**Fischer’s Linear Discriminant Analysis(FLDA)**

**FLDM1’s and FLDM2’s accuracies are approximately 58.70%. Both accuracies are not too far apart from each other as per the Machine Learning Analysis of given data because just by changing the order of features (permutations) in the given feature vector, will not change the accuracy of the predicted data by a large margin with respect to both models. From this, we can conclude that there is no meaningful difference in both the models in terms of accuracies too.**

**Logistic Regression**

**y = wx + w0**

**putting this y in sigmoid function to calculate probability.**

**p = sigmoid(-y)**

**Implementation**

Training

1. Initialise the weights and the bias
2. Predict/estimate using the sigmoid function
3. Calculate the error
4. Use the gradient decent to figure out the global minima
5. Repeat n times.

Testing

1. Put the values from the testing split into the LR
2. Classify and check prediction with the given classification and estimate accuracy.

Logistic Regression got the highest probability in our case as it is best suited to the given data.

We found out that the highest accuracy and consistency was achieved in case of feature engineering 2. While the lowest was achieved in case of unnormalized data. Also the consistency of the algorithms was questionable in case of feature engineering 1 and unnormalized data.

We also found out that Batch\_Gradient\_Descent algorithm was far more consistent in its result with both feature engineering and unnormalized data while stochastic\_Gradient\_Descent algorithm was quite erratic even in normalized data. Although the Max accuracy was that of stochastic\_Gradient\_Descent (Peak value not the average) while batch had the lowest peak value.

The Order of the Algorithms are Batch, Stochastic and Mini\_Batch

For 0.5 as the decision probability threshold

0.9148936170212766

0.9521276595744681

0.9042553191489362

0.9148936170212766

0.9521276595744681

0.9042553191489362

0.9148936170212766

0.9468085106382979

0.9095744680851063

For 0.3 as the decision probability threshold

0.7127659574468085

0.9095744680851063

0.4787234042553192

0.7127659574468085

0.9042553191489362

0.4787234042553192

0.7127659574468085

0.898936170212766

0.4787234042553192

For 0.4 as the decision probability threshold

0.824468085106383

0.9361702127659575

0.7819148936170213

0.824468085106383

0.9361702127659575

0.776595744680851

0.824468085106383

0.9414893617021277

0.7819148936170213

For 0.6 as the decision probability threshold

0.9574468085106383

0.9840425531914894

0.9574468085106383

0.9574468085106383

0.9680851063829787

0.9574468085106383

0.9574468085106383

0.973404255319149

0.9574468085106383

For 0.7 as the decision probability threshold

0.9627659574468085

0.9840425531914894

0.9627659574468085

0.9627659574468085

0.9840425531914894

0.9627659574468085

0.9627659574468085

0.9840425531914894

0.9574468085106383

We can see that there was a massive drop in accuracy and consistency when we switched to 0.3 as the decision probability threshold while 0.7 gave the max accuracy and consistency. We can also see that peak accuracy of all the 3 implementations increased with maximum peak from stochastic and batch and mini batch at almost the same level. We can also see that 0.6 and 0.7 gave almost the same result. As can be seen from the graph that 0.6 and above give almost the same result above with the accuracy drops due to overfitting/underfitting.

