

# Classification of the MNIST dataset

**Pattern Recognition**  
**ICP**

**Sajina Pathak**

**Supervisor: Dr. Georgios Anagnostopoulos**



# Outline

1. Dataset source
2. Data Visualization
3. Classification Algorithms implemented
4. Comparision with baseline methods
5. Conclusion

# Objective

- Implement classification algorithm taught in the Pattern Recognition class.
- Image Classification on sign language.

# Approach

- Image Classification using
  - LDA (Linear Discriminant Analysis)
  - LDA with PCA (Principal Component Analysis)
  - SVM (Support Vector Machine)
  - CNN (Convolutional Neural Network)

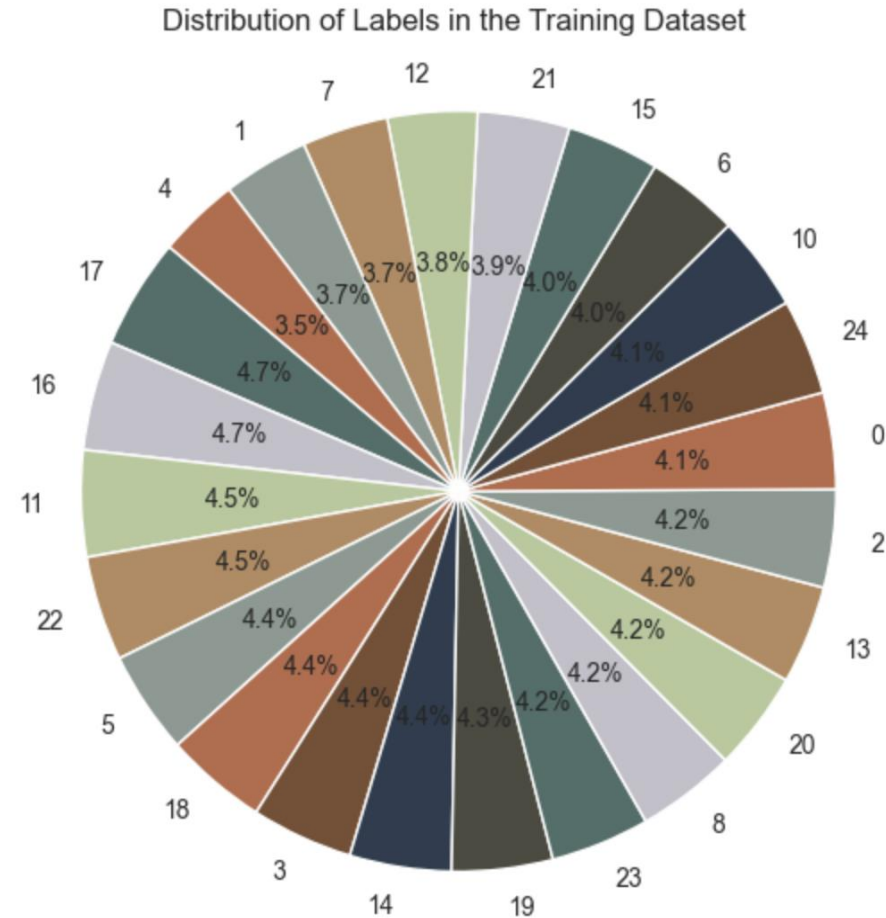
# DATASETS



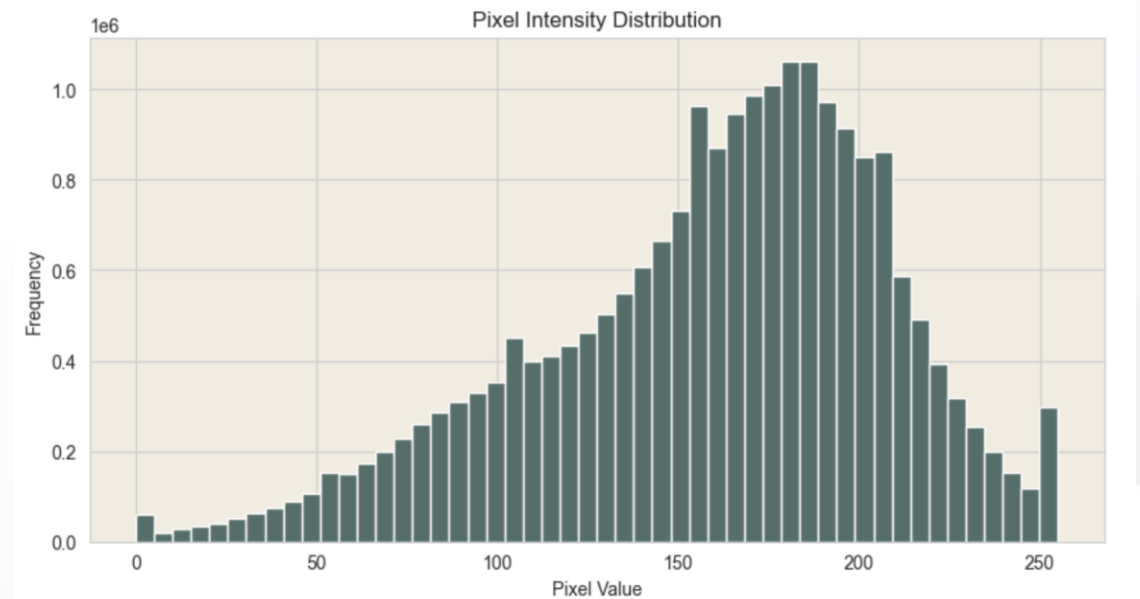
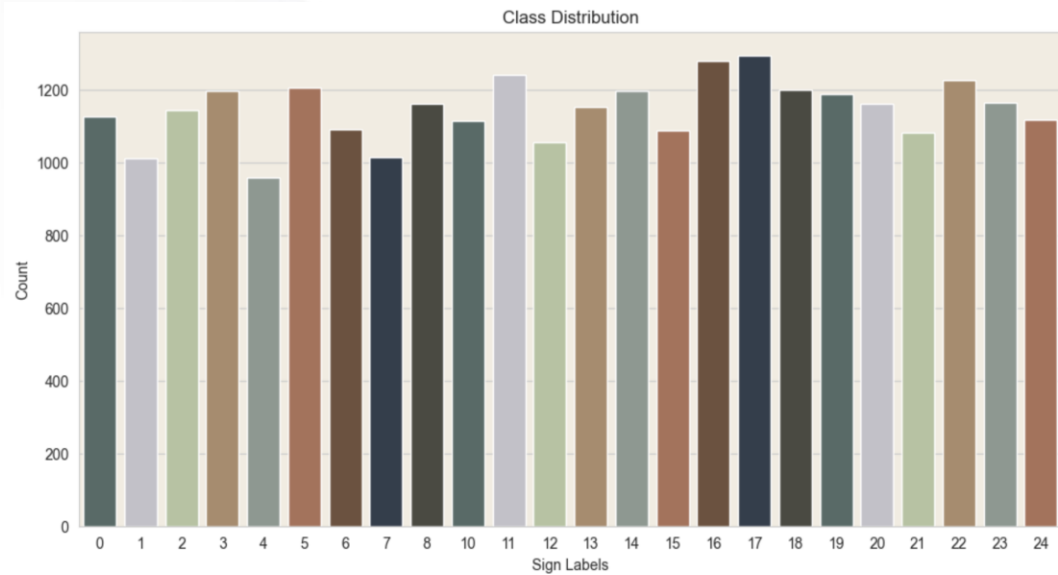
- MNIST(Modified National Institute of Standards and Technology) image dataset.
- The American Sign Language letter database of hand gestures represent a multi-class problem with 24 classes of letters (excluding J and Z which require motion).
- Training data (27,455 cases) and Test data (7172 cases).
- Pixel to pixel 784 which represent a single 28x28 pixel image with grayscale values between 0-255.
- <https://www.kaggle.com/datasets/datamunge/sign-language-mnist>

# Data Visualization

- Number of data per label.



