PRML MINI PROJECT: EVALUATION II

BioCreative III Protein-Protein Interaction (PPI)

Devendra Kumar - 13EC115
Karishma Mulani - 13EC124
Kevin D Souza - 13EC256
S Giritheja - 13EE141
(GROUP 10)

SUMMARY OF THE DATASET:

The BioCreative III Protein-Protein Interaction (PPI) project included detecting articles describing complex biological events like PPIs was addressed in the Article Classification Task (ACT), where participants were asked to implement tools for detecting PPI-describing abstracts. Therefore the BCIII-ACT corpus was provided, which includes a training, development and test set of over 12,000 PPI relevant and non-relevant PubMed abstracts labelled manually by domain experts and recording also the human classification times. Text mining was done on the data and the first 1406 words with the highest frequency were extracted and the frequency of these words for every data point was the feature set. Training data had 4000 points, validation data had 2280 and test data had 2000 points. There are two classes which are binary (0 or 1).

WHY SUPPORT VECTOR CLASSIFIER AND NOT NEURAL NETWORKS OR LOGISTIC REGRESSION:

Due to the large number or features and the limited amount of data we decided to go ahead with SVM as it is a convex function and goes to the global maxima. So a more accurate result is obtained using SVM without a greater computational expense. Also as the number of data points is comparable to the number of features, an SVM with a Gaussian kernel is expected to give a better performance.

EVALUATION OF PERFORMANCE BY VARYING REGULARISATION FOR A SUPPORT VECTOR CLASSIFIER WITH LINEAR KERNEL:

A few variations are shown in the table below for the validation data:

C value	True Positive	False Positive	True Negative	False Negative	Precision	Recall	F-Measure	Accuracy
0.002	0	0	1140	1140	NaN	0	NaN	50%
0.02	0	0	1140	1140	NaN	0	NaN	50%
0.2	0	0	1140	1140	NaN	0	NaN	50%
0.5	2	2	1138	1138	0.5	0.001754	0.0034965	50%
1	48	44	1096	1092	0.52174	0.042105	0.077922	50.1754%
15	220	212	928	920	0.50926	0.19298	0.27990	50.3509%
30	254	243	897	886	0.51107	0.31032	0.31032	50.4825%
35	248	244	896	892	0.50407	0.21754	0.30392	50.1754%
45	255	252	888	885	0.50296	0.22368	0.30965	50.1316%
60	255	258	882	885	0.49708	0.22368	0.30853	49.8684%
200	270	261	879	870	0.50847	0.23684	0.32316	50.3947%
500	271	262	878	869	0.60844	0.23772	0.32397	50.3947%

Beyond this it takes a lot of time for computation. It was observed that for really low values of C everything got classified to one class (0) as expected as the regularization is very high and our output will be insensitive to our input.

So, it goes into a scenario of underfit for less value of C. So although the accuracy improved, it didn’t really depend on the data. Hence, we took into consideration the F measure to judge the performance of the classifier. It was found out, the value of C for optimum accuracy and F measure was around 30. So it is not an overfit problem as decrease in value of C should have increased the performance if it were so. We are solving an underfit issue.

