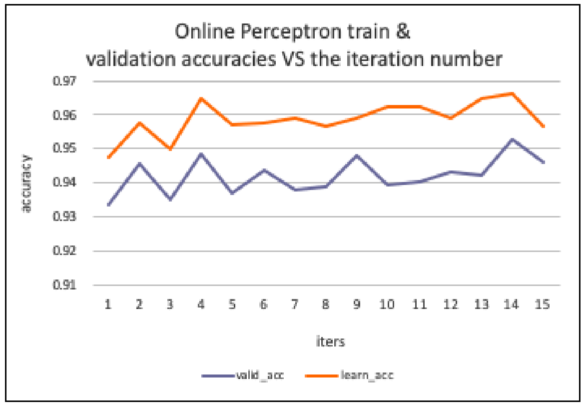
Perceptron algorithm for Optical Character Recognition

Jing Huang (33.3%)
Yuan-Hao Cheng (33.3%)
Shuhan Fan (33.4%)

Part 1 (20 pts) : Online Perceptron

a)

	v_acc/1	l_acc/1
1	0.933701657458563	0.947422258592471
2	0.94536525475752	0.957651391162029
3	0.934929404542664	0.949877250409165
4	0.948434622467771	0.96481178396072
5	0.936771025168815	0.956833060556464
6	0.943523634131369	0.957651391162029
7	0.937998772252916	0.958878887070376
8	0.938612645794966	0.956628477905073
9	0.947820748925721	0.959083469721767
10	0.939226519337016	0.962152209492635
11	0.940454266421117	0.962152209492635
12	0.942909760589318	0.959083469721767
13	0.942295887047268	0.964607201309329
14	0.952731737262124	0.966243862520458
15	0.94597912829957	0.956423895253682



No, it doesn't. Because the dataset is not linearly separable, the training accuracy couldn't reach to 100%. Also, the accuracy will go off when the model has a large margin by using the online perceptron.

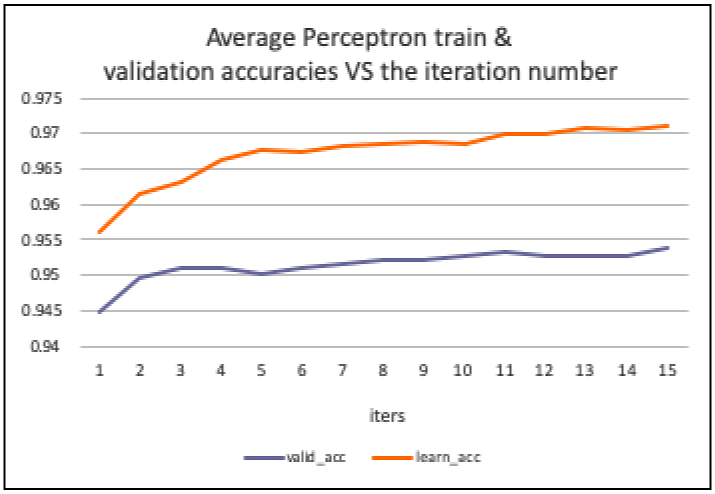
b) See the oplabel.csv file.

Part 2 (20 pts): Average Perceptron

a)

	v_acc/2	l_acc/2
1	0.944751381215469	0.956219312602291
2	0.949662369551872	0.961538461538461
3	0.950890116635973	0.96317512274959
4	0.950890116635973	0.966243862520458
5	0.950276243093922	0.967675941080196
6	0.950890116635973	0.967266775777414
7	0.951503990178023	0.968289689034369
8	0.952117863720073	0.968494271685761
9	0.952117863720073	0.968903436988543
10	0.952731737262124	0.968494271685761
11	0.953345610804174	0.969926350245499
12	0.952731737262124	0.969926350245499
13	0.952731737262124	0.970744680851063

14	0.952731737262124	0.970540098199672
15	0.953959484346224	0.970949263502455



b) The training accuracy and validation accuracy of the average model is performing better than the online perceptron. They have higher accuracies and smoother curves. Because the average model uses the average weight vector instead of using all weight vectors.

c) See the aplabel.csv file.

