PROJECT 2 Similarity Between Hand-written Samples using Different ML Models

Krishna Sehgal
50291124

1. OBJECTIVE

The goal of this project is to use machine learning to find similarity between the handwritten samples of the known and the questioned writer by using linear regression, Logistic Regression and Neural Networks.

There are two tasks:

1. Train a linear regression model on handwritten samples using a closed-form solution and stochastic gradient descent (SGD).

2. Train a logistic regression model on handwritten samples using a gradient descent solution

3. Train model on handwritten samples using Neural Networks

2. SOFTWARE USED

- Anaconda- Navigator
- Jupyter Notebook
- Python 3.6

3. LIBRARIES

- Sklearn
- Pandas
- Random
- TensorFlow
- NumPy
- CSV
- Math
- Matplot

- Keras

4. DATA SET

Our dataset uses “AND” images samples extracted from CEDAR Letter dataset. Image snippets of the word “AND” were extracted from each of the manuscript using transcript-mapping function of CEDAR-FOX. Figure 1. shows examples of the “AND” image fragments.

There are two types of Data-Set

- Human-Observed Data Set
- GSC Dataset using Feature Engineering

5. HYPER PARAMETERS

- Number of Clusters (M)
- Learning Rate
- Regularizer (λ)
- Batch Size
- Number of Epochs
- Drop_Out

6. TASK

Linear Regression is a kind of supervised learning technique in which training is done on an existing data-set. It is used to find relationship between a dependent variable and one or more independent variables. The training data given to us on hand-written samples of which we will use 80% of our data-set for training, 10% for validation and 10% to test whether model is working correctly or not.

6.1 Train a linear regression model on handwritten samples using a closed-form solution and stochastic gradient descent (SGD)

Neural networks that are used to solve a classification problem makes use of the equation $y=wx+b$ where w stands for weights, x for independent variable and b for bias. In case of Linear Regression, we cannot make use of this equation to predict value on a particular input this is because the function would be increasing and decreasing and is not linear so we can represent such curve with a set of curves known as basis function represented by $\Phi(x)$.

Step 1: K - means Clustering

Since we are working on a large data-set, it is difficult to apply linear regression directly on such a large data-set so we first perform clustering which is a unsupervised form of learning. In our model we have use K-means clustering where we choose random points as centroids of M number of clusters and after every iteration we improve our cluster set by finding mean repetitively. So we will first reduce our data-set by the process of K-means clustering.

Step 2: Linear Regression Model

Linear Regression model has the form,

$T=W^{T}\Phi(x)$

where T : Target Class W : Weight Vector Φ(x) : Basis Function

This equation will be used to predict value of weights since we know the basis function and our target class T having values 0,1,2. To find weights we will use the equation,

$W=\Phi^{-1}(x)\ T$

In order to find value of Φ^{-1} we use Moore Penrose pseudo-inverse of the matrix Φ which is show as,

$\Phi = (\Phi^{\top}\Phi)^{-1}\Phi^{\top}$

The closed form solution can be shown as,

$W= (\Phi^{\top}\Phi)^{-1}\Phi^{\top}\ T$

W that stands for weights will be a 10x1 matrix. $T= w_1\ \Phi_1(x)+ w_2\ \Phi_2(x)+ w_3\ \Phi_3(x)+.....w_{10}\Phi_{10}(x)$ Φ is the design matrix,

We have used a Gaussian Radial Basis Function that is shown by

where μ_j = Centroid of each cluster $\sum^{-1}_j$ = Variance - Covariance Matrix

In order to calculate Φ, we have to perform a matrix multiplication that should result into a scale 1x1 value. In order to do so we should have the following dimensions of terms in the gaussian radial basis function,

$$\mathbf{\Phi} = \begin{bmatrix} \phi_0(\mathbf{x}_1) & \phi_1(\mathbf{x}_1) & \phi_2(\mathbf{x}_1) & \cdots & \phi_{M-1}(\mathbf{x}_1) \\ \phi_0(\mathbf{x}_2) & \phi_1(\mathbf{x}_2) & \phi_2(\mathbf{x}_2) & \cdots & \phi_{M-1}(\mathbf{x}_2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \phi_0(\mathbf{x}_N) & \phi_1(\mathbf{x}_N) & \phi_2(\mathbf{x}_N) & \cdots & \phi_{M-1}(\mathbf{x}_N) \end{bmatrix}$$

$$\phi_j(\mathbf{x}) = \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_j)^{\top}\Sigma_j^{-1}(\mathbf{x}-\boldsymbol{\mu}_j)\right)$$

In the Gaussian Radial Basis function we have used a variance- covariance matrix which is a result of matrix multiplication of the feature square matrix with itself. Since we require co-variance that is the matrix multiplication of a feature with itself so we will make all the other variances as zero thus resulting in a diagonal covariance matrix shown by,

$$\Sigma = \begin{pmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_D^2 \end{pmatrix}$$

where σ stands for the spread of each basis function.

Use of Regularizer

We make use of regularizer to avoid the problem of over-fitting that is, if the basis functions try to cover each and every point present in the graph then there might be some redundant values or points due to which our model will learn wrong things and will make wrong predictions based on these.

