SMAI Project Report

Initially, the training accuracy was coming out to be huge as compared to the testing accuracy, which showed overfitting, which was occurring because the split between the train, validation and test data wasn't proper as the shuffle was True (by default). Then it was made false and the error was corrected.

Also, first RAW data was sent to the Logistic Regression Classifier, then PCA transformed data and then LDA transformed data. The observation is that the Testing Accuracy and F1 - Score was seen to be improved over the sequential features mentioned above.

Also, in PCA, first no. of components was been varied with values 50, 80, 100. Among which the best value was observed for "100" and then the value of C was varied (keeping no. of principal components = "100") as 0.8, 1.0, 1.5.

Also, the Accuracy and F1-score was seemed to be the same in all the cases.

Building Classifier from the data:

In Kernel SVM, the time taken to run for the whole was very huge, so trained for single batch files initially and then tested for the whole dataset.

In Decision Trees, the classifier always gave the training accuracy of "1.0" when no depth was mentioned and therefore early stopping was used to rectify this problem.

In Logistic Regression, the time taken was less as compared to Kernel SVM, but it gave lesser test accuracy. Increase in C value led to overfitting.

In Multi-Layer Perceptron, when no. of layers, no. of epochs were more, and no. of hidden neurons per layer was also more, then there was overfitting. But reduction in them and in the no. of epochs rectified this problem

1. **Logistic Regression**

RAW	Hyper Parameters	Training Accuracy	Validation Accuracy	Testing Accuracy	Testing F1-Score
1.	C = 1.5	0.3243	0.2893	0.2971	0.2971
2.	C = 1.0	0.3258	0.2923	0.2991	0.2991
3.	C = 0.8	0.3240	0.2908	0.2941	0.2941

PCA	Hyper Parameters	Training Accuracy	Validation Accuracy	Testing Accuracy	Testing F1-Score
1.	C = 1.5, n_comp = 100	0.31535	0.2931	0.3018	0.3018
2.	C = 1.0, n_comp = 50 C = 1.0, n_comp = 80 C = 1.0, n_comp = 100	0.3038 0.31025 0.31185	0.3006 0.3004 0.3044	0.3033 0.3034 0.3037	0.3033 0.3034 0.3037
3.	C = 0.8, n_comp = 100	0.309625	0.3048	0.3015	0.3015

LDA	Hyper Parameters	Training Accuracy	Validation Accuracy	Testing Accuracy	Testing F1-Score
1.	C = 1.5	0.341775	0.3502	0.3414	0.3414
2.	C = 1.0	0.344325	0.3438	0.3404	0.3404
3.	C = 0.8	0.344225	0.3415	0.3403	0.3403

The parameters are taken as (solver = “lbfgs”, multiclass = “multinomial”).
For RAW data, the best value for testing accuracy and F1-score was observed for C = 0.1.
For PCA data, the best value for testing accuracy and F1-score was observed for C = 0.1.
For LDA data, the best value for testing accuracy and F1-score was observed for C = 1.5.

2. Kernel SVM

RAW	Hyper Parameters	Training Accuracy	Validation Accuracy	Testing Accuracy	Testing F1-Score
1.	C = 1.5, gamma = 10	0.1777	0.1302	0.0914	0.0914
2.	C = 1.0, gamma = 0.001	0.1514	0.1263	0.1071	0.1071
3.	C = 0.8, gamma = 5	0.1601	0.1408	0.0941	0.0941

PCA	Hyper Parameters	Training	Validation	Testing	Testing

		Accuracy	Accuracy	Accuracy	F1-Score
1.	C = 1.5, gamma = 10	0.211775	0.1302	0.0914	0.0914
2.	C = 1.0, gamma = 0.001	0.1114	0.1063	0.1171	0.1171
3.	C = 0.8, gamma = 5	0.2040	0.1208	0.1041	0.1041

