Supervised Learning with Neural Networks

신경망이란?

신경망은 생물의 신호를 주고받는 방식을 활용해 만든 학습 알고리즘이다.

우리가 주택의 가격을 예측한다고 해보자.

신경망이 하는 일은 주택의 크기를 입력으로 받아서 함수를 계산하고 가격이라는 결과값을 도출하는 것이다.

주택의 크기뿐만 아니라 침실의 크기, 가족의 인원, 우편번호, 재산 등 다양한 feature들을 더 고려하여 주택의 가격을 예측할 수도 있다.

이렇게 더 많은 특성들을 고려하면 더 큰 신경망을 가지게 된다.

이때 우리는 많은 데이터들을 가지고 x,y 값만을 주고, 그 내부의 뉴런들의 값은 컴퓨터가 스스로 학습하게 한다.

분야에 따라 서로 다른 신경망이 적용됨.

ex) 부동산- standard, 이미지 -cnn , 음성인식,언어 - rnn, 자율주행- custom된 혹은 hybrid network를 사용

Neural Network examples

-Standard NN

-Convolutional NN

-Recurrnet NN

Strurctured Data - database로 정리된 데이터, 열로 나열됨.

Unstructured Data - 이미지, 텍스트, 음성

신경망 덕분에 컴퓨터가 unstructured data를 이해하는 정도가 빠르게 발전함.

그렇다면 딥러닝이 왜 이렇게 급부상하게 되었을까?

traditional learning algorithm(SVM, logistic regression..)의 경우 data수가 증가하면 성능이 늘어나지만, 일정량의 data이상이 주어지면 성능이 정체함. 점차 사람들이 인터넷 사회에서 살게 되면서 데이터수도 방대하게 증가하게됨. 따라서 기존의 알고리즘만으로는 현재의 방대한 데이터를 감당하기 어려움. 그러나 복잡한 Neural network을 사용하면 할수록 성능이 증가함.

큰 성능을 발휘하기 위해서는 [1.충분히 큰 신경망 2.많은 데이터]가 필요함.

여기서 큰 신경망은 많은 hidden unit, 많은 파라미터 등을 의미함.

물론 큰 신경망으로 큰 성능 향상을 이루었지만, 그건 한계가 있음.

+ data 수가 적다면 다른 복잡하고 더 큰 신경망보다 SVM이 나을 수도 있음. data 수가 적은 경우 알고리즘의 상대적 순위가 잘 정의되어있지 않기 때문이다.

Deep Learning Process에 기여하는 요소

1. Data
2. Computation
   - 빠른 계산 속도=> 빠른 결과
   specialized HW (ex. GPU), 빠른 네트워크
3. Algorithms

ex) Sigmoid 함수-> Relu 함수

sigmoid에서 경사하강법을 사용하면 기울기가 거의 0 인 부분에서 학습속도가 매우 느려지는 문제 발생 (파라미터가 매우 천천히 바뀜)

Logistic Regression(로지스틱 회귀)

이진 분류를 위한 알고리즘

Binary Classification(이진 분류)란?

이진분류에서는 특성 벡터 x 를 가지고 그에 대한 label y 가 0 인지 아닌지를 분류할 수 있도록 하는 것이 목적이다. 예를 들어, 고양이 사진이 하나 있을 때, 고양이인지 아닌지 분류하는 문제. 고양이라면 (1)을 출력하고 고양이가 아니면 (0)을 출력하고자 하는 것이다.

그렇다면 우리가 사진을 고양이인지 아닌지 분류할 때 Logisitic Regression 이 어떻게 사용되는지 알아보자.

그렇다면 컴퓨터가 사진을 어떤 방식으로 저장하고 있고, 우리가 이 데이터를 어떻게 처리할 것인지 알아보아야 한다.

사진은 어떤 방식으로 컴퓨터에 저장될까?

한 픽셀은 RGB 의 값을 가진다. 따라서 컴퓨터는 64*64 픽셀의 이미지를 저장할 때, RGB 채널 3 개의 64*64 행렬로 저장한다. 각 픽셀의 강도 값들은 RGB 채널에 각각의 값을 가지고 있으므로, 해당 사진의 feature vector 의 차원은 64*64*3 이다.

일단 기호들을 알아보자.

x : n_x 차원 상의 벡터, feature vector.

ˆy : 예측값. 이진 분류에서는 P(y=1|x)라고 표현할 수 있음. y 가 1 일 확률을 뜻한다.

y : 실제값

* 파라미터인 w 는 똑같이 n_x 차원 상의 벡터이고 b 는 실수이다.

로지스틱 회귀를 학습할 수 있는 **cost function** 에 대해 알아보자.

입력값인 x 와 파라미터 w,b 가 주어졌을 때 어떻게 y 값을 예측할 수 있을까??

$$\hat{y} = \sigma(w^T * x + b)$$

sigmoid 가 없는 ˆy = w^T * x + b 는 이진 분류를 위한 좋은 알고리즘은 아니다. 왜냐하면 이진 분류는 0 또는 1 일 확률로 나타낼 수 있는데 해당 선형함수로는 음수나 1 보다 큰 수 가 표현되므로, 0~1 사이의 확률값을 나타내기에 적절하지 않음. 따라서 해당 함수에 sigmoid 를 적용하여 출력하면 0~1 사이의 값으로 표현할 수 있다! 따라서 이진분류를 수행해야 하는 경우 위의 방식처럼 0~1 사이의 값으로 나타낼 수 있는 함수를 씌워준다.