# Data Visualization



# Linear Discriminant Analysis

		Confusion Matrix																
True Labels	0	259	0	0	0	0	0	0	0	0	0	0	29	21	0	0	0	0
	1	0	281	0	54	0	2	0	0	0	0	40	0	0	0	0	0	46
	2	0	0	251	0	0	0	0	0	0	0	1	0	0	39	0	0	0
	3	0	11	0	154	3	0	0	0	7	0	0	0	6	0	0	0	0
	4	0	22	0	0	293	0	0	0	0	0	0	88	23	0	0	0	6
	5	0	0	0	21	0	112	0	0	0	0	20	3	0	0	0	0	5
	6	3	0	0	20	0	0	170	13	1	0	0	0	1	6	0	0	20
	7	0	0	0	0	0	11	60	231	0	0	0	20	1	0	3	6	37
	8	0	0	0	1	17	0	0	0	47	0	17	19	37	16	0	21	2
	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	10	0	19	0	40	0	6	0	0	2	0	109	0	0	0	0	0	95
	11	0	16	0	5	0	1	0	2	0	0	0	109	0	0	0	0	0
	12	69	0	0	0	55	0	6	0	22	0	0	0	98	83	21	0	0
	13	84	21	0	0	0	0	0	0	9	0	0	0	35	66	0	0	10
	14	0	0	1	3	17	20	0	21	15	0	0	1	0	0	102	20	0
	15	0	0	0	0	12	38	0	15	31	0	13	18	17	0	0	64	54
	16	20	0	0	0	0	3	0	18	0	0	0	0	21	0	0	0	81
	17	0	0	0	18	0	0	0	0	0	0	26	5	0	0	0	0	21

```

Accuracy: 0.4337702175125488
precision recall f1-score support
0 0.565502 0.782477 0.656527 331.0
1 0.682039 0.650463 0.665877 432.0
2 0.940075 0.809677 0.870017 310.0
3 0.383085 0.628571 0.476043 245.0
4 0.694313 0.588353 0.636957 498.0
5 0.500000 0.453441 0.475584 247.0
6 0.627306 0.488506 0.549273 348.0
7 0.770000 0.529817 0.627717 436.0
8 0.265537 0.163194 0.202151 288.0
9 0.000000 0.000000 0.000000 0.0
10 0.339564 0.329305 0.334356 331.0
11 0.516588 0.521531 0.519048 209.0
12 0.276836 0.248731 0.262032 394.0
13 0.267206 0.226804 0.245353 291.0
14 0.618182 0.414634 0.496350 246.0
15 0.566372 0.184438 0.278261 347.0
16 0.305660 0.493902 0.377622 164.0
17 0.049881 0.145833 0.074336 144.0
18 0.243094 0.357724 0.289474 246.0
19 0.275785 0.495968 0.354467 248.0
20 0.264516 0.308271 0.284722 266.0
21 0.373541 0.277457 0.318408 346.0
22 0.200000 0.393204 0.265139 206.0
23 0.461538 0.337079 0.389610 267.0
24 0.616766 0.310241 0.412826 332.0
macro avg 0.432135 0.405585 0.402486 NaN
weighted avg 0.484798 0.433770 0.442672 NaN
    
```

