Scikit-learn을 이용한 필기체 숫자 데이터의 알고리즘 별 성능분석

2017029616 박병현

-초록

이 실험은 Sckit-learn에서 제공하는 필기체 숫자 데이터를 활용하여 세가지 알고리즘의 성능과 정확도를 분석하고, Training data의 개수가 정확도에 미치는 영향을 파악합니다. Training data 및 Testing data는 dataset.load_digits() 메소드를 사용하였습니다.

-문제 정의:

-Scikit-learn에서 제공하는 메소드를 통해 주어진 필기체 숫자 데이터를 학습하고 예측하는 문제를 진행합니다.

-dataset.load_digits() 메소드를 통해 필기체 숫자에 대한 샘플데이터를 확보합니다

-parameters

1) n_class :int, default=10

-class의 종류의 개수를 설정합니다. default는 0~9까지 10개의 class를 가짐. 이번 과제에서는 default값을 사용합니다.

2) return_X_y: bool, default = False

-return값을 data와 target값을 tuple로 return할지 혹은 하나의 객체로 묶은 Bunch object로 return할지를 결정하는 값입니다.. 이번 과제에서는 default값을 사용합니다.

3) as_frame: bool, default = False

-이번 과제에서는 default값을 사용합니다.

-return

Bunch object: 과제에 필요한 attribute을 담고 있는 객체입니다.

attributes

1.data: (1797,64)의 ndarray. 8x8에 해당하는 각 숫자데이터의 픽셀 값이 64의 길이를 가진 1차원 배열로 변경됨. 그러한 1차원 배열이 1797개 존재합니다.

2.target: (1797, 1)의 ndarray. 각 데이터가 어떤 class에 해당하는지에 대한 정보를 담습니다.

-class 별 target값의 개수: 개수(target 값)

178(0), 182(1), 177(2), 183(3), 181(4), 182(5), 181(6), 179(7), 174(8), 180(9)

-알고리즘 종류: 이번 실험에서 다음과 같은 알고리즘들에 대해 실험을 진행합니다.

1) SVM (Support Vector Machine) 알고리즘

-데이터 공간에서 가장 가까운 데이터와 가장 큰 폭으로 분류되는 경계를 찾는 알고리즘이다

-Gamma 값을 통해 거리에 따른 데이터들의 가중치를 결정합니다.

-C 값을 통해 Training data에 얼마나 fitting 할지를 결정합니다.

2) Logistic Regression: 분류를 통해 특정 클래스에 속할 확률을 결정하는 모델

-max_iter parameter가 존재합니다. 해당 횟수 이내에 수렴하지 않는다면 Error message를 보냅니다.

-Error message를 없애기 위해 max_iter값을 10,000으로 설정하였습니다.

3) KNN(K-Nearest Neighborhood) 알고리즘: 가까운 K개의 데이터의 값을 계산하여 예측하는 알고리즘

-실험

1) 학습 알고리즘 별 학습 소요시간

-학습 알고리즘 별로 샘플 데이터들을 학습하는데 시간이 얼마나 걸리는 지 측정하고 비교합니다.

-898개의 데이터를 학습하는 시간을 100회 측정 후, 평균 값을 결과값으로 합니다.

-학습 소요시간의 단위는 second(초)입니다.

2) 학습 알고리즘 별 정확도

- 각 알고리즘이 전체 샘플 중 정확하게 예측한 값의 비율인 정확도를 비교합니다.

- 597개의 testing data를 사용합니다

2) 학습량 별 정확도

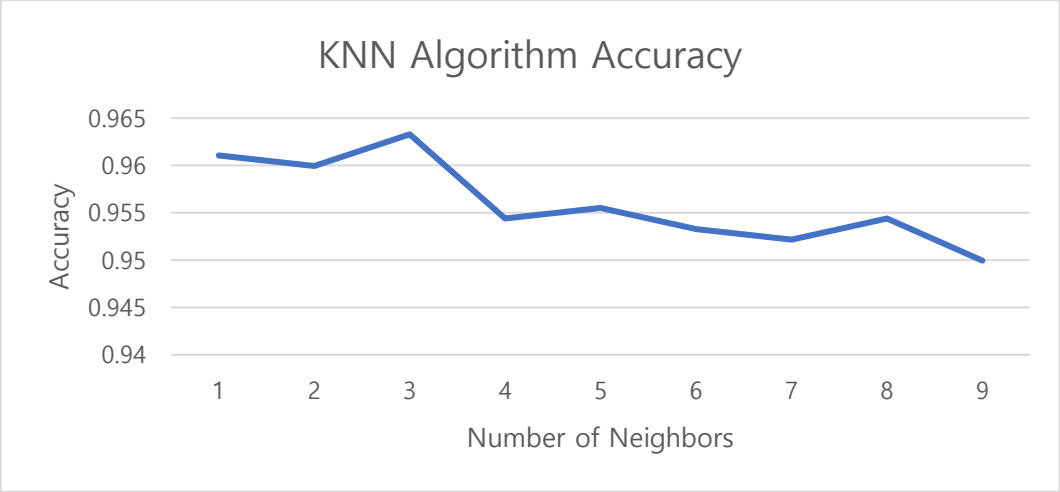
-Training data의 개수를 300, 600, 900개로 다르게 하며 학습량 별 정확도가 어떻게 변하는지 확인합니다.

-597개의 testing data를 사용합니다

-실험 결과

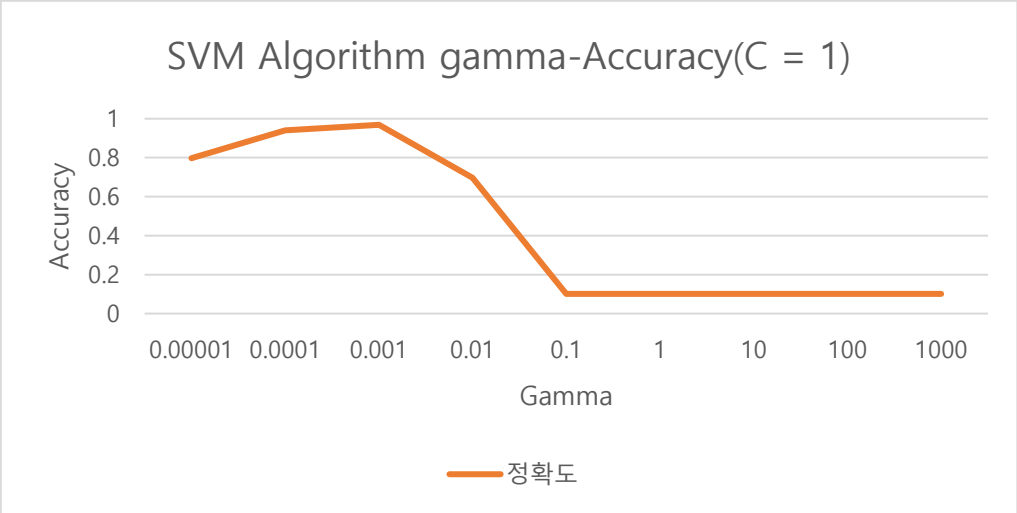
사전 실험) 각 알고리즘 별로 가장 높은 accuracy를 도출하는 parameter을 찾기

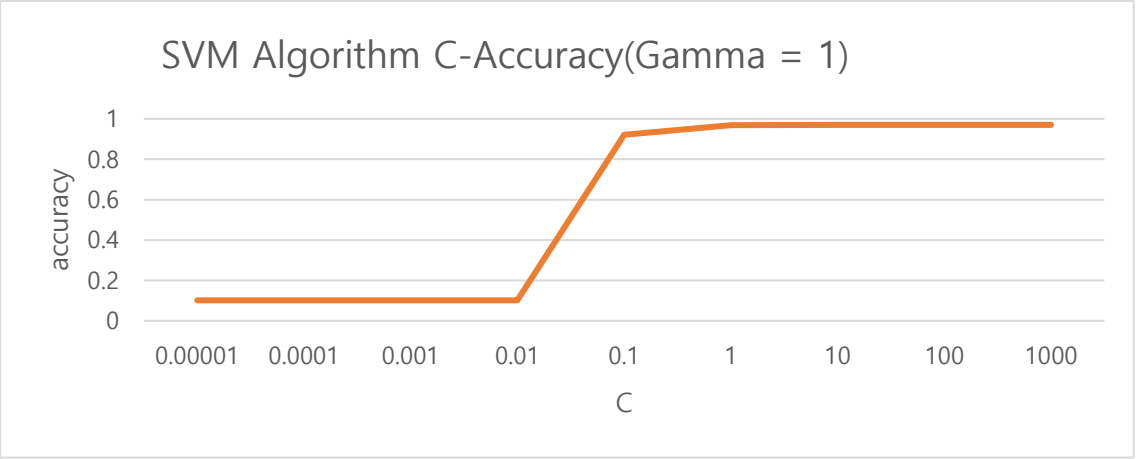
-KNN: 가장 가까운 data들의 개수를 지정할 n_neighbors parameter의 최적값 찾기



⇨ n_neighbors = 3일 때 가장 높은 정확도를 가짐

-SVM 알고리즘의 C, gamma parameter의 최적값 찾기

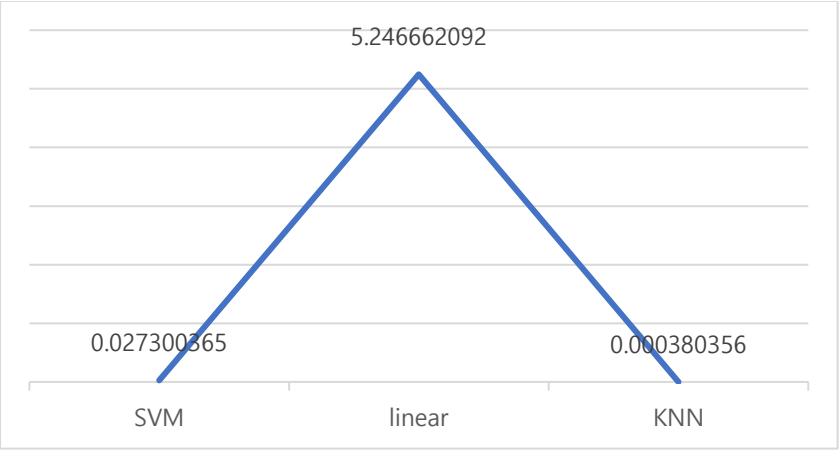




⇨ Gamma = 0.001, C = 10 에서 가장 높은 정확도를 가집니다.

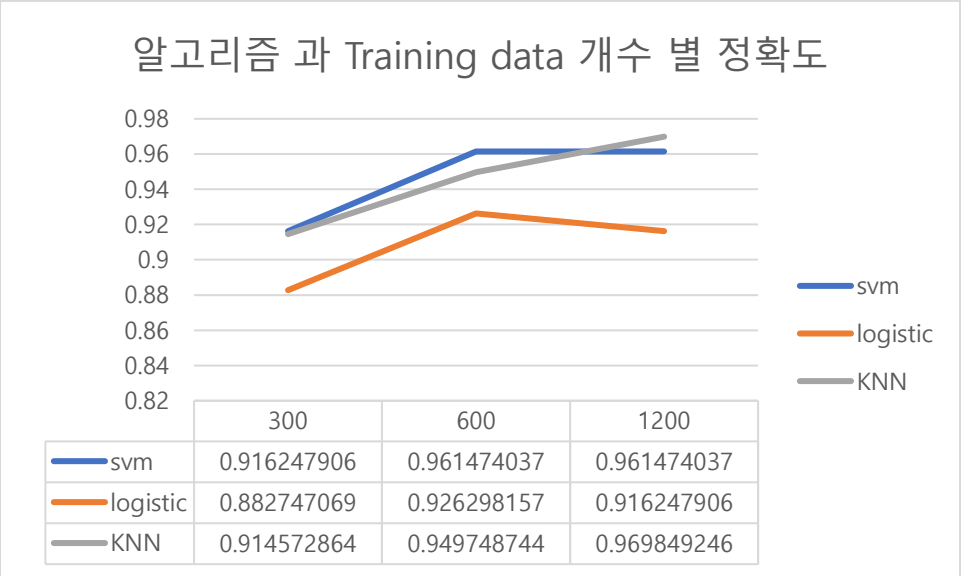
실험 1) 학습 알고리즘 별 학습 소요 시간

- SVM 알고리즘의 학습 소요시간: 0.027300364971160 89 sec
- logistic regression: 5.246662092208862 sec
- KNN: 0.0003803563117980957 sec



분석: 저는 KNN알고리즘의 속도가 느릴 것이라고 예측했습니다. 하지만 이번 실험에서 logistic regression이 가장 느린 결과가 나왔습니다. 이는 w값이 수렴하기 위해 시행하는 gradient descent 횟수를 10000번으로 설정했기 때문에 위와 같은 결과가 나왔다고 분석했습니다.

실험 2,3) 학습 알고리즘 및 Training data 개수 별 정확도



	300	600	1200
svm	0.916247906	0.961474037	0.961474037
logistic	0.882747069	0.926298157	0.916247906
KNN	0.914572864	0.949748744	0.969849246

분석: logistic regression의 정확도가 가장 낮고, SVM과 KNN은 유사한 정확도를 가지게 나왔습니다. 또한 학습량이 많아질수록 전반적인 정확도가 상승하였습니다. 특히 600->1200보다 300->600개로 증가할 때 더욱 가파른 상승치를 보였습니다. 이는 300개의 data개수는 이번 실험 데이터의 학습에 있어서 매우 부족한 양이라는 것을 알 수 있습니다. 하지만 logistic regression의 경우 1200개의 training data를 학습하게 될 경우 정확도가 더 낮아지는 결과가 나왔습니다. 이는 Training data에 과하게 overfitting 되었기 때문에 이와 같은 결과가 도출되었다고 분석하였습니다.

결론:

Scikit-learn에서 주어진 필기체 숫자 데이터에 대해서 학습 속도는 logistic regression이 가장 느렸고, KNN 및 SVM으로 이어졌습니다. 이는 logistic regression 방식으로 학습 시 수렴하는 값을 찾는데 많은 반복이 필요하기 때문입니다. KNN은 모든 데이터들에 대해 거리를 계산하기 때문에 계산량이 SVM보다 많아 상대적으로 더 오랜 시간이 걸렸습니다.

정확도 측면에서 역시 logistic regression이 가장 낮았습니다. SVM과 KNN은 training data의 개수에 따라 정확도 순위가 달라졌지만, 전반적으로 유사한 정확도를 보였습니다. 하지만 Training data의 개수에 따라 정확도가 달라지는 것으로 보아, 더 많은 Training 및 Testing data가 존재했다면 다른 결과가 나올 수 있다고 예상해 볼 수 있습니다. Logistic regression의 경우 학습 데이터가 과도하게 많아지면 Overfitting이 일어나 정확도가 감소하는 현상이 나타났습니다.