LDA	Hyper Parameters	Training Accuracy	Validation Accuracy	Testing Accuracy	Testing F1-Score
1.	C = 1.5, gamma = 10	0.51775	0.5102	0.5014	0.5014
2.	C = 1.0, gamma = 0.001	0.5114	0.5103	0.5071	0.5071
3.	C = 0.8, gamma = 5	0.5140	0.5108	0.5041	0.5041

3. Multi-Layer Perceptron

RAW	Hyper Parameters	Training Accuracy	Validation Accuracy	Testing Accuracy	Testing F1-Score
1.	hidden_layer_sizes=(512, 256,128,32,16), activation='relu', max_iter=20 **Overfitting: As training accuracy >> Testing accuracy**	0.542175	0.427	0.4303	0.4303
2.	hidden_layer_sizes=(256,128,32,16), activation='tanh', max_iter=20	0.48335	0.3874	0.395	0.395
3.	hidden_layer_sizes=(256,128,32,16), activation='tanh', max_iter=10	0.42895	0.3937	0.3911	0.3911

PCA	Hyper Parameters	Training Accuracy	Validation Accuracy	Testing Accuracy	Testing F1-Score

1.	n_comp = 80 hidden_layer_sizes=(512, 256,128,32,16), activation='relu', max_iter=20 **Overfitting: As training accuracy >> Testing accuracy**	0.9179	0.4317	0.4433	0.4433
2.	n_comp = 80 hidden_layer_sizes=(256,128,32,16), activation='relu', max_iter=20 **Less Overfitting**	0.715725	0.4423	0.4391	0.4391
3.	n_comp = 100 hidden_layer_sizes=(256,128,32,16), activation='tanh', max_iter=10	0.4376	0.3919	0.3961	0.3961

LDA	Hyper Parameters	Training Accuracy	Validation Accuracy	Testing Accuracy	Testing F1-Score
1.	hidden_layer_sizes=(512,256,128,32,16), activation='tanh', max_iter=20 **Overfitting: As training accuracy >> Testing accuracy**	0.4297	0.3465	0.3364	0.3364
2.	hidden_layer_sizes=(256,128,32,16), activation='tanh', max_iter=20 **Less Overfitting**	0.37925	0.3442	0.3472	0.3472
3.	hidden_layer_sizes=(256,128,32,16), activation='relu', max_iter=10	0.37205	0.3569	0.3546	0.3546

4. Decision Trees

RAW	Hyper Parameters	Training Accuracy	Validation Accuracy	Testing Accuracy	Testing F1-Score
1.	criterion = “gini” **Overfitting: As training accuracy >> Testing accuracy**	1.0	0.2271	0.23	0.23
2.	criterion = “gini” max_depth = 8	0.310375	0.2531	0.2493	0.2493
3.	criterion = “gini” max_depth = 10	0.39545	0.2515	0.2595	0.2595

PCA	Hyper Parameters	Training Accuracy	Validation Accuracy	Testing Accuracy	Testing F1-Score
1.	n_comp = 80 criterion = “gini” **Overfitting: As training accuracy >> Testing accuracy**	1.0	0.2292	0.2284	0.2284
2.	n_comp = 80 criterion='gini', max_depth = 8	0.32375	0.2791	0.2935	0.2935
3.	n_comp = 80 criterion='gini', max_depth = 10	0.3857	0.2842	0.2834	0.2834

LDA	Hyper Parameters	Training Accuracy	Validation Accuracy	Testing Accuracy	Testing F1-Score
1.	criterion = “gini” **Overfitting: As training accuracy >> Testing accuracy**	1.0	0.2277	0.2206	0.2206
2.	criterion='gini', max_depth = 8	0.3289	0.2994	0.2996	0.2996

3.	criterion='gini', max_depth = 10	0.37685	0.2843	0.2962	0.2962

For input from command language interpreter (CLI) the following format is to be followed:

```
mini_proj2.py <Data_method> <Classifier_method> <Input_method>
```