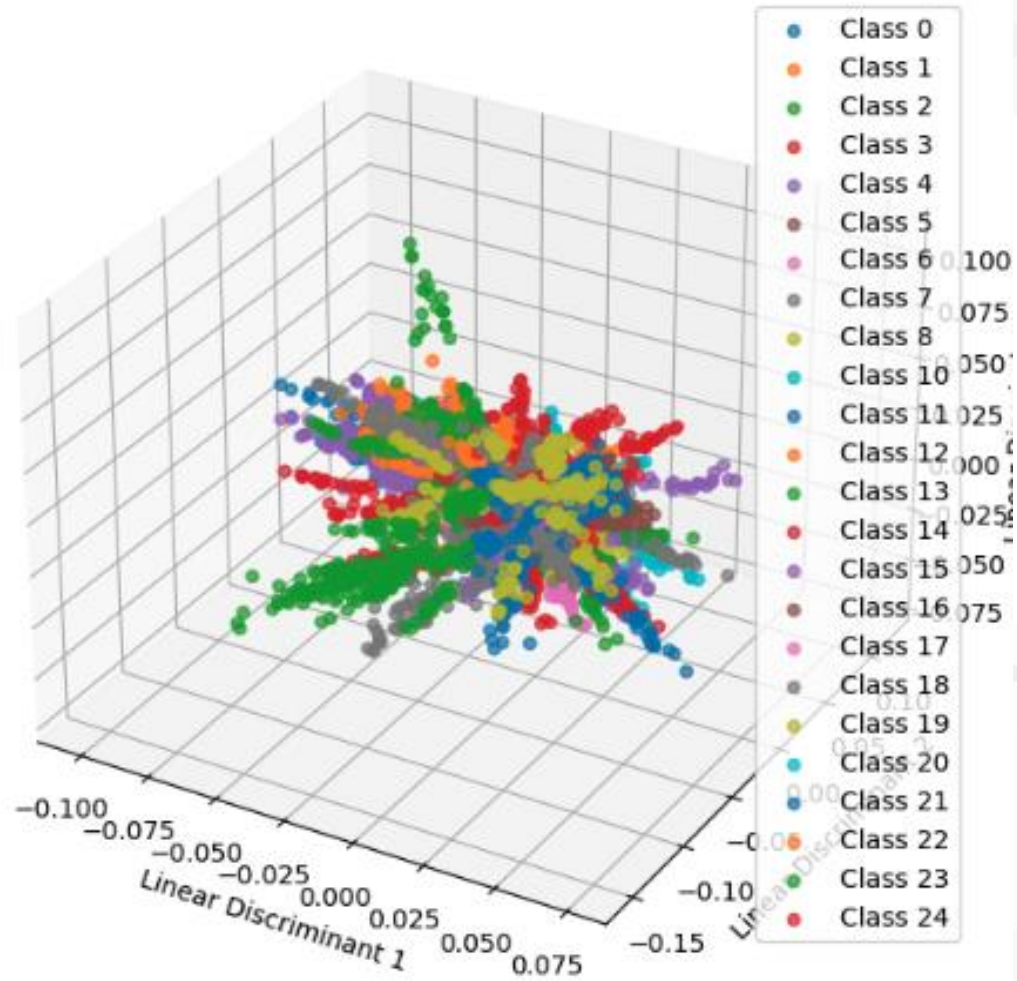


# LDA with PCA

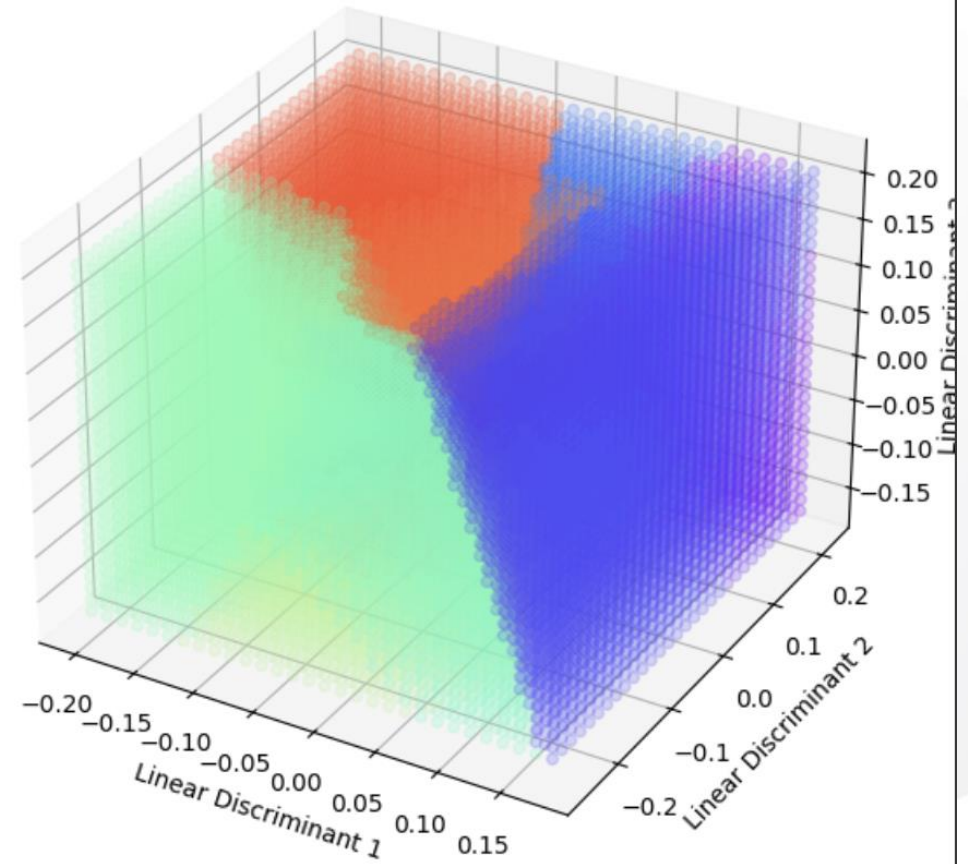
True Labels	6	0	0	0	20	0	0	203	18	0	0	0	0	21	14	0	19	0	0	52	0	0	0	1	0
	7	0	0	0	0	16	0	45	348	0	0	0	0	0	0	0	0	0	22	0	0	0	5	0	
	8	3	0	0	0	0	0	0	0	190	0	0	0	21	0	0	12	0	21	0	0	0	0	0	41
	9	0	0	0	9	0	21	0	0	24	159	0	0	0	0	0	0	87	3	0	0	0	8	0	20
	10	0	0	0	0	0	0	0	0	0	0	209	0	0	0	0	0	0	0	0	0	0	0	0	0
	11	24	0	0	0	47	0	0	0	0	0	0	135	45	0	0	39	0	104	0	0	0	0	0	0
	12	42	0	0	4	14	0	0	0	0	0	0	15	139	10	0	33	0	2	32	0	0	0	0	0
	13	0	0	17	0	21	21	8	0	0	0	0	0	0	138	0	20	0	0	5	4	0	0	12	0
	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	324	22	0	0	1	0	0	0	0	0
	15	0	0	0	0	0	3	0	0	0	0	0	21	0	0	0	140	0	0	0	0	0	0	0	0
	16	0	0	0	0	0	0	0	0	0	0	21	0	0	0	0	0	41	62	0	20	0	0	0	0
	17	0	0	0	3	22	0	0	0	40	0	0	57	27	0	0	3	0	43	0	18	0	12	0	21
	18	0	0	1	0	0	0	0	0	21	0	40	0	0	0	21	0	0	0	104	20	0	0	41	0
	19	0	17	0	39	0	3	0	0	0	41	0	0	0	0	0	0	56	0	0	80	30	0	0	0
	20	0	11	0	3	0	35	0	0	0	16	0	0	0	0	19	0	32	0	0	50	131	32	0	17
	21	0	20	0	0	0	3	0	0	0	20	0	0	0	0	0	0	22	0	0	17	1	123	0	0
	22	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	8	24	20	0	0	48	166	0
	23	0	0	0	17	0	0	0	0	17	0	9	0	0	0	0	0	63	9	42	0	2	22	0	151
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
		Predicted Labels																							

