Digital VLSI Design
Model based

Aim:

To train multiple models on the given Datasets, replicate hspice model using machine learning and get good accuracies.

Model -1:

Neural Network using Deep Learning Libraries

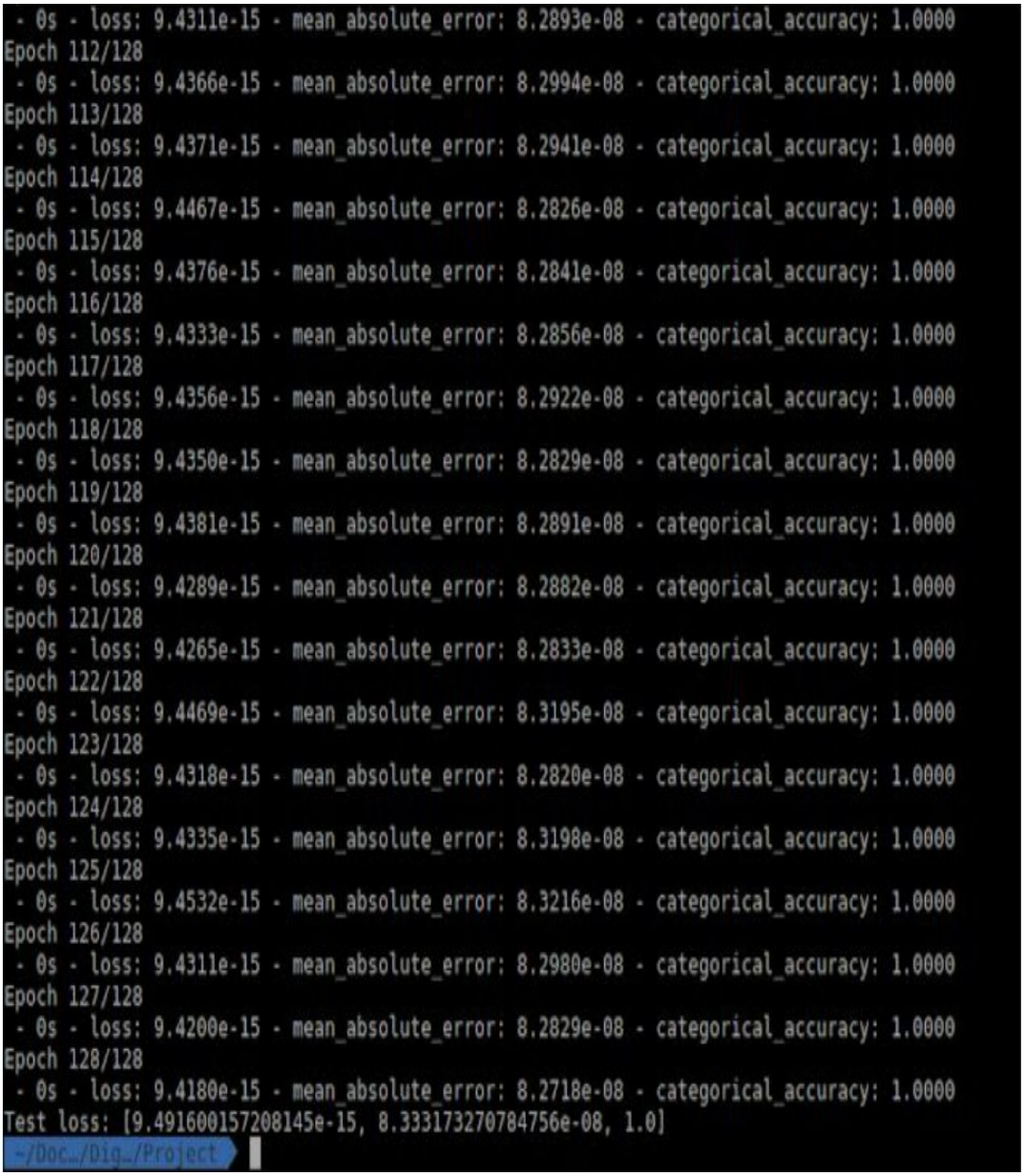
Converted the Dataset given from an xlsx file to a csv file.

