :



Observation result:

It is clearly observable that **categorical accuracy and binary accuracy** are the most dominant one , hence giving the better results.