Introduction

In this project we find similarity between the handwritten samples of the known and the questioned writer by using linear regression.

Each instance in the training data consists of set of input features for each hand-written "AND" sample. The features are obtained from two different sources: Human Observed features, GSC features.The target values are scalars that can take two values {1:same writer, 0:different writers}.

For each of these data sets we curate the training data by two methods:

1. Concatenating the features of two images being compared.
2. Subtracting the feature values of the two images being compared.

Description of the Linear Regression Model Used:

Our regression model looks is as follows:

$y(x, w) = w^T\phi(x)$,
Where $\phi(x)$ is the basis function, in our case we have used the radial basis function.

Why use radial basis function?

Since, our data is complex and the features hold a non-linear relation with the target variable, we need a model that is non-linear in **x,** so that we can fit a non-linear curve to our data, which will in turn reduce the cost function(error) and hence, potentially increase the accuracy. However, we still call it a linear model because its linear in **w.**

Gaussian radial basis function:

The Gaussian Radial basis function is defined as: $\phi_j(x) = \exp(-1/2 (x - \mu_j)^T \Sigma^{-1}{}_j (x - \mu_j))$

Here, μ_j is the mean of the cluster j, as we first divide our training data set into k clusters, using k-means. The value of this k is a hyper parameter, which we will tune during this

project.

The gaussian radial basis function actually computes the similarity of each data point with each of the cluster centroids (mean of each cluster). This similarity measure is basically given by the euclidean distance between the cluster centroid and the data point. Since Euclidean distance is in the power of 2, this gives us a higher degree (degree =2) non-linear curve to fit our data and hence, enables us to model a complex data set.

Hence, we use the **design matrix** $\phi_j(x)$, instead of simple feature vectors, in our model.

However, our model is linear with respect to **w,** the weights.

Implementational Objective for our model:

While fitting the curve to our data, our main objective is to minimize the cost which is defined by the total error made by our model. This total error or cost of our model is defined by Root-Mean-Square Error (RMS). To prevent overfitting, we add the regularization term as well and hence our overall cost function becomes:

$E(w) = E_D(w) + \lambda E_W(w)$
Where, λ is our regularization parameter which we will be tuning for optimal value.

We train our models using Gradient Descent as follows:

Starting with the random weights, $w^{(0)}$, we iteratively update the weights as :

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} + \Delta \mathbf{w}^{(\tau)}$$

where $\mathbf{w}^{(\tau+1)}$ is the weight matrix for $\tau+1$ th iteration of the update rule.

$$\Delta \mathbf{w}(\tau) = -\eta\ (\tau)\nabla E.$$

Here η is the learning rate, which is another hyperparameter that decides how fast the model learns or how aggressively we update the weights in each iteration.

After repeating the update rule for sufficient number of times, we get the most optimal values for weights, which can then be used to make predictions on unseen data.

$\nabla E = \nabla E_D + \lambda \nabla E_{W,}$
where $\nabla E_D = -(t_n - w^{(\tau)T}\, \phi(x_n))\, \phi(x_n)$, which is the differential for cost function without regularization term.
$\nabla E_W = w^{(\tau)}$ is the differential for regularization term.

Description of the Logistic Regression Model Used:

For the logistic regression we use the ‘sigmoid’ function to output the probabilities that indicate whether or not the sample belongs to a class, that is, classification. In our case, value of 1 represents a match, and a value 0 implies handwritings are different. The sigmoid function takes an input which is the dot product of design matrix and the weights, similar to what we had in logistic regression.

Then the model is trained using the gradient descent algorithm.

Performance:

After tuning various hyper parameters like learning rate, number of iterations, regularization parameter, etc. we found the following performance metrics for the most optimal hyper parameters:

For Human Observed Data Set:

Metrics For Best Performing Linear Regression Model On Feature Concatenated Data Set:

```
Train Accuracy: 0.6750524109014675
CV Accuracy: 0.6383647798742138
Test Accuracy: 0.7053291536050157
E_rms TRAINING: 0.49051
E_rms VALIDATION: 0.49174
E_rms TEST: 0.48871
```

Metrics For Best Performing Linear Regression Model On Feature Subtracted Data Set:

```
Train Accuracy: 0.8427672955974843
CV Accuracy: 0.8113207547169812
Test Accuracy: 0.8369905956112853
E_rms TRAINING: 0.46094
E_rms VALIDATION: 0.46586
E_rms TEST: 0.46029
```

Metrics For Best Performing Logistic Regression Model On Feature Concatenated Data Set:

```
Logistic Regression Loss On Training Data:
0.36788413820929255
ERMS ERMS Loss On Training Data: 0.33806033157156323
Accuracy On Training Data: 0.8542976939203354
ERMS Loss On CV Data: 0.33400884972241557
Accuracy On CV Data: 0.8710691823899371
ERMS Loss On Test Data: 0.31899720154649436
Accuracy On Test Data: 0.890282131661442
```

Metrics For Best Performing Logistic Regression Model On Feature Subtracted Data Set:

```
Logistic Regression Loss On Training Data: nan
ERMS Loss On Training Data: 0.2166171722663769
Accuracy On Training Data: 0.9433962264150944
ERMS Loss On CV Data: 0.22561939879489268
Accuracy On CV Data: 0.9433962264150944
ERMS Loss On Test Data: 0.1712133548725683
Accuracy On Test Data: 0.9686520376175548
```

For GSC Data Set:

Metrics For Best Performing Linear Regression Model On Feature Concatenated Data Set:

```
Train Accuracy: 0.5192294293875844
CV Accuracy: 0.5176269769386191
Test Accuracy: 0.5193855157278713
E_rms TRAINING: 0.55057
E_rms VALIDATION: 0.55195
E_rms TEST: 0.55079
```

Metrics For Best Performing Linear Regression Model On Feature Subtracted Data Set:

```
Train Accuracy: 0.5545646670846978
CV Accuracy: 0.5523583919738034
Test Accuracy: 0.556588985264918
E_rms TRAINING: 0.51973
E_rms VALIDATION: 0.52044
E_rms TEST: 0.51994
```

Metrics For Best Performing Logistic Regression Model On Feature Concatenated Data Set:

```
Logistic Regression Loss On Training Data:
0.35762210364924907
ERMS Loss On Training Data: 0.3275012269540261
Accuracy On Training Data: 0.8722682830534848
ERMS Loss On CV Data: 0.3305439936666008
Accuracy On CV Data: 0.8683898836480178
ERMS Loss On Test Data: 0.3290761184026051
Accuracy On Test Data: 0.8703800466785104
```

Metrics For Best Performing Logistic Regression Model On Feature Subtracted Data Set:

```
Logistic Regression Loss On Training Data:
0.35709949920804535
ERMS Loss On Training Data: 0.32449612369421893
Accuracy On Training Data: 0.8888734062565318
ERMS Loss On CV Data: 0.32802699333681967
Accuracy On CV Data: 0.8861213683550477
ERMS Loss On Test Data: 0.3262736664936357
Accuracy On Test Data: 0.8868220294701641
```

Conclusion:

Hence, from the above mentioned statistics, Logistic Regression Model On Feature Subtracted Data Set gives the best accuracy for Human Observed data and Logistic Regression Model On Feature Subtracted Data Set gives the best accuracy for GSC data.