Homework 3 Report

1. PERCEPTRON

The experimental data for perceptron classifier are given in the following table:

				perceptron experimental data					
				original word sequence			filter out stop words		
	learning rate	iterations	dataset	accuracy on ham	accuracy on spam	total accuracy	accuracy on ham	accuracy on spam	total accuracy
1	0.05	5	1	57.76%	96.15%	68.20%	78.45%	98.46%	83.89%
1	0.05	5	2	67.43%	100.00%	78.07%	87.95%	100.00%	91.89%
1	0.05	5	3	0.00%	100.00%	72.01%	5.26%	99.74%	73.30%
2	0.05	10	1	71.26%	97.69%	78.45%	97.99%	84.62%	94.35%
2	0.05	10	2	85.02%	91.95%	87.28%	93.49%	91.95%	92.98%
2	0.05	10	3	73.68%	96.16%	89.87%	82.89%	95.65%	92.08%
3	0.05	20	1	95.40%	88.46%	93.51%	97.99%	84.62%	94.35%
3	0.05	20	2	93.81%	88.59%	92.11%	93.49%	91.95%	92.98%
3	0.05	20	3	87.50%	94.63%	92.63%	83.55%	96.16%	92.63%
4	0.05	30	1	95.40%	88.46%	93.51%	97.99%	84.62%	94.35%
4	0.05	30	2	93.81%	88.59%	92.11%	93.49%	91.95%	92.98%
4	0.05	30	3	87.50%	94.63%	92.63%	83.55%	96.16%	92.63%
5	0.01	5	1	57.76%	96.92%	68.41%	76.44%	98.46%	82.43%
5	0.01	5	2	69.38%	99.33%	79.17%	90.23%	95.97%	92.11%
5	0.01	5	3	0.00%	100.00%	72.01%	68.42%	97.70%	89.50%
6	0.01	10	1	72.41%	97.69%	79.29%	97.99%	83.08%	93.93%
6	0.01	10	2	83.71%	95.97%	87.72%	94.46%	87.92%	92.32%
6	0.01	10	3	73.03%	96.16%	89.69%	84.21%	95.40%	92.27%
7	0.01	20	1	93.97%	93.08%	93.72%	97.99%	83.08%	93.93%
7	0.01	20	2	93.49%	91.28%	92.76%	94.46%	87.92%	92.32%
7	0.01	20	3	87.50%	95.14%	93.00%	84.21%	95.65%	92.45%
8	0.01	30	1	93.97%	93.08%	93.72%	97.99%	83.08%	93.93%
8	0.01	30	2	93.49%	91.28%	92.76%	94.46%	87.92%	92.32%
8	0.01	30	3	87.50%	95.14%	93.00%	84.21%	95.65%	92.45%
9	0.001	10	1	84.77%	98.46%	88.49%	97.70%	84.62%	94.14%
9	0.001	10	2	85.34%	98.66%	89.69%	95.11%	90.60%	93.64%
9	0.001	10	3	70.39%	96.68%	89.32%	84.21%	95.40%	92.27%
10	0.001	20	1	94.25%	91.54%	93.51%	97.70%	84.62%	94.14%
10	0.001	20	2	32.57%	99.33%	54.39%	95.11%	90.60%	93.64%
10	0.001	20	3	84.21%	95.65%	92.45%	84.21%	95.40%	92.27%
11	0.001	30	1	94.25%	91.54%	93.51%	97.70%	84.62%	94.14%
11	0.001	30	2	90.23%	93.29%	91.23%	95.11%	90.60%	93.64%
11	0.001	30	3	84.21%	95.65%	92.45%	84.21%	95.40%	92.27%
12	0.001	5	1	52.30%	99.23%	65.06%	77.01%	98.46%	82.85%
12	0.001	5	2	54.72%	97.32%	68.64%	75.24%	100.00%	83.33%
12	0.001	5	3	0.00%	100.00%	72.01%	63.82%	97.70%	88.21%
13	0.0002	10	1	80.17%	100.00%	85.56%	93.10%	93.08%	93.10%
13	0.0002	10	2	84.04%	98.66%	88.82%	93.49%	91.28%	92.76%
13	0.0002	10	3	70.39%	96.68%	89.32%	82.89%	95.65%	92.08%
14	0.0002	20	1	95.69%	89.23%	93.93%	96.26%	93.08%	95.40%
14	0.0002	20	2	91.21%	90.60%	91.01%	93.49%	91.28%	92.76%
14	0.0002	20	3	88.82%	93.61%	92.27%	83.55%	95.65%	92.27%
15	0.0002	25	1	95.69%	89.23%	93.93%	96.26%	93.08%	95.40%
15	0.0002	25	2	91.21%	90.60%	91.01%	93.49%	91.28%	92.76%
15	0.0002	25	3	88.82%	93.61%	92.27%	83.55%	95.65%	92.27%
16	0.0002	15	1	95.69%	89.23%	93.93%	96.26%	93.08%	95.40%
16	0.0002	15	2	86.32%	97.32%	89.91%	93.49%	91.28%	92.76%
16	0.0002	15	3	43.42%	98.72%	83.24%	83.55%	95.65%	92.27%
17	0.0006	5	1	53.45%	98.46%	65.69%	79.02%	98.46%	84.31%
17	0.0006	5	2	63.84%	100.00%	75.66%	86.97%	100.00%	91.23%
17	0.0006	5	3	0.00%	100.00%	72.01%	5.26%	99.74%	73.30%
18	0.0006	10	1	87.93%	96.92%	90.38%	97.70%	84.62%	94.14%
18	0.0006	10	2	79.15%	95.97%	84.65%	90.55%	94.63%	91.89%
18	0.0006	10	3	77.63%	96.16%	90.98%	82.89%	95.65%	92.08%
19	0.0006	20	1	94.25%	89.23%	92.89%	97.70%	84.62%	94.14%
19	0.0006	20	2	93.16%	85.91%	90.79%	91.53%	92.62%	91.89%
19	0.0006	20	3	84.87%	94.12%	91.53%	83.55%	96.16%	92.63%
20	0.0006	25	1	94.25%	89.23%	92.89%	97.70%	84.62%	94.14%
20	0.0006	25	2	93.16%	85.91%	90.79%	91.53%	92.62%	91.89%
20	0.0006	25	3	84.87%	94.12%	91.53%	83.55%	96.16%	92.63%