The closed form solution with an add-on regularizer term to remove redundant terms is given by,

where The coefficient λ governs the relative importance of the regularization term. In the given closed form solution all the redundant values will be ignored by the model. Regularization can be motivated as a technique to improve the generalizability of a learned model. Regularization is used in order to prevent the problem of overfitting.

Error - root mean square

Root mean square represents the square root of the second sample moment of the differences between predicted values by our model and observed values or the quadratic mean of these differences. We have calculated the ERMS value after training, validation and testing so as to check whether our model has been improved or not by checking on sets of data.

In stochastic gradient descent (SGD) we are updating weights in the equation after every datapoint so as to minimize the error. Minimization of error is done by differentiating the error function so as to minimize it and update it accordingly.

Comparison between Gradient Descent and Stochastic Gradient Descent

$$\mathbf{w}^* = (\lambda \mathbf{I} + \mathbf{\Phi}^\top \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top \mathbf{t}$$

In GD optimization, we compute the cost gradient based on the complete training set; hence, we sometimes also call it *batch GD*. In case of very large datasets, using GD can be quite costly since we are only taking a single step for one pass over the training set -- thus, the larger the training set, the slower our algorithm updates the weights and the longer it may take until it converges to the global cost minimum.

The term "stochastic" comes from the fact that the gradient based on a single training sample is a "stochastic approximation" of the "true" cost gradient. Due to its stochastic nature, the path towards the global cost minimum is not "direct" as in GD, but may go "zig-zag" if we are visualizing the cost surface in a 2D space. However, it has been shown that SGD almost surely converges to the global cost minimum if the cost function is convex

Learning rate

Learning Rate refers to the amount of time taken by our algorithm to learn. If we reduce value of learning rate, classifier takes longer time to learn but with an improved accuracy.

6.2 Train a logistic regression model on handwritten samples using a gradient descent solution

Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Unlike linear regression which outputs continuous number values, logistic regression transforms its output using the logistic sigmoid function to return a probability value which can then be mapped to two or more discrete classes.

In order to map predicted values to probabilities, we use the sigmoid function. The function maps any real value into another value between 0 and 1. In machine learning, we use sigmoid to map predictions to probabilities. Our current prediction function returns a probability score between 0 and 1.

We can't use the same cost function as we did for linear regression because our prediction function is non-linear (due to sigmoid transform). Squaring this prediction as we do in MSE results in a non-convex function with many local minimums. If our cost function has many local minimums, gradient descent may not find the optimal global minimum.

Instead of Mean Squared Error, we use a cost function called Cross-Entropy, also known as Log Loss. Cross-entropy loss can be divided into two separate cost functions: one for y=1 and one for y=0. To minimize our cost, we use Gradient Descent just like before in Linear Regression.

6.3 Train model on handwritten samples using Neural Networks

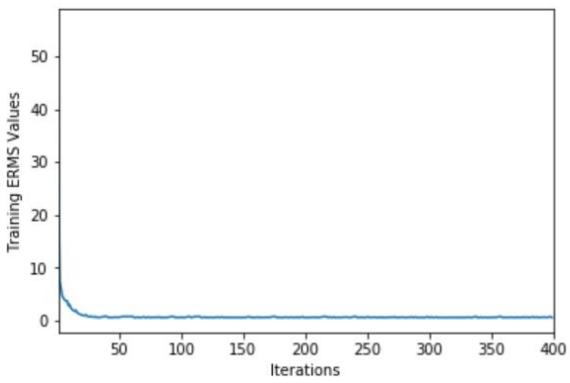
A neuron takes a group of weighted inputs, applies an activation function, and returns an output.
Inputs to a neuron can either be features from a training set or outputs from a previous layer's neurons. Weights are applied to the inputs as they travel along synapses to reach the neuron. The neuron then applies an activation function to the "sum of weighted inputs" from each incoming synapse and passes the result on to all the neurons in the next layer. **Bias** terms are additional constants attached to neurons and added to the weighted input before the activation function is applied. Bias terms help models represent patterns that do not necessarily pass through the origin. Bias terms typically accompany weights and must also be learned by your model.
Input Layer holds the data your model will train on. Each neuron in the input layer represents a unique attribute in your dataset.
Hidden Layer sits between the input and output layers and applies an activation function before passing on the results. There are often multiple hidden layers in a network. In traditional networks, hidden layers are typically fully-connected layers — each neuron receives input from all the previous layer's neurons and sends its output to every neuron in the next layer. This contrasts with how convolutional layers work where the neurons send their output to only some of the neurons in the next layer.
The final layer in a network. It receives input from the previous hidden layer, optionally applies an activation function, and returns an output representing your model's prediction.
A neuron's input equals the sum of weighted outputs from all neurons in the previous layer. Each input is multiplied by the weight associated with the synapse connecting the input to the current neuron.
Activation functions live inside neural network layers and modify the data they receive before passing it to the next layer. **Activation functions** give neural networks their power — allowing them to model complex non-linear relationships. By modifying inputs with non-linear functions neural networks can model highly complex relationships between features. Popular activation functions include relu and sigmoid.
A l**oss function, or cost function**, is a wrapper around our model's predict function that tells us "how good" the model is at making predictions for a given set of parameters. The loss function has its own curve and its own derivatives. The slope of this curve tells us how to change our parameters to make the model more accurate! We use the model to make predictions. We use the cost function to update our parameters.

