Hand Writing Comparison using Linear Regression, Logistic Regression and Neural Network

Manish Reddy Challamala
Department of Computer Science
University at buffalo
UBIT Name: manishre
Person Number : 50289714
manishre@buffalo.edu

November 2, 2018

Abstract

To train a model to compare the hand written samples a known and questioned writer using Linear regression, Logistic regression and Neural network.

1 Introduction

The goal of this project is to find similarities between hand written copies of two different writers by using CEDAR as data source. Where each instance in the CEDAR "AND" training data consists of set of input features for each handwritten "AND" sample.
The features are obtained from two different sources:

1. Human Observed features: Features entered by human document examiners manually
2. GSC features: Features extracted using Gradient Structural Concavity (GSC) algorithm.

The above feature data sets are trained by using following methods:

1. Linear Regression using Stochastic Gradient Descent.
2. Logistic Regression using Stochastic Gradient descent.
3. Neural networks.

2 Theory

2.1 Linear Regression Model using Stochastic Gradient Descent

Linear Regression is a technique to analyze the linear relationship between the independent(x) and dependent(y) variables.Here x is input data and y is predicted values.

$$BasicEquation : y = (a_0 + a_1)x$$

Here a_0 *and* a_1 are weights
The polynomial equation for linear regression is given by

$$t = w_0x_0 + w_1x_1 + .. + w_mx_m = w_0 + \sum_{i=0}^{m} w_ix_i = w^Tx \tag{1}$$

$$t = W^T\phi(x)$$

where;
t is vector of predicted outputs
W $=(w_0, w_1, ..., w_{(M-1)})$ is weight vector
The w weight vector (vector of coefficients): which gives the polynomial relationship between the output and the input and represents the y- axis intercept of the model.

Here the optimal weights are found by using stochastic gradient descent:

2.1.1 Stochastic gradient descent:

1. In stochastic Gradient Descent [SGD], We initialize the random weights.
2. SGD takes the derivative of each training data instance and calculates the update weights immediately iteratively.

$$for(each\ data\ points)\{ \tag{2}$$
$$w_1 = w_1 + \eta(\Delta(w_1)) \tag{3}$$
$$... \tag{4}$$
$$w_{10} = w_{10} + \eta(\Delta(w_{10}))\} \tag{5}$$

where,
η is the learning rate
In the above equations we have to calculate theΔw

$$\Delta w = [(t - t)^2] \tag{6}$$
$$\Delta w = 1/2(t - w^T\phi(x)) \tag{7}$$
$$\Delta w = 1/2(t - w_1\phi_1(x) + w_2\phi_2(x) + w_3\phi_3(x) + ... + w_{10}\phi_{10}(x))^2 \tag{8}$$
$$\Delta w = \frac{\partial E}{\partial w_1} = [-\phi_2(x)(t - w^T\phi(x))] \tag{9}$$

where, $w_1 = w_1 + \eta(-\phi_{10}(x)(t - w^T\phi(x)))$
similarly for all other weights from w_2 to w_{10}
$w_{10} = w_{10} + x(-\phi_{10}(x)(t - w^T\phi(x)))$
ϕ is basis function and $\phi(x)$ is a design matrix

Design Matrix The design matrix is a n-by-p dimension matrix, where n = no of samples and p is number of features.

Basis Function For calculating a basis function we are using the probability density function

$$\phi_1(x_1) = e^{-1/2((x_1-\mu_1)^2)/\sigma^2} \tag{10}$$

where, μ is the Mean

$$\sigma^2 = \sum = covarianceMatrix$$
$$\sum = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 & \sigma_{13}^2 & \dots & \sigma_{1n}^2 \\ \sigma_{21}^2 & \sigma_{22}^2 & \sigma_{23}^2 & \dots & \sigma_{2n}^2 \\ \dots & \dots & \dots & \dots & \dots \\ \sigma_{d1}^2 & \sigma_{d2}^2 & \sigma_{d3}^2 & \dots & \sigma_{dn}^2 \end{bmatrix} \tag{11}$$

So the above equation (2) can be rewritten has,

$$\phi_1(x_1) = e^{-1/2((x_1-\mu_1)^2)/\sigma^2} \tag{12}$$

2.1.2 Regularization

(a) By using the Feature Expansion we can fit the data even better by using higher degree polynomials, but there is a problem that we may over fit the data.