The table contains 20 suitable combinations of iteration and learning rate, considering both original word sequence and filter out stop words settings of ham/spam text, all on three datasets. The best performance for each iteration-learning rate combination of both 2 word sequence settings were given in red.

In order to compare with naive Bayes and logistic regression classifier, reports about the performance of these two classifier on both 3 datasets are given bellow:

multinomial naive Bayes experimental data

	original word sequence			filter out stop words		
dataset	accuracy on ham	accuracy on spam	total accuracy	accuracy on ham	accuracy on spam	total accuracy
1	96.84%	90.00%	94.98%	96.55%	86.92%	93.93%
2	97.39%	88.59%	94.52%	97.07%	87.92%	94.08%
3	88.82%	97.70%	95.21%	91.45%	98.98%	96.87%

logistic regression experimental data (ita=0.01, lambda=0.001, iteration=20)

	original word sequence			filter out stop words		
dataset	accuracy on ham	accuracy on spam	total accuracy	accuracy on ham	accuracy on spam	total accuracy
1	93.39%	96.92%	94.35%	97.13%	93.08%	96.03%
2	93.81%	96.64%	94.74%	94.79%	98.66%	96.05%
3	83.55%	100.00%	95.40%	87.50%	100.00%	96.50%

Analysis:

From the three tables, we can find that compared with naive Bayes and logistic regression, the perceptron classifier is more sensitive to stop words, we can see from the first table that after filtering out the stop words, perceptron classifier achieved best performance using less iterations for the same learning rate. While the influence of the stop words to the naive Bayes and logistic regression classifier are much less.

The reason behind this is that for naive Bayes and logistic regression classifier, the process of prediction or learning the classifier contains the comparison between P(y=1) and P(y=0), these will reduce the influence of stops words on the final prediction results. While in perceptron, weight vector is directly learned from training data and it will assign the stop words with big W because their frequent appearance in both ham and spam, it is bad for the prediction.

2. NEURAL NETWORK

The experimental data is given bellow. From the results, we can draw following conclusions:

1. suitable hidden nodes number can helps to get good classifier, two few hidden nodes contributes to poor performance, while large number of hidden nodes also harm the performance, but less severe than very small number of hidden nodes do.

2. small number of iterations is bad for classifier performance, it is because that the learning process is not convergent, while large number of iteration is less harmful when given small learning rate, because the weight values are vibrating close to the local optima for large number of iterations, so the performance will not fall significantly.

3. large learning rate is bad for performance, when small learning rate used, more iterations should be provided for ensuring converge to local optima.

4. momentum provided a faster way to make the learning process converges, if the learning rate is suitable (a small value), the influence of momentum to performance is slighter compared with the former 3 parameters.

5. filter out stop words improves the performance, but not significant as that shows in perceptron, showing that neural network is more powerful than perceptron.

neural networks experimental data (activation=logistic, batch_size=1, solver=sgd)

	#hidden units	iterations	learning rate	momentum	datase1	original sequence total accuracy	filter stop-words total accuracy
0	10	200	0.01	0.9	1	95.19%	95.61%
1	1	200	0.01	0.9	1	84.31%	94.77%
1	20	200	0.01	0.9	1	94.35%	95.61%
1	100	200	0.01	0.9	1	93.72%	96.03%
2	10	1	0.01	0.9	1	89.54%	91.84%
2	10	20	0.01	0.9	1	95.40%	95.82%
2	10	100	0.01	0.9	1	94.35%	95.61%
3	10	200	0.1	0.9	1	68.20%	94.77%
3	10	200	0.001	0.9	1	94.35%	95.61%
3	10	200	0.0001	0.9	1	94.14%	94.77%
4	10	200	0.01	0.001	1	94.56%	95.61%
4	10	200	0.01	0.1	1	94.35%	95.40%
4	10	200	0.01	0.4	1	94.35%	95.40%

3. COLLABORATIVE FILTERING

The following is the experiment data for collaborative filtering:

collaborative filtering experimental data

sample size	mean absolute error	root mean square error
1800	0.7446210608	0.943908057

The original test set is big and time consuming to run whole, so a random sample was done to help saving time while also show the performance of the collaborative filtering algorithm.

As the data shows, the mean absolute error is around 0.74 and the root mean square error is about 0.94.