3	0.59	0.57	0.58	245
4	0.78	0.87	0.82	498
5	0.67	0.83	0.74	247
6	0.71	0.58	0.64	348
7	0.95	0.74	0.83	436
8	0.66	0.63	0.64	288
10	0.47	0.50	0.48	331
11	0.73	1.00	0.84	209
12	0.66	0.31	0.42	394
13	0.51	0.45	0.48	291
14	0.81	0.57	0.67	246
15	0.89	0.94	0.91	347
16	0.47	0.85	0.60	164
17	0.11	0.28	0.16	144
18	0.19	0.29	0.23	246
19	0.37	0.46	0.41	248
20	0.43	0.31	0.36	266
21	0.82	0.33	0.47	346
22	0.44	0.52	0.48	206
23	0.59	0.63	0.61	267
24	0.65	0.45	0.53	332
accuracy			0.63	7172
macro avg	0.63	0.62	0.61	7172
weighted avg	0.67	0.63	0.63	7172

LDA-transformed Data 3D Scatter Plot



Decision Boundaries in LDA-Reduced 3D Space



# Hyper parameter tuning

	19	0.37	0.46
	20	0.43	0.31
	21	0.82	0.33
	22	0.44	0.52
	23	0.59	0.63
	24	0.65	0.45
accuracy			
macro avg	0.63	0.62	
weighted avg	0.67	0.63	

Summary of Best Model:

Best Hyperparameters:

Learning Rate: 0.1

Number of Iterations: 2000

Regularization Strength: 0.01

Best Testing Accuracy: 62.98%

Training with learning\_rate=0.01, num\_iterations=5000, reg\_strength=0.005  
Testing accuracy: 56.05%

Training with learning\_rate=0.01, num\_iterations=5000, reg\_strength=0.01  
Testing accuracy: 55.69%

Training with learning\_rate=0.01, num\_iterations=5000, reg\_strength=0.02  
Testing accuracy: 55.05%

Training with learning\_rate=0.02, num\_iterations=2000, reg\_strength=0.005  
Testing accuracy: 55.28%

Training with learning\_rate=0.02, num\_iterations=2000, reg\_strength=0.01  
Testing accuracy: 55.09%

Training with learning\_rate=0.02, num\_iterations=2000, reg\_strength=0.02  
Testing accuracy: 54.56%

Training with learning\_rate=0.02, num\_iterations=3000, reg\_strength=0.005  
Testing accuracy: 56.76%

Training with learning\_rate=0.02, num\_iterations=3000, reg\_strength=0.01  
Testing accuracy: 56.09%

Training with learning\_rate=0.02, num\_iterations=3000, reg\_strength=0.02  
Testing accuracy: 55.31%

Training with learning\_rate=0.02, num\_iterations=5000, reg\_strength=0.005  
Testing accuracy: 61.60%