(b) So by over-fit of data, The model works well for training data but for a unseen data the model fails to predict the output.

(c) To over come this effect we are using regularization.

$$E_D(w) = 1/2 \sum_{n=1}^{N} t_n - w^T\phi(x_n)^2 \tag{13}$$
$$E(w) = E_d(w) + \lambda E_w(w) \tag{14}$$
$$E_w(w) = 1/2 w^T w \tag{15}$$
$$\tag{16}$$

λ is learning rate.

(d) Regularization method explicitly takes care of curve not to over-fit the data by adding error to the model.

3 Logistic Regression Model with stochastic gradient descent

The logistic regression model is a classification model with linear regression algorithm which tries to predict y for a given x.

1. The logistic regression gives the probability to which output each input belongs.

2. To generate probability for each input logistic regression takes a function that maps the output between the range of 0 and 1 for all values of input x.

3. The hypothesis for logistic regression is given below

$$h_\theta = g(\theta^T x) = \frac{1}{1 + e^{-z}} \tag{17}$$

   where
   θ is weights

$$g(z) = \frac{1}{1 + e^{-z}} \tag{18}$$

4. The above equation is called the logistic function or the sigmoid function.

5. we calculate the loss function by taking the derivative of logistic function.

$$l(\theta) = logL(\theta) = \sum_{i=1}^{N} y^i log(h(x^i)) + (1 - y^i) log(1 - h(x^i)) \tag{19}$$

6. we calculate the maximum likelihood of the data by minimizing the loss function either by increasing or decreasing the weights that best fit for our data.we can achieve this by taking the partial derivative of loss function.Which is nothing but using the SGD and update the weights iteratively.

$$J(\theta) = \frac{1}{2} \sum_{i=1}^{N} (y^i - \theta^T x^i)^2 \tag{20}$$

$$\theta_j := \theta_j + \alpha(y^i - h_\theta(x^i)) x_j^i$$

4 Neural Network

The neural network model is a classification model which tries to predict output y for a given input x

1. The neural network model contain two phase:
   1 Learning Phase
   2. Prediction Phase
2. In learning phase, the neural network takes inputs and corresponding outputs.
3. The neural network process the data and calculates the optimal weights to gain the training experience.
4. In prediction process the neural network is presented with the unseen data where it predicts a output by using its past training experience on that given data.

5 Experimental Setup:

The experimental setup consists of three steps:

1. Data Pre-processing
2. Linear Regression using Stochastic gradient descent
3. Logistic regression using stochastic gradient descent
4. Neural Network

5.0.1 Data Pre-Processing:

1. In data pre-processing, we have 2 data sets: 1. Human observed dataset and 2.GSC observed dataset.
2. Firstly, we are trying to generate 4 feature sets from these 2 data sets one is by concatenating the features of two writers and other is by subtracting the features two writers.
3. Total we will be having 4 feature sets generated from the 2 given data sets, Where we split the raw data in accordance to our program requirement.
4. we are splitting 80% of raw data has training data set , In remaining 20% we are splitting 10% for validation data and another 10% for testing data set from all the 4 feature sets.
5. In this process, we are reading the raw data from a file 'HumanObserved-Features-Data.csv file and the target data from same-pairs.csv and diff-pairs.csv file.

6. we are combining the target values from the same-pair and diff-pair file and merging the feature data accordingly to the image id of the writers from humanObserverd-Feature-Data file. which forms our human concatenated dataset.On the other hand, we subtract the features of two writers respectively, which gives us the Human subtracted data set. The same process is applied for GSC dataset.

7. Dimensions for each Dataset is given Below:
   where rows = no of samples and columns = no of features.
   splitting the raw data: 80% for training data, 10% for validation data, 10% for testing data.

