머신러닝 모델들에 대한 성능 비교 보고서

이승태
반도체시스템공학과
2017313107

요 약

머신러닝이란 어떠한 작업(task)에 대해 경험(experience)을 통하여 성능(performance)를 높이는 컴퓨터 알고리즘의 연구이다. 4차 산업혁명이 진행됨에 따라 머신러닝에 대한 연구가 활발하게 진행되고 있다. 이 보고서에서는 모델들에 관련된 간단한 설명과 필자가 직접 만든 신경망 모델(MLP)과 전통적인 모델들(Naive Bayes, KNN, SVM)을 통해, 문서 분류(text categorization)를 이용하여 모델들의 성능을 비교하여 분석한다.

1. 서 론

최근 빅데이터, 머신러닝의 최적화 이론, 컴퓨터의 성능 등이 비약적으로 상승하면서 머신러닝의 학습 속도, 학습 능력이 같이 증가하게 되었다. 이에 따라 과거와 비교했을 때 소프트웨어 엔지니어들 사이에서 많은 관심을 받게 되었다. 실제로 얼굴인식, 음성인식, 주식시장 예측, 자율주행 자동차, 자연어 처리 등 다양한 분야에서 사용되고 있고 이는 먼 미래의 이야기가 아님을 알 수 있다.

대표적인 머신러닝 알고리즘으로는 다음과 같이 Naive Bayes, KNN, SVM, MLP 4가지를 예를 들 수 있으며. 간단하지만 현재까지도 많이 사용되고 있는 알고리즘이다. 먼저 이에 대한 간단한 설명과 차이점에 대하여 알아보도록 하겠다.

2. 머신러닝 모델들

(1) Naive Bayes

Bayes 법칙에 기반한 학습 알고리즘으로 각 데이터들의 특징(feature)과 출력(output)이 나올 확률 따라 값을 예측한다. Naive Bayes의 데이터들의 특징(feature)들이 서로 독립적이라는 가정을 안고 시작한다. 이로 인해 모델이 단순화되고 빠르게 작동하는 경향이 있지만 그만큼 에러가 발생할 수 있기 때문에 사용시에 주의가 필요하다.

(2) KNN

어떤 data의 vector에 대해 이 벡터와 가장 가까운 거리(보통 Euclidean distance를 사용한다.)에 있는 training data, k개를 찾아낸다. 그 중 가장 많이 label된 값을 선택하여 값을 예측하게 된다. 이는 새로운 데이터가 입력되었을 때만 계산을 진행하므로 'lazy learner' 라고 불리며, data가 들어 왔을 때, 모든 training data에 대해 검사를 진행하므로 느린 running time을 갖는다.

(3) SVM

support vector machine이라고 불리는 SVM은 margin을 최대한으로 늘리는 surface σ를 결정하며, 이 surface를 이용하여 data를 분류하는 머신러닝 기법이다. 이는 사용자가 parameter들을 tuning해줄 필요가 없고 적은 데이터로도 surface의 결정이 가능하며 성능 또한 매우 좋게 평가 된다.

(4) MLP

Neural Network의 일종으로, 여러 개의 hidden layer와 1개의 output layer를 갖는다. 이러한 성질로 인해 xor과 같은 non-linear한 값들도 분류가 가능하게 된다.

3. MLP와 전통적인 모델의 차이(이론)

(1) 신경망 (MLP) vs 통계 모델 (Naive Bayes)

Naive Bayes는 MLP에 비해 비교적 낮은 성능을 보인다. 하지만 Naive Bayes는 통계적인 모델로 모델 구축이 MLP에 비해 훨씬 더 간단하다. 결정해야 할 파라미터의 개수도 MLP에 비해 매우 적다고할 수 있다.

(2) 신경망 (MLP) vs 인스턴스 모델 (KNN)
MLP는 KNN과 비슷한 성능을 보이며, 결정해야 할 파라미터의 개수도 MLP에 비해 적다.

(3) 신경망 (MLP) vs 커널 모델 (SVM)
이론적인 측면에서 보았을 때, MLP는 SVM보다 결정해야 할 하이퍼파라미터의 개수도 적기 때문에 더 빠른 모델을 구축할 수 있다. 또한 SVM은 MLP에 비해 성능도 뛰어나며, 비교적 적은 training data에도 잘 작동한다.