- Divided the given data into Training Data and Testing Data in the ratio 75 : 25
- Created a three layer neural network and trained the data on different activation functions, loss functions and optimizers.
- When the activation function was ReLu, the loss function increased to infinity. Sigmoid activation function also had a very high loss. Finally, Tanh activation function was found to give the best results.
- Comparing different optimizers such as sgd, Adam and Adadelta, the best results were achieved by using Adadelta.
- Between different loss functions like mean_squared_error,mean_squared_logarithmic_error and mean_absolute_percentage_error, the mean_squared_error method had given the best loss function
- Compared the outputs to that of the HSPICE model and calculated the accuracy.

Results: Achieved a model analogous to HSPICE which is faster with an accuracy of 99% with the given data.

Code file :

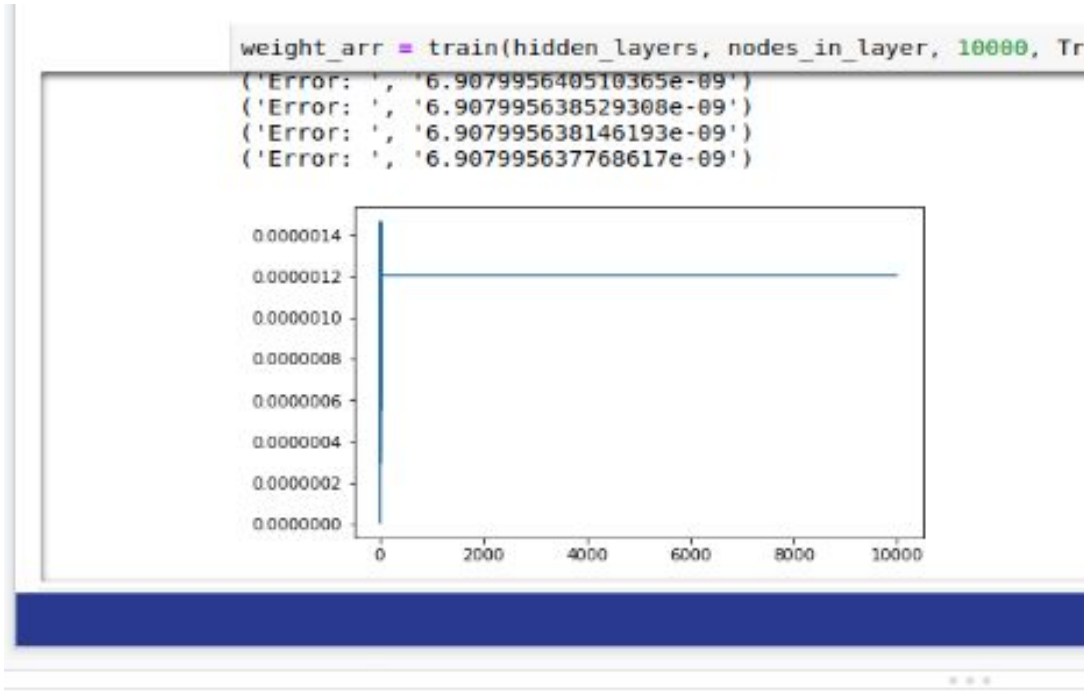
Command:



Model -2 :

Neural Network without Deep Learning libraries

- We have created a neural network using only numpy library.
- Error plot looks like shown below