   (a) Human Observed Concatenated Dataset
   1. Dimension of Data frame after Merging: 1582 X 21 [Merged 791 from same pair and 791 from diffpair file]
   2. Feature Matrix: 1582 X 18 [removed image id in this step]
   3. Target Vector : 1582 X 1
   4. Training Feature Matrix: 1266 X 18
   5. Validation Feature Matrix: 158 X 18
   6. Testing Feature matrix: 156 X 18

   (b) Human Observed subtracted Dataset
   1. Dimension of Data frame after Merging: 1582 X 21 [Merged 791 from same pair and 791 from diffpair file]
   2. Data frame after subtracting the features: 1582 X 12
   3. Feature Matrix: 1582 X 9 [removed image id in this step]
   4.Target Vector : 1582 X 1
   5. Training Feature Matrix: 1266 X 9
   6. Validation Feature Matrix: 158 X 9
   7. Testing Feature matrix: 156 X 9

   (c) GSC Observed Concatenated Dataset
   1. Dimension of Data frame after Merging: 1024 X 1027 [Merged 791 from same pair and 791 from diffpair file]
   2. Feature Matrix: 1024 X 1024 [removed image id in this step]
   3. Target Vector : 1024 X 1
   4. Training Feature Matrix: 820 X 1024
   5. Validation Feature Matrix: 102 X 1024
   6. Testing Feature matrix: 100 X 1024

   (d) GSC Observed subtracted Dataset
   1. Dimension of Data frame after Merging: 1024 X 1027 [Merged 791 from same pair and 791 from diffpair file]
   2. Data frame after subtracting the features: 1024 X 512
   3. Feature Matrix: 1024 X 512 [removed image id in this step]
   4.Target Vector : 1024 X 1
   5. Training Feature Matrix: 820 X 512
   6. Validation Feature Matrix: 102 X 512
   7. Testing Feature matrix: 100 X 512

5.0.2 Linear regression with stochastic gradient descent:

1. In linear regression we are clustering the data using the k-means clustering library and finding 15 means.
2. By getting the mean for each cluster we are substituting it in the probability basis function formula equation[10] to generate a scalar value for a feature vector accordingly.
3. we create the phi matrix for training, validation and testing whose dimensions are displayed above for each data set respectively.
4. we are initializing the random weights and iterating the values to converge it to the optimum weights by using Stochastic gradient descent.
5. At each iteration, we need to calculate the error function, where the error function value tells us in which direction we need to converge.

5.0.3 Logistic Regression using Stochastic Gradient Descent :

1. In logistic Regression, initialize the random weights.
2. Calculating the optimal weights by plugging the input features and the transpose of weight matrix to the sigmoid function equation[17]
3. By using SGD, we are calculating the loss function at each iteration and updating weights by using equations [19 and 20].
4. After, getting the optimized weights we are predicting the outputs for the new unseen data and compare the predicted output with the actual output which gives our accuracy.

5.0.4 Neural network:

1. Creating a model with 3 layers, 1. input layer 2. hidden layer 3. output layer
2. No of nodes for each layer is given below:
   1.No of nodes in input layer = No of features in the data set
   2. No of nodes in hidden layer 256
   3.No of nodes in output layer 2 [because we are classifying it into only two classes]
3. Activation functions used in hidden layer is relu [rectified linear unit] because it introduces the non linearity in the network and softmax function is used on the output layer to predict the target class.
4. By using the above parameters we create a model.
5. we run this model by plugging in the appropriate data to it and train the model.
6. After training the model we test the model using the unseen data to predict that which class output belongs.

6 Experimental Results:

				Project 2 Experimental Results					
	LINEAR REGRESSION				LOGISTIC REGRESSION			NEURAL NETWORK	
	ERMS_TRAINING	ERMS_VALIDATION	ERMS_TESTING	TEST ACCURACY	Loss_Training	Loss_Validation	TEST ACCURACY	TEST LOSS	TEST ACCURACY
Human Concatenated Dataset	0.50635	0.50507	0.49296	62.17949	0.6837	0.2697	83	0.20407	0.9873
Human Subtraction Dataset	0.49939	0.50033	0.49862	51	0.6889	0.5706	60	0.5188	0.7974
GSC Concatenated Dataset	0.64798	0.59843	0.63326	60	0.5668	0.0607	94	0.2366	0.9705
GSC Subtracted Dataset	0.713	0.51622	0.53943	52	0.66	0.1218	93	0.3327	0.9705

