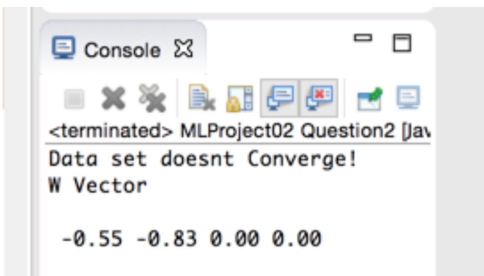
1. Introduction

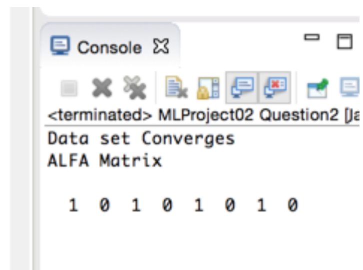
The project aims at implementation of the four perceptron algorithms and evaluation of the digit recognition data set using these four algorithms and Support Vector machines to analyze the performance of the algorithms and its results.

2. Results and Analysis using Perceptron Algorithms

2.1 Question B

The dat set given in the question 2 of the assignment is evaluated with the Linear Perceptron algorithm implemented. The result proved that the data doesn't converge at this feature space with number of Epochs set to 2, 5, 10. The same data set converges with Polynomial Kernel Perceptron Algorithm with degree 2 and No of Epochs as 3.





2.2 Question C

For Digit Recognition, The training file “optdigits.tra” has features for digits 0 to 9. So, it is divided into 9 training files each for a digit setting the class to 1 for instances of the digit and -1 for instances of the other digits.

2.3 Question D

2.3.1 Optimal Parameters

The Linear perceptron algorithm is trained with the training data and tested with the development data to obtain a best value for T among {1,5,10,20}. The best value for T that is obtained on the development data is 5 with an accuracy of 90.8%. Refer table 1.1 for values of sigma and their accuracies.

Table 1.1

Accuracy %’s for Different values of T				
Epochs	1	5	10	15
Accuracy(%)	88	90.8	87.9	89.6

The Polynomial Kernel Perceptron algorithm is trained with the training data and tested with the development data to yield a best value for d as 5 with an accuracy of 98.3%. Refer table 1.2 for values of sigma and their accuracies.

Table 1.2

Accuracy %'s for Different values of d					
Exponent(d)	2	3	4	5	6
Accuracy(%)	94.69	96.5	97	98.3	97.39

The Gaussian Kernel Perceptron Algorithm trained with training data and tested with development data resulted the best sigma as 0.5 with and Accuracy of 99.6%. Refer table 1.3 for values of sigma and their accuracies.

Table 1.3

Accuracy %'s for Different values of Sigma					
Sigma	0.5	2	3	5	10
Accuracy(%)	99.6	97.1	97.2	97.2	98.2

All The kernels and Weight Vectors that are used for the above computations are normalized except Gaussian Kernel as it is not subjective to Normalization.

2.3.2 Training Times and Support Vectors

Table 1.4 shows the Training times for different Models and the values reveals that Gaussian models take more time compared to other models. This is because the Gaussian Kernel is e^(-||X-Y||^2/2*s*s). So, the no of steps involved in the Gaussian kernel computation is more compared to others.

Table 1.4

Model	Training Time(Milli Sec's)
Linear Perceptron	242
Avg. Linear Perceptron	714
Polynomial Kernel Perceptron	18128
Avg. Poly Kernel Perceptron	31735
Gaussian Kernel Perceptron	226456
Avg. Gaussian Kernel Perceptron	227918

Table 1.5 shows the no of support vectors for different Models implementing different versions of Perceptron Algorithm. It is observed that Gaussian Algorithms has got more no of support vectors compared to polynomial kernels Algorithms. Its observed that no of support vectors remains same for both normal and average kernel algorithms.

Table 1.5

Model for Digit	**Polynomial Kernel Peceptron**	**Avg. Kernel Polynomial Perceptron**	**Gaussian Kernel Perceptron**	**Avg. Gaussain Kernel Perceptron**
0	22	22	1194	1194
1	103	103	1176	1176
2	46	46	1142	1142
3	88	88	1107	1107
4	45	45	1136	1136
5	50	50	1101	1101
6	46	46	1175	1175
7	42	42	1143	1143
8	145	145	1059	1059
9	141	141	1119	1119

2.3.3 Comparison of Confusion Matrices on Test Data

1. Gaussian and Averaged Gaussian Kernel Perceptron

Gaussian Kernel Perceptron Confusion Matrix

	TD0	TD1	TD2	TD3	TD4	TD5	TD6	TD7	TD8	TD9	
SD0	178	46	71	82	83	101	42	94	112	92	901
SD1	0	135	0	0	0	0	0	0	0	0	135
SD2	0	0	106	0	0	0	0	0	0	0	106
SD3	0	0	0	100	0	0	0	0	0	0	100
SD4	0	0	0	0	98	0	0	0	0	0	98
SD5	0	0	0	0	0	81	0	0	0	1	82
SD6	0	0	0	0	0	0	139	0	0	0	139
SD7	0	0	0	0	0	0	0	85	0	0	85
SD8	0	1	0	0	0	0	0	0	62	0	63
SD9	0	0	0	1	0	0	0	0	0	87	88
Sum	178	182	177	183	181	182	181	179	174	180	1797
%	100.00	74.10	59.88	54.64	54.14	44.50	76.79	47.48	35.63	53.88	