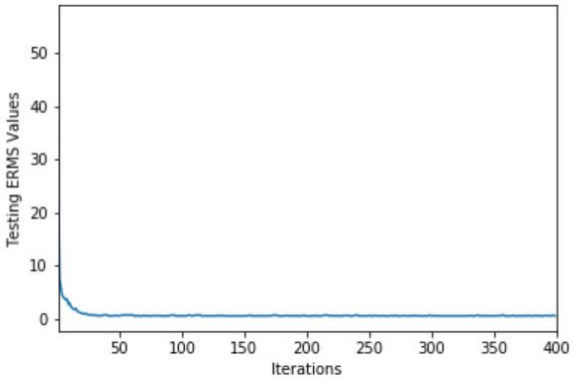
7. GRAPHICAL REPRESENTATION

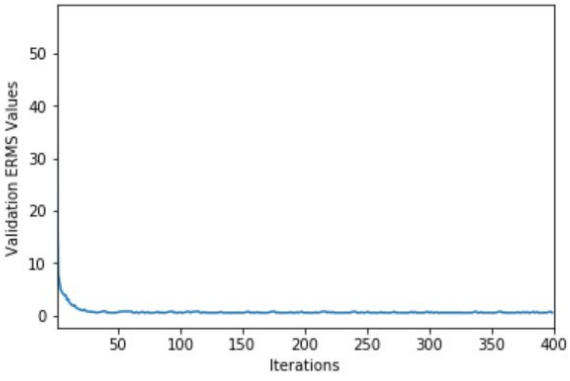
7.1 Linear Regression

7.1.1 Human Observed Data-set using Concatenation

```
UBITname      = Krishna Sehgal
Person Number = 50291124
--------------------------------------------------
--------------------------------------------------
-------Closed Form with Radial Basis Function-------
--------------------------------------------------
M = 10
Lambda = 0.9
E_rms Training   = 0.498493315845173
E_rms Validation = 0.505048998344562
E_rms Testing    = 0.502304579939376
Accuracy Training   = 53.712480252764614
Accuracy Validation = 49.685534591194966
Accuracy Testing    = 50.0
--------------------------------------------------
-------------Please Wait for 2 mins!----------------
--------------------------------------------------
----------Gradient Descent Solution--------------------
M = 15
Lambda  = 0.0001
eta=0.01
E_rms Training   = 0.50018
E_rms Validation = 0.49216
E_rms Testing    = 0.49838
```