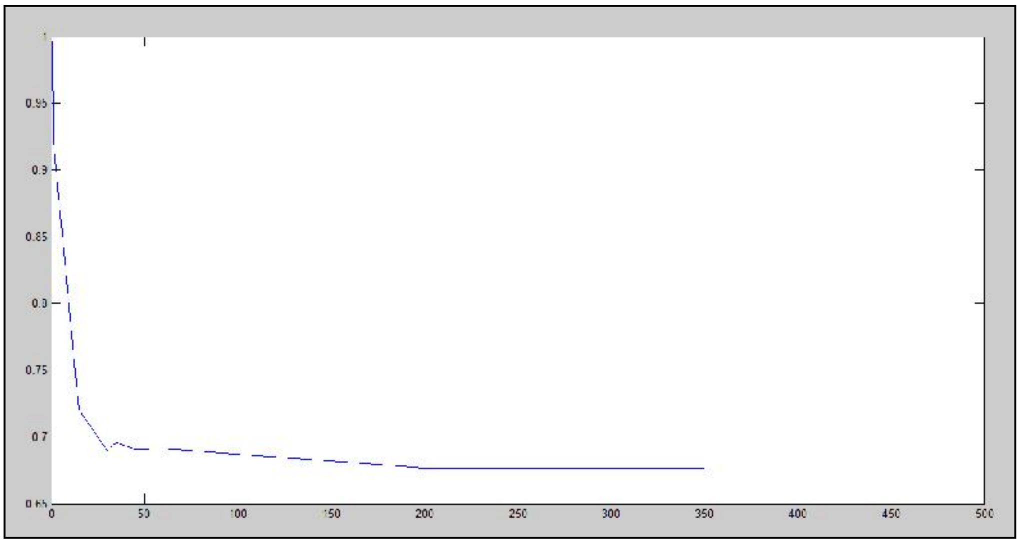

Figure 1: Plot of C v/s (1-FMeasure)

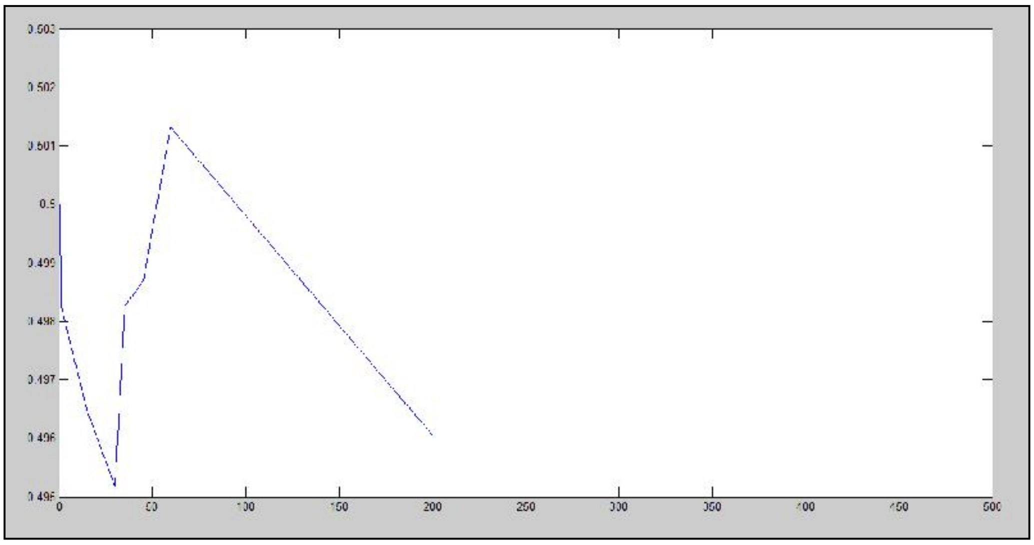

Figure 2: Plot of C v/s (1-Accuracy)

Hence, the F-Measure gives us a better understanding with the variation in C

A few variations of the accuracy with varying C on the test data are shown in the table below:

C value	True Positive	False Positive	True Negative	False Negative	Precision	Recall	F-Measure	Accuracy
0.0002	0	0	1674	326	NaN	0	NaN	83.7%
0.02	0	0	1674	326	NaN	0	NaN	83.7%
0.2	0	0	1674	326	NaN	0	NaN	83.7%
0.5	3	3	1671	323	0.5	0.0092025	0.018072	83.7%
1	10	55	1619	316	0.15385	0.030675	0.051151	81.45%
15	60	318	1356	266	0.15873	0.18405	0.17045	70.8%
30	65	355	1319	261	0.15476	0.19939	0.17426	69.2%
35	65	374	1300	261	0.14806	0.19939	0.16993	68.25%
45	64	385	1289	262	0.14254	0.19632	0.16516	67.65%
60	65	382	1292	261	0.14541	0.19939	0.16818	67.85%
200	73	405	1269	253	0.15272	0.22393	0.18159	67.1%
500	76	405	1269	250	0.15800	0.23313	0.18835	67.25%