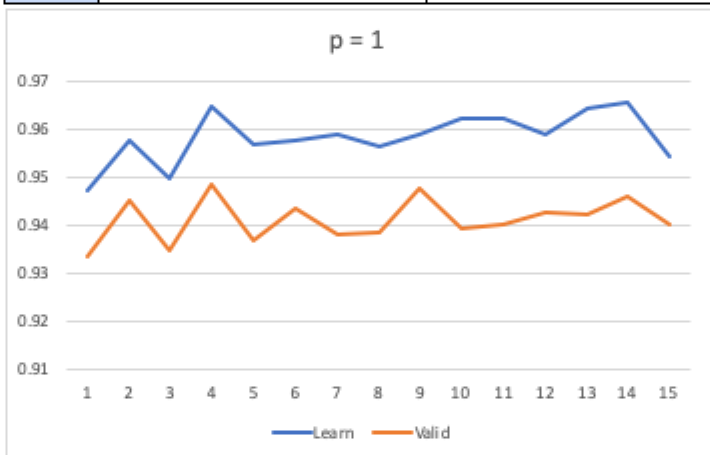
Part 3 (40 pts). Polynomial Kernel Perceptron

a)

1) P = 1

	l_acc/3	v_acc/3
1	0.947422	0.933702
2	0.957651	0.945365
3	0.949877	0.934929
4	0.964812	0.948435
5	0.956833	0.936771
6	0.957651	0.943524
7	0.958879	0.937999
8	0.956628	0.938613

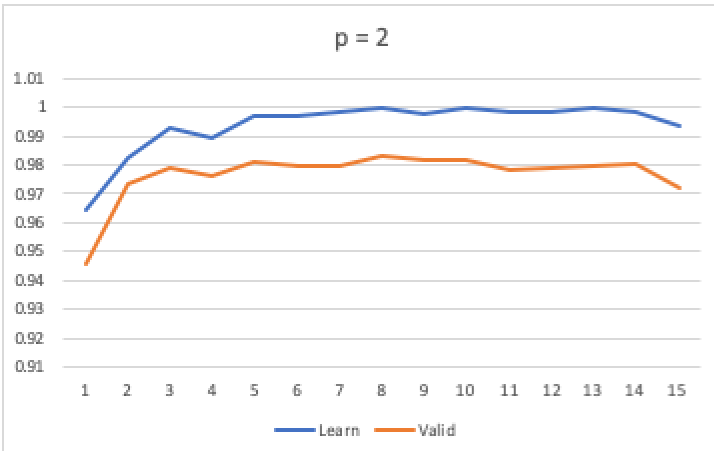
9	0.959083	0.947821
10	0.962152	0.939227
11	0.962152	0.940454
12	0.959083	0.94291
13	0.964607	0.942296
14	0.96563	0.945979
15	0.954583	0.940454



P = 2

	l_acc/3	v_acc/3
1	0.964403	0.945979
2	0.982406	0.973603
3	0.992635	0.979128
4	0.989157	0.976059
5	0.996727	0.98097
6	0.996727	0.979742
7	0.998363	0.979742
8	0.999591	0.983425
9	0.99775	0.981584
10	0.999591	0.981584

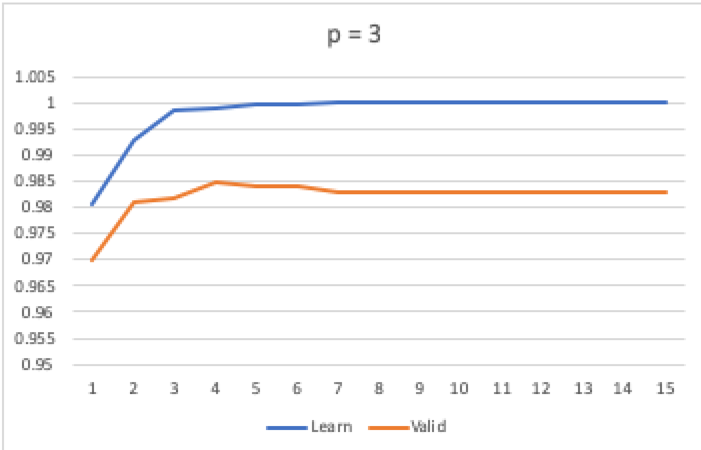
11	0.998773	0.978514
12	0.998773	0.979128
13	0.999591	0.979742
14	0.998363	0.980356
15	0.993249	0.972376



P = 3

	l_acc/3	v_acc/3
1	0.980769	0.96992
2	0.99284	0.98097
3	0.998363	0.981584
4	0.998977	0.984653
5	0.999795	0.984039
6	0.999591	0.984039
7	1	0.982812
8	1	0.982812
9	1	0.982812
10	1	0.982812
11	1	0.982812
12	1	0.982812