Training with learning\_rate=0.02, num\_iterations=5000, reg\_strength=0.01

# Logistic Regression Model

```
Iteration 0, Loss: 3.1781  
Iteration 100, Loss: 2.3069  
Iteration 200, Loss: 1.9253  
Iteration 300, Loss: 1.7007  
Iteration 400, Loss: 1.5470  
Iteration 500, Loss: 1.4323  
Iteration 600, Loss: 1.3419  
Iteration 700, Loss: 1.2679  
Iteration 800, Loss: 1.2056  
Iteration 900, Loss: 1.1520  
Training Accuracy: 76.39%  
Testing Accuracy: 63.43%
```

# SVM

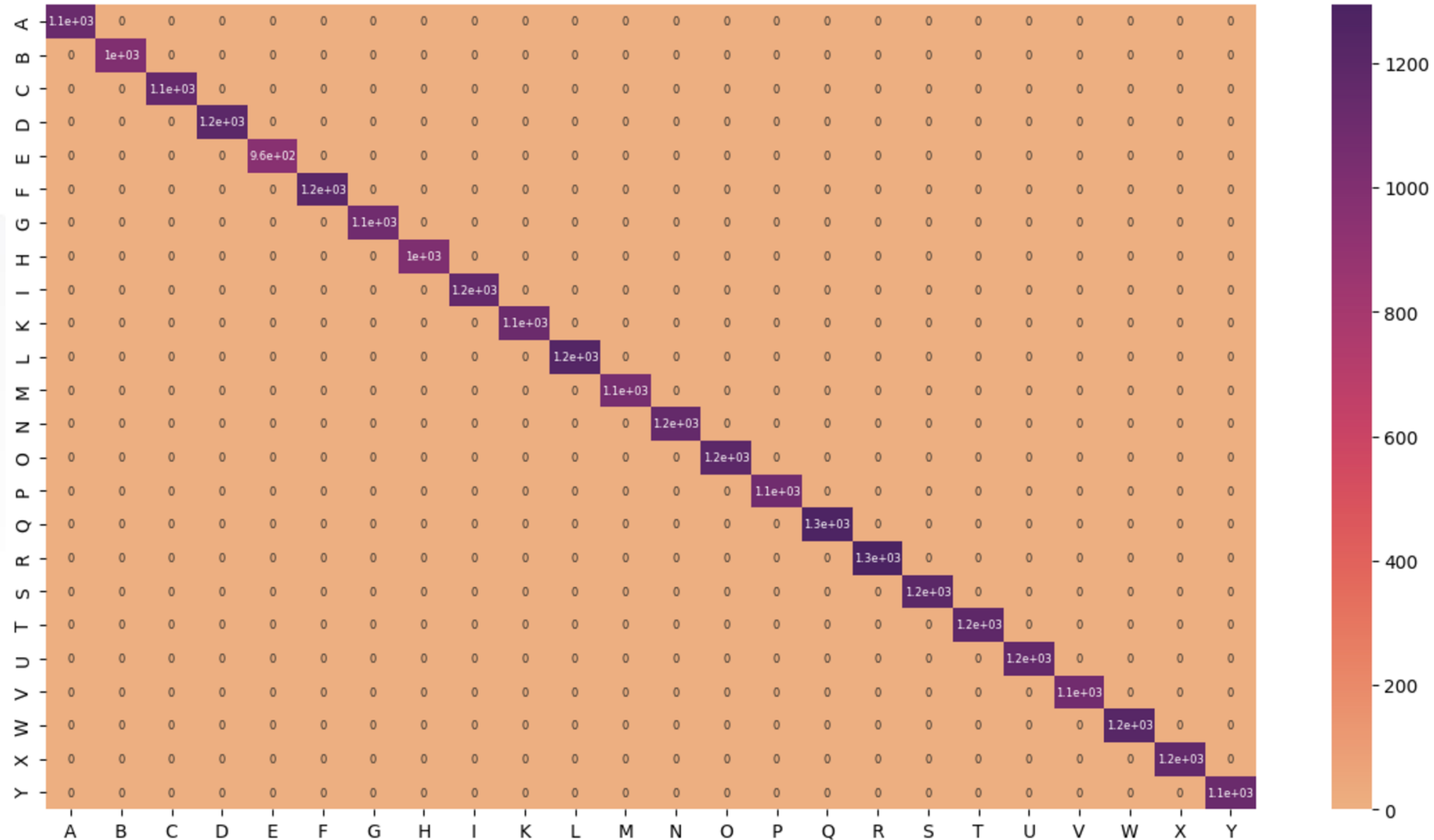
## Classification Report:

	precision	recall	f1-score	support
0	0.82	0.98	0.89	331
1	0.88	0.81	0.84	432
2	0.87	0.85	0.86	310
3	0.62	0.62	0.62	245
4	0.78	0.87	0.83	498
5	0.66	0.81	0.73	247
6	0.79	0.58	0.67	348
7	0.95	0.80	0.87	436
8	0.65	0.66	0.66	288
10	0.49	0.48	0.48	331
11	0.73	1.00	0.85	209
12	0.59	0.34	0.43	394
13	0.51	0.48	0.50	291
14	0.73	0.56	0.63	246
15	0.89	0.93	0.91	347
16	0.49	0.85	0.62	164
17	0.13	0.28	0.18	144
18	0.13	0.17	0.15	246
19	0.37	0.42	0.40	248
20	0.38	0.30	0.34	266
21	0.77	0.38	0.51	346
22	0.45	0.60	0.51	206
23	0.59	0.62	0.60	267
24	0.60	0.45	0.52	332
accuracy			0.63	7172
macro avg	0.62	0.62	0.61	7172
weighted avg	0.66	0.63	0.64	7172



True Labels	0	323	0	0	0	0	0	0	0	0	0	0	0	3	4	0	0	0	1	0	0	0	0	0	0	
	1	0	349	0	0	0	0	0	0	0	75	0	0	0	0	0	0	0	0	0	0	0	8	0	0	
	2	0	0	264	0	0	16	0	0	0	0	5	0	0	19	0	0	0	0	0	0	0	0	6	0	
	3	0	0	0	153	0	0	0	0	0	15	0	0	14	0	0	0	0	5	0	0	6	0	52	0	
	4	0	0	0	0	435	0	0	0	0	0	0	0	0	0	0	0	0	63	0	0	0	0	0	0	
	5	0	0	20	0	0	201	0	0	0	0	1	0	0	5	0	0	0	0	0	0	0	20	0	0	
	6	0	0	0	20	0	0	203	18	0	0	0	0	21	14	0	19	0	0	52	0	0	0	1	0	
	7	0	0	0	0	16	0	45	348	0	0	0	0	0	0	0	0	0	0	22	0	0	0	5	0	
	8	3	0	0	0	0	0	0	190	0	0	0	0	21	0	0	12	0	21	0	0	0	0	0	41	
	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	10	0	0	0	9	0	21	0	0	24	0	159	0	0	0	0	0	87	3	0	0	0	8	0	20	
	11	0	0	0	0	0	0	0	0	0	0	209	0	0	0	0	0	0	0	0	0	0	0	0	0	
	12	24	0	0	0	47	0	0	0	0	0	0	135	45	0	0	39	0	104	0	0	0	0	0	0	
	13	42	0	0	4	14	0	0	0	0	0	0	15	139	10	0	33	0	2	32	0	0	0	0	0	
	14	0	0	17	0	21	21	8	0	0	0	0	0	0	138	0	20	0	0	5	4	0	0	12	0	
	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	324	22	0	0	1	0	0	0	0	0	
	16	0	0	0	0	0	3	0	0	0	0	0	0	21	0	0	0	140	0	0	0	0	0	0	0	
	17	0	0	0	0	0	0	0	0	0	0	21	0	0	0	0	0	41	62	0	20	0	0	0	0	
	18	0	0	0	3	22	0	0	0	40	0	0	0	57	27	0	0	3	0	43	0	18	0	12	0	21
	19	0	0	1	0	0	0	0	0	21	0	0	40	0	0	0	21	0	0	0	104	20	0	0	41	0
	20	0	17	0	39	0	3	0	0	0	0	41	0	0	0	0	0	56	0	0	80	30	0	0	0	
	21	0	11	0	3	0	35	0	0	0	0	16	0	0	0	0	19	0	32	0	0	50	131	32	0	17
	22	0	20	0	0	0	3	0	0	0	0	20	0	0	0	0	0	22	0	0	17	1	123	0	0	
	23	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	8	24	20	0	0	48	166	0	
	24	0	0	0	17	0	0	0	0	17	0	0	9	0	0	0	0	63	9	42	0	2	22	0	151	
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
	Predicted Labels																									

