인공지능

k-Nearest Neighbor 알고리즘을 사용한 Iris classification

B511086 백성진

1. 과제에서 사용된 k-Nearest Neighbor 알고리즘 소개
2. 테스트 결과

1. 과제에서 사용된 k-Nearest Neighbor 알고리즘 소개

본 과제에서 사용한 k-Nearest Neighbor(이하 knn)알고리즘에서 사용한 방법은 다음과 같습니다.

테스트 데이터 y와 훈련용 데이터 x의 모든 원소들의 거리를 구합니다.
이때, 거리를 구하기 위해 사용한 방법은 유클리디안 거리 계산 방법입니다.
거리를 계산한 후, 미리 준비한 딕셔너리가, 계산된 거리를 Key로 가지고 Value로 x의 인덱스를 가지도록 저장합니다.

훈련용 데이터 x와의 거리 계산을 마친 후, 기록된 딕셔너리를 Key를 피봇으로 하여 오름차순 정렬합니다. 정렬 함수를 마친 후 반환되는 값을 튜플을 원소로 가지는 리스트로 받아, 리스트의 0번째 원소가 y와 가장 짧은 유클리디안 거리를 가지도록 합니다.

리스트에는 훈련용 데이터 x의 길이만큼의 데이터가 들어있습니다. 이를 사전에 지정한 k개의 원소만 취합니다. 즉, k가 3이라면, 리스트에서 0, 1, 2 까지만 가지도록 설정합니다.

이제 k개의 가장 근접한 원소가 무엇인지 알았으므로, majority vote 방식을 사용하든, weighted majority vote 방식을 사용하든 전혀 무리가 없습니다.

우선, majority vote 방식을 사용하기 위해서는 k번 루프를 돌아야 합니다.
루프 안에서, 새로운 리스트의, 근접 리스트의 각 원소가 가지는 class번째 원소의 값을 1씩 증가 시킵니다. 결과적으로 루프를 탈출한 후, 가장 많은 투표를 받은 꽃의 종이 분류된 iris의 종이 됩니다.

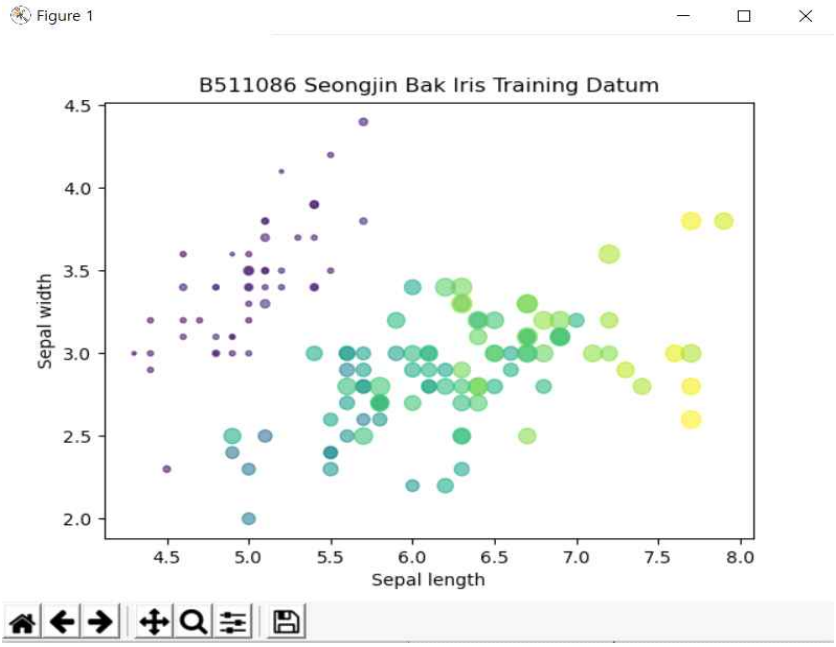
weighted majority vote 방식을 위해서는 가중치 값을 구하는 수식을 우선 정해야 합니다. 본 과제에서, weight는 실제 거리의 역수의 로그 값을 가집니다. 가까운 거리를 가지는 것 일수록 높은 가중치 값을 가지는데, 이에 로그 값을 취한다면 0과 1 사이로 정규화 되므로 파격적인 가중치를 가지는 것을 방지할 수 있습니다.
majority vote 방식과 비슷하게 루프를 돌 때, class 마다 1씩 증가시키는 방식 대신, weight를 곱한 후 그 값을 더하는 방식을 채택했습니다.

2. 테스트 결과

2.1 데이터 도표

* 데이터들은 x 축에 sepal_length, y 축에 sepal_width 를 지정하였으며, 점의 크기는 petal_width 의 스칼라 값 만큼 크게, 색은 petal_length 스칼라 값의 색상을 지정하였습니다.

2.1.1 트레이닝 데이터 산포도



2.1.2 테스트 데이터 산포도