4. text categorization을 이용한 성능측정

text categorization은 어떠한 document가 있을 때 그 document의 category를 예측하는 방법이다. 주어진 training data는 500개의 document이고 text data는 100개의 document이다. 이 데이터들을 이용하여 여러 머신러닝 모델들로 document의 category를 예측하였다.
각 모델의 능력을 평가하기 위해 bag of words, tf, tfidf를 사용하였으며, 여러 가지 하이퍼파라미터값을 변경해가며 accuracy를 측정하였다.

(1) 신경망 (MLP)
MLP에서 Learning rate라 불리는 알파 값과 hidden layer의 node 개수를 바꾸어 갔고 mini batch기법을 이용하여 accuaracy를 측정했다.
Learning rate가 0.003, 0.01, 0.03일 때, hidden layer의 node갯수가 30, 35, 40, 45, 50일 때 bag of words, tf, tfidf를 비교해보았다. 다음 표1. 2. 3과 그림 1, 2, 3은 각각 bag of words, tf, tfidf의 learning rate와 hidden layer의 node 개수에 따른 accuracy를 표와 차트로 나타낸 것이다.

	0.003	0.01	0.03
30	77%	71%	69%
35	73%	72%	71%
40	70%	73%	73%
45	74%	67%	71%
50	69%	73%	68%

표1. learning rate와 hidden layer의 node 개수에 따른 bag of word의 accuracy

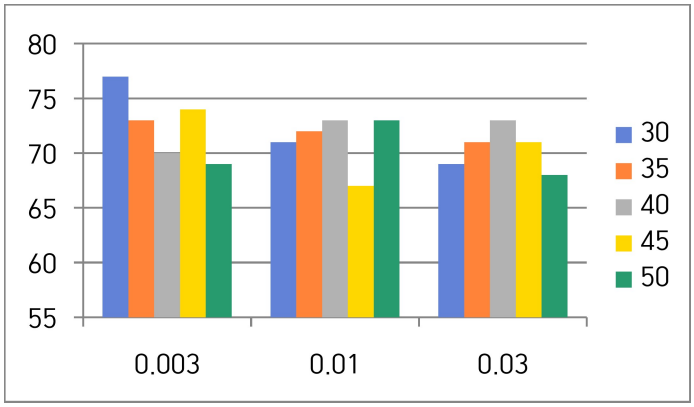

그림1. learning rate와 hidden layer의 node 개수에 따른 bag of word의 accuracy

	0.003	0.01	0.03
30	72%	75%	73%
35	71%	70%	74%
40	70%	71%	71%
45	71%	76%	74%
50	71%	72%	71%

표2. learning rate와 hidden layer의 node 개수에 따른 tf의 accuracy

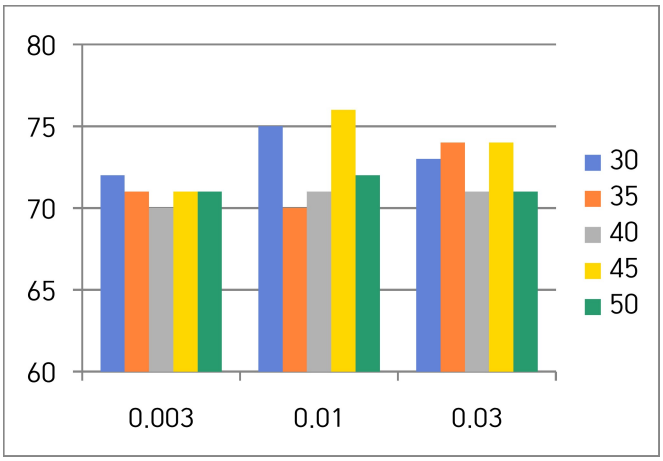

그림2. learning rate와 hidden layer의 node 개수에 따른 tf의 accuracy

	0.003	0.01	0.03
30	60%	65%	67%
35	59%	66%	65%
40	71%	69%	62%
45	65%	71%	66%
50	65%	66%	63%