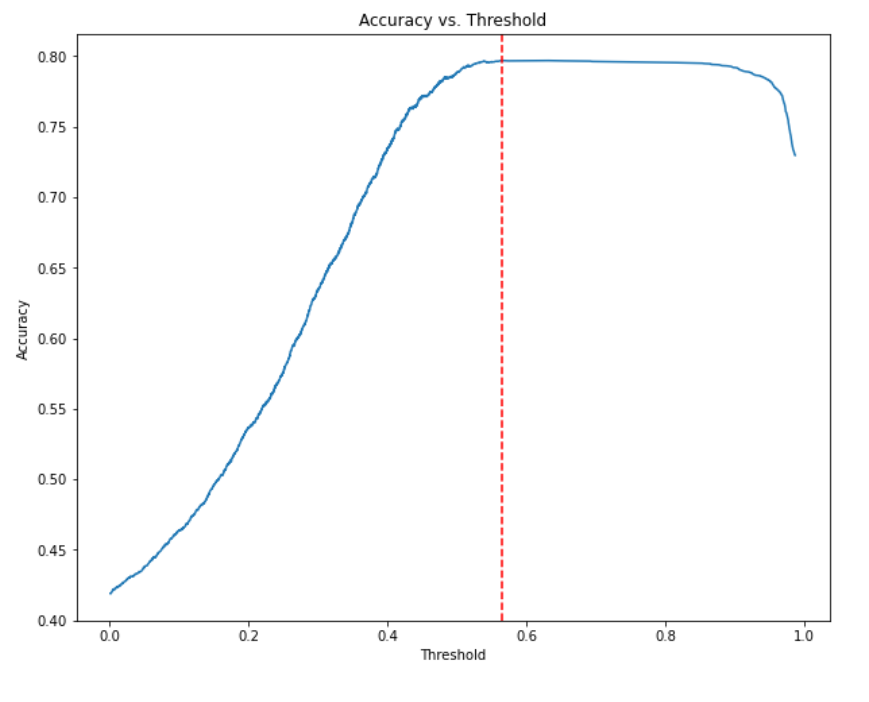
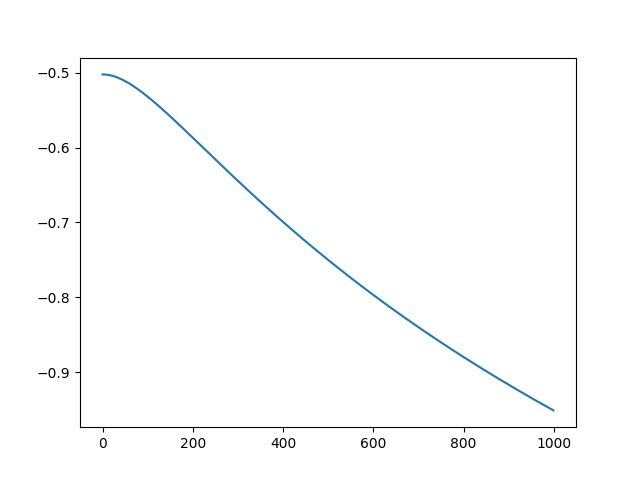
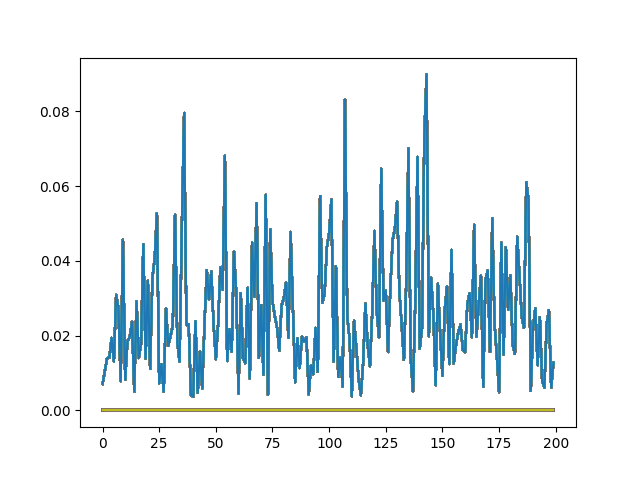


Image Source: towardsdatascience.com

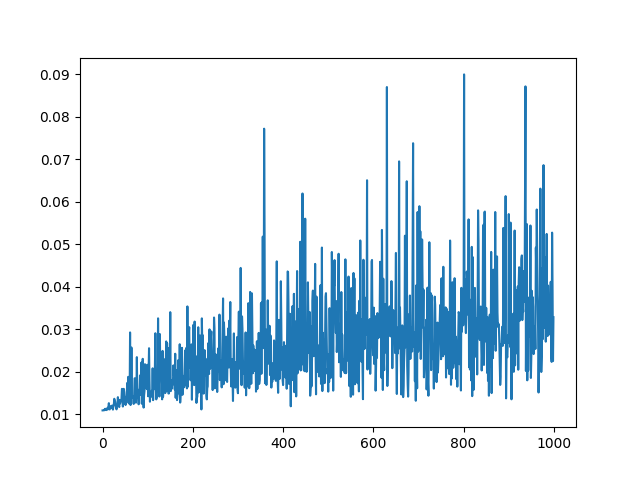
**Cost Vs Epoch function for Logistic Regression Batch implementation**



**Cost Vs Epoch function for Logistic Regression Batch implementation**

****

**Cost Vs Epoch function for Logistic Regression Batch implementation**

****

**Perceptron**

wt+1=wt+ yi\*xi

**Learning Task 1:**

Data is approximately 82% separable for 1000 epochs (accuracy =82%)

Between PM1 and PM2. PM2’s accuracy varies above and below that of PM1’s with the average accuracy being approximately equal to PM1’s accuracy. This shows that some set of tuples from the data are better at separating the data than others.

**Learning Task 2:**

The accuracy of PM3 is approximately 93% which is significantly higher than that of PM1. This is because normalization would improve the accuracy of the model as it balances the impact of each feature.

**Learning Task 3:**

There is no significant difference in the accuracy of the models PM4 and PM1 as the effective dataset is the same, only a different permutation of the weight vector elements will be given as the output.

**Comparative Analysis**

After training and testing on 10 different samples of the same data(population) using the models mentioned in the Assignment, we found that accuracy of Logistic Regression Model is highest among all. Hence, we can conclude that it is the best performing model. We can say this because it is less prone to overfitting, is robust to noise, i.e., it is less sensitive to outliers in the data unlike other models like Perceptron and Fischer’s Linear Discriminant Analysis. Also, at the same time, it can be trained efficiently even on a large dataset like ours.