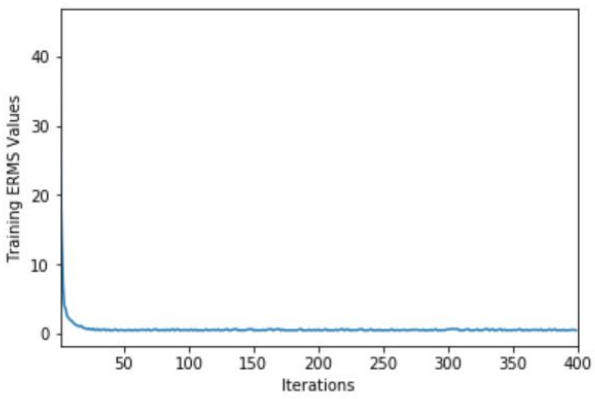


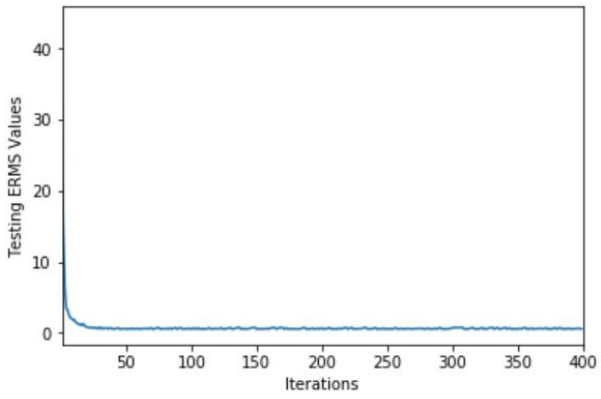


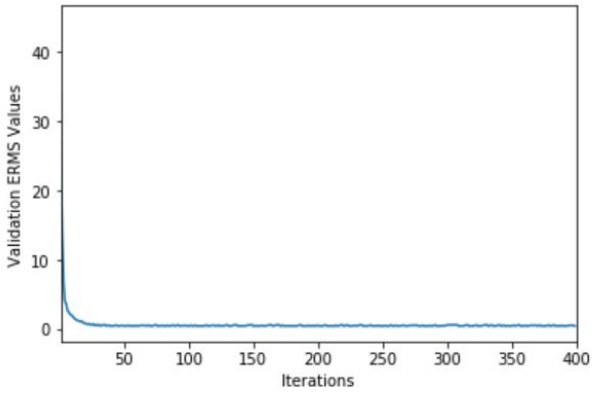


7.1.2 Human Observed Data-set using Subtraction

```
UBITname      = Krishna Sehgal
Person Number = 50291124
--------------------------------------------------
--------------------------------------------------
-------Closed Form with Radial Basis Function-------
--------------------------------------------------
M = 10
Lambda = 0.9
E_rms Training   = 0.49898742283039343
E_rms Validation = 0.5025539252009872
E_rms Testing    = 0.5019735639851859
Accuracy Training   = 53.001579778830965
Accuracy Validation = 49.056603773584904
Accuracy Testing    = 50.0
--------------------------------------------------
--------------Please Wait for 2 mins!----------------
--------------------------------------------------
----------Gradient Descent Solution--------------------
M = 15
Lambda  = 0.0001
eta=0.01
E_rms Training   = 0.49969
E_rms Validation = 0.4989
E_rms Testing    = 0.49932
```

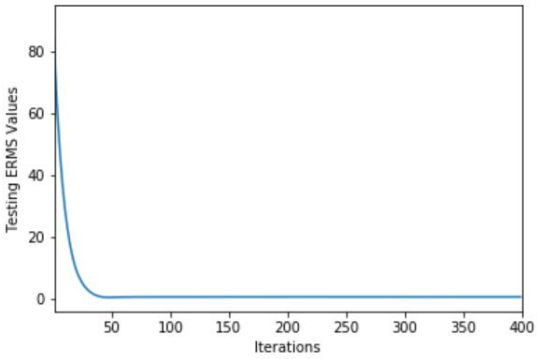


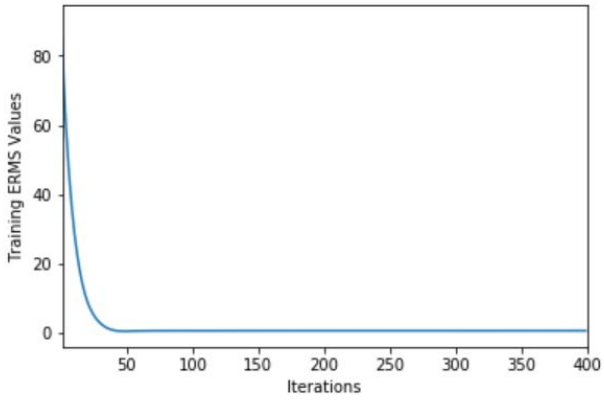


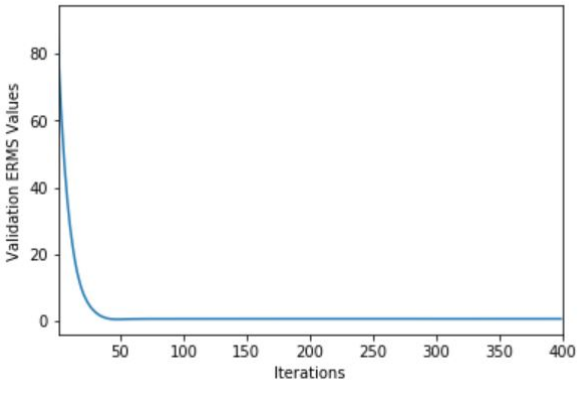


7.1.3 GSC Data-set using Concatenation

```
UBITname      = Krishna Sehgal
Person Number = 50291124
--------------------------------------------------
--------------------------------------------------
-------Closed Form with Radial Basis Function-------
--------------------------------------------------
M = 10
Lambda = 0.9
E_rms Training   = 0.5400642812693723
E_rms Validation = 0.529642832242614
E_rms Testing    = 0.5318249284583857
Accuracy Training   = 50.1913499344692
Accuracy Validation = 53.01600615083525
Accuracy Testing    = 52.16692296938348
--------------------------------------------------
-------------Please Wait for 2 mins!----------------
--------------------------------------------------
----------Gradient Descent Solution--------------------
M = 15
Lambda  = 0.0001
eta=0.01
E_rms Training   = 0.54242
E_rms Validation = 0.53198
E_rms Testing    = 0.53441
```

