GNU

Lab Seminar: 2024. 02.02

Advanced Learning Algorithms

(Advice for applying machine learning & Decision trees)

IDEALAB
Improving lives through learning

Su-Gyeong Ban
School of Computer Science
Gyeongsang National University (GNU)

GNU 경상국립대학교 Gyeongsang National University

Contents

- Feedback
- Evaluating a model
- Bias and variance
- Precision and Recall
- Decision trees
- Random forest

Feedback (1/5)

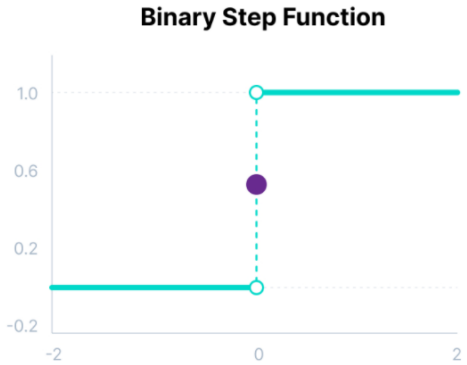
- **Neural Networks**
  - 단점 : 논리적이지 않음
    - 강의 자막 해석의 문제
    - "**이 강의에서 신경망에 대해 가르치는 이유는 신경망이 다양한 기계 학습 문제에 매우 잘 작동하기 때문이지 논리적 동기 때문이 아님**" 이라는 내용이 "신경망은 어려운 머신러닝 문제에서 잘 작동하지만 논리적이지 않음" 이라고 잘 못 해석되었음

Feedback (2/5)

- **Activation Functions (1/4)**
  - Single layer perceptron 다음 뉴런으로 신호 전달 유무 결정
  - Multi layer perceptron 다음 뉴런으로 보낼 신호의 강도 결정
    - 전달하는 신호의 강도를 정하는 방법이 활성화 함수
    - 활성화 함수는 훈련 과정에 계산량 多, 역전파에서도 사용됨으로 연산에 대한 효율성 중요

Feedback (3/5)

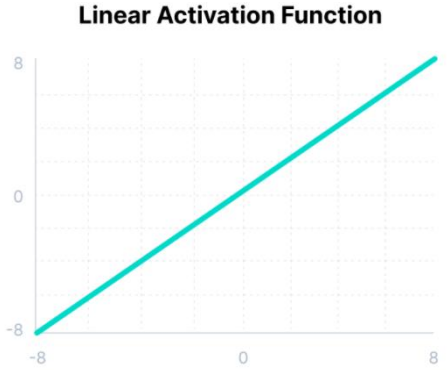
- **Activation Functions (2/4)**
  - Activation의 분류 : Binary step function, Linear activation function, Non-linear activation function
    - **Binary step function**
      - 임계치를 기준으로 출력해주는 함수
      - 다중 값 출력 할 수 없어 다중 클래스 분류 문제에 사용 X
      - 함수의 기울기가 0이기 때문에 역전파 과정에서 문제가 됨



$$f(x) = \begin{cases} 1 \; for \; x & (x \geq 0) \\ 0 \; for \; x, & (x < 0) \end{cases}$$

Feedback (4/5)

- **Activation Functions (3/4)**
  - **Linear activation function**
    - 다중 출력이 가능하다는 장점, 두 개의 문제점 존재
    - 문제점 1. 역전파 사용 불가능
      - 역전파 : 활성함수를 미분하여 이를 이용해 손실값을 줄이기 위한 과정
      - 선형함수의 미분값은 상수이기에 입력값과 상관없는 결과를 얻음
      - 그렇기 때문에 예측과 가중치 사이의 상호관계에 대한 정보를 얻을 수 없음
    - 문제점 2. 은닉층을 무시하고, 얻을 수 있는 정보 제한



Feedback (5/5)

- **Activation Functions (4/4)**
  - **Non-linear activation function**
    - 두 종류 함수의 단점 때문에 활성화 함수는 비선형 함수를 주로 사용
    - 입출력간의 복잡한 관계를 만들어 입력에서 필요한 정보를 얻음
      - 비정형적인 데이터에 특히 유용
    - 비선형 활성화 함수의 장점
      - 입력과 관련있는 미분값을 얻으므로 역전파를 가능하게 함
      - 출력이 여러 레이어를 통과하는 입력의 비선형 조합이 되므로 뉴런의 여러 레이어를 쌓을 수 있음