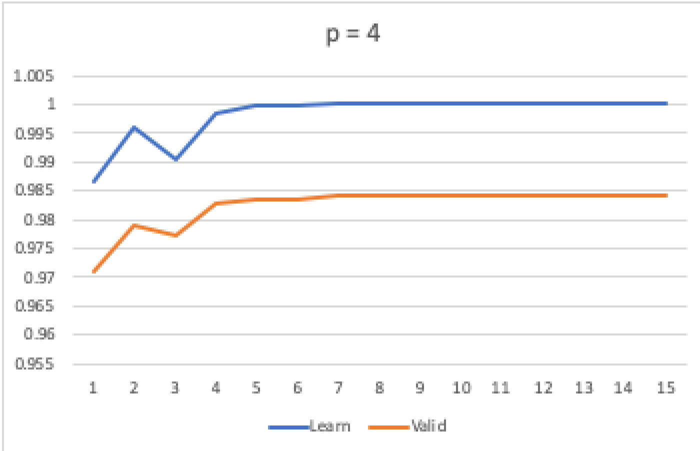
13	1	0.982812
14	1	0.982812
15	1	0.982812



P = 4

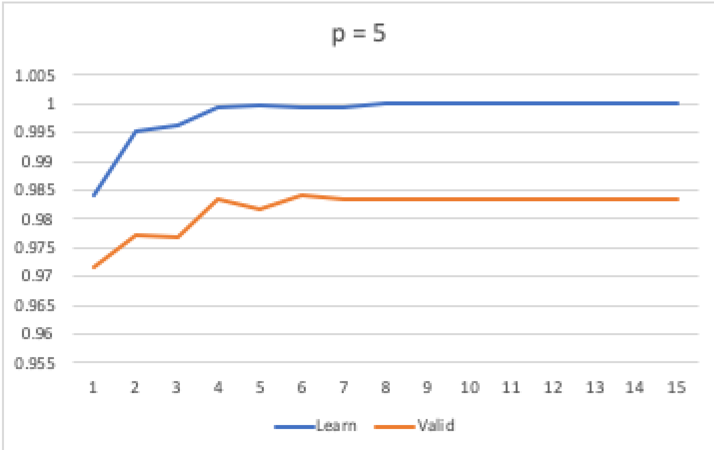
	l_acc/3	v_acc/3
1	0.9867021	0.971148
2	0.9961129	0.979128
3	0.9903846	0.977287
4	0.9983633	0.982812
5	0.9997954	0.983425
6	0.9997954	0.983425
7	1	0.984039
8	1	0.984039
9	1	0.984039
10	1	0.984039
11	1	0.984039
12	1	0.984039
13	1	0.984039
14	1	0.984039

15	1	0.984039



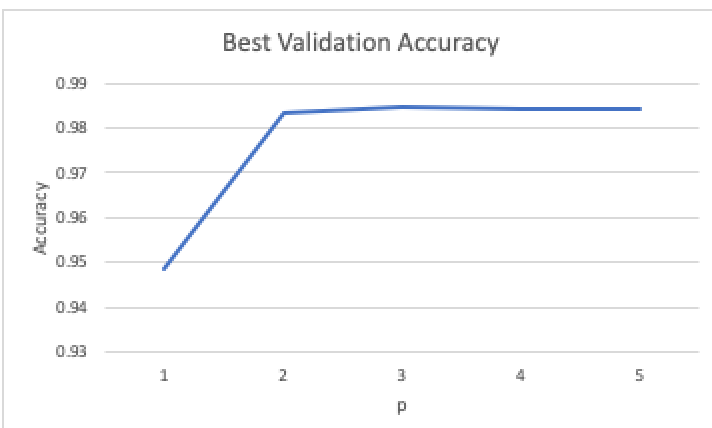
P = 5

	l_acc/3	v_acc/3
1	0.984247	0.971762
2	0.995295	0.977287
3	0.996113	0.976673
4	0.999386	0.983425
5	0.999795	0.981584
6	0.999386	0.984039
7	0.999386	0.983425
8	1	0.983425
9	1	0.983425
10	1	0.983425
11	1	0.983425
12	1	0.983425
13	1	0.983425
14	1	0.983425
15	1	0.983425



2)

p	Best Accuracy
1	0.948435
2	0.983425
3	0.984653
4	0.984039
5	0.984039



We are creating higher dimensional space while we are using higher p, but the accuracy will reach a limit when the p is high enough. To sum up, We can make the data linearly separable easier with an appropriate p. As a result, the performance of the algorithm will be boosted if we use a p which is high enough to reach the limit of accuracy.

b) See the kplabel.csv file.