$Table[1]\ Experimental Results$

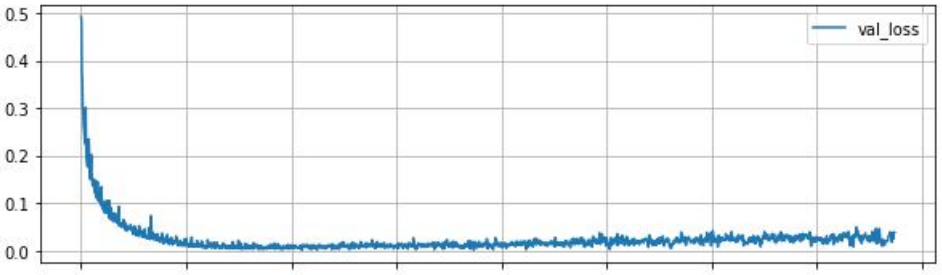
6.0.1 Graphs:

Neural Network:

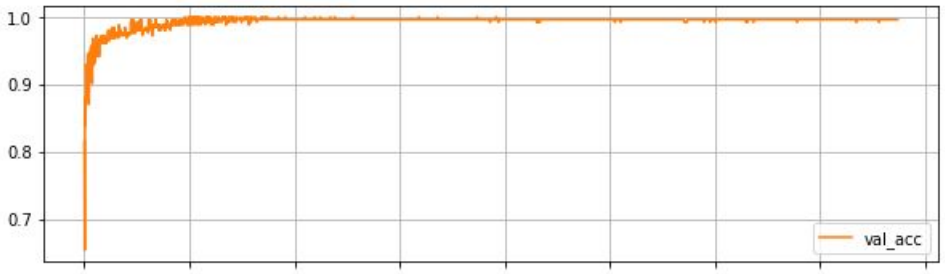
Human Observed Concatenated Dataset

```
18
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_60 (Dense)             (None, 256)               4864
_________________________________________________________________
activation_57 (Activation)   (None, 256)               0
_________________________________________________________________
dropout_29 (Dropout)         (None, 256)               0
_________________________________________________________________
dense_61 (Dense)             (None, 2)                 514
_________________________________________________________________
activation_58 (Activation)   (None, 2)                 0
=================================================================
Total params: 5,378
Trainable params: 5,378
Non-trainable params: 0
_________________________________________________________________
```

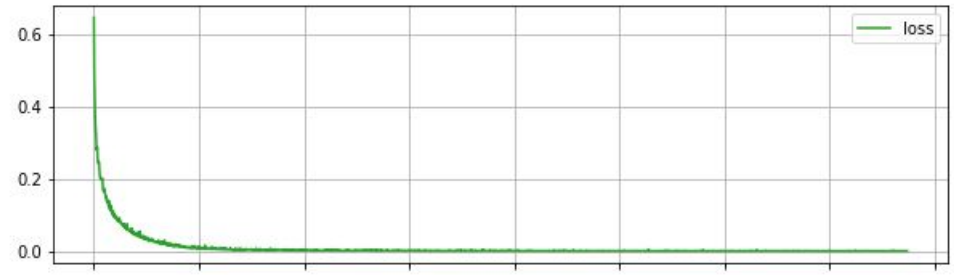
$fig[1]: Model Parameters of Human Observed Concatenated Dataset$



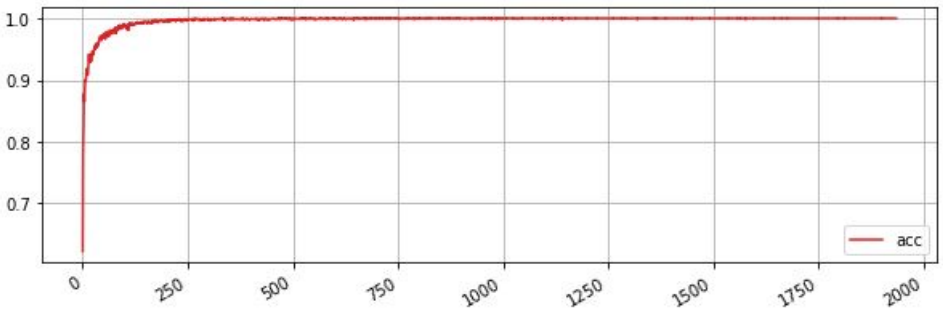
$fig[2]:\ validationlossforHumanObservedConcatenatedDataset$