Average Gaussain Kernel Perceptron Confusion Matrix

	TD0	TD1	TD2	TD3	TD4	TD5	TD6	TD7	TD8	TD9	Sum
SD0	178	46	71	82	83	101	42	94	112	92	901
SD1	0	135	0	0	0	0	0	0	0	0	135
SD2	0	0	106	0	0	0	0	0	0	0	106
SD3	0	0	0	100	0	0	0	0	0	0	100
SD4	0	0	0	0	98	0	0	0	0	0	98
SD5	0	0	0	0	0	81	0	0	0	1	82
SD6	0	0	0	0	0	0	139	0	0	0	139
SD7	0	0	0	0	0	0	0	85	0	0	85
SD8	0	1	0	0	0	0	0	0	62	0	63
SD9	0	0	0	1	0	0	0	0	0	87	88
Sum	178	182	177	183	181	182	181	179	174	180	1797
%	100.00	74.10	59.88	54.64	54.14	44.50	76.79	47.48	35.63	53.88	

The Confusion Matrix clearly shows that both Gaussian Kernel Perceptron Algorithm and Averaged Gaussian Kernel Perceptron Algorithm find digit 8 classification hard.

2. Linear Perceptron and Averaged Linear Perceptron

	Perceptron Confusion Matrix										
	TD0	TD1	TD2	TD3	TD4	TD5	TD6	TD7	TD8	TD9	Sum
SD0	169	0	0	0	0	0	0	0	0	0	169
SD1	0	146	2	0	0	0	0	0	11	1	160
SD2	0	12	174	3	0	1	0	0	0	0	190
SD3	0	0	1	166	0	0	0	1	1	0	169
SD4	1	1	0	0	178	0	2	5	0	4	191
SD4	8	5	0	4	0	180	3	11	12	4	227
SD5	0	3	0	0	0	0	175	0	1	0	179
SD6	0	0	0	0	0	0	0	150	0	0	150
SD7	0	8	0	3	3	0	1	7	146	7	175
SD8	0	7	0	7	0	1	0	5	3	164	187
SD9	178	182	177	183	181	182	181	179	174	180	1797
%	94.94	80.02	98.30	90.70	98.34	98.90	97.20	83.79	83.90	91.10	
	Average Perceptron Confusion Matrix										
	TD0	TD1	TD2	TD3	TD4	TD5	TD6	TD7	TD8	TD9	Sum
SD0	174	0	0	0	0	0	0	0	0	0	174
SD1	0	161	1	1	3	0	2	0	17	3	188
SD2	0	9	173	2	0	0	0	0	0	0	184
SD3	0	0	1	168	0	1	0	0	1	3	174
SD4	1	0	0	0	176	0	2	1	0	4	184
SD4	3	1	0	3	0	179	0	7	4	3	200
SD5	0	0	0	0	0	1	176	0	1	0	178
SD6	0	0	2	2	1	0	0	167	0	0	172
SD7	0	5	0	3	1	0	1	2	148	9	169
SD8	0	6	0	4	0	1	0	2	3	158	174
SD9	178	182	177	183	181	182	181	179	174	180	1797
%	97.75	88.46	97.74	91.80	97.23	98.35	97.23	93.29	85.05	87.77	

The Confusion Matrix clearly shows that the using Linear Perceptron Algorithm its difficult to classify digit 1 and for Averaged Linear Perceptron Algorithm its difficult to classify digit 8. This is concluded based