# CNN



# Testing

Predicted: 6, True: 6



Predicted: 13, True: 0



Predicted: 7, True: 14



Predicted: 19, True: 19

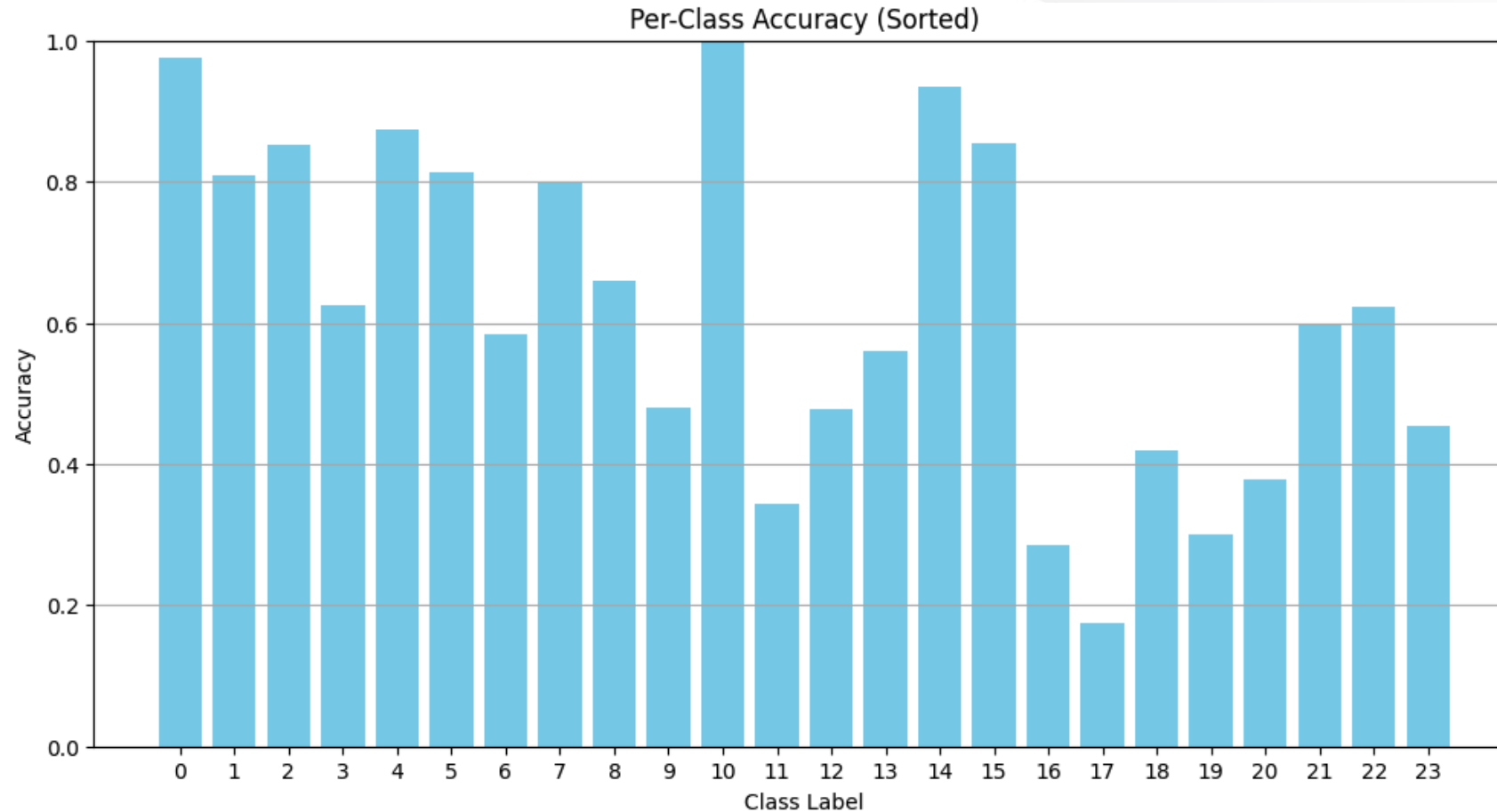


Predicted: 19, True: 10





# Prediction Accuracy



# Observation

Model	Implementation	Metrics			
		Accuracy	Precision	Recall	F1 Score
LDA	Custom	43.38%	0.48	0.43	0.40
LDA with PCA	Custom	61.88%	0.66	0.62	0.63
Logistic Regression	Custom	62.98%	0.67	0.63	0.63
SVM	Library	63%	0.66	0.63	0.64
CNN (Baseline)	Library	100%	1	1	1



# Conclusion

- Implement and test various Classifier model and classify Images in American Sign Language.
- Construct the model with accuracy above 76%.
- Hyper parameter tuning, PCA can increase the performance of a model.



# References

- [https://www.youtube.com/watch?v=YYEJ\\_GUguHw](https://www.youtube.com/watch?v=YYEJ_GUguHw)
- <https://github.com/vaibhavbichave/American-Sign-Language/blob/master/Sign%20Language%20MNIST.ipynb>