표3. learning rate와 hidden layer의 node 개수에 따른 tfidf의 accuracy

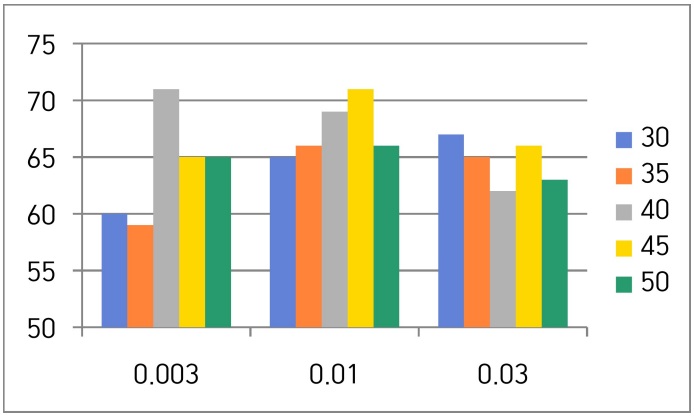

그림3. learning rate와 hidden layer의 node 개수에 따른 tfidf의 accuracy

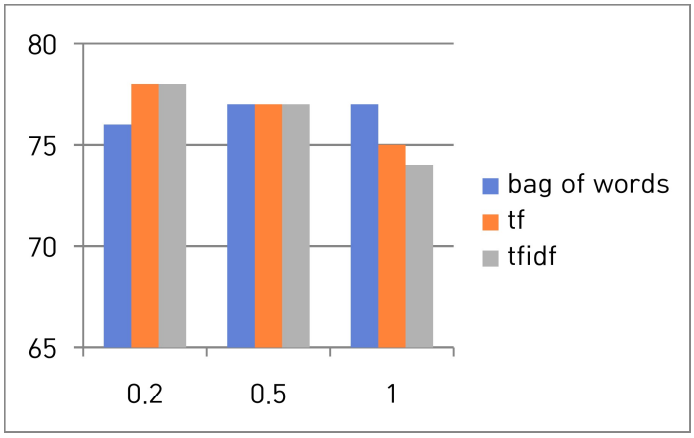

그림4. alpha 값에 따른 accuracy

평균적으로 tf가 가장 높은 성능을 보여주었지만, bag of words에서 learning rate가 0.003, hidden layer의 node개수가 30일 때 77%로 가장 높은 성능을 보여주었다.

Text categorization은 feature 개수가 상당히 많기 때문에 컴퓨터가 바꿔줘야 할 parameter(가중치)의 개수가 많다. 그러므로 한번 실행할 때마다 평균적으로 약 23분이 소요되었다. MLP는 local optima problem이 존재하기 때문에 global optimal을 찾기가 매우 힘들다. 또한 바꿔줘야 할 하이퍼파라미터도 많았기 때문에 최적화된 모델을 찾기가 매우 힘들었다.

(2) Naive Bayes

Naive Bayes에서 laplace smoothing을 이용하여 alpha의 값은 0.2, 0.5, 1로 세팅하였고 MLP와 마찬가지로 bag of words, tf, tfidf를 이용하여 성능을 측정하였다.

다음 표4는 alpha값에 따른 accuracy표 이다.

	bag of words	tf	tfidf
0.2	76%	78%	78%
0.5	77%	77%	77%
1	77%	75%	74%

표4. alpha 값에 따른 accuracy

대체적으로 alpha값이 감소할수록 accuracy는 증가했다. alpha값이 0.2일 때 tf와 tfidf에서 가장 높은 성능을 나타내 주었다.

Naive bayes는 모델을 설계하고 계산하는 과정이 많이 복잡하지 않아 평균 1분이라는 빠른 시간에 결과값을 도출했다.

(3) KNN

KNN에도 마찬가지로 bag of words, tf, tfidf를 사용하여 성능을 측정하였고, K값을 5, 11, 15로 바꾸어가며 성능을 측정하였다. 또한 거리를 재는 방법으로, Euclidean distance, cosine similarity가 있는데 이 두 방법을 통해 성능을 비교해 보았다.