3. Polynomial and Averaged Polynomial Kernel Perceptron

	Kernel Perceptron Confusion Matrix										
	TD0	TD1	TD2	TD3	TD4	TD5	TD6	TD7	TD8	TD9	
SD0	178	0	0	0	0	1	0	0	0	0	179
SD1	0	182	4	0	7	2	0	1	7	1	204
SD2	0	0	170	0	0	0	0	1	0	0	171
SD3	0	0	0	176	0	1	0	0	0	2	179
SD4	0	0	0	0	172	1	2	0	0	0	175
SD5	0	0	0	2	0	175	0	0	0	1	178
SD6	0	0	0	0	0	0	178	0	0	0	178
SD7	0	0	1	0	0	0	0	173	0	0	174
SD8	0	0	2	5	2	0	1	1	166	2	179
SD9	0	0	0	0	0	2	0	3	1	174	180
Sum	178	182	177	183	181	182	181	179	174	180	1797
%	100.00	100.00	96.04	96.17	95.02	96.15	98.34	96.64	95.40	96.66	
	Average Kernel Perceptron Confusion Matrix										
	TD0	TD1	TD2	TD3	TD4	TD5	TD6	TD7	TD8	TD9	Sum
SD0	178	0	0	0	0	1	0	0	0	0	179
SD1	0	181	5	0	3	0	0	0	7	0	196
SD2	0	0	172	1	0	0	0	0	0	0	173
SD3	0	0	0	173	0	0	0	0	0	1	174
SD4	0	0	0	0	176	1	2	0	0	0	179
SD5	0	0	0	2	0	179	0	2	0	1	184
SD6	0	0	0	0	0	0	178	0	0	0	178
SD7	0	0	0	1	1	0	0	173	0	0	175
SD8	0	1	0	4	1	0	1	1	166	1	175
SD9	0	0	0	2	0	1	0	3	1	177	184
Sum	178	182	177	183	181	182	181	179	174	180	1797
%	100.00	99.45	97.17	94.50	97.23	98.35	98.34	96.64	95.40	98.33	

The Confusion Matrix clearly shows that both Polynomial Kernel Perceptron Algorithm and Averaged polynomial Kernel Perceptron Algorithm find digit 8 classification hard.

2.3.4 Overall Performance on Test Data

The overall performance of the the six models are computed and tabulated in Table 1.6. The Gaussian and Average Gaussian Kernel models has the least performance on the digit recognition data set with an accuracy of 69.61%. The best performance is observed through Averaged polynomial kernel model with an accuracy of 97.55%

Table 1.6

Model	Correctly Classified Samples	Test data Set Size	Accuracy %
Perceptron Algorithm	1648	1797	91.7
Avg. Perceptron	1680	1797	93.48
Polynomial Kernel Perceptron	1744	1797	97.05
Avg. Polynomial Kernel Perceptron	1753	1797	97.55
Gaussian Kernel Perceptron	1251	1797	69.61
Avg. Gaussian Kernel Perceptron	1251	1797	69.61

3. Results and Analysis using Support Vector Machines

3.1 Question E

Confusion Matrix for Linear Kernel SVM

Linear Kernel SVM Confusion Matrix											
	TD0	TD1	TD2	TD3	TD4	TD5	TD6	TD7	TD8	TD9	Sum
SD0	174	0	0	0	0	1	0	0	0	0	175
SD1	0	152	1	1	7	0	2	0	15	3	3
SD2	0	16	174	4	0	0	0	0	0	1	1
SD3	0	0	1	167	0	0	0	0	1	4	4
SD4	1	0	0	0	172	0	1	1	0	5	5
SD5	3	1	0	3	0	179	0	7	6	3	3
SD6	0	1	0	0	0	1	177	0	2	0	0
SD7	0	0	1	3	1	0	0	167	0	0	0
SD8	0	3	0	3	1	0	1	1	143	4	4
SD9	0	9	0	2	0	1	0	3	7	160	160
Sum	178	182	177	183	181	182	181	179	174	180	1797
%	97.75	83.51	98.30	91.25	95.02	98.35	97.79	94.41	82.18	88.88	

The confusion matrix says that the hardest digit to classify by the Linear Kernel SVM is 8 with an accuracy of 82.18%.

The Overall Accuracy of the Linear Kernel SVM Model is 1665/1797= 92.65%. Its improved by .95% when compared to Linear Perceptron Model.

Support Vectors count using Linear Perceptron with SVM

Model	Model for Digit1	Model for Digit1	Model for Digit2	Model for Digit3	Model for Digit4	Model for Digit5	Model for Digit6	Model for Digit7	Model for Digit8	Model for Digit9
Support Vectors Count	124	238	202	251	212	234	155	157	395	378

Confusion Matrix for Gaussian Kernel SVM

Gaussian Kernel SVM Confusion Matrix											
	TD0	TD1	TD2	TD3	TD4	TD5	TD6	TD7	TD8	TD9	Sum
SD0	0	0	0	0	0	0	0	0	0	0	0
SD1	0	0	0	0	0	0	0	0	0	0	0
SD2	0	0	0	0	0	0	0	0	0	0	0
SD3	178	182	177	183	181	182	181	179	174	180	1797
SD4	0	0	0	0	0	0	0	0	0	0	0
SD5	0	0	0	0	0	0	0	0	0	0	0
SD6	0	0	0	0	0	0	0	0	0	0	0
SD7	0	0	0	0	0	0	0	0	0	0	0
SD8	0	0	0	0	0	0	0	0	0	0	0
SD9	0	0	0	0	0	0	0	0	0	0	0
Sum	178	182	177	183	181	182	181	179	174	180	1797
%											

The Gaussian Kernel SVM yields weird results with the sigma taken as 0.5 which is the best optimal value that we have got on development data. The same behavior is observed in the Confusion Matrix of the Perceptron with Gaussian kernel.