Evaluating a model (1/2)

- **학습 알고리즘 성능 향상 방법**
  - 더 많은 학습 데이터 셋 학보
  - 피처 셋 감소 : 과적합 방지
  - 추가적인 피처 설계
  - 고차 다항식 추가
  - 정규화 파라미터 람다 값 바꾸기

Evaluating a model (2/2)

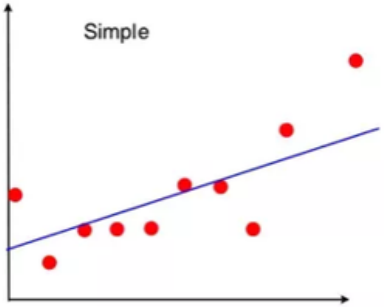
- **Train/Validation/Test Sets**
  - Train : 60%, Validation : 20%, Test : 20%
    - 학습 셋으로 모델을 학습함
    - 교차 검증 셋으로 모델을 선택함
    - 테스트 셋으로 모델을 평가함

Bias & Variance (1/6)

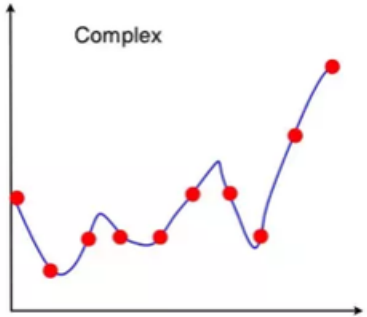
- **What is bias?**
  - 모델을 통해 얻은 예측 값과 실제 값 사이의 차이
    - 편향이 높다면, 과소 적합을 야기 함
- **What is variance?**
  - 주어진 데이터로 학습한 모델이 예측한 값의 변동성
    - 분산이 크면, 과대 적합을 야기 함

Bias & Variance (2/6)

- **bias**
  - 왼쪽 그래프의 예측 값과 실제 값 차이는 오른쪽 그래프보다 큼
  - 오른쪽 그래프의 예측 값과 실제 값의 차이는 0으로 편향이 0임
- **variance**
  - 왼쪽 그래프가 오른쪽 그래프보다 더 작음
  - 왼쪽 그래프는 일반화가 잘되어 있어 예측 값이 일정한 패턴을 나타냄



high bias, low variance



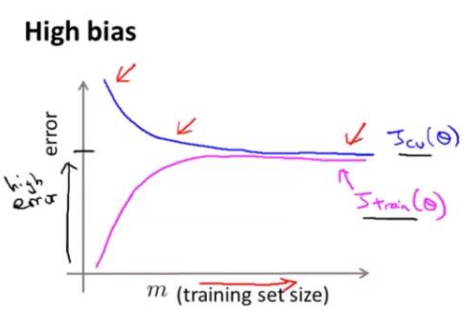
low bias, high variance

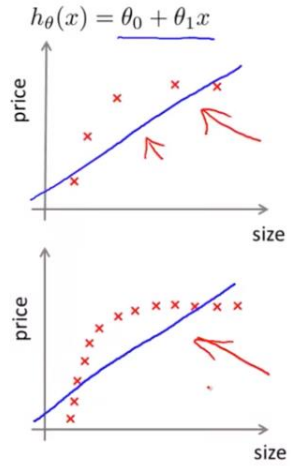
Bias & Variance (3/6)

- **Regularization and bias/variance**
  - 편향과 분산 사이의 균형을 유지하여 모델의 일반화 성능을 향상시키는 역할
    - 1. 정규화 파라미터 람다가 10000으로 매우 큰 경우
      - 모든 파라미터는 막대한 페널티를 받아 가설 $h_\theta(x) = \theta_0$ 과 거의 같음
      - 가설 그래프는 다소 평평한 일직선
      - 편향이 높고 데이터 셋에 적합하지 않음
    - 2. 정규 파라미터 람다가 0으로 매우 작은 경우
      - 정규화 항의 값은 0이 되고 모든 파라미터의 값은 그대로 유지
      - 학습 데이터를 모두 만족하는 곡선
      - 분산이 높고 데이터 셋에 과적합 함