다음 표5,6와 그림 5,6는 각각 cosine similarity와 Euclidean distance를 사용한 K값과 에 따른 accuracy 표와 차트 이다.

	bag of words	tf	tfidf
5	67%	58%	67%
11	71%	70%	68%
15	74%	71%	73%

표5. cosine similarity를 이용한 K값에 따른 accuracy

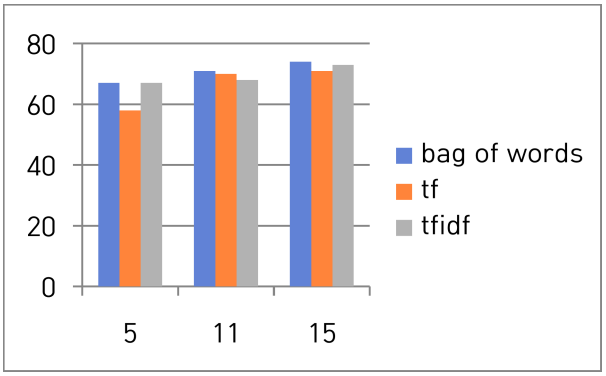

그림5. cosine similarity를 이용한 K값에 따른

accuracy

	bag of words	tf	tfidf
5	31%	42%	65%
11	40%	49%	72%
15	38%	41%	75%

표6. Euclidean distance를 이용한 K값에 따른 accuracy

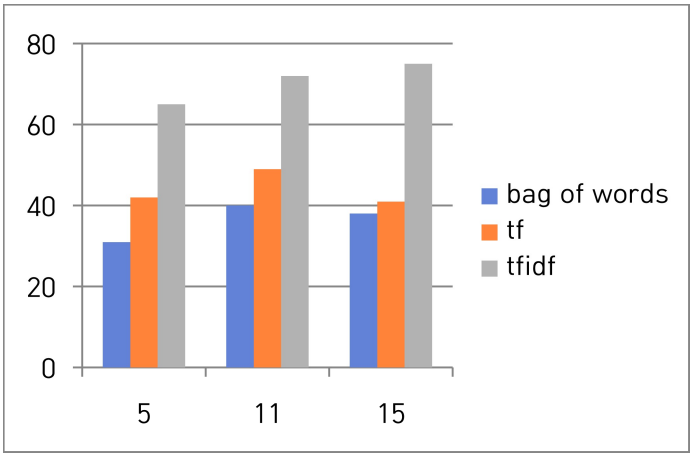

그림6. Euclidean distance를 이용한 K값에 따른 accuracy

대체적으로 K값이 증가할수록 accuracy가 증가했지만 K값이 어느 정도 커지면 증가를 멈추고 감소할 것이다. 대체적으로 cosine similarity가 평균적으로 훨씬 좋은 성능을 보여주지만 예외적으로 tfidf에서는 Euclidean distance가 더 높은 성능을 보여주었다. 가장 좋은 성능을 가진 모델은 Euclidean distance와 tfidf, K값은 15를 선택한 모델로 accuracy는 75%가 나왔다.

이 모델들은 train에서 아무것도 하지 않지만 다른 모델과 달리 test하는데 시간이 오래 걸린다.(lazy learner) 평균적으로 1번 학습하고 테스트하는데 까지 5분의 시간이 소요되었다.

(4) SVM

SVM은 linear하게 값을 나누는 방법과 gaussian distribution 즉, kernel을 이용하여 값을 나누는 방법이 존재하는데 여기서는 feature갯수가 10000개 이상이고 data의 개수는 500개 정도 되어서 feature개수가 data의 개수보다 훨씬 많으므로 linear모델을 사용하게 되었다. 여기서 하이퍼파라미터인 C값을 조절하여 모델의 accuracy를 조절할 수 있는데 C값이 클수록 모델이 overfiting될 가능성이 약간 존재하고 작을수록 underfitting될 가능성이 약간 존재한다. 여기서는 C값을 각각 1, 10, 100으로 설정하여 성능을 비교해보았다.