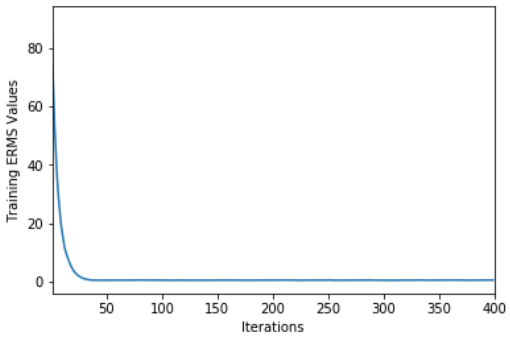


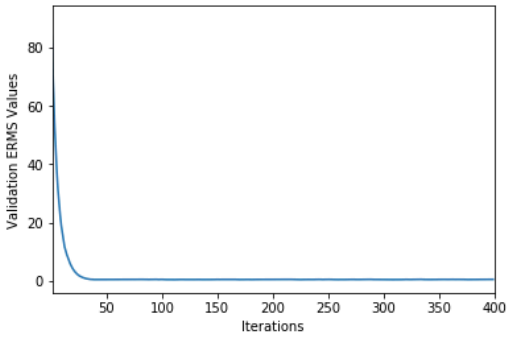


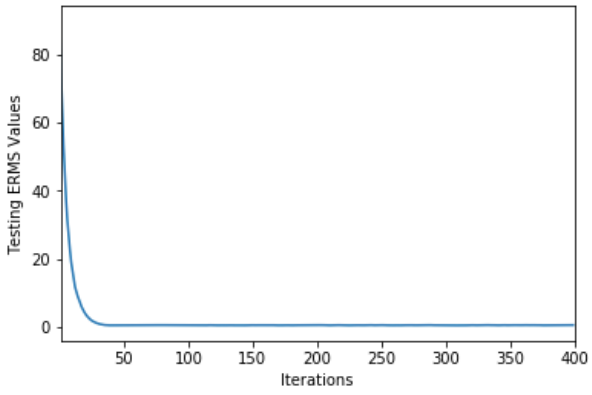


7.1.4 GSC Data-set using Subtraction

```
UBITname      = Krishna Sehgal
Person Number = 50291124
--------------------------------------------------
--------------------------------------------------
-------Closed Form with Radial Basis Function-------
--------------------------------------------------
M = 10
Lambda = 0.9
E_rms Training   = 0.5128883833980322
E_rms Validation = 0.5130758571673361
E_rms Testing    = 0.5140745924066413
Accuracy Training   = 50.246395806028836
Accuracy Validation = 49.98951562172363
Accuracy Testing    = 49.72738711030337
--------------------------------------------------
--------------Please Wait for 2 mins!----------------
--------------------------------------------------
----------Gradient Descent Solution--------------------
M = 15
Lambda  = 0.0001
eta=0.01
E_rms Training   = 0.55225
E_rms Validation = 0.55072
E_rms Testing    = 0.5537
```



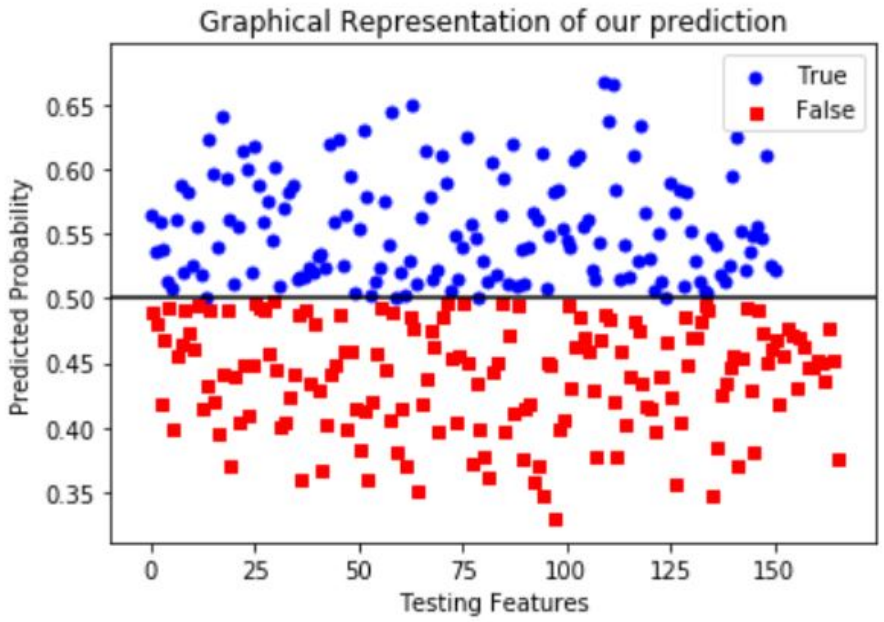




7.2 Logistic Regression

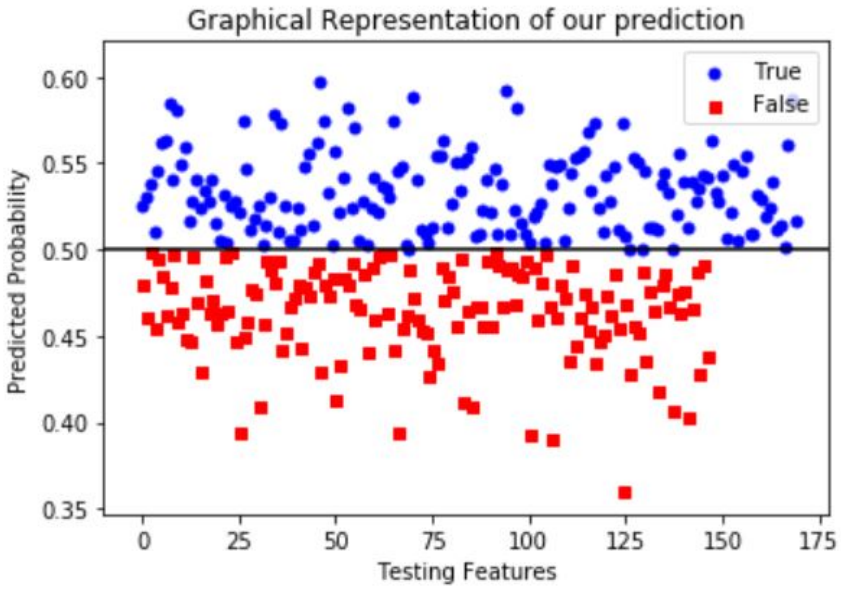
7.2.1 Human Observed Data-set using Concatenation

```
UBITname      = Krishna Sehgal
Person Number = 50291124
--------------------------------------------------
--------------------------------------------------
-------Logistic Regression-------
--------------------------------------------------
----------Gradient Descent Solution--------------------
Learning Rate  = 0.05
Accuracy:
51.73501577287066
```