Bias & Variance (4/6)

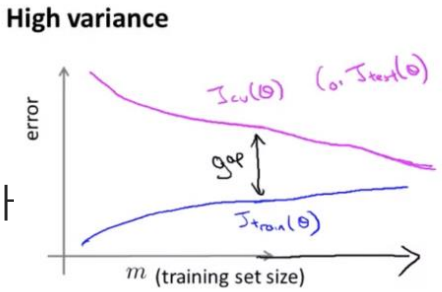
- **Learning curves (1/2)**
  - **High bias**
    - 교차검증 오류 : 학습 셋이 커질수록 오류가 줄어들고 특정한 수의 학습예제에 도달하면 그래프가 팽팽해짐
    - 학습 오류 : 처음엔 학습 오류가 적다가 편향이 높아질수록 교차 검증 오류와 가까워짐
      - 따라서 편향이 크면 더 많은 학습 데이터를 구하는 것은 도움이 되지 않음

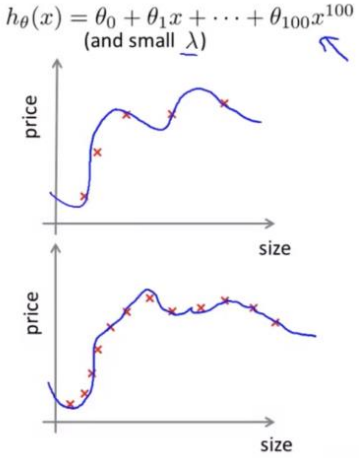




Bias & Variance (5/6)

- **Learning curves (2/2)**
  - **High variance**
    - 교차검증 오류 : 학습 예제가 증가함에 따라 감소하지만 여전히 큼
    - 학습 오류 : 학습 예제가 증가할수록 오류가 증가하지만 여전히 낮음
    - 즉, 높은 분산이 있는 알고리즘은 학습 오류와 교차 검증 오류 사이에 큰 갭이 있음
      - 더 많은 학습 데이터를 추가하게 되면 두 그래프가 서로 수렴함을 알 수 있음
      - 따라서 분산이 크면 더 많은 학습 데이터를 구하는 것이 도움이 됨





Bias & Variance (6/6)

- **학습 알고리즘 성능 향상 방법들과 각각의 역할**
  - 더 많은 학습 예제 수집 : 높은 분산 교정
  - 피처 셋 크기 감소/증가 : 높은 분산 교정/높은 편향 교정
  - 다항식 추가 : 높은 편향 교정
  - 정규화 파라미터 람다 값 감소/증가 : 높은 편향 교정/높은 분산 교정

Precision and Recall (1/2)

- **What is Precision and Recall?**
  - 머신러닝 분류 모델의 성능을 측정하는 데 사용하는 지표
  - 특정 클래스를 얼마나 정확하게 식별하는지를 나타냄
    - Precision : 모델이 positive로 예측한 샘플 중 얼마나 정확하게 실제 positive를 식별하는지 나타냄
    - Recall : 실제 positive인 샘플 중 얼마나 많은 샘플을 모델이 식별하는지 나타냄

$$(Precision) = \frac{TP}{TP + FP}$$

$$(Recall) = \frac{TP}{TP + FN}$$

Precision and Recall (1/2)

- **Trading off Precision and Recall**
  - 정밀도가 높을수록 재현율이 낮아짐
  - 재현율이 높을수록 정밀도가 낮아짐
    - 정밀도와 재현율 모두 높은 것이 좋지만 trade-off 관계에 있어 함께 늘리기 어려움
    - 이를 해결하기 위해 F1 score를 사용

      - F1 score : 정밀도와 재현율의 조화평균 (0~1 사이 값이며 1에 가까울수록 분류 성능 좋음)

$$F_1 \text{ Score:}$$

$$2\,\frac{Precision * Recall}{Precision + Recall} = 2\,\frac{PR}{P+R}$$

Decision Trees (1/3)

- **What is Decision tree?**
  - 기계학습에서 사용되는 지도학습 알고리즘 중 하나
  - 특성들을 기반으로 데이터를 분할하고, 각 분할된 영역에서의 예측 값을 결정하는 모델
    - 장점 : 직관적이고 해석이 쉬움
    - 단점 : 규칙을 추가하며 서브트리를 만들어 나갈수록 모델이 복잡해지고, 과적합에 빠지기 쉬움
      - 과적합을 피하기 위한 방법에 가지치기와 앙상블 기법이 있음

Decision Trees (2/3)

- **Pruning and Ensemble Methods**
  - Pruning (가지치기)
    - 단일 모델인 결정 트리의 크기를 제어하여 과적합 방지하고 복잡성을 줄이는 방식
      - 단일 모델의 복잡성을 줄이는 데 중점을 둠
  - Ensemble Methods (앙상블 기법)
    - 여러 모델의 다양성을 활용하여 일반화 성능을 향상시키는 방식
      - 다양한 모델의 예측을 결합하여 성능을 향상시키는 데 중점을 둠

Decision Trees (3/3)

- **Impurity**
  - 특정 노드에서의 데이터의 다양성을 측정하는 지표
    - 높을수록 데이터가 여러 클래스로 나누어져 있음
    - 지니 지수
      - 데이터의 불순도를 측정하는 다른 지표
      - 지니 지수의 최댓값은 0.5, 낮을수록 좋음

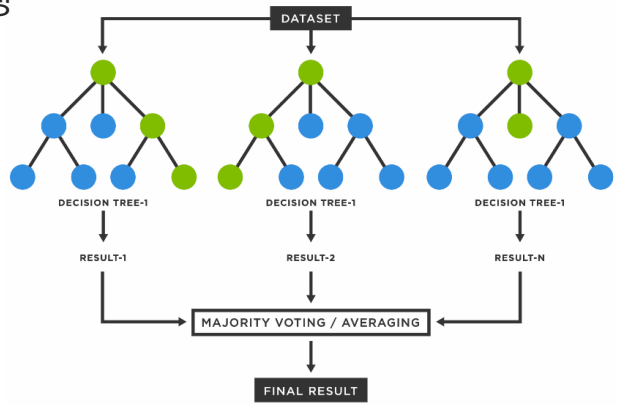
$$I(A) = 1 - \sum_{k=1}^{m} p_k^2$$

    - 엔트로피 지수
      - 데이터의 불확실성을 측정하는 지표
      - 엔트로피 지수가 낮을수록 좋음

$$E = -\sum_{i=1}^{k} p_i \log_2(p_i)$$

Random forest (1/2)

- **What is Random forest?**
  - 앙상블 학습 기법 중 하나
  - 다양한 종류의 데이터에 적용할 수 있는 유연성, 일반적으로 다른 모델보다 안정적인 예측 제공
  - 여러 개의 의사결정나무를 조합하여 더 강력한 모델을 생성하는 방법
    - 각각의 의사결정나무는 서로 다른 특성 부분집합을 사용하여 독립적으로 학습하고, 그 예측을 결합하여 최종 예측을 수행



Random forest (2/2)

- **Example**
  - iris data set을 사용한 간단한 분류 문제
    - 데이터 셋을 로드하고 test, train로 분리
    - 'RandomForestClassifier'을 사용하여 모델을 생성하고 학습 시킴
    - 'n_estimators' : 포레스트 내 트리의 개수 지정
    - Test 데이터에 대한 예측 수행 후 정확도 출력

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
```

```
# iris 데이터 로드
iris = load_iris()
X = iris.data
y = iris.target

# 데이터를 학습용과 테스트용으로 나눔
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# 랜덤 포레스트 모델 생성 및 학습
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

# 테스트 데이터에 대한 예측
y_pred = rf_model.predict(X_test)

# 정확도 출력
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

경상국립대학교

Gyeongsang National University

Improving lives through learning

IDEALAB