다음으로는 학습이 잘 되고 있는지를 확인할 수 있는 **loss(error) function** 에 대해 알아보자.

$$L(\hat{y}, y) = (\hat{y} - y)^2 / 2$$

* 로지스틱 회귀문제에서는 잘 사용하지 않는다. 왜냐하면 매개변수들을 학습하기 위해 풀어야 할 최적화 함수가 nonconvex 이기 때문이다. 즉, 여러 개의 local minimum 값을 가지게 되어 gradient descent 를 사용하였을 때 전역적인 최적값을 찾을 수 없을 수 있기 때문이다.

우리는 최적화 문제가 CONVEX 해질 수 있는 또다른 LOSS FUNCTION 을 정의하여 해결할 수 있다.

L(ˆy , y) = - (ylogˆy + (1-y)log(1-ˆy))
그렇다면 왜 제곱오차함수를 사용하지 않고 아래의 함수를 사용하는 것일까? 두 가지 case 를 통해 이해해보자.

근본적으로 loss function 을 통해 구한 에러값은 최대한 작아야 한다.

case 1) y=1

L: -ylogˆy 이고, 그러면 logˆy 의 값이 최대한 커져야 한다. 따라서 ˆy 의 값이 커져야 한다. 그러나 예측값(ˆy)은 시그모이드 함수 값이므로 1 보다 커질 수 없다.

case 2) y=0

L:- log(1-ˆy) 이고, 그러면 log(1-ˆy)의 값이 최대한 커져야 하므로, ˆy 의 값이 최대한 작아야 한다. 그러나 예측값은 0 보다 작아질 수 없다.

** 이는 비공식적인 검증!! 하지만 이러한 성질을 가진 함수들은 많음. 그런데 왜 해당 함수를 쓰는건지 공식적인 증명을 찾아보자.**

Cross-Entropy loss function 과 sigmoid 를 함께 사용하여 로지스틱 회귀 모델을 학습하면 convex 한 손실 함수로 변형할 수 있게 되고, 이에 gradient descent 를 통해 파라미터를 업데이트 할 수 있게 된다!

Loss function 은 훈련 샘플 하나에 관하여 정의되어서 그 하나가 얼마나 잘 예측되었는지를 보여주는 것이고, **cost function** 은 전체 데이터셋에 대해 얼마나 잘 예측되었는지 측정해주는 함수이다.

$$J(w,b) = -\frac{1}{m}\Sigma_{i=1}^{i=m}(y^{(i)}\log\hat{y}^{(i)} + (1-y^{(i)})\log(1-\hat{y}^{(i)}))$$

J(w,b)= Σ L(ˆy (i), y(i)) /m

비용함수는 손실함수를 각각의 샘플에 적용한 값의 평균이다. 따라서 모델의 파라미터를 조정하는 데 사용되는 것은 비용함수라고 볼 수 있다.

우리의 목적은 실제값과 비슷한 예측값을 도출하는 것이기 때문에, 해당 비용함수 J 를 최소화해주는 매개변수 w 와 b 를 찾는 것이 중요하다. 로지스틱 회귀는 아주 작은 신경망과 같다...! 이를 다음강의에서 보도록 하자.

Gradient Descent(경사하강법)

우리는 손실함수 J(w,b)를 최소화하는 w,b 를 찾아야 한다. 이를 찾을 때 사용하는 알고리즘 중 하나이다. 처음에 시작하는 지점은 보통 0 이다. (초기화 값:0) Convex 함수의 경우 어디서 시작해도 항상 같은 지점에 도달하기에 도달 속도에 차이가 있을 수는 있으나

시작점의 의미가 크지 않다. 시작점으로부터 가장 가파른 기울기를 가진 방향(=가장 빨리 내려올 수 있는 방향)으로 한단계씩 내려간다.

α : learning rate. 경사하강법을 반복할 때 한 step 의 크기.

dw : J(w)의 해당 w 값에서의 미분계수(비용함수의 기울기, J 가 w 방향으로 얼만큼 기울어졌는지를 나타냄). 이 값이 클수록 w 값이 크게 변함.

- $w : w - \alpha\frac{dJ(w,b)}{dw}$
- $b : b - \alpha\frac{dJ(w,b)}{db}$

Derivatives

삼각형을 그려보면 알 수 있다. 밑변과 높이의 비율이 달라짐에 따라 기울기가 바뀜.

도함수를 통해 원함수의 값이 얼만큼 더해지는지를 알 수 있다.

강의 들으면서 생긴 궁금증

1. 왜 불연속적인 값으로 분류하면서 회귀라고 부를까?

보통 우리가 회귀(Regression)라고 말할 때는 주로 연속적인 값을 예측하는 것을 의미합니다. 그러나 로지스틱 회귀에서의 "회귀"는 주로 "로지스틱 함수"에 대한 회귀를 의미합니다. 로지스틱 함수는 입력 변수의 선형 결합을 통해 출력 변수(확률)를 예측하기 위해 사용됩니다. 따라서 여기서의 회귀는 입력 변수와 출력 변수 간의 관계를 로지스틱 함수로 모델링하는 과정을 의미합니다. (logistic function = sigmoid)

2. 왜 loss function 으로 저 함수를 사용하는지 아직 이해가 안되었따..

3. 경사하강법은 convex 함수에서만 쓸 수 있는건가? Logistic regression 에서 어떻게 gradient descent 를 쓸 수 있는거지?

로지스틱 회귀에서는 Cross-Entropy loss function 과 sigmoid 를 사용하여 로 convex 한 손실 함수로 만든 다음에 gradient descent 를 사용하기 때문에 가능하다.