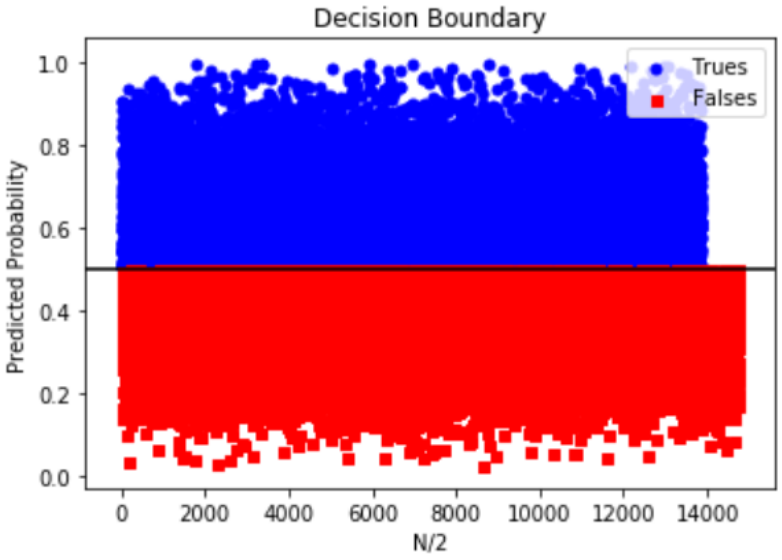


7.2.2 Human Observed Data-set using Subtraction

```
UBITname      = Krishna Sehgal
Person Number = 50291124
--------------------------------------------------
--------------------------------------------------
-------Logistic Regression-------
--------------------------------------------------
----------Gradient Descent Solution--------------------
Learning Rate  = 0.05
Accuracy:
50.78864353312302
```



7.2.3 GSC Data-set using Concatenation



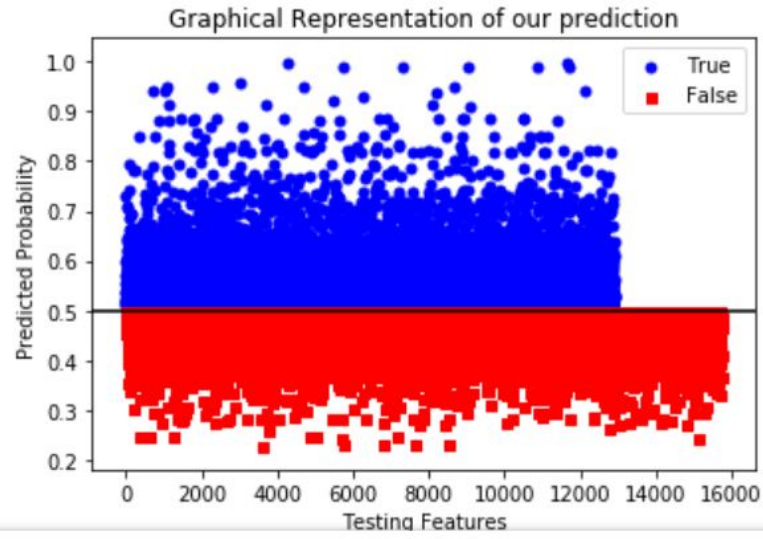

```
61.05266836752526
```

Accuracy = 61.05

Learning Rate = 0.05

7.2.4 GSC Data-set using Subtraction

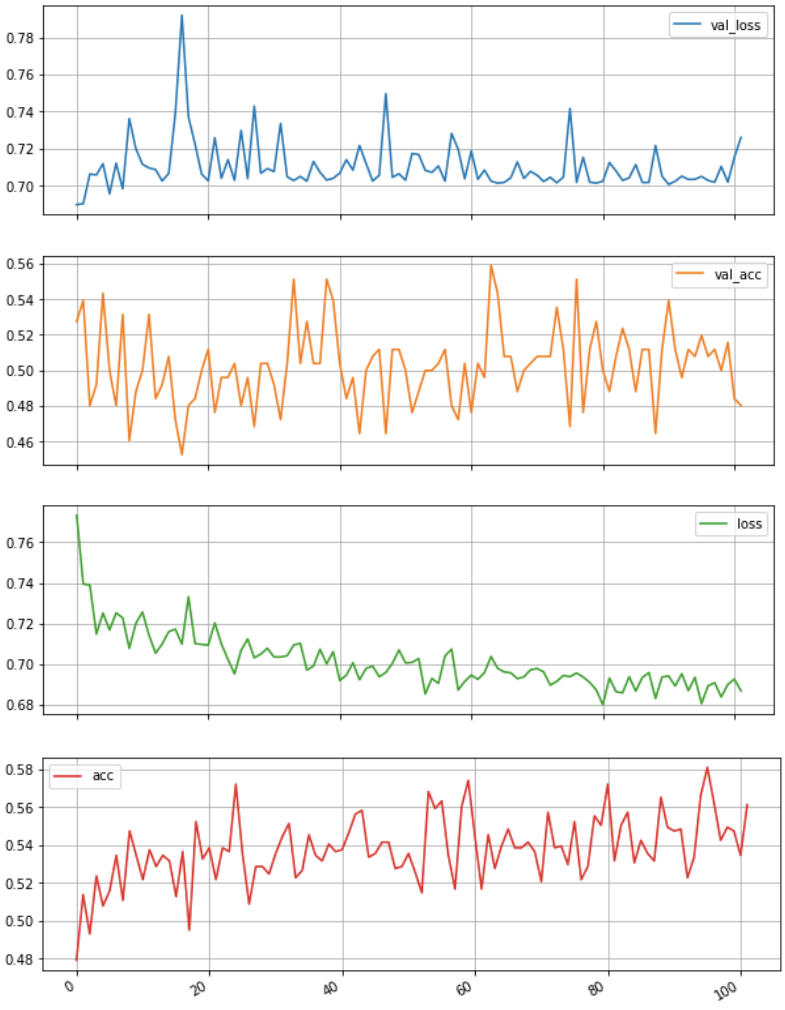
```
UBITname      = Krishna Sehgal
Person Number = 50291124
--------------------------------------------------
--------------------------------------------------
-------Logistic Regression-------
--------------------------------------------------
----------Gradient Descent Solution--------------------
Learning Rate  = 0.05
Accuracy:
49.98077796805648
```



7.3 Neural Network

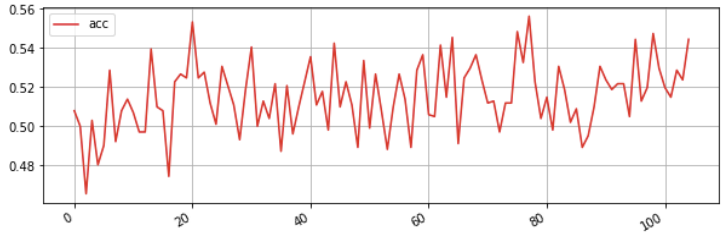
7.3.1 Human Observed Data-set using Concatenation

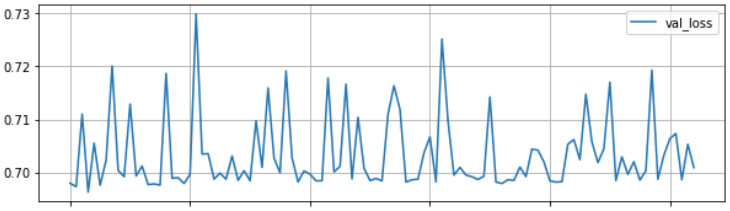
```
_________________________________________________________________
Errors: 155  Correct :162
UBITname      = Krishna Sehgal
Person Number = 50291124
------------------------------
-------Neural Networks-------
------------------------------
Testing Accuracy: 51.10410094637224
```

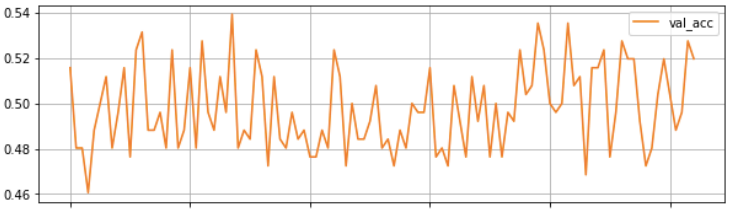


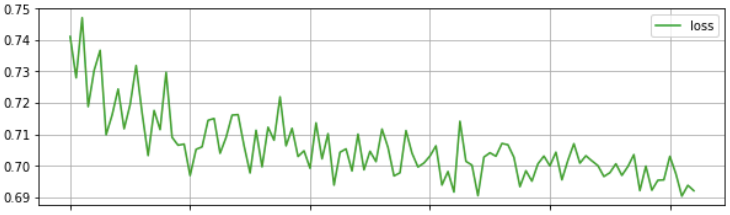
7.3.2 Human Observed Data-set using Subtraction

```
UBITname      = Krishna Sehgal
Person Number = 50291124
--------------------------------
-------Neural Networks-------
--------------------------------
Testing Accuracy: 54.25867507886435
```



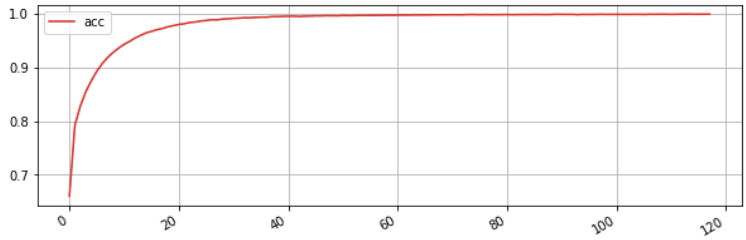


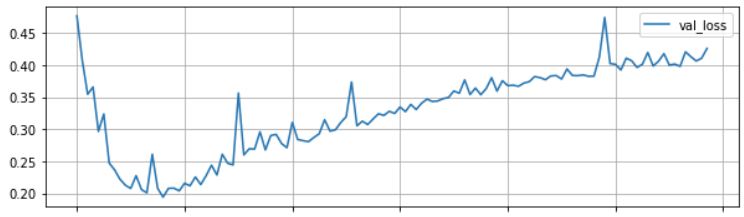


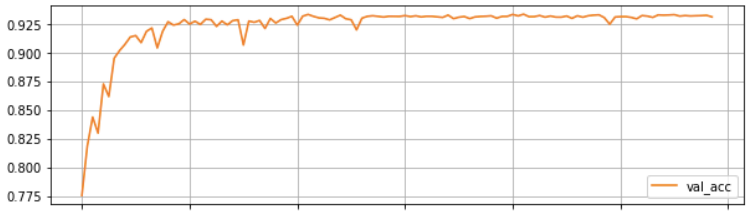


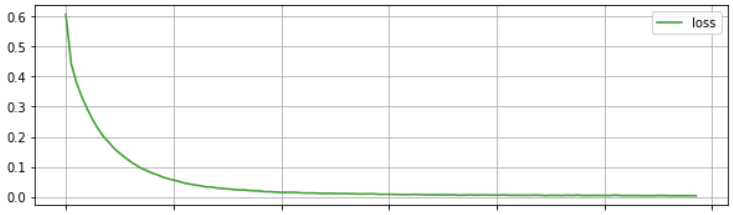
7.3.3 GSC Data-set using Concatenation

```
UBITname      = Krishna Sehgal
Person Number = 50291124
------------------------------
-------Neural Networks-------
------------------------------
Testing Accuracy: 93.3282074581484
```



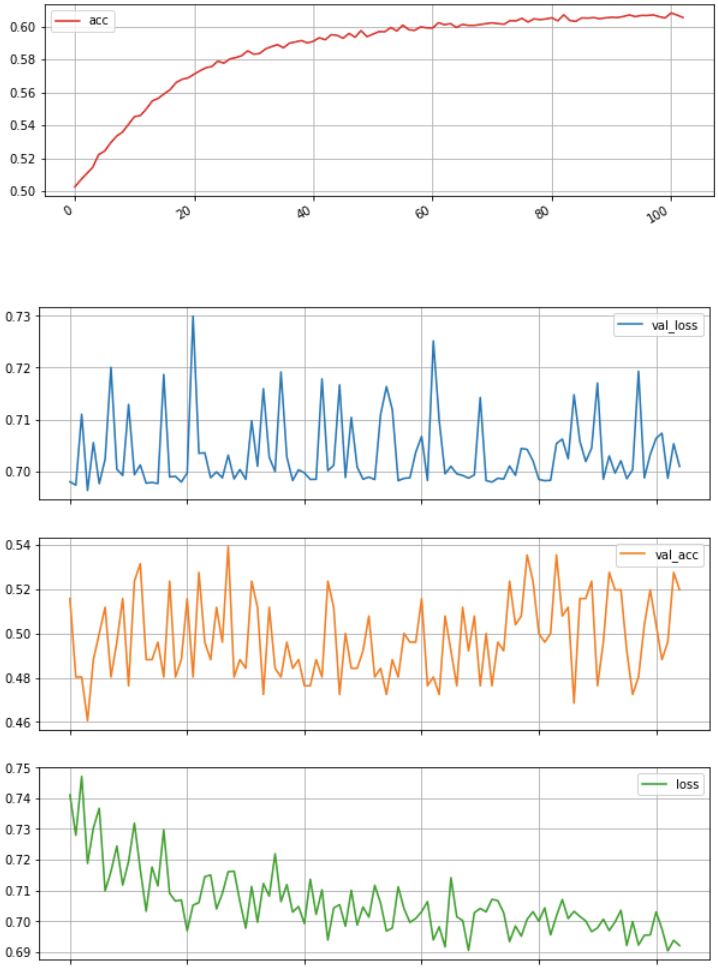






7.3.4 GSC Data-set using Subtraction

```
------------------------------
-------Neural Networks-------
------------------------------
Testing Accuracy: 49.44256107363786
```



8. CONCLUSION

We have used Logistic Regression, Linear Regression and Neural Networks to predict accuracy of our model to find similarity between the handwritten samples of the known and the questioned writer.