$fig[3]:\ validationAccuracyforHumanObservedConcatenatedDataset$



$fig[4]:\ lossforHumanObservedConcatenatedDataset$



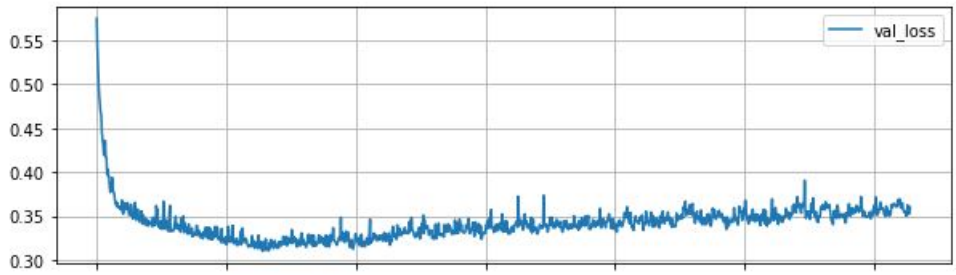
$fig[5]:\ AccuracyforHumanObservedConcatenatedDataset$

Human Observed Subtracted Dataset

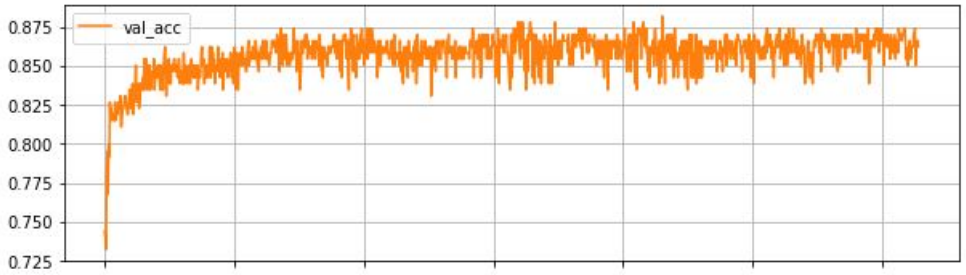
Layer (type)	Output Shape	Param #
dense_54 (Dense)	(None, 256)	2560
activation_51 (Activation)	(None, 256)	0
dropout_26 (Dropout)	(None, 256)	0
dense_55 (Dense)	(None, 2)	514
activation_52 (Activation)	(None, 2)	0

Total params: 3,074
Trainable params: 3,074
Non-trainable params: 0

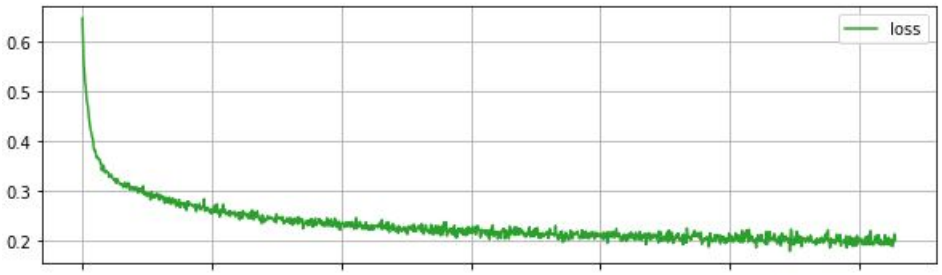
$fig[6]:\ ModelParametersofHumanObservedSubtractedDataset$



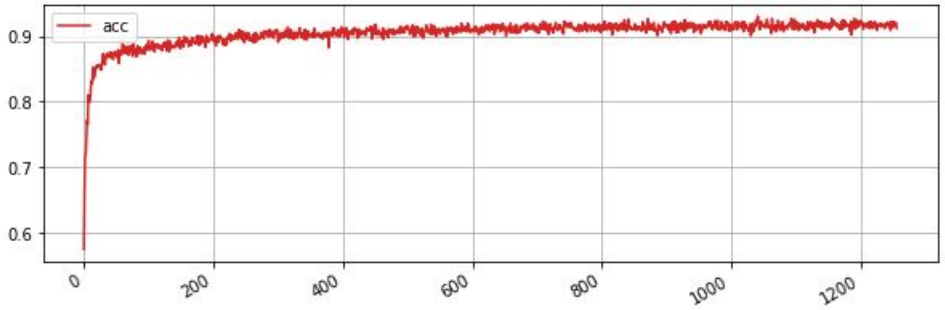
$fig[7]:\ validationlossforHumanObservedSubtractedDataset$



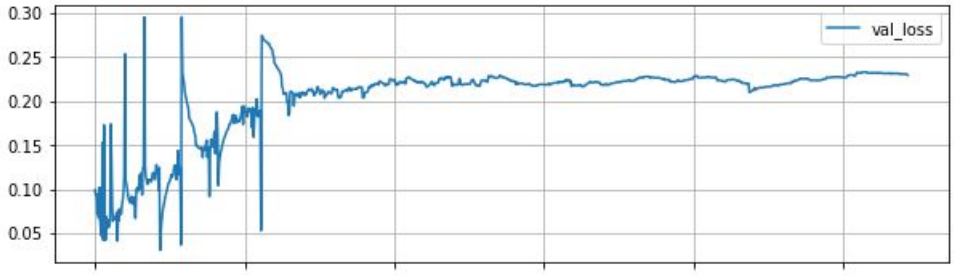
$fig[8]:\ validationAccuracyforHumanObservedSubtractedDataset$



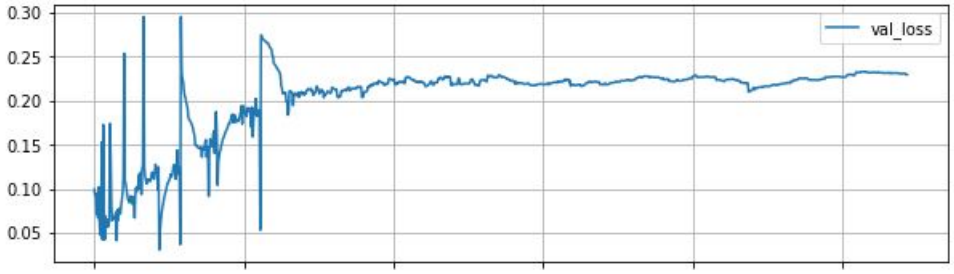
$fig[9]:\ lossforHumanObservedSubtractedDataset$



$fig[10]:\ AccuracyforHumanObservedSubtractedDataset$



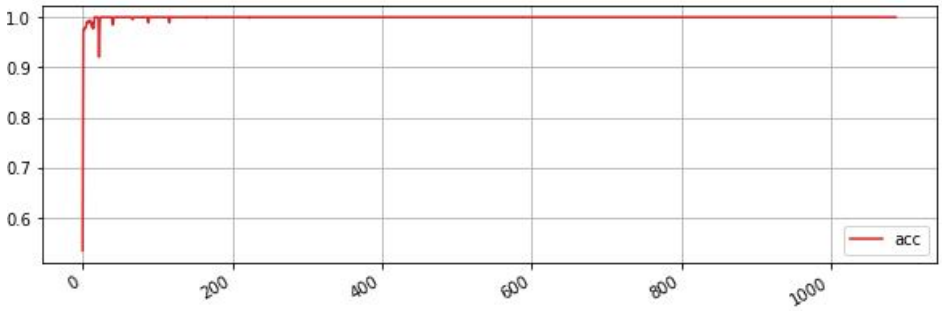
$fig[11]:\ ModelParametersofGSCObservedConcatenatedDataset$



$fig[12]:\ validationlossforGSCObservedConcatenatedDataset$

$fig[13]:\ validationAccuracyforGSCObservedConcatenatedDataset$

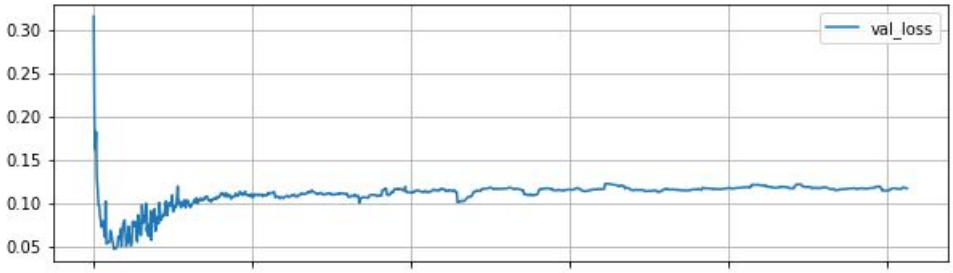
$fig[14]:\ lossforGSCObservedConcatenatedDataset$



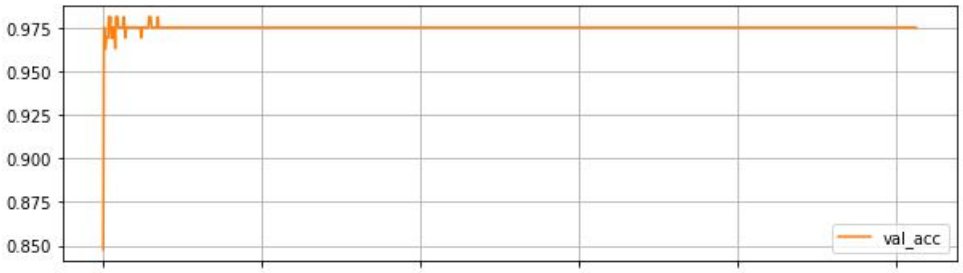
$fig[15]:\ AccuracyforGSCObservedConcatenatedDataset$

```
512
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_58 (Dense)             (None, 256)               131328
_________________________________________________________________
activation_55 (Activation)   (None, 256)               0
_________________________________________________________________
dropout_28 (Dropout)         (None, 256)               0
_________________________________________________________________
dense_59 (Dense)             (None, 2)                 514
_________________________________________________________________
activation_56 (Activation)   (None, 2)                 0
=================================================================
Total params: 131,842
Trainable params: 131,842
Non-trainable params: 0
_________________________________________________________________
```

$fig[16]:\ ModelParametersofGSCObservedSubtractedDataset$

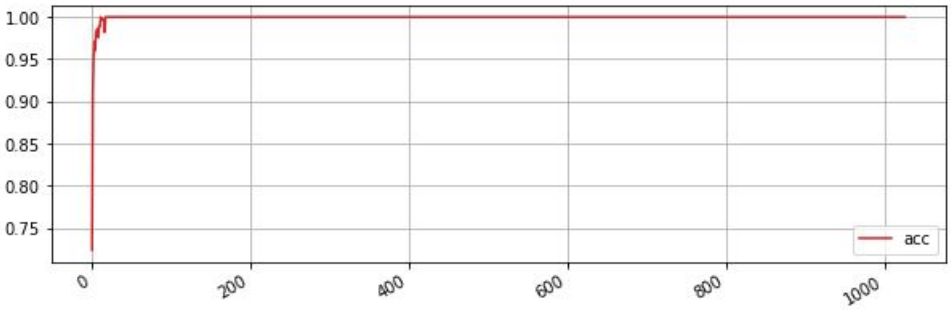


$fig[17]:\ validationlossforGSCObservedSubtractedDataset$



$fig[18]:\ validationAccuracyforGSCObservedSubtractedDataset$

$fig[19]:\ lossforGSCObservedSubtractedDataset$



$fig[20]:\ AccuracyforGSCObservedSubtractedDataset$

7 Conclusion:

From the above table[1]: Experimental Results.

1. For Human observed concatenated dataset, the accuracy of the neural network model is high compared to logistic and linear regression whose values are ranging at 98% , 83% , 62% respectively.Similarly for GSC concatenated Dataset, accuracy of the neural network model is high compared to logistic and linear regression models ranging at 97%, 94%, 60% respectively.

2. So while Comparing the Human and GSC concatenated dataset I conclude that the overall accuracy of the neural network model is high compared to Logistic regression and Linear Regression.

3. On other hand,For Human observed subtracted dataset, the accuracy of the neural network model is high compared to logistic and linear regression whose values are ranging at 79% , 60% , 51% respectively.Similarly for GSC subtracted Dataset, accuracy of the neural network model is high compared to logistic and linear regression models ranging at 97%, 93%, 52% respectively.

4. So while Comparing the Human and GSc subtracted dataset I conclude that the overall accuracy of the neural network model is high compared to Logistic regression and Linear Regression.

5. From my observations, I conclude that time complexity of the neural network model is less compared to logistic regression model. Similarly the time complexity for logistic regression model is less compared to liner regression model.

8 References:

1. https://beckernick.github.io/logistic-regression-from-scratch/
2. https://medium.com/@martinpella/logistic-regression-from-scratch-in-python-124c5636b8ac