ker notes

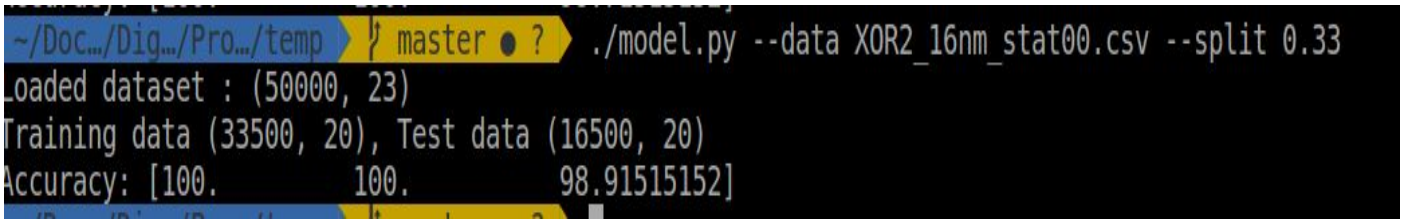
- Data from the 16nm model file of XOR2 gate is taken and preprocessed initially, normalised by subtracting means,dividing with maximum of the data rows in the given dataset.
- Sigmoid, Relu, Softmax, tanh activation functions are coded, used
- Best results are found with tanh function and a Cross entropy loss is used.
- Code file : Neural_Net.ipynb
- Run it on jupyter notebook

Model - 3 :

Sickit libraries based Regression implementation

Linear Regression:

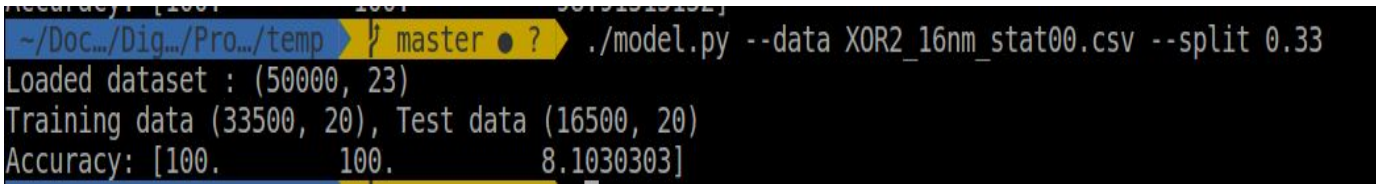
This had the maximum accuracy of 100% 100% 98% among all the models.



Ridge Regression:

Ridge regression addresses some of the problems of Ordinary Least Squares by imposing a penalty on the size of coefficients. The ridge coefficients minimize a penalized residual sum of squares,

$$\min_w ||Xw - y||_2^2 + \alpha ||w||_2^2$$



Lasso Regression:

Mathematically, it consists of a linear model trained with ℓ_1 prior as regularizer. The objective function to minimize is:

$$\min_w \frac{1}{2n_{samples}} ||Xw - y||_2^2 + \alpha ||w||_1$$

```
~/Doc…/Dig…/Pro…/temp  master ● ?  ./model.py --data XOR2_16nm_stat00.csv --split 0.33
Loaded dataset : (50000, 23)
Training data (33500, 20), Test data (16500, 20)
Accuracy: [100.         100.          98.76969697]
```

Lasso Lars Regression:

```
~/Doc…/Dig…/Pro…/temp  master ● ?  ./model.py --data XOR2_16nm_stat00.csv --split 0.33
Loaded dataset : (50000, 23)
Training data (33500, 20), Test data (16500, 20)
Accuracy: [100.         100.          98.77575758]
```

Model - 4 :

To reduce overfitting on the given dataset of 50,000 samples, introduction of dropout layers is done.

The term “dropout” refers to dropping out units (hidden and visible) in a neural network.

Introducing dropout layers:

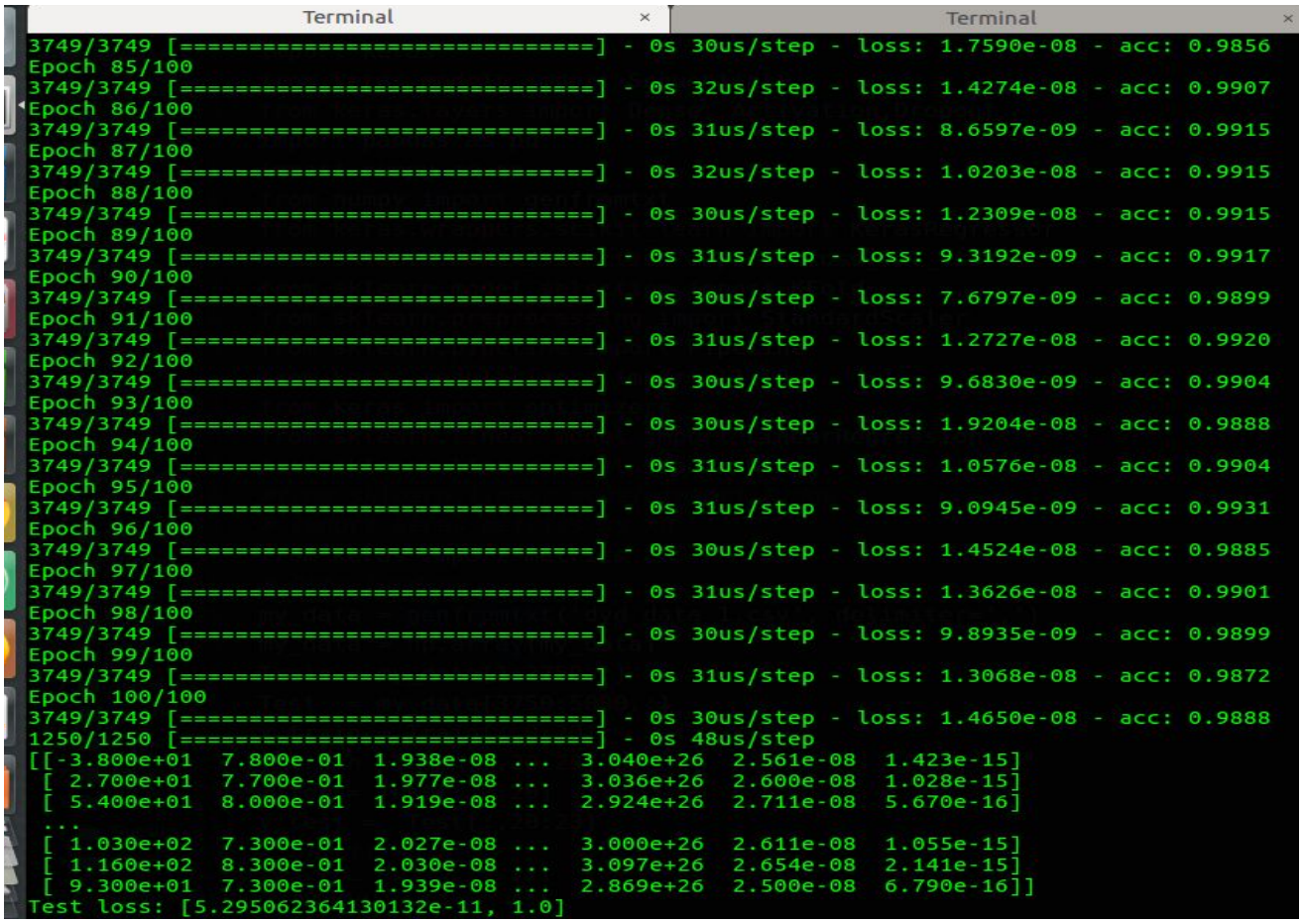
Dropout rate is selected to be 0.25 after every neuron layer

- This helps in increasing the robustness of the model
- Also the efficiency of the model to work on any dataset given is improved

Varying activation functions:

- Relu : This activation function in the inner 2 layers is used
- Softmax : Activation at the output layer
  This combined usage of activation functions brought a good accuracy of about 98.9 to 99.2% .

Optimizer used : AdamaxFile : dvd_m22.py can run the file in folder with command “python3 <file_name>”



Neural network implementation using SGD optimizer:

A learning rate of 0.01 is used along with

Momentum parameter - 0.9 , this has been introduced to accelerate SGD in the relevant direction and dampens oscillations.

Decay element - 1e-6

While using all these parameters, in order to clip the gradient from raising above a particular value and gain a consistent accuracy , gradient clipping is done.

A clipnorm of 1.0 is introduced to stop the gradient from growing above threshold.
Activation functions : relu,relu, softmax
File : dvd_m21.py

- can run the file in folder with command python3 <file_name>
- Results: