CSE574 Introducion to Machine Learning

by Prof: Sargur Srihari

Project 2 Report.

Submitted by

Sai Varun Alapati
Ubit : saivarun
Personal Number :50290571

Table of Contents

Introduction :

The aim of the project is to use machine learning to solve a problem that arises in FBI and to help the Bureau and police departments to solve criminal cases dealing with evidence provided by handwritten documents such as wills and ransom notes. We formulate this as a problem of linear regression where we map a set of input features x to a real-valued scalar target y(x,w).

We find the similarity between the handwritten samples of the known and the questioned writer by using linear regression, Logistic Regression and Neural Networks.

Problem Definition :

In this project, We find the similarity between the handwritten samples of the known and the questioned writer by using linear regression, Logistic Regression and Neural Networks.

Each instance in the CEDAR \AND" training data consists of set of input features for each hand-written \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 target values are scalars that can take two values { 1: same writer, 0:dfferent writers}. Although

the training target values are discrete we use linear regression, Logistic Regression and Neural Networks to obtain real values which is more useful for finding similarity.

Dataset Description and Pre Processing :

Source of Dataset :

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.

Figure 1

	and	and	and
Sample ID [XXXXy_numZ]	0001a_num1	0001a_num2	0002a_num1
Writer Number [XXXX]	Writer 0001	Writer 0001	Writer 0002
Page Number [y]	Page 1	Page 1	Page 1
Sample Number [Z]	Sample 1	Sample 2	Sample 1

Human Observed Dataset Descrition

The Human Observed dataset shows only the cursive samples in the data set, where for each image the features are entered by the human document examiner. There are total of 18 features for a pair of handwritten \AND" sample (9 features for each sample).

The entire dataset consists of 791 same writer pairs and 293,032 different writer pairs(rows). The shows two sample rows derived using the three csv _les for human observed dataset:The meaning of each column of above dataset are as follows :

1. The first column is the relevance label of the row. It takes one of the discrete values 0, 1 or 2. The larger the relevance label, the better is the match between query and document. Note that objective output y of our linear regression will give a continuous value rather than a discrete one– so as to give a fine-grained distinction.
2. The second column qid is the query id. It is only useful for indexing the dataset and not used in regression.
3. The following 46 columns are the features. They are the 46-dimensional input vector x for our linear regression model. All the features are normalized to fall in the interval of [0; 1].

Figure 2

img_id_A	img_id_B	f_{A1}	f_{A2}	f_{A3}	f_{A4}	f_{A5}	f_{A6}	f_{A7}	f_{A8}	f_{A9}	f_{B1}	f_{B2}	f_{B3}	f_{B4}	f_{B5}	f_{B6}	f_{B7}	f_{B8}	f_{B9}	t
1121a_num1	1121b_num2	2	1	1	3	2	2	0	1	2	2	1	1	0	2	2	0	3	2	1
1121a_num1	1386b_num1	2	1	1	3	2	2	0	1	2	3	1	1	0	2	2	0	1	2	0

There are two settings under which we performed linear regression, Logistic Regression and Neural Networks :
Setting 1: Feature Concatenation [18 features]
Setting 2: Feature subtraction [9 features]

GSC Dataset using Feature Engineering

Gradient Structural Concavity algorithm generates 512 sized feature vector for an input handwritten "AND"image. The dataset is named as \GSC-Features-Data. The entire dataset consists of 71,531 same writer pairs and 762,557 different writer pairs(rows). Figure 3. shows two sample rows derived using the three csv files for GSC dataset:

Figure 3: GSC Dataset Example

img_id_A	img_id_B	f_{A1}	f_{A2}	f_{A3}	f_{A4}	f_{A5}	f_{A6}	. . .	f_{A512}	f_{B1}	f_{B2}	f_{B3}	f_{B4}	f_{B5}	f_{B6}	. . .	f_{B512}	t
1121a_num1	1121b_num2	0	1	1	0	1	0	. . .	0	0	1	1	0	0	1	. . .	1	1
1121a_num1	1386b_num1	0	1	1	0	1	0	. . .	0	1	1	1	0	1	0	. . .	0	0

There are two settings under which we performed linear regression, Logistic Regression and Neural Networks :
Setting 1: Feature Concatenation [1024 features]
Setting 2: Feature subtraction [512 features]

Data Preprocessing steps :

Extracting Feature Values and labels form the data :

There are three types of data portioning techniques

1. Unseen Writer Partitioning: In this method there exists no writer which is present in both the training (Tr) and testing (Ts) writer set simultaneously. Hence, any test writer would not be a part of training set and vice-versa.
2. Shuffled Writer Partitioning: In this method, entire dataset is first shuffled. Hence, there are X writers which are concurrent in both the training (Tr) and testing (Ts) writer set. Hence, given a test writer may or may not be present in the training set.
3. Seen Writer Partitioning: In this method, we train over 80% of each writers samples and test over the remaining 20% samples of each writer.

We partitioned the data using Shuffled writer partitioning where 50% dataset consists of same pair writers and 50% consists of different pair writers which doesn't make dataset skewed.

The two types of data set contains the below setting for Human Observed Data.
Setting 1: Feature Concatenation [18 features]
Setting 2: Feature subtraction [9 features]
For Gsc we took 5000 data samples from same pair writers and another 5000 samples from different pair writers and we increased the data samples for the better erms and accuracy.
The two types of data set contains the below setting for GSC Data.
Setting 1: Feature Concatenation [1024 features]
Setting 2: Feature subtraction [512 features]
We saved the datasets into csv file that contains the feature vectors, labels and the target values.

Target Values :

The Fourth column of the csv file contains the target values having values 0 and 1, where 1 says the two images are from same writer and 0 says the two images are from different writers.

Feature vectors :

The entire HOD dataset consists of 791 same writer pairs and 293,032 different writer pairs(rows). There are 9 feature vectors for each image. So the Setting 1: Feature Concatenation contains 18 features and Setting 2: Feature subtraction contains 9 features.
The entire GSC dataset consists of 71,531 same writer pairs and 762,557 different writer pairs(rows). There are 512 feature vectors for each image. So the Setting 1: Feature Concatenation contains 1024 features and Setting 2: Feature subtraction contains 512 features.

Data Division:

We divided the data into a training set, a validation set and a testing set. The training set takes around 80% of the total. The validation set takes about 10% . The testing set takes the rest without overlapping with the above.

Training Linear Regression Model :

we train a linear regression model onLeToR dataset using the following methods :

$$y(x, w) = \omega^T \phi(x)$$

1. Linear Regression usingClosed-form solution
2. Linear Regression using Stochastic gradient descent (SGD).

Linear Regression using Stochastic gradient descent (SGD) :

- It is the process of mininmizing a function by following the gradients of the cost function.
- This involves knowing the form of the cost as well as the derivative so that from a given point you know the gradient and can move in that direction, e.g. downhill towards the minimum value.

- In Machine learning we can use a similar technique called stochastic gradient descent to minimize the error of a model on our training data.
- The stochastic gradient descent algorithm takes a random initial value. Then it updates the value of w using
  $W^{(\tau+1)} = W^{(\tau)} + \Delta W^{(\tau)}$

where $\Delta w(\tau) = -\eta^{(\tau)} \Delta E$ is called the weight updates. It goes along the opposite direction of the gradient of the error. $\eta(\tau)$ is the learning rate, deciding how big each update step would be. Because of the linearity of differentiation, we have

$$\nabla E = \nabla E_D + \lambda \nabla E_W$$

$$\nabla E_D = -(t_n - \mathbf{w}^{(\tau)\top} \phi(\mathbf{x}_n))\phi(\mathbf{x}_n)$$
$$\nabla E_W = \mathbf{w}^{(\tau)}$$

Experimental Values :

Hyperparameters Used :

The following are the hyper parameters used in this model :

- M - The number of basis functions used in the model.
- C_Lambda – This is the regularization factor which helps to reduce the overfitting of the model by adding some random weights to the desired weights.
-

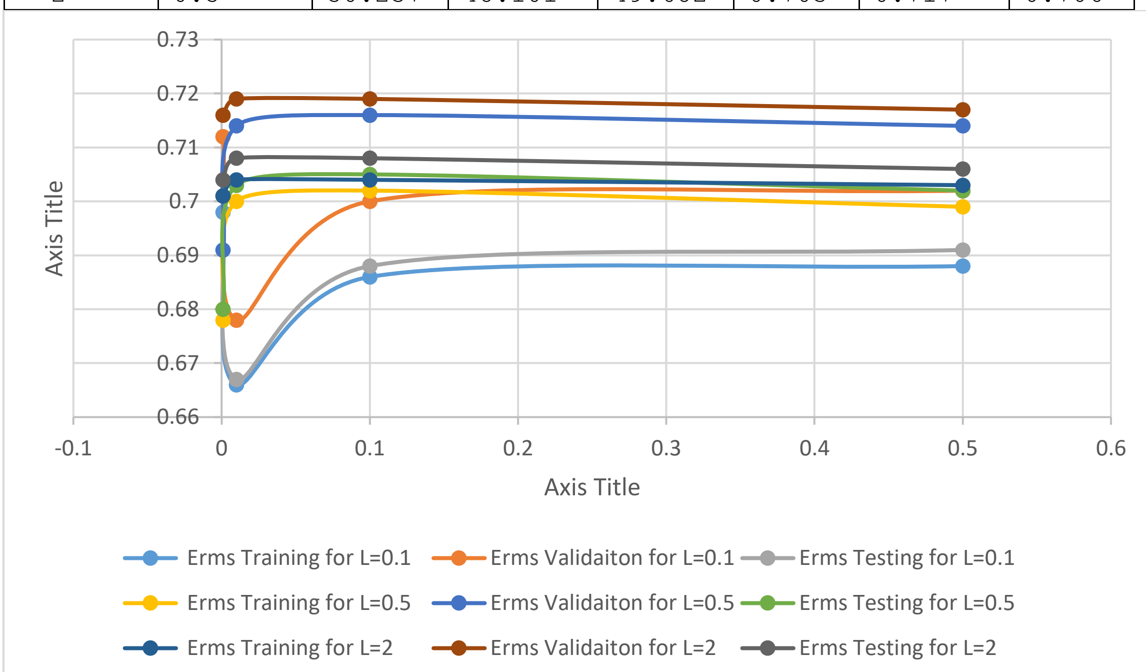
Varying Number of Basis Functions and Regularization, Eta :

These are the number of basis funcitons used in the model are varied from 1 and 15 for different Lambda Values and eta values:

For Hod feature Concatenation -

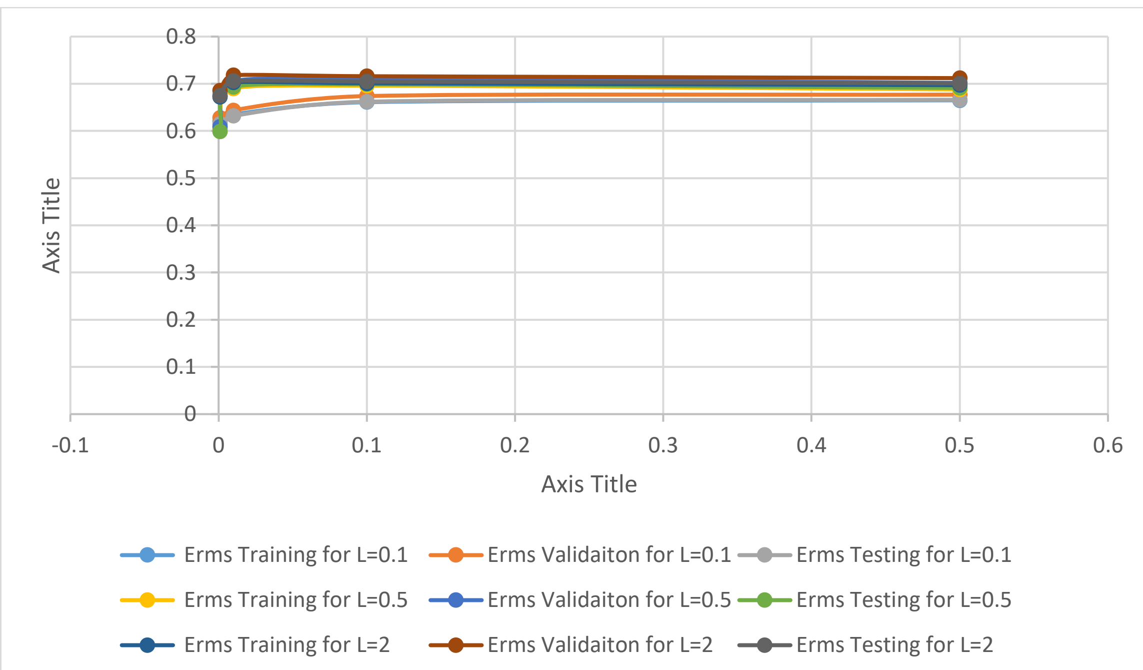
For M=1, the following values are observed from this model:

C_Lambda	Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy	Emrs Training	Erms Validaiton	Erms Testing
0.1	0.001	50.237	48.101	49.682	0.698	0.712	0.701
0.1	0.01	50.237	48.101	49.682	0.666	0.678	0.667
0.1	0.1	50.237	48.101	49.682	0.686	0.7	0.688
0.1	0.5	50.237	48.101	49.682	0.688	0.702	0.691
0.5	0.001	50.237	48.101	49.682	0.678	0.691	0.68
0.5	0.01	50.237	48.101	49.682	0.7	0.714	0.703
0.5	0.1	50.237	48.101	49.682	0.702	0.716	0.705
0.5	0.5	50.237	48.101	49.682	0.699	0.714	0.702
2	0.001	50.237	48.101	49.682	0.701	0.716	0.704
2	0.01	50.237	48.101	49.682	0.704	0.719	0.708
2	0.1	50.237	48.101	49.682	0.704	0.719	0.708
2	0.5	50.237	48.101	49.682	0.703	0.717	0.706



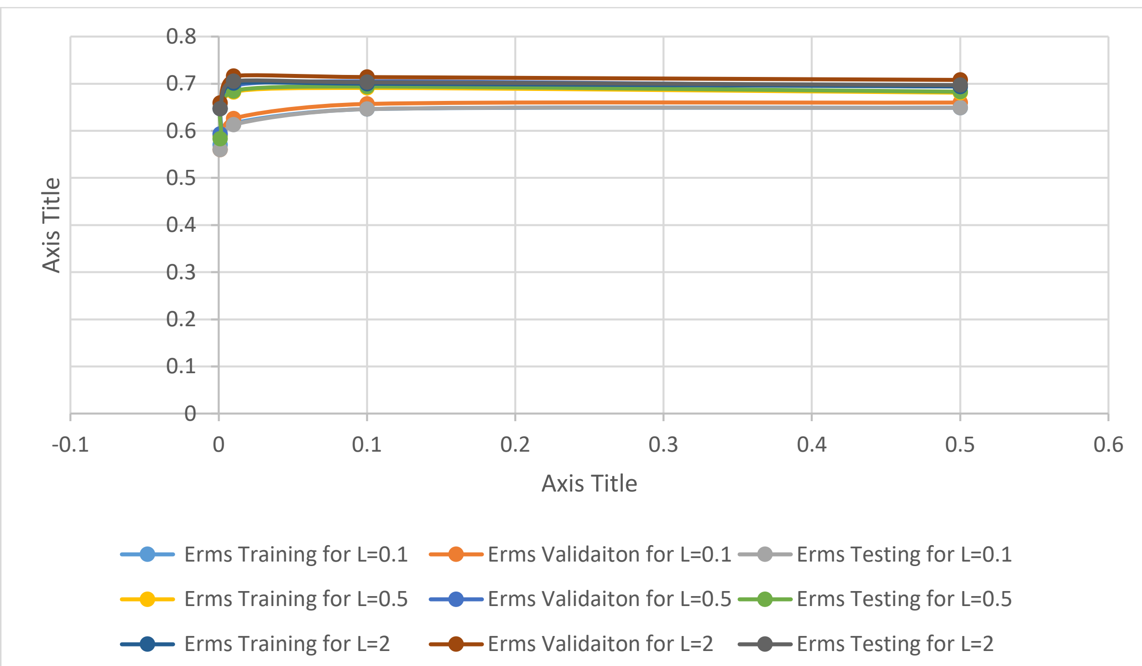
For M=5, the following values are observed from this model:

C_Lambda	Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy	Emrs Training	Erms Validaiton	Erms Testing
0.1	0.001	50.79	49.367	49.682	0.62	0.628	0.615
0.1	0.01	50.395	48.101	49.682	0.635	0.644	0.632
0.1	0.1	50.237	48.101	49.682	0.661	0.674	0.662
0.1	0.5	50.237	48.101	49.682	0.665	0.677	0.666
0.5	0.001	51.659	51.266	52.229	0.603	0.609	0.599
0.5	0.01	50.237	48.101	49.682	0.69	0.705	0.693
0.5	0.1	50.237	48.101	49.682	0.696	0.71	0.699
0.5	0.5	50.237	48.101	49.682	0.688	0.702	0.691
2	0.001	50.237	48.101	49.682	0.672	0.686	0.675
2	0.01	50.237	48.101	49.682	0.703	0.718	0.706
2	0.1	50.237	48.101	49.682	0.701	0.716	0.705
2	0.5	50.237	48.101	49.682	0.697	0.712	0.701



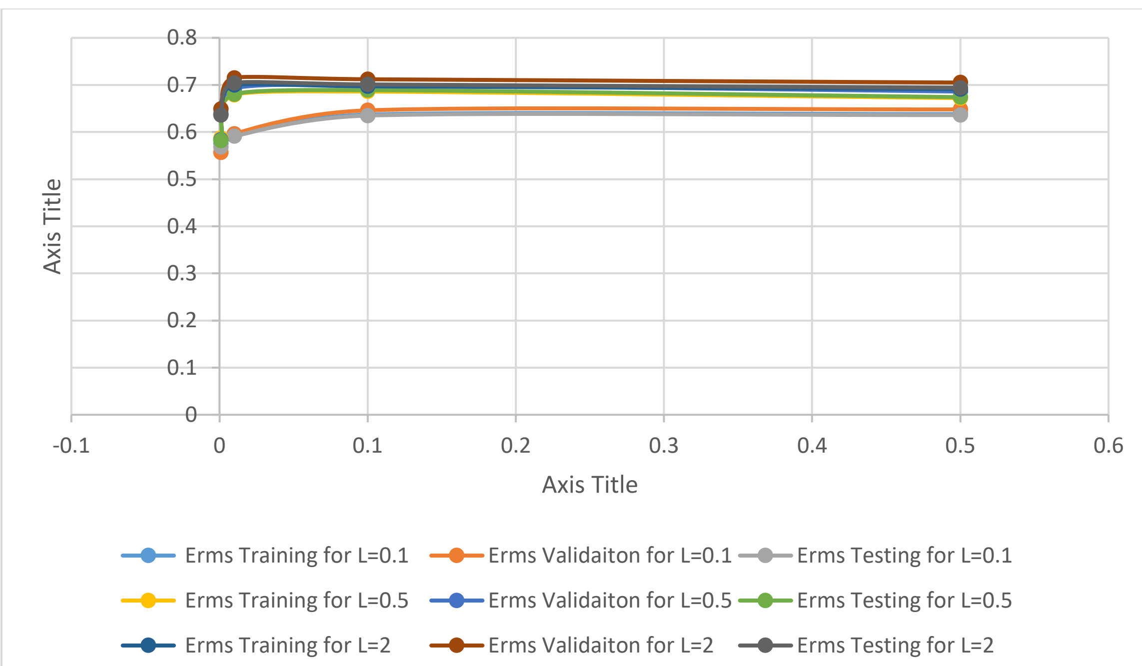
For M=10 , the following values are observed from this model:

C_Lambda	Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy	Emrs Training	Erms Validaiton	Erms Testing
0.1	0.001	54.423	56.962	59.236	0.57	0.56	0.56
0.1	0.01	50.948	49.367	49.682	0.616	0.626	0.613
0.1	0.1	50.237	48.101	49.682	0.646	0.657	0.646
0.1	0.5	50.237	48.101	49.682	0.649	0.66	0.649
0.5	0.001	52.528	52.532	53.503	0.589	0.593	0.583
0.5	0.01	50.237	48.101	49.682	0.682	0.695	0.684
0.5	0.1	50.237	48.101	49.682	0.691	0.705	0.694
0.5	0.5	50.237	48.101	49.682	0.681	0.694	0.683
2	0.001	50.237	48.101	49.682	0.647	0.659	0.647
2	0.01	50.237	48.101	49.682	0.701	0.716	0.705
2	0.1	50.237	48.101	49.682	0.7	0.714	0.703
2	0.5	50.237	48.101	49.682	0.694	0.708	0.697



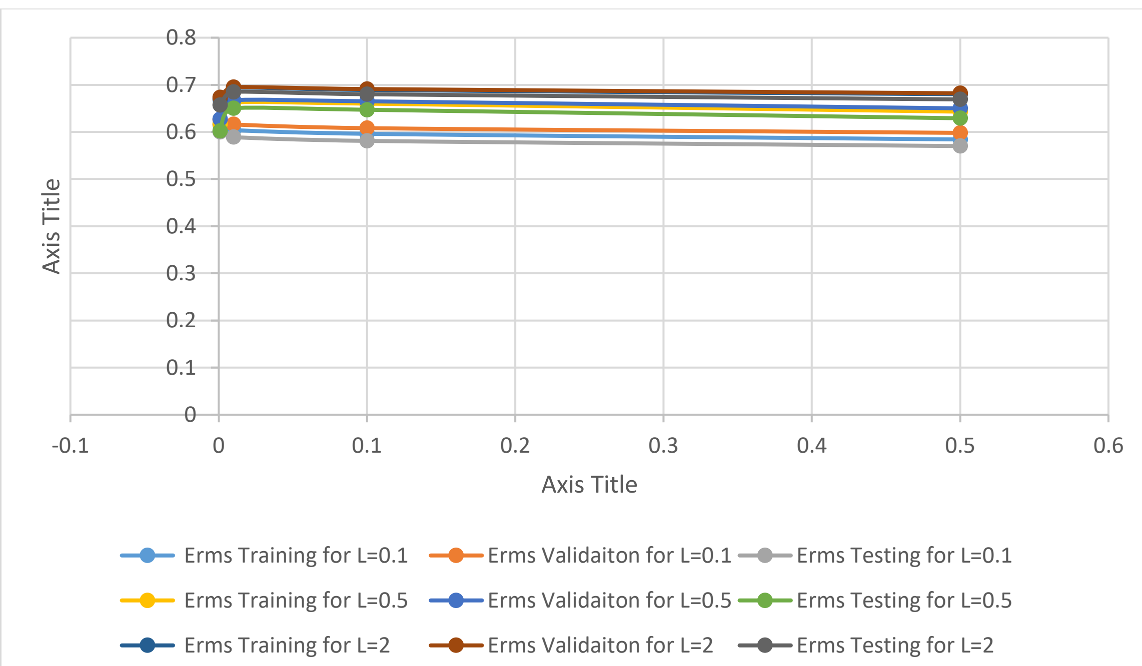
For M=15, the following values are observed from this model:

C_Lambda	Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy	Emrs Training	Erms Validaiton	Erms Testing
0.1	0.001	54.581	56.962	57.962	0.576	0.557	0.568
0.1	0.01	51.896	53.165	52.229	0.596	0.596	0.591
0.1	0.1	50.237	48.101	49.682	0.637	0.646	0.635
0.1	0.5	50.237	48.101	49.682	0.638	0.648	0.636
0.5	0.001	52.923	54.43	54.14	0.587	0.584	0.582
0.5	0.01	50.237	48.101	49.682	0.679	0.692	0.68
0.5	0.1	50.237	48.101	49.682	0.686	0.7	0.689
0.5	0.5	50.237	48.101	49.682	0.672	0.686	0.674
2	0.001	50.237	48.101	49.682	0.638	0.649	0.636
2	0.01	50.237	48.101	49.682	0.7	0.715	0.704
2	0.1	50.237	48.101	49.682	0.697	0.712	0.701
2	0.5	50.237	48.101	49.682	0.691	0.705	0.694



For M=1, the following values are observed from this model:

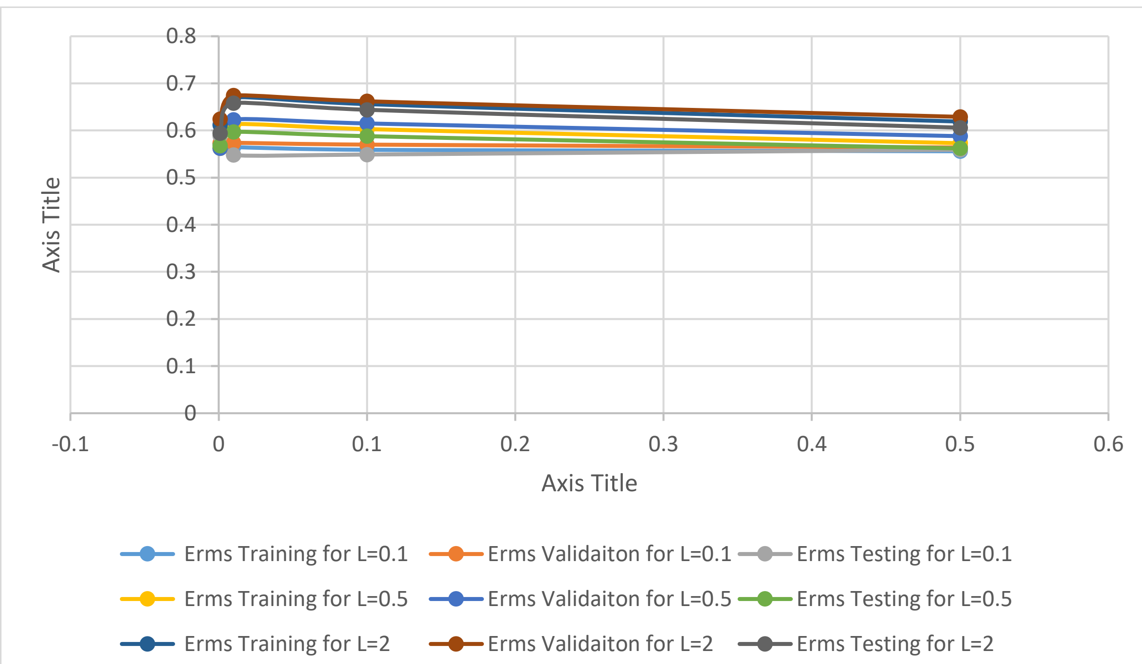
C_Lambda	Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy	Emrs Training	Erms Validaiton	Erms Testing
0.1	0.001	49.842	50.0	51.592	0.615	0.626	0.6
0.1	0.01	49.605	50.633	49.682	0.604	0.616	0.589
0.1	0.1	49.842	50.633	47.771	0.596	0.608	0.581
0.1	0.5	49.684	49.367	49.045	0.584	0.598	0.57
0.5	0.001	49.842	50.0	50.955	0.617	0.627	0.602
0.5	0.01	49.842	50.0	50.955	0.662	0.667	0.65
0.5	0.1	49.842	50.0	50.955	0.66	0.665	0.647
0.5	0.5	49.842	50.0	50.955	0.643	0.65	0.629
2	0.001	49.842	50.0	50.955	0.669	0.673	0.657
2	0.01	49.842	50.0	50.955	0.694	0.695	0.685
2	0.1	49.842	50.0	50.955	0.689	0.691	0.68
2	0.5	49.842	50.0	50.955	0.68	0.682	0.669



For Hod feature Subraction

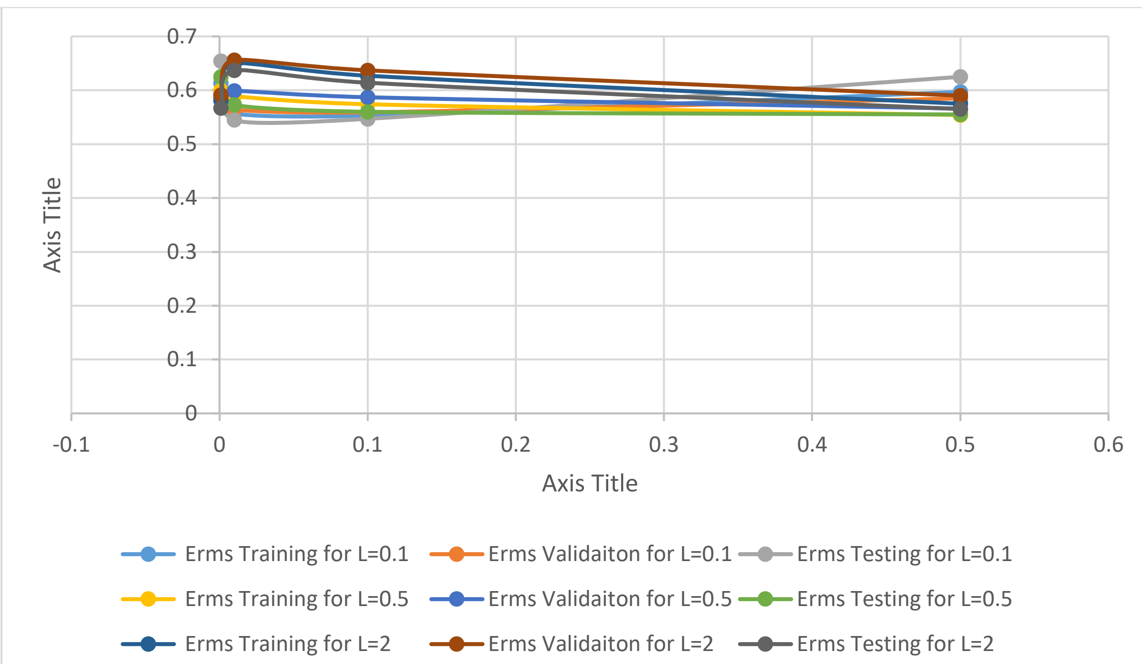
For M=5, the following values are observed from this model:

C_Lambda	Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy	Emrs Training	Erms Validaiton	Erms Testing
0.1	0.001	51.975	49.367	52.866	0.572	0.573	0.589
0.1	0.01	50.158	53.165	52.866	0.565	0.574	0.548
0.1	0.1	51.264	54.43	52.229	0.559	0.57	0.549
0.1	0.5	52.212	53.797	56.688	0.556	0.564	0.559
0.5	0.001	51.659	56.329	52.866	0.562	0.562	0.568
0.5	0.01	49.842	50.0	50.955	0.613	0.623	0.597
0.5	0.1	49.842	50.0	50.955	0.603	0.615	0.588
0.5	0.5	49.763	50.633	48.408	0.573	0.588	0.562
2	0.001	49.842	50.0	50.955	0.612	0.624	0.594
2	0.01	49.842	50.0	50.955	0.67	0.674	0.658
2	0.1	49.842	50.0	50.955	0.656	0.662	0.644
2	0.5	49.842	50.0	50.955	0.619	0.629	0.606



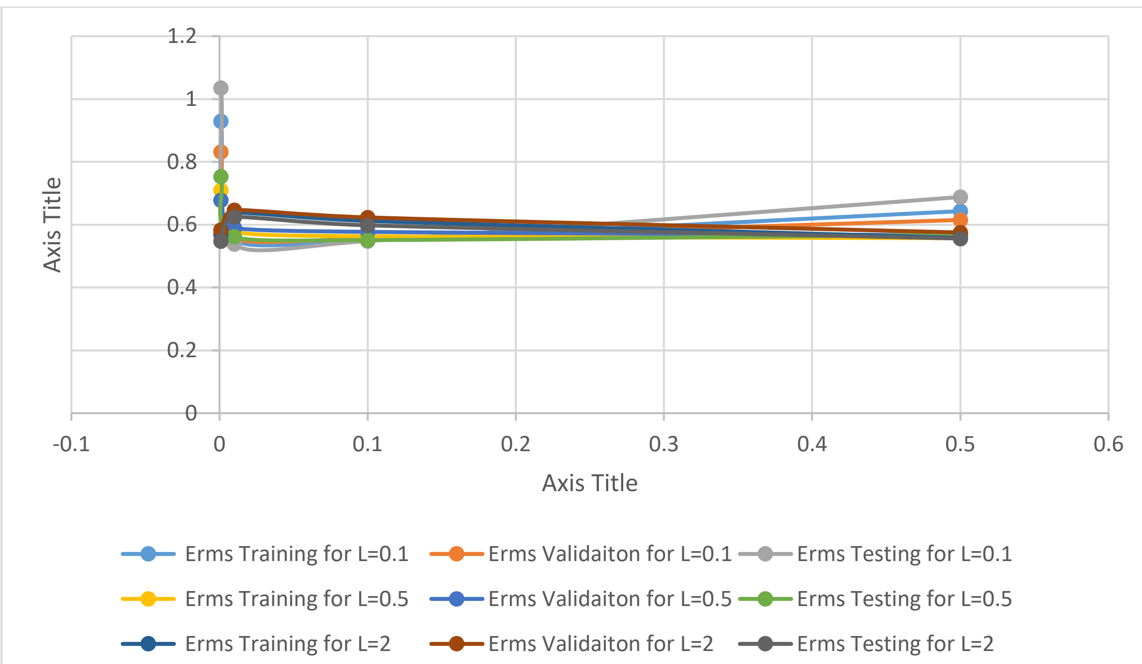
For M=10, the following values are observed from this model:

C_Lambda	Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy	Emrs Training	Erms Validaiton	Erms Testing
0.1	0.001	51.185	50.0	48.408	0.613	0.592	0.654
0.1	0.01	51.185	51.899	54.14	0.557	0.564	0.545
0.1	0.1	51.501	53.797	54.14	0.553	0.559	0.547
0.1	0.5	51.58	49.367	52.229	0.597	0.585	0.625
0.5	0.001	51.106	51.266	53.503	0.6	0.587	0.625
0.5	0.01	49.684	50.633	49.682	0.589	0.599	0.573
0.5	0.1	50.079	51.266	49.045	0.574	0.587	0.56
0.5	0.5	52.607	53.165	55.414	0.554	0.566	0.555
2	0.001	49.289	50.633	47.134	0.581	0.591	0.567
2	0.01	49.842	50.0	50.955	0.65	0.656	0.637
2	0.1	49.842	50.0	50.955	0.627	0.637	0.614
2	0.5	49.368	50.633	47.771	0.575	0.59	0.565



For M=15, the following values are observed from this model:

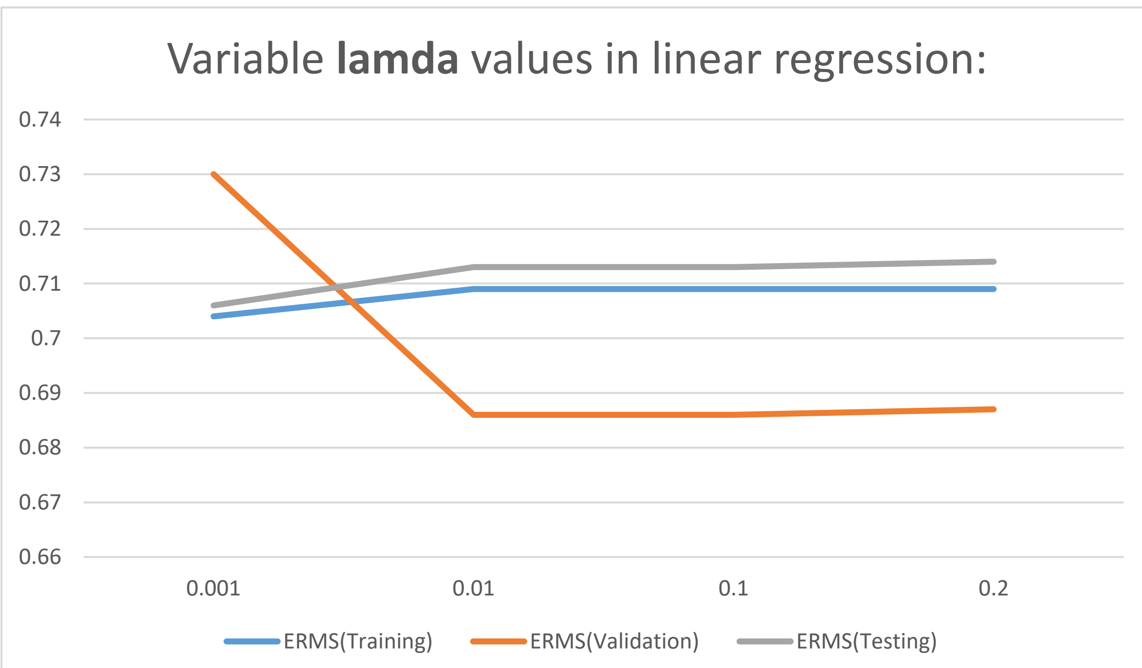
C_Lambda	Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy	Emrs Training	Erms Validaiton	Erms Testing
0.1	0.001	38.547	39.873	39.49	0.929	0.832	1.035
0.1	0.01	51.106	53.165	54.777	0.552	0.558	0.539
0.1	0.1	52.291	52.532	54.14	0.551	0.556	0.548
0.1	0.5	50.0	44.304	50.955	0.643	0.615	0.688
0.5	0.001	47.314	44.937	49.045	0.71	0.678	0.753
0.5	0.01	49.842	50.0	48.408	0.578	0.59	0.562
0.5	0.1	49.526	50.633	49.682	0.563	0.577	0.551
0.5	0.5	53.16	52.532	55.414	0.557	0.566	0.568
2	0.001	49.763	49.367	50.318	0.569	0.583	0.548
2	0.01	49.842	50.0	50.955	0.638	0.646	0.624
2	0.1	49.842	50.0	50.955	0.612	0.623	0.598
2	0.5	51.106	52.532	51.592	0.558	0.575	0.556



For GSC Concatenation Dataset Experimental values are :

For M=10, the following values are observed from this model:

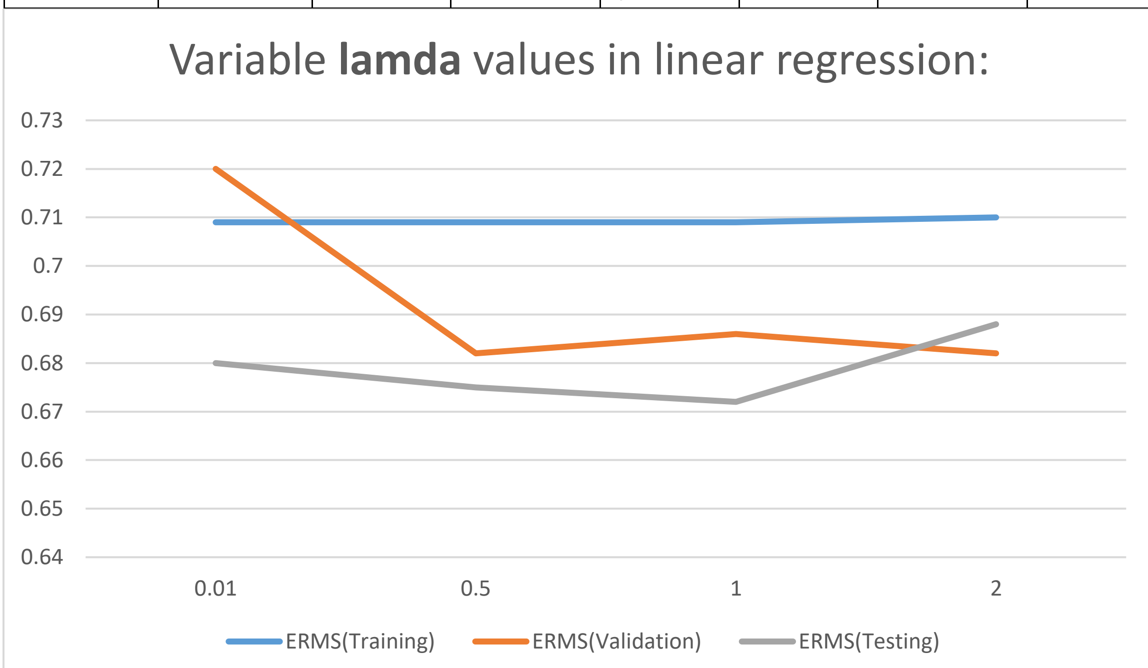
C_Lambda	Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy	Emrs Training	Erms Validaiton	Erms Testing
0.001	0.0001	49.75	52.953	49.149	0.709	0.686	0.713
0.001	0.001	50.4	46.647	50.15	0.704	0.73	0.706
0.01	0.001	49.75	52.953	49.149	0.709	0.686	0.713
0.1	0.001	49.75	52.953	49.149	0.709	0.686	0.713
0.2	0.001	49.74	52.95	49.15	0.709	0.687	0.714



For GSC Subraction Dataset Experimental values are :

For M=10, the following values are observed from this model:

C_Lambda	Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy	Emrs Training	Erms Validaiton	Erms Testing
0.001	0.0001	49.625	52.953	50.15	0.71	0.686	0.706
0.001	0.001	49.763	48.148	53.754	0.709	0.72	0.68
0.01	0.001	49.75	52.953	49.149	0.709	0.682	0.675
0.1	0.001	49.75	52.953	49.149	0.709	0.686	0.672
0.1	0.001	49.75	52.953	49.149	0.710	0.682	0.688



Observations :

Effect of Lambda :

- From above observations we noticed that for lower lambda values the error is huge. Since we are intaializing random weights in SGD this lambda factor plays an important role which adds random number of weights. So for higher rate the error is less which is recommended.

Effect of Learning rate:

- From above observations we noticed that , If the learning rate value is too high, your model will be simple, but you run the risk of under fitting your data. Your model won't learn enough about the training data to make useful predictions.

- If the learning rate value is too low, your model will be more complex, and you run the risk of overfitting your data. Your model will learn too much about the particularities of the training data, and won't be able to generalize to new data.

Effect of the number of Basis Functions:

- From above observations we noticed that, increase in the basis function increased the emrs for lower lambda values. So it is strictly recommended to keep lamda values high while increasing M.

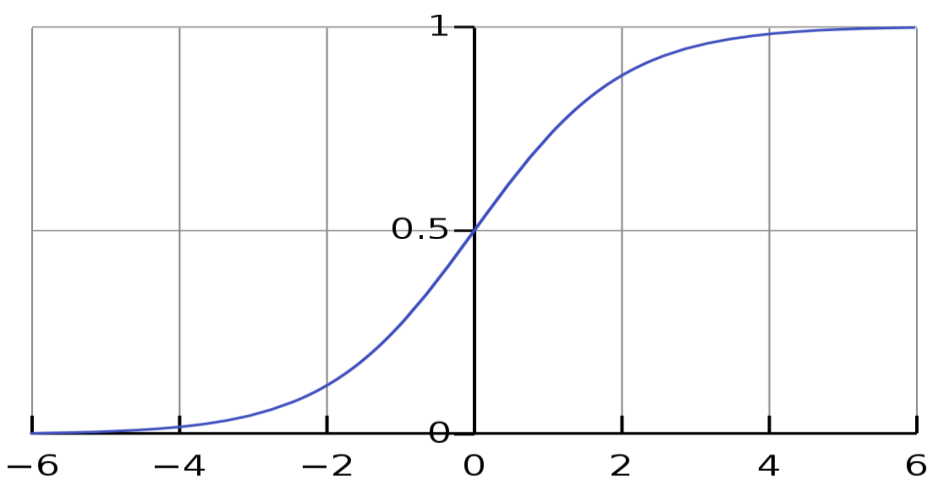
We observed that the Erms is minimum for the lambda = 0.1 and Learning rate = 0.001 for Human observed dataset Concatention Mode. Although the computation time is huge for Gsc datasets we got almost the same accuracy as the Human observed dataset.

Training Logistic Regression Model :

we train a linear regression model onLeToR dataset using the following methods :

$$y(x, w) = \boldsymbol{\sigma}(\omega^T \phi(x)) = 1/(1 + e\hat{}(-\omega^T \phi(x)))$$

A function takes inputs and returns outputs. To generate probabilities, logistic regression uses a function that gives outputs between 0 and 1 for all values of X. This is called the the *logistic function*.



Cost function : cost function we used is 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

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathrm{Cost}(h_\theta(x^{(i)}), y^{(i)})$$

$$\mathrm{Cost}(h_\theta(x), y) = -\log(h_\theta(x)) \quad \text{if } y = 1$$

$$\mathrm{Cost}(h_\theta(x), y) = -\log(1 - h_\theta(x)) \quad \text{if } y = 0$$

The combine equation is

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} [y^{(i)} \log(h_\theta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))]$$

Logistic Regression using gradient descent (SGD) :

- It is the process of mininmizing a function by following the gradients of the cost function.
- This involves knowing the form of the cost as well as the derivative so that from a given point you know the gradient and can move in that direction, e.g. downhill towards the minimum value.
- In Machine learning we can use a similar technique called stochastic gradient descent to minimize the error of a model on our training data.
- The gradient descent algorithm takes a random initial value. Then it updates the value of w using
  $W^{(\tau+1)} = W^{(\tau)} + \Delta W^{(\tau)}$

where $\Delta w(\tau) = -\eta^{(\tau)} * X * (y - y^{l})$ is called the weight updates. It goes along the opposite direction of the gradient of the error. $\eta(\tau)$ is the learning rate, deciding how big each update step would be. Because of the linearity of differentiation and y is the actual output where as y^{l} is the predicted output.

Experimental Values :

Hyperparameters Used :

The following are the hyper parameters used in this model :

- M - The number of basis functions used in the model.
- C_Lambda – This is the regularization factor which helps to reduce the overfitting of the model by adding some random weights to the desired weights.

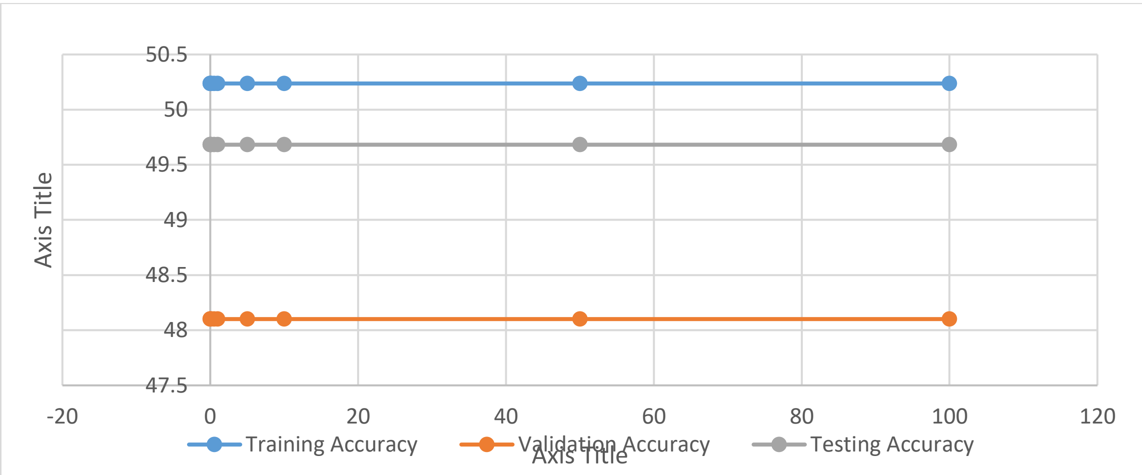
Varying Number of Basis Functions and Regularization, Eta :

These are the number of basis funcitons used in the model are kept as 10 for different Lambda Values and eta values:

Hod Feature Concatination

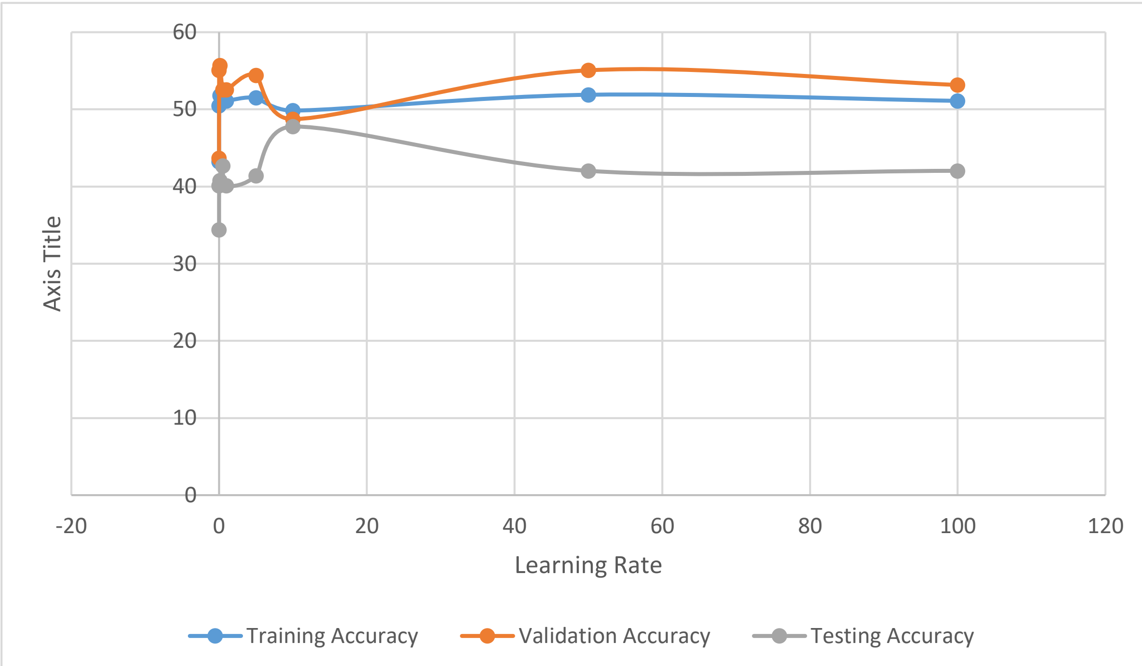
Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy
0.001	50.237	48.101	49.682
0.01	50.237	48.101	49.682
0.1	50.237	48.101	49.682

0.5	50.237	48.101	49.682
1	50.237	48.101	49.682
5	50.237	48.101	49.682
10	50.237	48.101	49.682
50	50.237	48.101	49.682
100	50.237	48.101	49.682



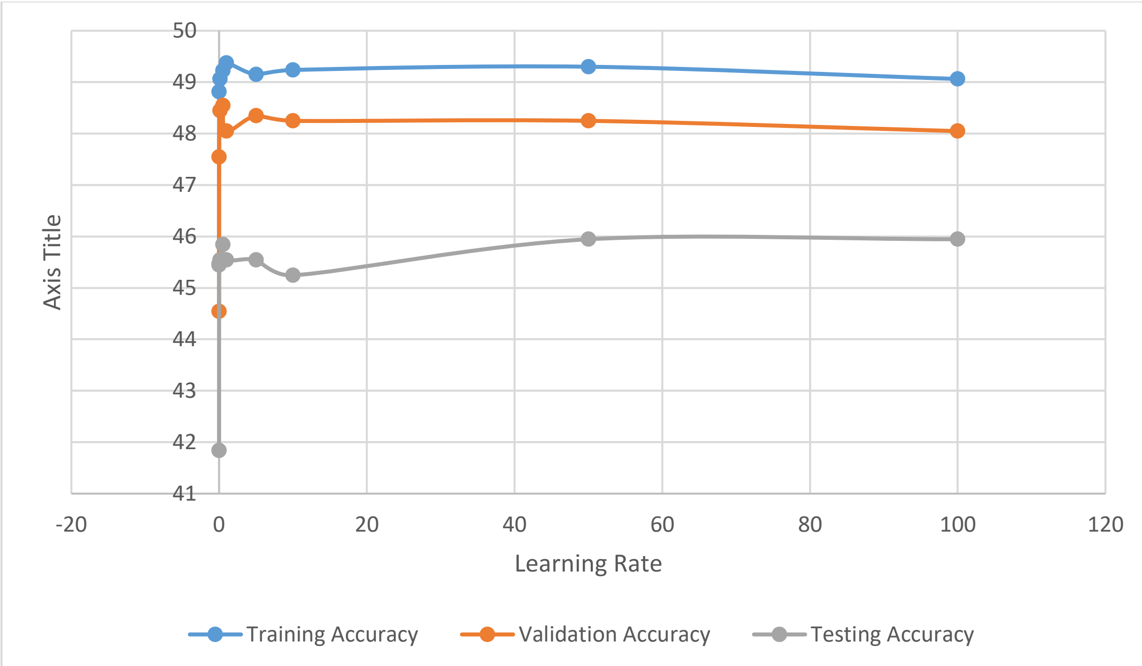
For Hod Feature Subraction

Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy
0.001	43.207	43.671	34.395
0.01	50.474	55.063	40.127
0.1	51.817	55.696	40.764
0.5	51.185	52.532	42.675
1	51.106	52.532	40.127
5	51.501	54.43	41.401
10	49.842	48.734	47.771
50	51.896	55.063	42.038
100	51.106	53.165	42.038



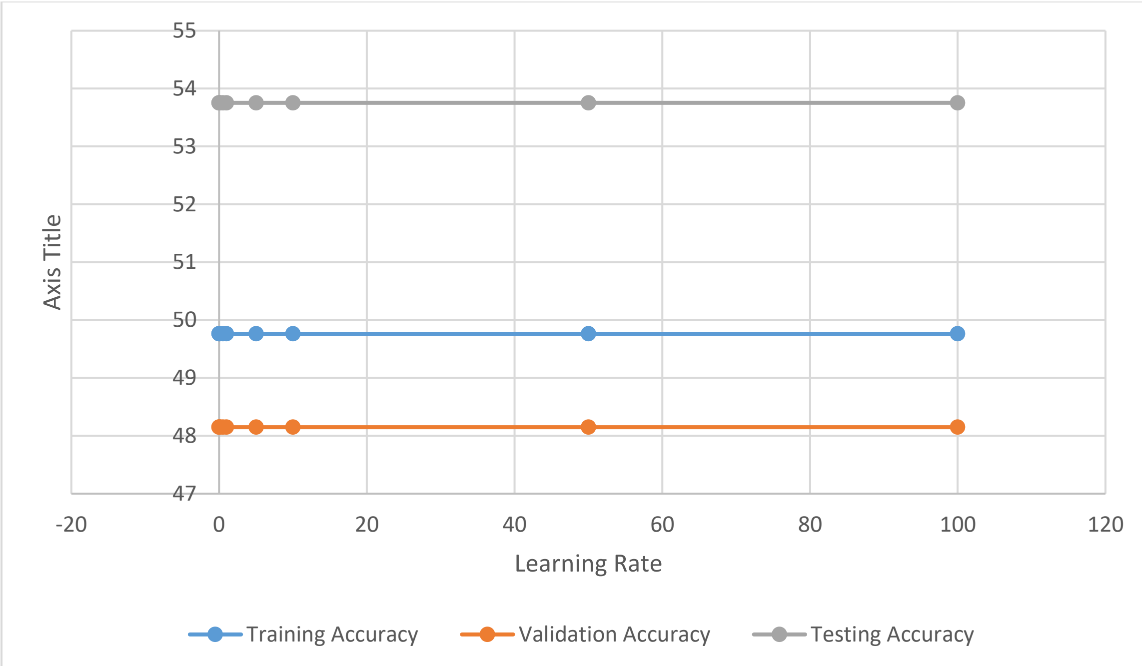
Gsc Feature Subraction

Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy
0.001	45.462	44.545	41.842
0.01	48.812	47.548	45.445
0.1	49.062	48.448	45.546
0.5	49.225	48.549	45.846
1	49.375	48.048	45.546
5	49.15	48.348	45.546
10	49.237	48.248	45.245
50	49.3	48.248	45.946
100	49.062	48.048	45.946



Gsc Feature Concatination

Learning Rate	Training Accuracy	Validation Accuracy	Testing Accuracy
0.001	49.762	48.148	53.754
0.01	49.762	48.148	53.754
0.1	49.762	48.148	53.754
0.5	49.762	48.148	53.754
1	49.762	48.148	53.754
5	49.762	48.148	53.754
10	49.762	48.148	53.754
50	49.762	48.148	53.754
100	49.762	48.148	53.754



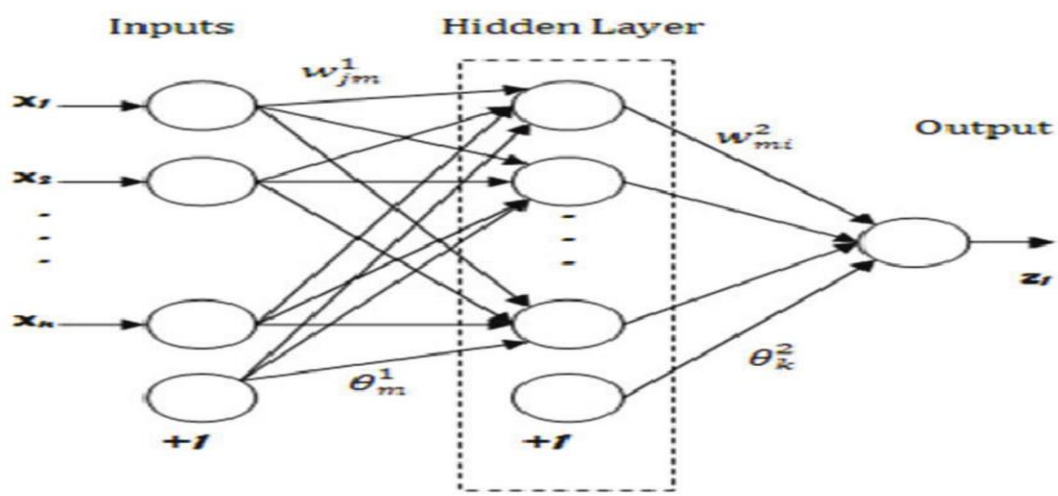
Observations :

We observed that the Erms is minimum for the Learning rate = 0.001 for Human observed dataset Concatention Mode. Although the computation time is huge for Gsc datasets we got almost the same accuracy as the Human observed dataset.

Neural Networks Model :

The learning system

Making a model helps to explain a system and to study the effects of different components, and to make predictions about behavior. The model is created in keras which uses high level API built on Tensor Flow. It is more user-friendly and easy to use as compared to Tensor flow**.** Using Sequential model which is just a linear stack of layers. The Artificial Neural Network model will have inputs which are features, one dense hidden layer consists of several nodes, activation function and 2nd layer as outputs as shown in below figure.



The activation function introduce non linear properties to the network and converts input signal of a node to an output signal which can be used as a input to the next dense layer. Later To prevent Neural network from overfitting we are adding dropouts to the hidden layer which drops random weights which are causing overfitting to the model. To reduce the influence of extreme values in the data without removing them from the data set we are adding softmax activation function. Later Categorical crossentropy Loss function is used to calculate the amount of inaccuracy.

Experimental Values :

Hyper parameters Used:

The following are the hyper parameters used in this model.

- Number of Neuron/Nodes : These are the number of nodes/neurons in the first dense layer.
- Droupouts : This is the percentage of dropout rate Activation function
- Optimiser : This the function name of the optimizer used
- Number of hidden layers : These are number of hidden layers in the overall neural network used.
- Input batch size : This the number of data samples sent into model per single batch
- Number of epochs :This is the number of iterations the model should run for the given data set.

- Validation data split – This is the percentage of the data samples used for validating the model in each epoch.

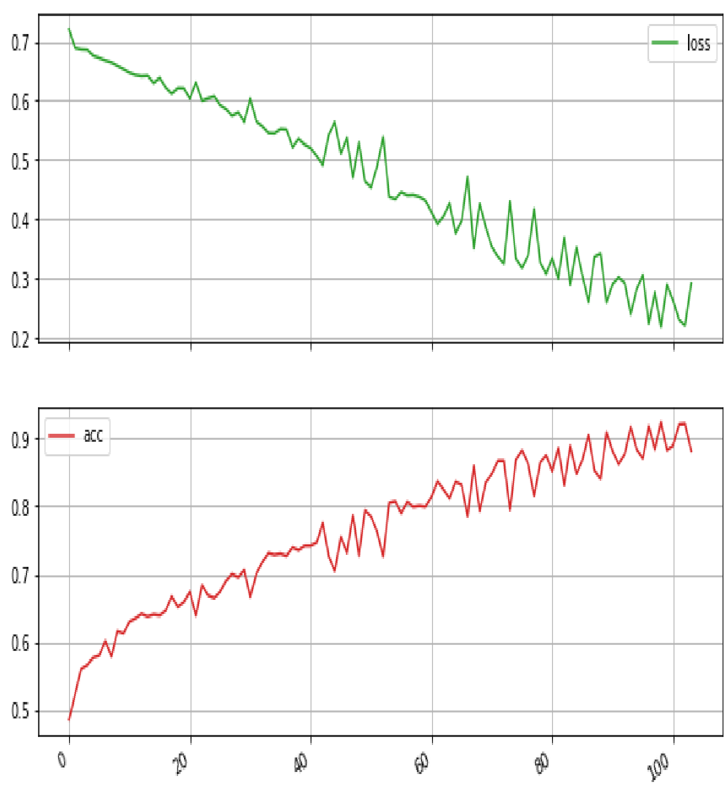
For Hod Subract the model values are :

Based on the experimental results of varying hyper parameters we have chosen the following hyp er parameters.

Number of Neuron	2048+1024
Droupouts	0.2
Activation function	Relu +sigmoid
Optimiser	Adadelta
Number of hidden layers	2
Input batch size	128
Number of epochs	10000
Validation data split –	0.15
Early Stopping	100

The accuracy measures are best which are shown in the below table and the graph with respect to e pochs are shown .

Accuracy Measures	Value
Training Loss	0.028
Training Accuracy	0.84
Validation Accuracy	0.71
Testing Accuracy	0.70



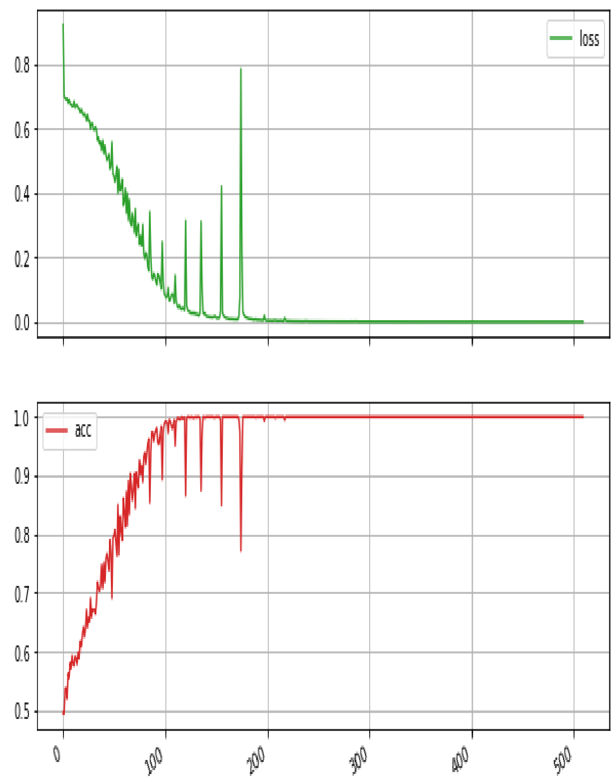
For Hod concat the model values are :

Based on the experimental results of varying hyper parameters we have chosen the following hyp er parameters.

Number of Neuron	2048+1024
Droupouts	0.2
Activation function	Relu +sigmoid
Optimiser	Adadelta
Number of hidden layers	2
Input batch size	128
Number of epochs	10000
Validation data split –	0.15
Early Stopping	500

The accuracy measures are best which are shown in the below table and the graph with respect to e pochs are shown .

Accuracy Measures	Value
Training Loss	0.0001
Training Accuracy	0.9900
Validation Accuracy	0.8400
Testing Accuracy	0.8200

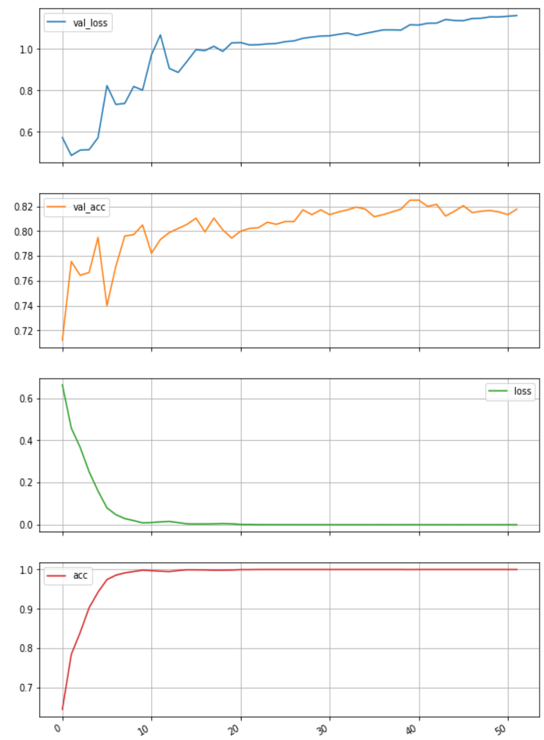


For Gsc subract the model values are :

Based on the experimental results of varying hyper parameters we have chosen the following hyp er parameters.

Number of Neuron	2048+1024
Droupouts	0.2
Activation function	Relu +sigmoid
Optimiser	Adadelta
Number of hidden layers	2
Input batch size	128
Number of epochs	10000
Validation data split –	0.15
Early Stopping	50

The accuracy measures are best which are shown in the below table and the graph with respect to epochs are sh own .



Accuracy Measures	Value
Training Loss	0.0001
Training Accuracy	0.1000
Validation Accuracy	0.8200
Testing Accuracy	0.81

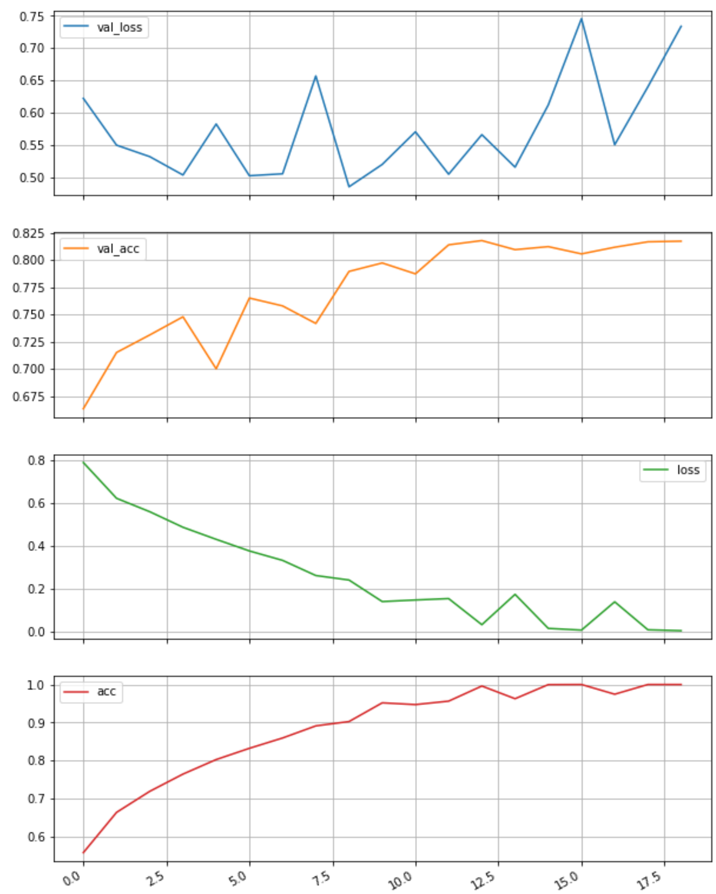
For Gscconcat the model values are :

Based on the experimental results of varying hyper parameters we have chosen the following hyp er parameters.

Number of Neuron	2048+1024
Droupouts	0.2
Activation function	Relu +sigmoid
Optimiser	Adadelta
Number of hidden layers	2
Input batch size	128
Number of epochs	10000
Validation data split –	0.15
Early Stopping	500

The accuracy measures are best which are shown in th e below table and the graph with respect to epochs are shown .

Accuracy Measures	Value
Training Loss	0.0001
Training Accuracy	0.1000
Validation Accuracy	0.860
Testing Accuracy	0.8500



Conclusion and Summary :

We observed that there isn't significant difference in the accuracies for any change in hyper parameters. The given dataset has the highest accuracies are obtained is 100 for training and testing accuracy is 90 for Neural Network models.