다음 표7은 C값에 따른 accuracy값의 비교이다.

	bag of words	tf	tfidf
1	79%	76%	79%
10	79%	76%	80%
100	79%	76%	80%

표7. C값에 따른 accuracy

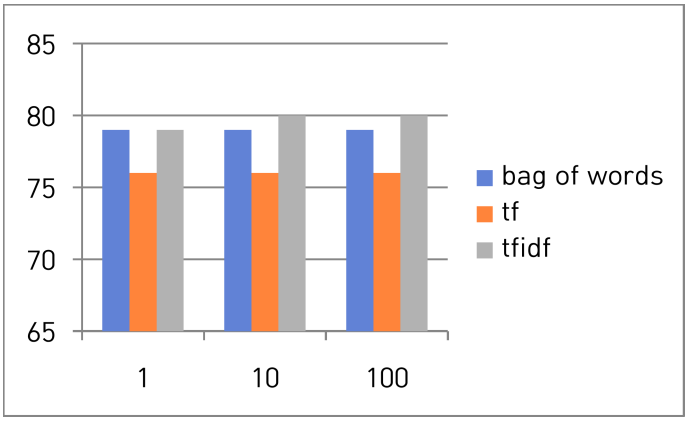

그림7. C값에 따른 accuracy

C값을 바꾸어 성능을 측정해 보았지만 accuracy는 거의 비슷한 값을 보여주었다. C값을 매우 높여 1000000의 값도 측정해 보았지만 C가 10, 100일 때와 차이를 보이지 않았다. 가장 높은 성능을 보인 모델은 tfidf, C값은 10, 100이었다.

시간은 매우 빠르게 측정되었는데 3초 미만정도로 측정이 되어 매우 빠른 train시간을 보여주었다. 또한 우리가 이 모델을 사용해야할 때 결정해줘야 할 parameter은 C값 1개 밖에 없다. 그것도 C값의 차이에 대해 accuracy 변화가 거의 없으므로 우리는 이 모델의 하이퍼파라미터를 결정하는데 많은 시간을 들일 필요가 없다는 것을 알 수 있다.

5. text categorization을 이용한 성능비교

(1) 신경망 (MLP) vs 통계 모델 (Naive Bayes)

Naive Bayes가 MLP에 비하여 학습 속도가 훨씬 더 빨랐고, 하이퍼파라미터의 개수도 Naive Bayes가 훨씬 적어 사람의 손이 덜 가는 모델이었다. 평균적인 accuracy는 Naive Bayes가 더 좋았다. 하지만 이는 data의 개수가 너무 적고 MLP의 global minimum을 찾기가 굉장히 어려워서 이러한 결과를 보여줬다고 생각한다.

(2) 신경망 (MLP) vs 인스턴스 모델 (KNN)
평균적인 accuracy는 cosine similarity로 비교하면 비슷하지만 Euclidean distance를 사용할 때에는 MLP가 더 좋은 accuracy를 보여주었다. 이는 기존 이론과 매우 비슷한 모습이다. 학습시간은 KNN도 길긴 했지만 MLP가 더 긴 학습시간을 보여주었다. 또한 KNN에서 사람이 결정 해줘야 하는 값은 K로 MLP보다 단순하게 모델을 결정해줄 수 있었다.

(3) 신경망 (MLP) vs 커널 모델 (SVM)
평균적인 accuracy는 SVM이 좋은 성능을 보여주었다. 마찬가지로 SVM은 적은 데이터로도 우수한 성적을 보여준다는 이론과 매우 유사한 모습을 보여주었다. 또한 MLP와 SVM은 학습시간도 현저하게 차이가 난다. 결정해야 할 하이퍼파라미터도 없으므로 사람이 모델을 만들 때 거의 개입을 하지 않으므로 훨씬 더 빠르게 모델을 구축할 수 있었다.

6. 결론

여러 가지 모델들을 비교해 보았을 때, SVM이 가장 성능도 좋고, 결정해줄 하이퍼파라미터도 없어서 빠르게 모델 구축이 가능했다. 나머지 MLP, KNN, Naive Bayes모델에 대해서는 모델을 구축하는데 MLP, KNN, Naive Bayes순으로 오래 걸렸다. 또한 이 세 모델의 성능 차이는 거의 없었다. 이는 data의 개수가 적기 때문에 발생한 문제라고 생각한다. 데이터의 개수를 늘리게 된다면 세 모델에 대해 성능 비교를 더 확실히 할 수 있을 것으로 예상된다.