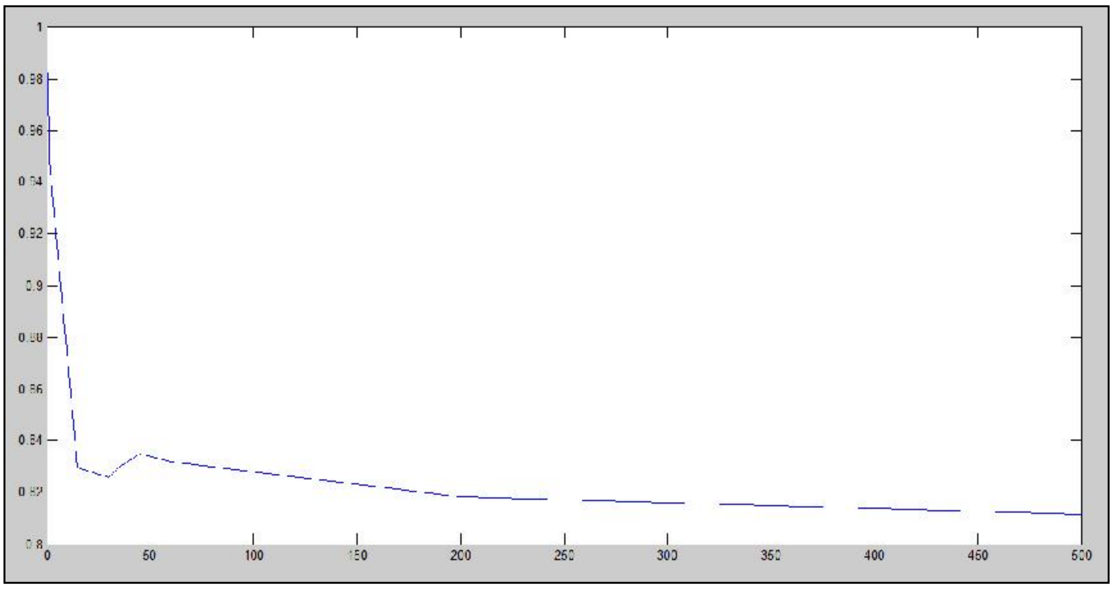

Figure 3: Plot of C v/s (1-FMeasure)

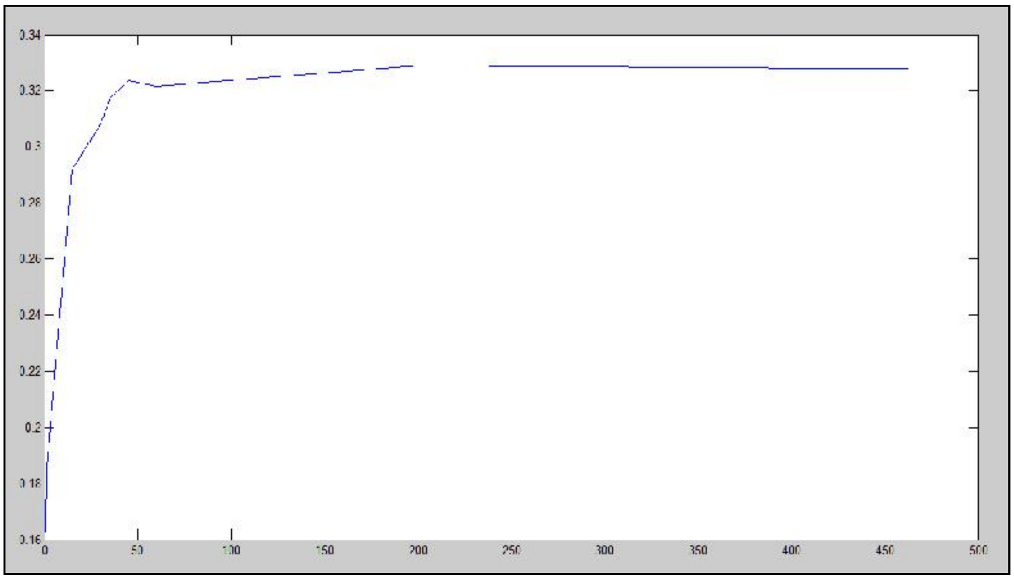

Figure 4: Plot of C v/s (1-Accuracy)

Based on the results seen in the graphs we choose the value of C which has a high F score and Accuracy, i.e., some sort of a trade-off. This turns out to be true for around C=30 as seen in the validation data as well. Beyond C=30 it is a case of overfit and below it is a case of underfit.

EFFECT OF VARYING SIGMA PARAMETER OF THE GAUSSIAN KERNEL:

As the value of sigma increased, the data was getting classified to one class (0) and hence the accuracy improved to 50%. With decreasing value of sigma, it was more sensitive to the data and not everything was classified to one class. This came at an expense with the accuracy.

A few variations with sigma for C=30 are shown in the table below. Here g=1/(2sigma)^2

Value of g	True Positive	False Positive	True Negative	False Negative	Precision	Recall	F-Measure	Accuracy
0.001	0	0	1140	1140	NaN	0	NaN	50%
0.01	6	3	1137	1134	0.66667	0.0052632	0.010444	50.1316%
0.02	56	57	1083	1084	0.049558	0.049123	0.089385	49.9561%
0.05	115	110	1030	1025	0.51111	0.10088	0.16850	50.2193%
0.0625	110	100	1040	1030	0.52381	0.096491	0.16296	50.4386%
0.08	94	95	1045	1046	0.49735	0.082456	0.14146	49.9561%
0.1	77	76	1064	1063	0.50327	0.067544	0.11910	50.0439%
0.3	7	10	1130	1133	0.41176	0.0061404	0.012100	49.8684%
0.5	1	0	1140	1139	1	8.7719e-4	0.0071528	50.0439%
1	0	0	1140	1140	NaN	0	NaN	50%
5	0	0	1140	1140	NaN	0	NaN	50%
10	0	0	1140	1140	NaN	0	NaN	50%
20	0	0	1140	1140	NaN	0	NaN	50%

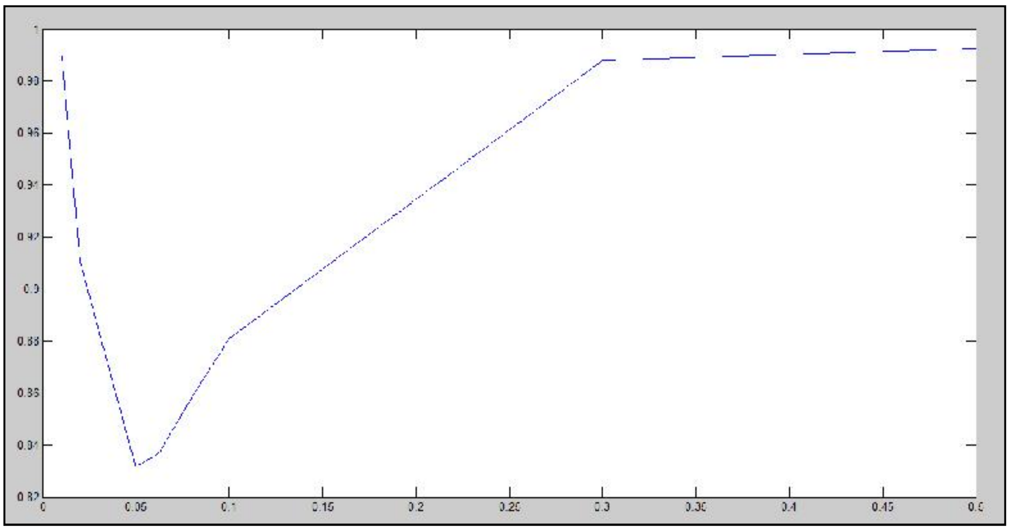

Figure 5: Plot of g v/s (1-FMeasure)

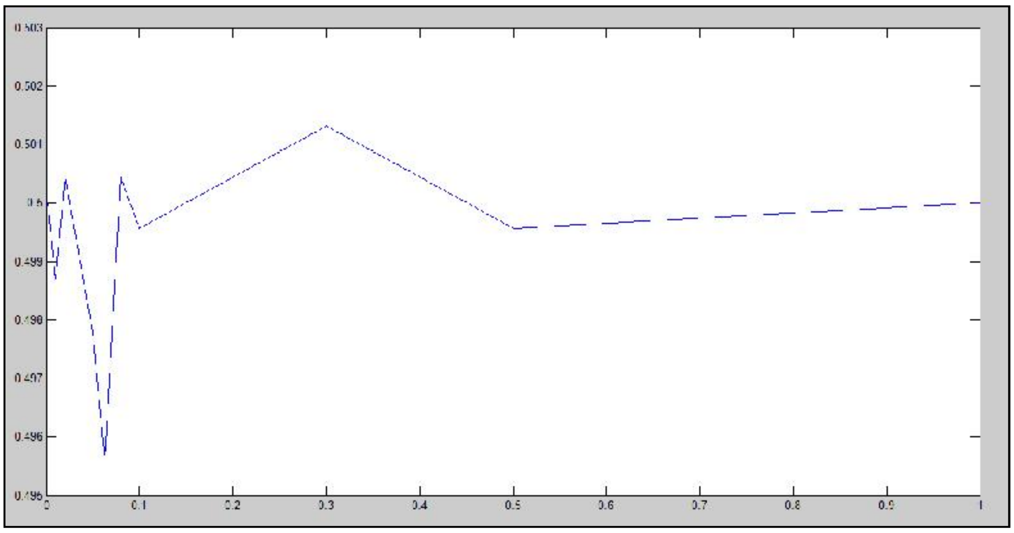

Figure 6: Plot of g v/s (1-Accuracy)

g is inversely proportional to sigma so we saw that for higher values of g a better performance is obtained. These g values are higher in comparison with the default value of 0.0007.Hence, decreasing the value of sigma (increasing value of g) increased the performance of the classifier. The optimum performance was achieved with g approximately equal to 0.08 keeping in mind the accuracy and F-Measure. This ensures that the overfitting of the data doesn’t occur. Beyond a value of 0.08 for g it is the case of overfit and below it, it is a case of underfit.

The effect on accuracy and F-Measure by varying the parameter g of the test data is as follows,

Value of g	True Positive	False Positive	True Negative	False Negative	Precision	Recall	F-Measure	Accuracy
0.001	0	0	1674	326	NaN	0	NaN	83.7%
0.01	3	11	1663	323	0.21429	0.0092025	0.0176467	83.3%
0.02	12	72	1602	314	0.14286	0.036810	0.058537	80.7%
0.05	26	151	1523	300	0.14689	0.079755	0.10338	77.45%
0.0625	24	144	1530	302	0.14286	0.073629	0.097166	77.7%
0.08	27	126	1548	299	0.17647	0.082822	0.11273	78.75%
0.1	24	108	1566	302	0.18182	0.073620	0.10480	79.5%
0.3	4	10	1664	322	0.28571	0.012270	0.023529	83.4%
0.5	1	1	1673	325	0.5	0.0030675	0.0060976	83.7%
1	0	0	1674	326	NaN	0	NaN	83.7%
5	0	0	1674	326	NaN	0	NaN	83.7%
10	0	0	1674	326	NaN	0	NaN	83.7%
20	0	0	1674	326	NaN	0	NaN	83.7%

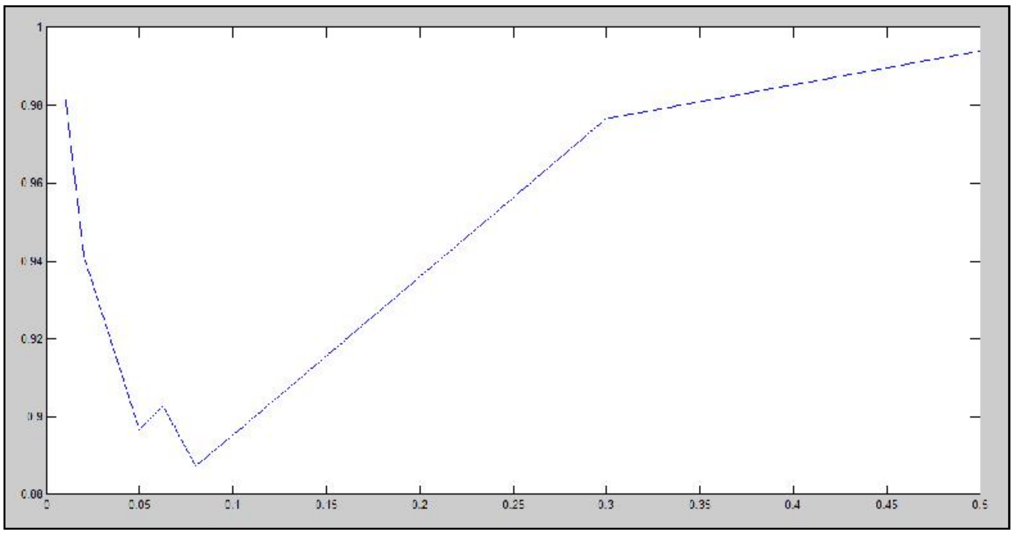

Figure 7: Plot of g v/s (1-FMeasure)

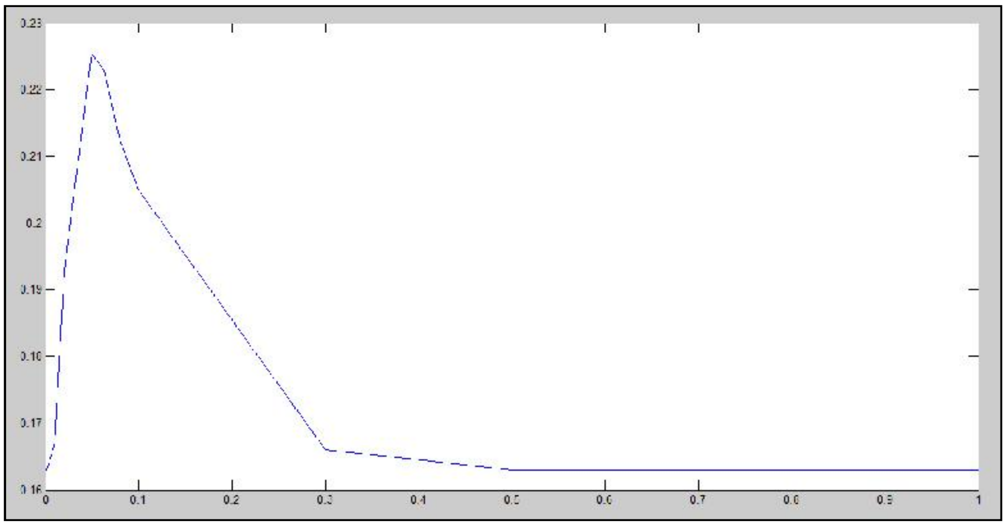

Figure 8: Plot of g v/s (1-Accuracy)

This is in coherence with choosing g=0.08 from our validation data performance.

OPTIMUM VALUE FOR CLASSIFICATION PARAMETERS:

Based on the validation the optimum value for the classification parameters were found out to be around:
C=30
And for the Gaussian Kernel g= 0.08
The accuracy for our test data was found out to be 78.75% and F-Measure of 0.11273 for a Gaussian kernel with the above set of parameters.

LEARNING CURVE:

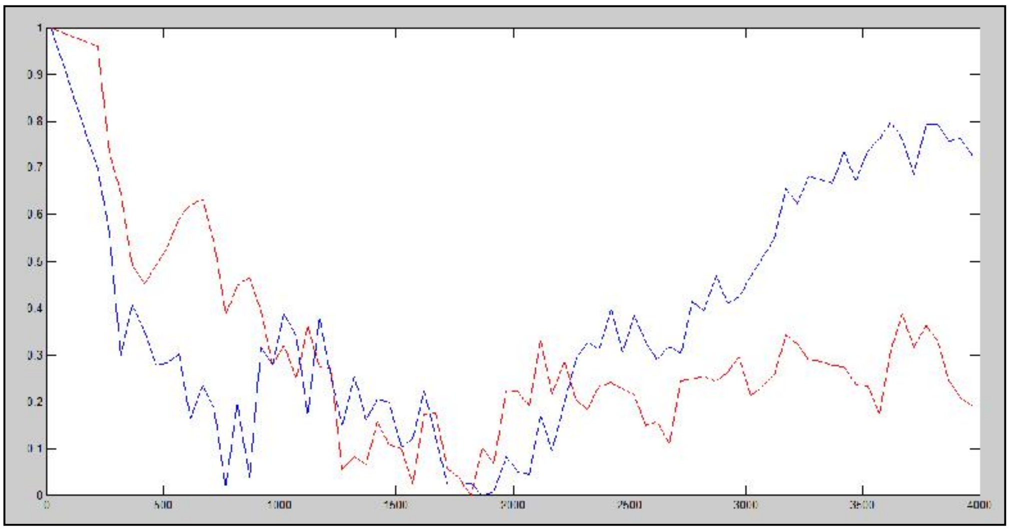

Figure 9: Plot of No. of training points v/s (1-FMeasure) for test data

The one in red is the Gaussian kernel with the optimum parameters and the one in blue is the linear kernel. As we can observe from the graph, the error has reduced and reduces with the increase in the number of samples.

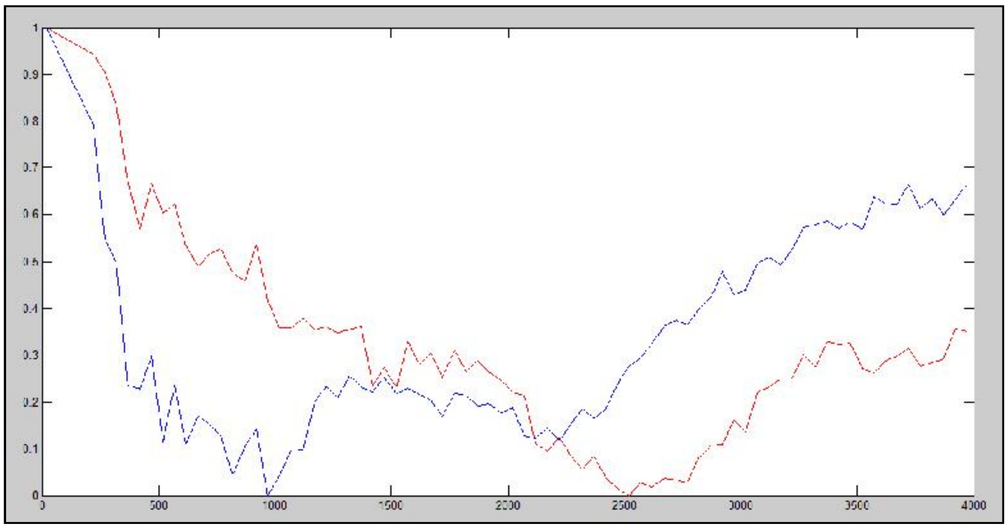

Figure 10: Plot of No. of training points v/s (1-FMeasure) for validation data

Similar observations were made for the validation data. Nothing much could be interpreted out of the accuracy measure due to the sparse dataset hence we took into consideration the F Measure.

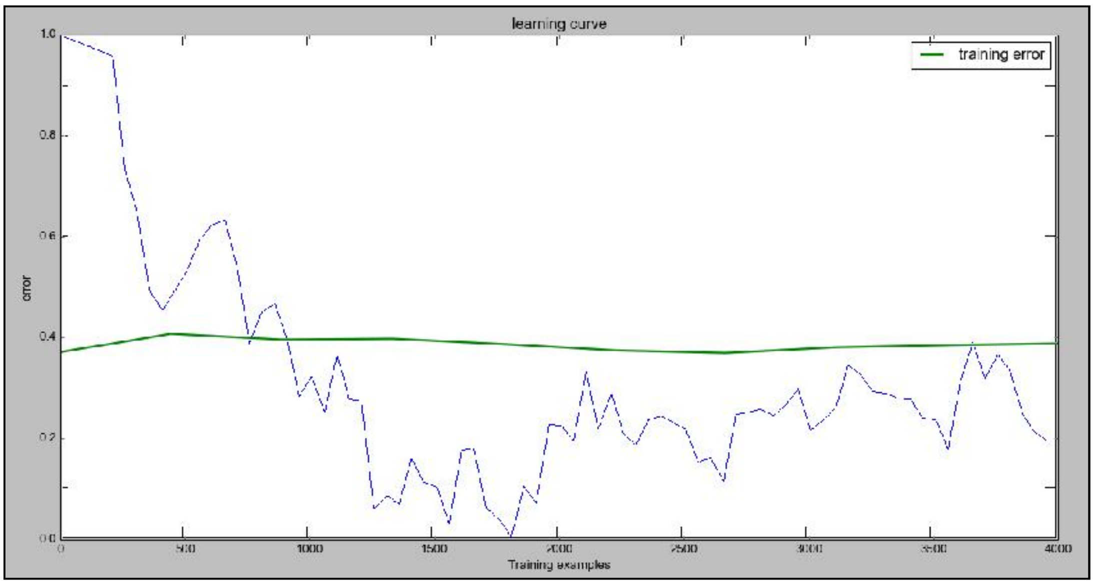

Figure 11: Learning curve for a simple Gaussian kernel (without optimum parameters)

We can conclude that it is a high bias case. If we were to smoothen out the error in test data we can see that it saturates at a low value of training data points. This is fixed by increasing the C value and decreasing the sigma value such that the error decreases and the performance improves.

OBSERVATIONS:

1. We observed that the performance measure for a simple linear kernel was less and it was a case of underfit of data.
2. To solve the issue of underfit we had to increase the value of C upto around 30 but not way beyond that as it might run into an overfit scenario.
3. By default the value of g=1/number of features=0.0007. As this is a case of underfit, we increased the value of g (decreased value of sigma) to 0.08 for a better performance. Beyond this we run into a situation of overfit.
4. Another way of solving the underfit issue is by introducing more features which are functions of the existing features.
5. As seen in the learning curve, the error decreased by a large amount when we implemented the kernel based SVM with the set of optimum parameters
6. Also from the learning curve we concluded that it is a high bias case (underfit) as it saturates at lower values of the data point and the error is high.

RESULT:

So we solved the issue of underfit of data by increasing C and decreasing sigma for a Gaussian Kernel based SVM to achieve an optimum performance of 78.75%. Although the Bayes' classification accuracy was 83.05% but this could be explained owing to the fact that the data set is skewed.